

Chapter 5

Analyzing Vulnerability Scans

THIS CHAPTER COVERS THE FOLLOWING COMPTIA PENTEST+ EXAM OBJECTIVES:

Domain 2: Information Gathering and Vulnerability Identification

- ✓ **2.3 Given a scenario, analyze vulnerability scanning results.**
 - Asset categorization
 - Adjudication
 - False positives
 - Prioritization of vulnerabilities
 - Common themes
 - Vulnerabilities
 - Observations
 - Best Practices
- ✓ **2.5 Explain weaknesses related to specialized systems.**
 - ICS
 - SCADA
 - Mobile
 - IoT
 - Embedded
 - Point-of-sale system
 - Application containers
 - RTOS

Domain 4: Penetration Testing Tools

- ✓ **4.2 Compare and contrast various use cases of tools.**
 - Use cases
 - Vulnerability scanning





Penetration testers spend a significant amount of time analyzing and interpreting the reports generated by vulnerability scanners, in search of vulnerabilities that may be exploited to gain a foothold on a target system. Although scanners are extremely effective at automating the manual work of vulnerability identification, the results that they generate require interpretation by a trained analyst. In this chapter, you will learn how penetration testers apply their knowledge and experience to the review of vulnerability scan reports.



Real World Scenario

Analyzing a Vulnerability Report

Let's again return to the penetration test of MCDS, LLC that we've been building over the last two chapters. You've now conducted an Nmap to perform your initial reconnaissance and developed a vulnerability scanning plan based upon those results.

After developing that plan, you ran a scan of one of the MCDS web servers and should have found three potential vulnerabilities. These vulnerabilities are discussed later in the chapter, but they are as follows:

- Internal IP disclosure (see Figure 5.19)
- CGI generic SQL injection (see Figure 5.21)
- SSLv3 Padding Oracle on Downgraded Legacy Encryption (POODLE) (see Figure 5.24)

As you read through this chapter, consider how you might exploit these vulnerabilities to attack the target system. We will return to this exercise in Lab Activity 5.3 to develop an exploitation plan.

Reviewing and Interpreting Scan Reports

Vulnerability scan reports provide analysts with a significant amount of information that assists with the interpretation of the report. In addition to the high-level report examples shown in Chapter 4, "Vulnerability Scanning," vulnerability scanners provide

detailed information about each vulnerability that they identify. Figure 5.1 shows an example of a single vulnerability reported by the Nessus vulnerability scanner.

FIGURE 5.1 Nessus vulnerability scan report

A **SSH Weak Algorithms Supported** **H** **Plugin Details**

B **Description**
Nessus has detected that the remote SSH server is configured to use the Arcfour stream cipher or no cipher at all. RFC 4253 advises against using Arcfour due to an issue with weak keys.

C **Solution**
Contact the vendor or consult product documentation to remove the weak ciphers.

D **See Also**
<https://tools.ietf.org/html/rfc4253#section-6.3>

E **Output**
The following weak server-to-client encryption algorithms are supported :
arcfour
arcfour128
arcfour256
The following weak client-to-server encryption algorithms are supported :
arcfour
arcfour128
arcfour256

F **Port** **Hosts**
22 (tcp / ssh) 52.200.148.151, 54.172.251.189, 54.174.107.98

H **Plugin Details**
Severity: Medium
ID: 90317
Version: \$Revision: 1.2 \$
Type: remote
Family: Misc.
Published: 2016/04/04
Modified: 2016/04/26

G **Risk Information**
Risk Factor: Medium
CVSS Base Score: 4.3
CVSS Vector: CVSS2#AV:N/AC:M/Au:N/C:P/I:N/A/N

Let's take a look at this report, section by section, beginning at the top left and proceeding in a counterclockwise fashion.

At the very top of the report, labeled A, we see two critical details: the *name of the vulnerability*, which offers a descriptive title, and the *overall severity* of the vulnerability, expressed as a general category, such as low, medium, high, or critical. In this example report, the scanner is reporting that a server's Secure Shell (SSH) service supports weak encryption algorithms. It is assigned to the medium severity category.

Next, in section B, the report provides a *detailed description* of the vulnerability. In this case, the vulnerability has a fairly short, two-sentence description, but these descriptions can be several paragraphs long depending on the complexity of the vulnerability. In this case, the description informs us that the server's SSH service only supports the insecure Arcfour stream cipher and explains that this service has an issue with weak encryption keys.

Section C of the report provides a *solution* to the vulnerability. When possible, the scanner offers detailed information about how system administrators, security professionals, network engineers, and/or application developers may correct the vulnerability. In this case, no detailed solution is available and administrators are advised to contact the vendor for instructions on removing the weak cipher support.

In section D, “See Also,” the scanner provides *references* where administrators can find more details on the vulnerability described in the report. In this case, the scanner refers the reader to Internet Engineering Task Force (IETF) Request for Comments (RFC) 4253, which describes the SSH protocol in great detail. It includes the following advice regarding the Arcfour cipher: “The Arcfour cipher is believed to be compatible with the RC4 cipher. Arcfour (and RC4) has problems with weak keys, and should be used with caution.”

The *output* of the report (E) shows the detailed information returned by the remote system when probed for the vulnerability. This information can be extremely valuable to an analyst because it often provides the verbatim output returned by a command. Analysts can use this to better understand why the scanner is reporting a vulnerability, identify the location of a vulnerability, and potentially identify false positive reports. In this case, the output section shows the specific weak ciphers supported by the SSH server.

The *port/hosts* section (F) provides details on the server(s) that contain the vulnerability as well as the specific services on that server that have the vulnerability. In this case, the same vulnerability exists on three different servers: those at IP addresses 10.12.148.151, 10.14.251.189, and 10.14.107.98. These three servers are all running an SSH service on TCP port 22 that supports the Arcfour cipher.

The *risk information* section (G) includes useful information for assessing the severity of the vulnerability. In this case, the scanner reports that the vulnerability has an overall risk of Medium (consistent with the tag next to the vulnerability title). It also provides details about how the vulnerability rates when using the Common Vulnerability Scoring System (CVSS). In this case, the vulnerability has a CVSS base score of 4.3 and has the following CVSS vector:

```
CVSS2#AV:N/AC:M/Au:N/C:P/I:N/A:N
```

We’ll discuss the details of CVSS scoring in the next section of this chapter.

The final section of the vulnerability report (H) provides details on the vulnerability scanner plug-in that detected the issue. This vulnerability was reported by Nessus plug-in ID 90317, which was published in April 2016.



Although this chapter focuses on interpreting the details of a Nessus vulnerability scan, the process is extremely similar for other vulnerability scanners. The reports generated by different products may vary in format, but they generally provide the same information. For example, Figure 5.2 shows the output of a Qualys vulnerability report, while Figure 5.3 shows the output of an OpenVAS report.

FIGURE 5.2 Qualys vulnerability scan report

4 OpenSSL oracle padding vulnerability(CVE-2016-2107) port 443/tcp over SSL


QID: 38626
Category: General remote services
CVE ID: CVE-2016-2107
Vendor Reference: [OpenSSL Security Advisory 20160503](#)
Bugtraq ID: 91787
Service Modified: 05/24/2016
User Modified: -
Edited: No
PCI Vuln: No
Ticket State:

THREAT:
 The OpenSSL Project is an Open Source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS) protocols as well as a general purpose cryptography library.
 OpenSSL contains the following vulnerability:
 A MITM attacker can use a padding oracle attack to decrypt traffic when the connection uses an AES CBC cipher and the server support AES-NI. Affected Versions: OpenSSL 1.0.2 prior to OpenSSL 1.0.2h OpenSSL 1.0.1 prior to OpenSSL 1.0.1i

IMPACT:
 A MITM attacker can use a padding oracle attack to decrypt traffic.

SOLUTION:
 OpenSSL version 1.0.2h and 1.0.1i have been released to address these issues. Refer to [OpenSSL Advisory](#) to obtain more information.
 Patch:
 Following are links for downloading patches to fix the vulnerabilities:
[OpenSSL Security Advisory 3rd May 2016](#)

COMPLIANCE:
 Not Applicable

EXPLOITABILITY:
 [The Exploit-DB](#)
Reference: CVE-2016-2107
Description: OpenSSL - Padding Oracle in AES-NI CBC MAC Check - The Exploit-DB Ref : 39768
Link: <http://www.exploit-db.com/exploits/39768>

ASSOCIATED MALWARE:
 There is no malware information for this vulnerability

RESULTS:
 No results available

FIGURE 5.3 OpenVAS vulnerability scan report

Dashboard Home Assets Hosts Networks Configuration Reports Administration Help

Result: TCP timestamps

Vulnerability TCP timestamps **Severity** 3 (Low) **Qid** 8076 **Host** 10.1.107.88 **Location** generic/tcp **Actions**

Summary
 The remote host implements TCP timestamps and therefore allows to compute the uptime.

Vulnerability Detection Result
 cv was detected that the host implements RFC1323.
 The following timestamps were retrieved with a delay of 1 second (s)-between:
 Packet 1: 3027989
 Packet 2: 3027989

Impact
 A side effect of this feature is that the uptime of the remote host can sometimes be computed.

Solution
Solution type: Mitigation
 To disable TCP timestamps on Linux add the line "net.ipv4.tcp_timestamps = 0" to /etc/sysctl.conf. Execute 'sysctl' if to apply the settings at runtime.
 To disable TCP timestamps on Windows execute 'netsh int tcp set global timestamps=disabled'.
 Starting with Windows Server 2008 and Vista, the timestamp can not be completely disabled.
 The default behavior of the TCP/IP stack on this Systems is to not use the Timestamp options when initiating TCP connections, but use them if the TCP peer that is initiating communication includes them in their synchroise (SYN) segment.
 See also: <http://www.microsoft.com/en-us/download/details.aspx?id=9152>

Affected Software/OS
 TCP/IPv4 implementations that implement RFC1323.

Vulnerability Insight
 The remote host implements TCP timestamps, as defined by RFC1323.

Vulnerability Detection Method
 Special IP packets are forged and sent with a little delay in between to the target IP. The responses are searched for a timestamps. If found, the timestamps are reported.
 Details: TCP timestamps (SOO: 1.3.6.1.4.1.2062.1.0.80981)
 Version used: sNessus 9

References
 Other: <http://www.ietf.org/rfc/rfc1323.txt>

Understanding CVSS

The *Common Vulnerability Scoring System (CVSS)* is an industry standard for assessing the severity of security vulnerabilities. It provides a technique for scoring each vulnerability on a variety of measures. Cybersecurity analysts often use CVSS ratings to prioritize response actions.

Analysts scoring a new vulnerability begin by rating the vulnerability on six different measures:

- Access vector
- Access complexity
- Authentication
- Confidentiality
- Integrity
- Availability

Each measure is given both a descriptive rating and a numeric score. The first three measures evaluate the exploitability of the vulnerability, whereas the last three evaluate the impact of the vulnerability.

Access Vector Metric

The *access vector metric* describes how an attacker would exploit the vulnerability and is assigned according to the criteria shown in Table 5.1.

TABLE 5.1 CVSS access vector metric

Value	Description	Score
Local (L)	The attacker must have physical or logical access to the affected system.	0.395
Adjacent Network (A)	The attacker must have access to the local network that the affected system is connected to.	0.646
Network (N)	The attacker can exploit the vulnerability remotely over a network.	1.000

Access Complexity Metric

The *access complexity metric* describes the difficulty of exploiting the vulnerability and is assigned according to the criteria shown in Table 5.2.

TABLE 5.2 CVSS access complexity metric

Value	Description	Score
High (H)	Exploiting the vulnerability requires “specialized” conditions that would be difficult to find.	0.350
Medium (M)	Exploiting the vulnerability requires “somewhat specialized” conditions.	0.610
Low (L)	Exploiting the vulnerability does not require any specialized conditions.	0.710

Authentication Metric

The *authentication metric* describes the authentication hurdles that an attacker would need to clear to exploit a vulnerability and is assigned according to the criteria in Table 5.3.

TABLE 5.3 CVSS authentication metric

Value	Description	Score
Multiple (M)	Attackers would need to authenticate two or more times to exploit the vulnerability.	0.450
Single (S)	Attackers would need to authenticate once to exploit the vulnerability.	0.560
None (N)	Attackers do not need to authenticate to exploit the vulnerability.	0.704

Confidentiality Metric

The *confidentiality metric* describes the type of information disclosure that might occur if an attacker successfully exploits the vulnerability. The confidentiality metric is assigned according to the criteria in Table 5.4.

TABLE 5.4 CVSS confidentiality metric

Value	Description	Score
None (N)	There is no confidentiality impact.	0.000
Partial (P)	Access to some information is possible, but the attacker does not have control over what information is compromised.	0.275
Complete (C)	All information on the system is compromised.	0.660

Integrity Metric

The *integrity metric* describes the type of information alteration that might occur if an attacker successfully exploits the vulnerability. The integrity metric is assigned according to the criteria in Table 5.5.

TABLE 5.5 CVSS integrity metric

Value	Description	Score
None (N)	There is no integrity impact.	0.000
Partial (P)	Modification of some information is possible, but the attacker does not have control over what information is modified.	0.275
Complete (C)	The integrity of the system is totally compromised and the attacker may change any information at will.	0.660

Availability Metric

The *availability metric* describes the type of disruption that might occur if an attacker successfully exploits the vulnerability. The availability metric is assigned according to the criteria in Table 5.6.

TABLE 5.6 CVSS authentication metric

Value	Description	Score
None (N)	There is no availability impact.	0.000
Partial (P)	The performance of the system is degraded.	0.275
Complete (C)	The system is completely shut down.	0.660



The Forum of Incident Response and Security Teams (FIRST) released CVSS version 3.0 in June 2015, but the new version of the standard has not yet been widely adopted. As of this writing, major vulnerability scanners still use CVSS version 2.0.

Interpreting the CVSS Vector

The CVSS *vector* uses a single-line format to convey the ratings of a vulnerability on all six of the metrics described in the preceding sections. For example, recall the CVSS vector presented in Figure 5.1:

CVSS2#AV:N/AC:M/Au:N/C:P/I:N/A:N

This vector contains seven components. The first section, CVSS2#, simply informs the reader (human or system) that the vector was composed using CVSS version 2. The next six sections correspond to each of the six CVSS metrics. In this case, the SSH cipher vulnerability in Figure 5.1 received the following ratings:

- Access Vector: Network (score: 1.000)
- Access Complexity: Medium (score: 0.610)
- Authentication: None (score: 0.704)
- Confidentiality: Partial (score: 0.275)
- Integrity: None (score: 0.000)
- Availability: None (score: 0.000)

Summarizing CVSS Scores

The CVSS vector provides good detailed information on the nature of the risk posed by a vulnerability, but the complexity of the vector makes it difficult to use in prioritization exercises. For this reason, analysts can calculate the CVSS *base score*, which is a single number representing the overall risk posed by the vulnerability. Arriving at the base score requires first calculating the *exploitability score*, *impact score*, and *impact function*.

Calculating the Exploitability Score

Analysts may calculate the exploitability score for a vulnerability using this formula:

$$\text{Exploitability} = 20 \times \text{AccessVector} \times \text{AccessComplexity} \times \text{Authentication}$$

Plugging in values for our SSH vulnerability, we get this:

$$\text{Exploitability} = 20 \times 1.000 \times 0.610 \times 0.704$$

$$\text{Exploitability} = 8.589$$

Calculating the Impact Score

Analysts may calculate the impact score for a vulnerability using this formula:

$$\text{Impact} = 10.41 \times (1 - (1 - \text{Confidentiality}) \times (1 - \text{Integrity}) \times (1 - \text{Availability}))$$

Plugging in values for our SSH vulnerability, we get this:

$$\text{Impact} = 10.41 \times (1 - (1 - 0.275) \times (1 - 0) \times (1 - 0))$$

$$\text{Impact} = 10.41 \times (1 - (0.725) \times (1) \times (1))$$

$$\text{Impact} = 10.41 \times (1 - 0.725)$$

$$\text{Impact} = 10.41 \times 0.275$$

$$\text{Impact} = 2.863$$

Determining the Impact Function Value

The impact function is a simple check. If the impact score is 0, the impact function value is also 0. Otherwise, the impact function value is 1.176. So, in our example case, the result is as follows:

$$\text{ImpactFunction} = 1.176$$

Calculating the Base Score

With all of this information at hand, we can now calculate the CVSS base score using this formula:

$$\text{BaseScore} = ((0.6 \times \text{Impact}) + (0.4 \times \text{Exploitability}) - 1.5) \times \text{ImpactFunction}$$

Plugging in values for our SSH vulnerability, we get this:

$$\text{BaseScore} = ((0.6 \times 2.863) + (0.4 \times 8.589) - 1.5) \times 1.176$$

$$\text{BaseScore} = (1.718 + 3.436 - 1.5) \times 1.176$$

$$\text{BaseScore} = 3.654 \times 1.176$$

$$\text{BaseScore} = 4.297$$

Rounding this result, we get a CVSS base score of 4.3, which is the same value found in Figure 5.1.

Categorizing CVSS Base Scores

Many vulnerability scanning systems further summarize CVSS results by using risk categories rather than numeric risk ratings. For example, Nessus uses the risk rating scale shown in Table 5.7 to assign vulnerabilities to categories based on their CVSS base scores.

TABLE 5.7 Nessus risk categories and CVSS scores

CVSS score	Risk category
Under 4.0	Low
4.0 or higher, but less than 6.0	Medium
6.0 or higher, but less than 10.0	High
10.0	Critical

Continuing with the SSH vulnerability example from Figure 5.1, we calculated the CVSS score for this vulnerability as 4.3. This places it into the Medium risk category, as shown in the screen header in Figure 5.1.

Validating Scan Results

Cybersecurity analysts interpreting reports often perform their own investigations to confirm the presence and severity of vulnerabilities. This adjudication may include the use of external data sources that supply additional information valuable to the analysis.

False Positives

Vulnerability scanners are useful tools, but they aren't foolproof. Scanners do sometimes make mistakes for a variety of reasons. The scanner might not have sufficient access to the target system to confirm a vulnerability, or it might simply have an error in a plug-in that generates an erroneous vulnerability report. When a scanner reports a vulnerability that does not exist, this is known as a *false positive error*.

Cybersecurity analysts should confirm each vulnerability reported by a scanner. In some cases, this may be as simple as verifying that a patch is missing or an operating system is outdated. In other cases, verifying a vulnerability requires a complex manual process that simulates an exploit. For example, verifying a SQL injection vulnerability may require actually attempting an attack against a web application and verifying the result in the backend database.

When verifying a vulnerability, analysts should draw on their own expertise as well as the subject matter expertise of others throughout the organization. Database administrators, system engineers, network technicians, software developers, and other experts have domain knowledge that is essential to the evaluation of a potential false positive report.

Documented Exceptions

In some cases, an organization may decide not to remediate a vulnerability for one reason or another. For example, the organization may decide that business requirements dictate the use of an operating system that is no longer supported. Similarly, development managers may decide that the cost of remediating a vulnerability in a web application that is exposed only to the internal network outweighs the security benefit.

Unless analysts take some action to record these exceptions, vulnerability scans will continue to report them each time a scan runs. It's good practice to document exceptions in the vulnerability management system so that the scanner knows to ignore them in future reports. This reduces the level of noise in scan reports and increases their usefulness to analysts.



Be careful when deciding to allow an exception. As discussed in Chapter 4, many organizations are subject to compliance requirements for vulnerability scanning. Creating an exception may violate those compliance obligations or go against best practices for security.

Understanding Informational Results

Vulnerability scanners often supply very detailed information when run using default configurations. Not everything reported by a vulnerability scanner actually represents a significant security issue. Nevertheless, scanners provide as much information as they are able to determine to show the types of information that an attacker might be able to gather when conducting a reconnaissance scan. This information also provides important reconnaissance for a penetration tester seeking to gather information about a potential target system.

Figure 5.4 provides an example of a high-level report generated from a vulnerability scan run against a web server. Note that about two-thirds of the vulnerabilities in this report fit into the “Info” risk category. This indicates that the plug-ins providing results are not even categorized according to the CVSS. Instead, they are simply informational results. In some cases, they are simply observations that the scanner made about the system, while in other cases they may refer to a lack of best practices in the system configuration. Most organizations do not go to the extent of addressing all informational results about a system because it can be difficult, if not impossible, to do so.

FIGURE 5.4 Scan report showing vulnerabilities and best practices



A penetration tester encountering the scan report in Figure 5.4 should first turn their attention to the high-severity SQL injection vulnerability that exists. That is a very serious vulnerability that may provide a direct path to compromising the system's underlying database. If that exploitation does not bear fruit, the seven medium-severity vulnerabilities may offer potential access. The remaining informational vulnerabilities are useful for reconnaissance but may not provide a direct path to compromise.

Many organizations will adopt a formal policy for handling these informational messages from a remediation perspective. For example, some organizations may decide that once a message appears in two or three consecutive scans, they will create a journal entry documenting the actions they took in response to the message or the reasons they chose not to take actions. This approach is particularly important for highly audited organizations that have stringent compliance requirements. Creating a formal record of the decision-making process satisfies auditors that the organization conducted due diligence.

Reconciling Scan Results with Other Data Sources

Vulnerability scans should never take place in a vacuum. Penetration testers interpreting these reports should also turn to other sources of security information as they perform their analyses. When available to a penetration tester, the following information sources may also contain valuable information:

- *Logs* from servers, applications, network devices, and other sources that might contain information about possible attempts to exploit detected vulnerabilities
- *Security information and event management (SIEM)* systems that correlate log entries from multiple sources and provide actionable intelligence
- *Configuration management systems* that provide information on the operating system and applications installed on a system

Each of these information sources can prove invaluable when a penetration tester attempts to reconcile a scan report with the reality of the organization's computing environment.

Trend Analysis

Trend analysis is also an important part of a vulnerability scanning program. Managers should watch for overall trends in vulnerabilities, including the number of new vulnerabilities arising over time, the age of existing vulnerabilities, and the time required to remediate vulnerabilities. Figure 5.5 shows an example of the trend analysis reports available in Nessus SecurityCenter.

FIGURE 5.5 Vulnerability trend analysis

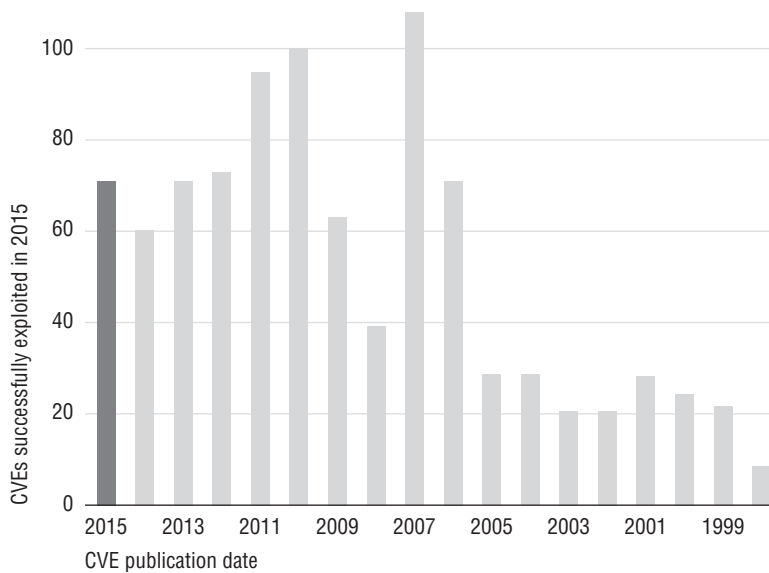
Source: Tenable Network Security, Inc.

Common Vulnerabilities

Each vulnerability scanning system contains plug-ins able to detect thousands of possible vulnerabilities, ranging from major SQL injection flaws in web applications to more mundane information disclosure issues with network devices. Though it's impossible to discuss each of these vulnerabilities in a book of any length, penetration testers should be familiar with the most commonly detected vulnerabilities and some of the general categories that cover many different vulnerability variants.

Chapter 4 discussed the importance of regularly updating vulnerability scanners to make them effective against newly discovered threats. Although this is true, it is also important to note that even old vulnerabilities can present significant issues to the security of organizations. Each year Verizon conducts a widely respected analysis of all the data breaches they investigated over the course of the prior year. Figure 5.6 shows some of the results from the 2016 *Data Breach Investigations Report*, the last year for which this information is available.

Figure 5.6 underscores the importance of addressing old vulnerabilities and the stark reality that many organizations fail to do so. Many of the vulnerabilities exploited during data breaches in 2015 had been discovered more than a *decade* earlier. That's an astounding statistic.

FIGURE 5.6 Vulnerabilities exploited in 2015 by year of initial discovery

Source: Verizon Data Breach Investigations Report

Server and Endpoint Vulnerabilities

Computer systems are quite complex. Operating systems run on both servers and endpoints comprising millions of lines of code, and the differing combinations of applications they run makes each system fairly unique. It's no surprise, therefore, that many of the vulnerabilities detected by scans exist on server and endpoint systems, and these vulnerabilities are often among the most complex to remediate. This makes them attractive targets for penetration testers.

Missing Patches

Applying security patches to systems should be one of the core practices of any information security program, but this routine task is often neglected due to a lack of resources for preventive maintenance. One of the most common alerts from a vulnerability scan is that one or more systems on the network are running an outdated version of an operating system or application and require security patch(es). Penetration testers may take advantage of these missing patches and exploit operating system weaknesses.

Figure 5.7 shows an example of one of these scan results. The server located at 10.64.142.211 has a remote code execution vulnerability. Though the scan result is fairly brief, it does contain quite a bit of helpful information:

- The description tells us that this is a flaw in the Windows HTTP stack.
- The service information in the Output section of the report confirms that the server is running an HTTPS service on TCP port 443.

- We see in the header that this is a critical vulnerability, which is confirmed in the Risk Information section, where we see that it has a CVSS base score of 10.
- We can parse the CVSS vector to learn a little more about this vulnerability:
 - *AV:N* tells us that the vulnerability can be exploited remotely by a hacker over the network.
 - *AC:L* tells us that the access complexity is low, meaning that a relatively unskilled attacker can exploit it.
 - *Au:N* tells us that no authentication is required to exploit the vulnerability.
 - *C:C*, *I:C*, and *A:C* tell us that someone exploiting this vulnerability is likely to completely compromise the confidentiality, integrity, and availability of the system.

FIGURE 5.7 Missing patch vulnerability

CRITICAL MS15-034: Vulnerability in HTTP.sys Could Allow Remote Code Execution (...)

Description
The version of Windows running on the remote host is affected by a vulnerability in the HTTP protocol stack (HTTP.sys) due to improperly parsing crafted HTTP requests. A remote attacker can exploit this to execute arbitrary code with System privileges.

Solution
Microsoft has released a set of patches for Windows 7, 2008 R2, 8, 8.1, 2012, and 2012 R2.

See Also
<https://technet.microsoft.com/en-us/library/security/MS15-034>

Output
No output recorded.

Port **Hosts**
1837/tcp/www 10.64.1.80(1)

Plugin Details

Severity:	Critical
ID:	82828
Version:	\$Revision: 1.5 \$
Type:	remote
Family:	Windows
Published:	2015/04/18
Modified:	2015/09/14

Risk Information

Risk Factor:	Critical
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:N/D/RL:OF/RC:C
CVSS Temporal Score:	6.7
IAVM Severity:	1



We won't continue to parse the CVSS vectors for each of the vulnerabilities discussed in this chapter. However, you may wish to do so on your own as an exercise in assessing the severity of a vulnerability.

Fortunately, there is an easy way to fix this problem. The Solution section tells us that Microsoft has released patches for the affected operating systems, and the “See Also” section provides a direct link to the Microsoft security bulletin (MS15-034) that describes the issue and solution in greater detail.

Mobile Device Security

The section “Server and Endpoint Vulnerabilities” refers to the vulnerabilities typically found on traditional servers and endpoints, but it’s important to note that mobile devices have a host of security issues of their own and must be carefully managed and patched to remain secure.

The administrators of mobile devices can use a mobile device management (MDM) solution to manage the configuration of those devices, automatically installing patches, requiring the use of encryption, and providing remote wiping functionality. MDM solutions may also restrict the applications that can be run on a mobile device to those that appear on an approved list.

That said, mobile devices do not typically show up on vulnerability scans because they are not often sitting on the network when those scans run. Therefore, administrators should pay careful attention to the security of those devices, even when they do not show up as requiring attention after a vulnerability scan.

Unsupported Operating Systems and Applications

Software vendors eventually discontinue support for every product they make. This is true for operating systems as well as applications. Once the vendor announces the final end of support for a product, organizations that continue running the outdated software put themselves at a significant risk of attack. The vendor simply will not investigate or correct security flaws that arise in the product after that date. Organizations continuing to run the unsupported product are on their own from a security perspective, and unless you happen to maintain a team of operating system developers, that’s not a good situation to find yourself in.

From a penetration tester’s perspective, reports of unsupported software are a treasure trove of information. They’re difficult for IT teams to remediate and offer a potential avenue of exploitation.

Perhaps the most famous end of support for a major operating system occurred in July 2015 when Microsoft discontinued support for the more-than-a-decade-old Windows Server 2003. Figure 5.8 shows an example of the report generated by Nessus when it identifies a server running this outdated operating system.

We can see from this report that the scan detected two servers on the network running Windows Server 2003. The description of the vulnerability provides a stark assessment of what lies in store for organizations continuing to run any unsupported operating system:

Lack of support implies that no new security patches for the product will be released by the vendor. As a result, it is likely to contain security vulnerabilities. Furthermore, Microsoft is unlikely to investigate or acknowledge reports of vulnerabilities.

FIGURE 5.8 Unsupported operating system vulnerability

The screenshot displays a Nessus scan result for a critical vulnerability. The main title is 'Microsoft Windows Server 2003 Unsupported Installation Detection'. The description states that support for this operating system ended on July 14th, 2015, and that the lack of support implies no new security patches will be released. The solution is to 'Upgrade to a version of Windows that is currently supported.' The 'See Also' section provides a link to a Nessus.org advisory. The 'Output' section shows 'No output recorded.' Below this is a table with columns 'Port' and 'Hosts'. The 'Port' column shows 'N/A' and the 'Hosts' column shows '162.246.142.218, 162.246.142.220'. On the right side, the 'Plugin Details' section lists: Severity: Critical, ID: 84729, Version: \$Revision: 1.4 \$, Type: combined, Family: Windows, Published: 2015/07/14, Modified: 2015/10/21. The 'Risk Information' section lists: Risk Factor: Critical, CVSS Base Score: 10.0, CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C. The 'Vulnerability Information' section lists: CPE: cpe:/o:microsoft/windows_2003_server, Unsupported by vendor: true.

The solution for organizations running unsupported operating systems is simple in its phrasing but complex in implementation: “Upgrade to a version of Windows that is currently supported” is a pretty straightforward instruction, but it may pose a significant challenge for organizations running applications that can’t be upgraded to newer versions of Windows. In cases where the organization must continue using an unsupported operating system, best practice dictates isolating the system as much as possible, preferably not connecting it to any network, and applying as many compensating security controls as possible, such as increased monitoring and implementation of strict network firewall rules.

Buffer Overflows

Buffer overflow attacks occur when an attacker manipulates a program into placing more data into an area of memory than is allocated for that program’s use. The goal is to overwrite other information in memory with instructions that may be executed by a different process running on the system.

Buffer overflow attacks are quite commonplace and tend to persist for many years after they are initially discovered. For example, the 2016 *Verizon Data Breach Investigation Report* identified 10 vulnerabilities that were responsible for 85 percent of the compromises in their study. Among the top ten were four overflow issues:

- CVE 1999-1058: Buffer overflow in Vermillion FTP Daemon
- CVE 2001-0876: Buffer overflow in Universal Plug and Play (UPnP) on Windows 98, 98SE, ME, and XP
- CVE 2002-0126: Buffer overflow in BlackMoon FTP Server 1.0 through 1.5
- CVE 2003-0818: Multiple integer overflows in Microsoft ASN.1 library



One of the listed vulnerabilities is an “integer overflow.” This is simply a variant of a buffer overflow where the result of an arithmetic operation attempts to store an integer that is too large to fit in the specified buffer.

The four-digit number following the letters *CVE* in each vulnerability title indicates the year that the vulnerability was discovered. In a study of breaches that took place in 2015, four of the top ten issues causing breaches were exploits of overflow vulnerabilities that were between 12 and 16 years old!

Cybersecurity analysts discovering a buffer overflow vulnerability during a vulnerability scan should seek out a patch that corrects the issue. In most cases, the scan report will directly identify an available patch.

Privilege Escalation

Privilege escalation attacks seek to increase the level of access that an attacker has to a target system. They exploit vulnerabilities that allow the transformation of a normal user account into a more privileged account, such as the root superuser account.

In October 2016, security researchers announced the discovery of a Linux kernel vulnerability dubbed Dirty COW. This vulnerability, present in the Linux kernel for nine years, was extremely easy to exploit and provided successful attackers with administrative control of affected systems.

In an attempt to spread the word about this vulnerability and encourage prompt patching of Linux kernels, security researchers set up the `dirtycow.ninja` website, shown in Figure 5.9. This site provides details on the flaw and corrective measures.

FIGURE 5.9 Dirty COW website



Arbitrary Code Execution

Arbitrary code execution vulnerabilities allow an attacker to run software of their choice on the targeted system. This can be a catastrophic event, particularly if the vulnerability allows the attacker to run the code with administrative privileges. *Remote code execution* vulnerabilities are an even more dangerous subset of code execution vulnerabilities because the attacker can exploit the vulnerability over a network connection without having physical or logical access to the target system.

Figure 5.10 shows an example of a remote code execution vulnerability detected by Nessus.

FIGURE 5.10 Code execution vulnerability

The screenshot displays a Nessus vulnerability report for MS14-066. The report is divided into several sections: Description, Solution, See Also, Output, Plugin Details, Risk Information, and Vulnerability Information. The Description section states that the remote Windows host is affected by a remote code execution vulnerability due to improper processing of packets by the Secure Channel (Schannel) security package. The Solution section mentions that Microsoft has released a set of patches for Windows 2000, Vista, 2008, 7, 2008 R2, 8, 2012, 8.1, and 2012 R2. The See Also section provides a link to the Microsoft Security Bulletin MS14-066. The Output section shows "No output recorded." The Plugin Details section lists the severity as Critical, ID as 79838, version as \$Revision: 1.42 \$, type as remote, family as Windows, published as 2014/12/01, and modified as 2016/10/20. The Risk Information section lists the risk factor as Critical, CVSS Base Score as 10.0, CVSS Vector as CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C, CVSS Temporal Vector as CVSS2#E:ND/RL:OF/RC:C, and CVSS Temporal Score as 8.7. The Vulnerability Information section lists the CPE as cpe:/o:microsoft:windows, exploit availability as true, exploit ease as Exploits are available, patch/pub date as 2014/11/11, and vulnerability pub date as 2014/11/11.

Port	Hosts
445 (SMB)	10.64.143.200

Plugin Details	
Severity:	Critical
ID:	79838
Version:	\$Revision: 1.42 \$
Type:	remote
Family:	Windows
Published:	2014/12/01
Modified:	2016/10/20

Risk Information	
Risk Factor:	Critical
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:ND/RL:OF/RC:C
CVSS Temporal Score:	8.7

Vulnerability Information	
CPE:	cpe:/o:microsoft:windows
Exploit Available:	true
Exploit Ease:	Exploits are available
Patch/Pub Date:	2014/11/11
Vulnerability Pub Date:	2014/11/11

Notice that the CVSS access vector in Figure 5.10 shows that the access vector for this vulnerability is network based. This is consistent with the description of a remote code execution vulnerability. The impact metrics in the vector show that the attacker can exploit this vulnerability to completely compromise the system.

Fortunately, as with most vulnerabilities detected by scans, there is an easy fix for the problem. Microsoft has issued patches for the versions of Windows affected by the issue and describes them in Microsoft Security Bulletin MS14-066.

Hardware Flaws

While most vulnerabilities affect operating systems and applications, occasionally vulnerabilities arise that directly affect the underlying hardware running in an organization. These may arise due to firmware issues or, in rarer cases, may be foundational hardware issues requiring remediation.

Firmware Vulnerabilities

Many hardware devices contain *firmware*: computer code stored in nonvolatile memory on the device, where it can survive a reboot of the device. Firmware often contains the device's operating system and/or configuration information. Just like any other code, the code contained in firmware may contain security vulnerabilities.

In many cases, this code resides out of sight and out of mind for the IT team because it is initially provided by the manufacturer and often lacks both an automatic update mechanism and any integration with enterprise configuration management tools. Cybersecurity analysts should carefully monitor the firmware in use in their organizations and develop an updating procedure that applies security updates as they are released.

For penetration testers, firmware vulnerabilities present a unique opportunity because they often remain unpatched. A tester may use a firmware vulnerability in a nonstandard computing device to gain a foothold on a network and then pivot to other systems.

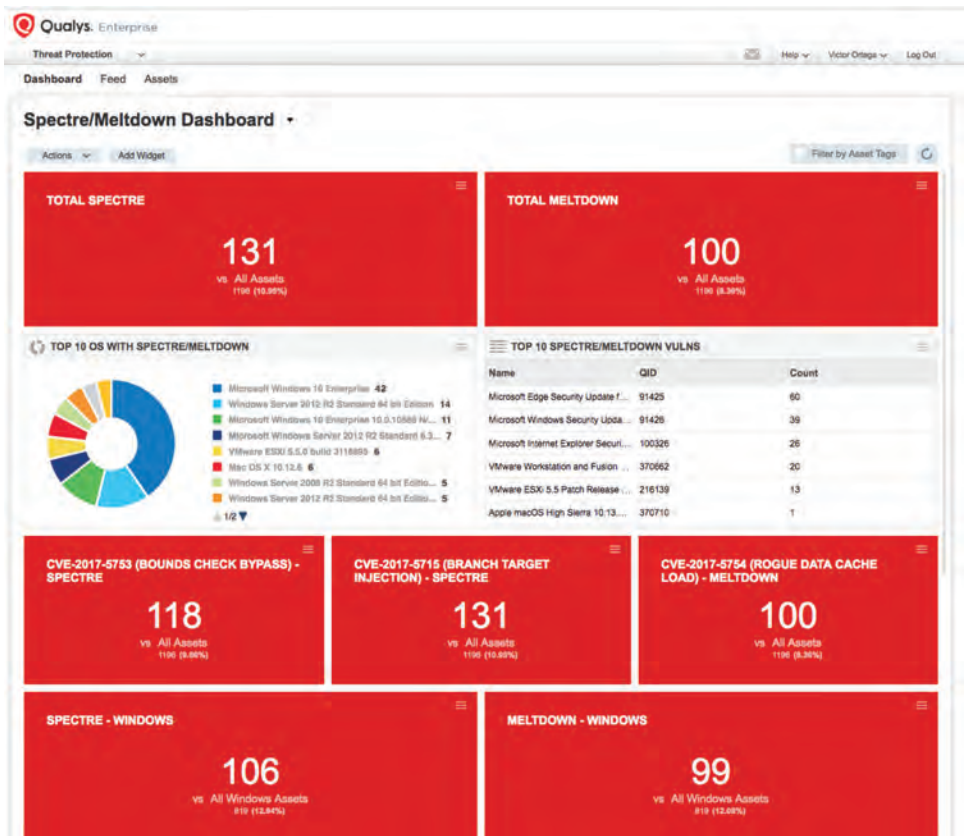
Spectre and Meltdown

Hardware may also contain intrinsic vulnerabilities that can be quite difficult to remediate. In 2017, security researchers announced the discovery of two related hardware vulnerabilities in nearly every microprocessor manufactured during the preceding two decades. These vulnerabilities, named Spectre and Meltdown, exploit a feature of the chips known as speculative execution to allow processes to gain access to information reserved for other processes.

Launching these attacks does require an initial presence on the system, but a penetration tester might exploit this type of vulnerability to engage in privilege escalation after establishing initial access to a system.

Detecting hardware-related vulnerabilities often requires the use of credentialed scanning, configuration management tools, or other approaches that leverage inside access to the system. When significant new vulnerabilities are discovered, scanning vendors often provide a customized dashboard, such as the one shown in Figure 5.11, to assist cybersecurity analysts in identifying, tracking, and remediating the issue.

FIGURE 5.11 Spectre and Meltdown dashboard from QualysGuard



Point-of-Sale System Vulnerabilities

The point-of-sale (POS) systems found in retail stores, restaurants, and hotels are lucrative targets for attackers and penetration testers alike. These systems often store, process, and/or transmit credit card information, making them highly valuable in the eyes of an attacker seeking financial gain.

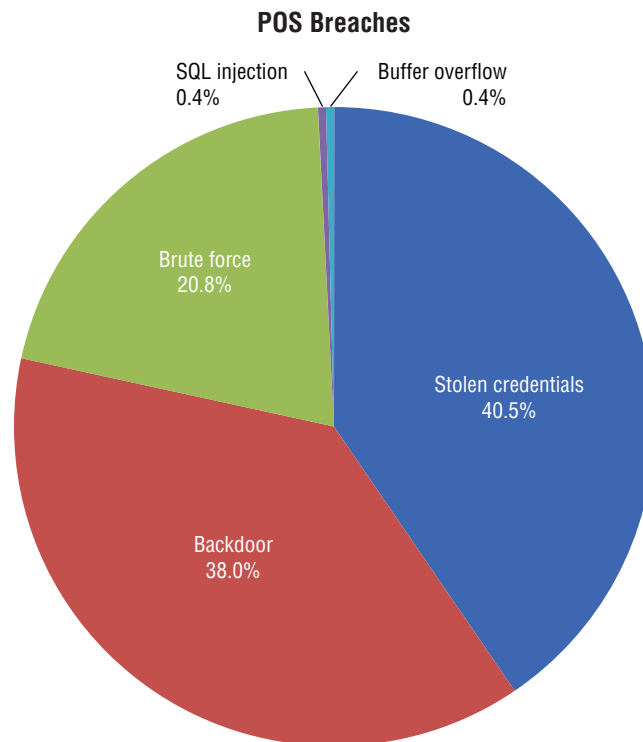
POS systems commonly run either standard or specialized versions of common operating systems, with many running variants of Microsoft Windows. They require the same level

of patching and security controls as any other Windows system and are subject to the same security vulnerabilities as those devices.

POS systems involved in credit and debit card transactions must comply with the Payment Card Industry Data Security Standard (PCI DSS), which outlines strict, specific rules for the handling of credit card information and the security of devices involved in those transactions.

The 2017 Verizon *Data Breach Investigations Report (DBIR)* did a special analysis of POS breaches and identified common trends in the types of attacks waged against POS systems, as shown in Figure 5.12.

FIGURE 5.12 POS Breach Types in 2017



Insecure Protocol Use

Many of the older protocols used on networks in the early days of the Internet were designed without security in mind. They often failed to use encryption to protect user-names, passwords, and the content sent over an open network, exposing the users of the protocol to eavesdropping attacks. Telnet is one example of an insecure protocol used to gain command-line access to a remote server. The File Transfer Protocol (FTP) provides the ability to transfer files between systems but does not incorporate security features. Figure 5.13 shows an example of a scan report that detected a system that supports the insecure FTP protocol.

FIGURE 5.13 FTP cleartext authentication vulnerability

The screenshot displays a vulnerability scan result for the issue 'FTP Supports Cleartext Authentication'. The severity is marked as 'Low'. The description states: 'The remote FTP server allows the user's name and password to be transmitted in cleartext, which could be intercepted by a network sniffer or a man-in-the-middle attack.' The solution provided is: 'Switch to SFTP (part of the SSH suite) or FTPS (FTP over SSL/TLS). In the latter case, configure the server so that control connections are encrypted.' The output shows: 'This FTP server does not support 'AUTH TLS''. A table lists the host '10.11.248.224' on port '21' as 'ftp'. To the right, 'Plugin Details' include: Severity: Low, ID: 34324, Version: \$Revision: 1.24 \$, Type: remote, Family: FTP, Published: 2006/10/01, Modified: 2015/06/23. 'Risk Information' shows: Risk Factor: Low, CVSS Base Score: 2.6, CVSS Vector: CVSS2#AV:N/AC:H/Au:N/C:P/I:N/A/N. 'Reference Information' lists CWE: 502, 503, 528, 931.

The solution for this issue is to simply switch to a more secure protocol. Fortunately, encrypted alternatives exist for both Telnet and FTP. System administrators can use the Secure Shell (SSH) as a secure replacement for Telnet when seeking to gain command-line access to a remote system. Similarly, the Secure File Transfer Protocol (SFTP) and FTP-Secure (FTPS) both provide a secure method to transfer files between systems.

Debug Modes

Many application development platforms support *debug modes* that give developers crucial information needed to troubleshoot applications in the development process. Debug modes typically provide detailed information on the inner workings of an application and server as well as supporting databases. Although this information can be useful to developers, it can inadvertently assist an attacker seeking to gain information about the structure of a database, authentication mechanisms used by an application, or other details. For this reason, vulnerability scans do alert on the presence of debug mode on scanned servers. Figure 5.14 shows an example of this type of scan result.

FIGURE 5.14 Debug mode vulnerability

The screenshot displays a vulnerability report for the 'ASP.NET DEBUG Method Enabled' issue. The report is divided into several sections: Description, Solution, See Also, Output, Plugin Details, and Risk Information.

Description: It is possible to send debug statements to the remote ASP scripts. An attacker might use this to alter the runtime of the remote scripts.

Solution: Make sure that DEBUG statements are disabled or only usable by authenticated users.

See Also: <http://support.microsoft.com/default.aspx?scid=kb;en-us;915157>

Output: The request shows a DEBUG statement being sent to a remote ASP script. The response shows the server's output, including headers like 'HTTP/1.1 200 OK', 'Cache-Control: private', 'Content-Type: text/html; charset=utf-8', 'Server: Microsoft-IIS/8.5', 'X-AspNet-Version: 4.0.30319', 'X-Powered-By: ASP.NET', and 'Date: Sat, 21 Oct 2016 05:53:07 GMT'.

Plugin Details:

- Severity: Medium
- ID: 33270
- Version: \$Revision: 1.12 \$
- Type: remote
- Family: CGI abuses
- Published: 2008/05/27
- Modified: 2013/01/25

Risk Information:

- Risk Factor: Medium
- CVSS Base Score: 5.0
- CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:N/I:P/A:N

In this particular example, the target system appears to be a Windows Server system supporting the ASP.NET development environment. The Output section of the report demonstrates that the server responds when sent a DEBUG request by a client.

Solving this issue requires the cooperation of developers and disabling debug modes on systems with public exposure. In mature organizations, software development should always take place in a dedicated development environment that is accessible only from private networks. Developers should be encouraged (or ordered!) to conduct their testing only on systems dedicated to that purpose, and it would be entirely appropriate to enable debug mode on those servers. There should be no need for supporting this capability on public-facing systems.

Network Vulnerabilities

Modern interconnected networks use a complex combination of infrastructure components and network appliances to provide widespread access to secure communications capabilities. These networks and their component parts are also susceptible to security vulnerabilities that may be detected during a vulnerability scan.

Missing Firmware Updates

Operating systems and applications aren't the only devices that require regular security updates. Vulnerability scans may also detect security problems in network devices that require firmware updates from the manufacturer to correct. These vulnerabilities result in

reports similar to the operating system missing patch report shown in Figure 5.7 earlier and typically direct administrators to the location on the vendor’s site where the firmware update is available for download.

SSL and TLS Issues

The *Secure Sockets Layer (SSL)* protocol and its successor, *Transport Layer Security (TLS)*, offer a secure means to exchange information over the Internet and private networks. Although these protocols can be used to encrypt almost any type of network communication, they are most commonly used to secure connections to web servers and are familiar to end users designated by the *S* in *HTTPS*.

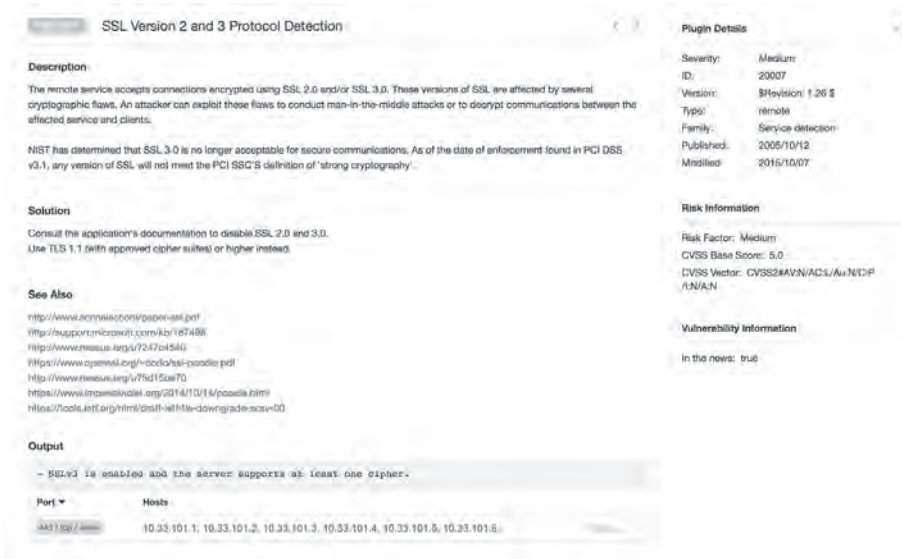


Many cybersecurity analysts incorrectly use the acronym SSL to refer to both the SSL and TLS protocols. It’s important to understand that SSL is no longer secure and should not be used. TLS is a replacement for SSL that offers similar functionality but does not have the security flaws contained in SSL. Be careful to use this terminology precisely and, to avoid ambiguity, question those who use the term *SSL* whether they are really referring to *TLS*.

Outdated SSL/TLS Versions

SSL is no longer considered secure and should not be used on production systems. The same is true for early versions of TLS. Vulnerability scanners may report that web servers are using these protocols, and cybersecurity analysts should understand that any connections making use of these outdated versions of SSL and TLS may be subject to eavesdropping attacks. Figure 5.15 shows an example of a scan report from a network containing multiple systems that support the outdated SSL version 3.

FIGURE 5.15 Outdated SSL version vulnerability



The administrators of servers supporting outdated versions of SSL and TLS should disable support for these older protocols on their servers and support only newer protocols, such as TLS version 1.2.

Insecure Cipher Use

SSL and TLS are commonly described as cryptographic algorithms, but in fact, this is not the case. The SSL and TLS protocols describe how cryptographic ciphers may be used to secure network communications, but they are not cryptographic ciphers themselves. Instead, they allow administrators to designate the cryptographic ciphers that can be used with those protocols on a server-by-server basis. When a client and server wish to communicate using SSL/TLS, they exchange a list of ciphers that each system supports and agree on a mutually acceptable cipher.

Some ciphers contain vulnerabilities that render them insecure because of their susceptibility to eavesdropping attacks. For example, Figure 5.16 shows a scan report from a system that supports the insecure RC4 cipher.

FIGURE 5.16 Insecure SSL cipher vulnerability

LOW SSL RC4 Cipher Suites Supported (Bar Mitzvah)

Description

The remote host supports the use of RC4 in one or more cipher suites. The RC4 cipher is flawed in its generation of a pseudo-random stream of bytes so that a wide variety of small biases are introduced into the stream, decreasing its randomness.

If plaintext is repeatedly encrypted (e.g., HTTP cookies), and an attacker is able to obtain many (i.e., tens of millions) ciphertexts, the attacker may be able to derive the plaintext.

Solution

Reconfigure the affected application, if possible, to avoid use of RC4 ciphers. Consider using TLS 1.2 with AES-GCM suites subject to browser and web server support.

See Also

<http://www.nessus.org/u/7217a3668>
<http://cc.yo.to/talks/2013.03.12/slides.pdf>
<http://www.hsp.ru.ac.uk/files/>
http://www.imperya.com/docs/Hijacking_SSL_when_using_RC4.pdf

Output

List of RC4 cipher suites supported by the remote server :

High Strength Ciphers (>= 112-bit key)

TLSv1	Kx=RSA	Au=RSA	Enc=RC4(128)	Mac=MD5
RC4-MD5	Kx=RSA	Au=RSA	Enc=RC4(128)	Mac=MD5
RC4-SHA	Kx=RSA	Au=RSA	Enc=RC4(128)	Mac=SHA1

The fields above are :

```
{OpenSSL ciphername}
Kx={key exchange}
Au={authentication}
Enc={symmetric encryption method}
Mac={message authentication code}
{export flag}
```

Plugin Details

Severity: Low
 ID: 65821
 Version: \$Revision: 1.9 \$
 Type: remote
 Family: General
 Published: 2013/04/05
 Modified: 2016/08/16

Risk Information

Risk Factor: Low
 CVSS Base Score: 2.6
 CVSS Vector: CVSS2#AV:N/AC:H/Au:N/C:P/IN:A/N
 CVSS Temporal Vector:
 CVSS2#E:ND/RL:OF/RC:C
 CVSS Temporal Score: 2.3

Vulnerability Information

Exploit Available: false
 Exploit Ease: No known exploits are available
 Vulnerability Pub Date: 2013/03/12
 In the news: true

Reference Information

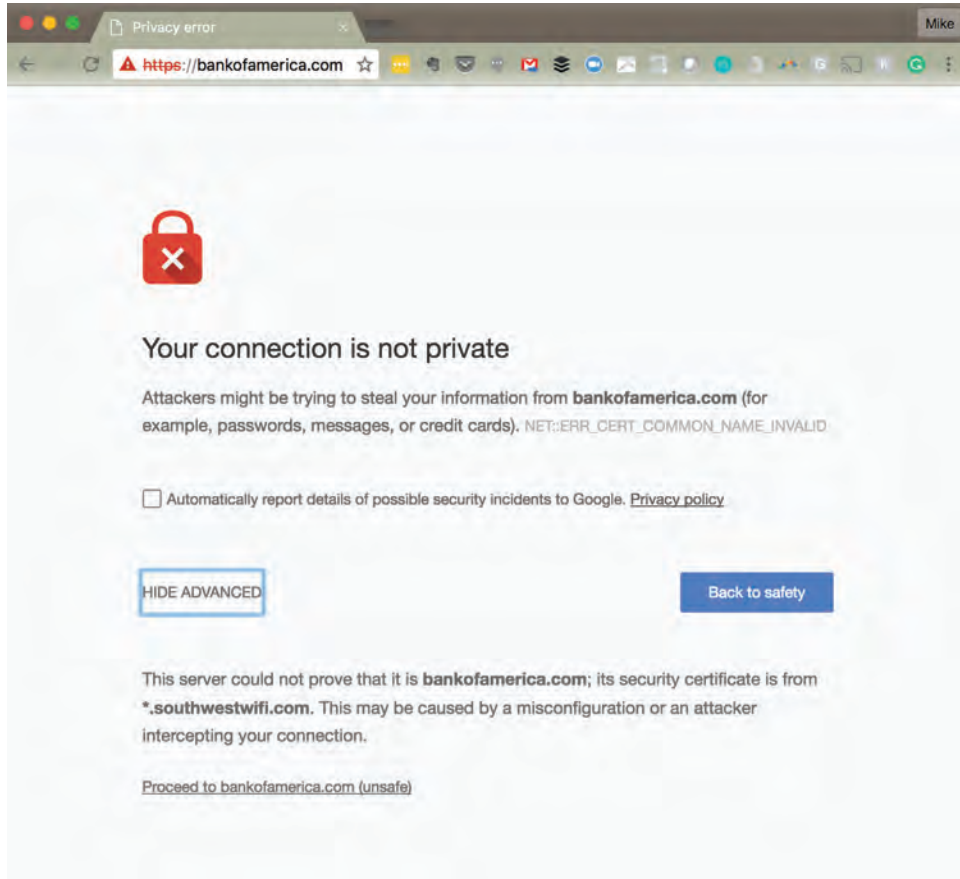
CVE: CVE-2013-2566, CVE-2015-2808
 OSVDB: 91160, 117855
 BID: 58786, 73684

Solving this common problem requires altering the set of supported ciphers on the affected server and ensuring that only secure ciphers may be used.

Certificate Problems

SSL and TLS rely on the use of digital certificates to validate the identity of servers and exchange cryptographic keys. Website users are familiar with the error messages displayed in web browsers, such as that shown in Figure 5.17. These errors often contain extremely important information about the security of the site being accessed but, unfortunately, are all too often ignored.

FIGURE 5.17 Invalid certificate warning



Vulnerability scans may also detect issues with the certificates presented by servers that support SSL and/or TLS. Common errors include the following:

Mismatch between the Name on the Certificate and the Name of the Server This is a very serious error because it may indicate the use of a certificate taken from another site. It's the digital equivalent of someone using a fake ID "borrowed" from a friend.

Expiration of the Digital Certificate Digital certificates have validity periods and expiration dates. When you see an expired certificate, it most likely means that the server administrator failed to renew the certificate in a timely manner.

Unknown Certificate Authority (CA) Anyone can create a digital certificate, but digital certificates are only useful if the recipient of a certificate trusts the entity that issued it. Operating systems and browsers contain instructions to trust well-known CAs but will show an error if they encounter a certificate issued by an unknown or untrusted CA.

The error shown in Figure 5.17 indicates that the user is attempting to access a website that is presenting an invalid certificate. From the URL bar, we see that the user is attempting to access `bankofamerica.com`. However, looking in the details section, we see that the certificate being presented was issued to `southwestwifi.com`. This is a typical occurrence on networks that use a captive portal to authenticate users joining a public wireless network. This example is from the in-flight WiFi service offered by Southwest Airlines. The error points out to the user that they are not communicating with the intended website owned by Bank of America and should not provide sensitive information.

Domain Name System (DNS)

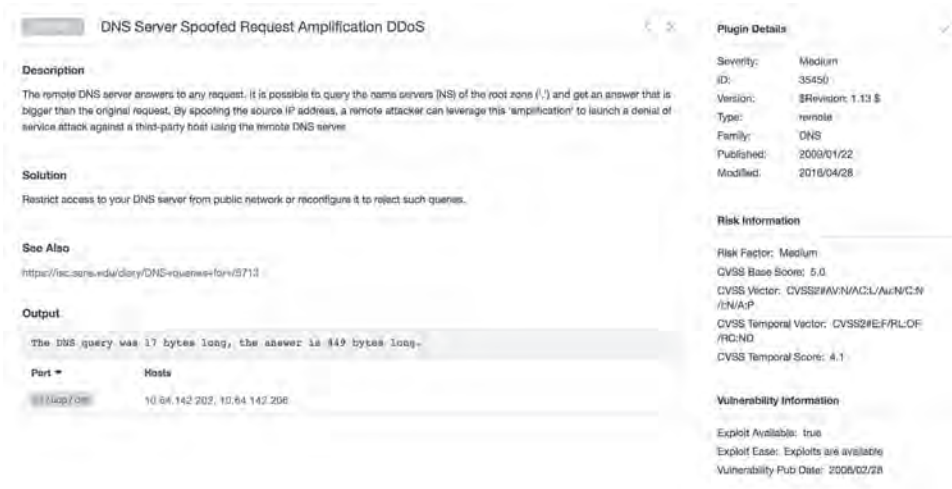
The *Domain Name System (DNS)* provides a translation service between domain names and IP addresses. DNS allows end users to remember user-friendly domain names, such as `apple.com`, and not worry about the mind-numbing IP addresses actually used by those servers.

DNS servers are a common source of vulnerabilities on enterprise networks. Despite the seemingly simple nature of the service, DNS has a track record of many serious security vulnerabilities and requires careful configuration and patching. Many of the issues with DNS services are those already discussed in this chapter, such as buffer overflows, missing patches, and code execution vulnerabilities, but others are specific to the DNS service.

Because DNS vulnerabilities are so prevalent, DNS servers are a common first target for attackers and penetration testers alike.

Figure 5.18 shows an example of a vulnerability scan that detected a *DNS amplification* vulnerability on two servers on an organization's network. In this type of attack, the attacker sends spoofed DNS requests to a DNS server that are carefully designed to elicit responses that are much larger in size than the original requests. These large response packets then go to the spoofed address where the DNS server believes the query originated. The IP address used in the spoofed request is actually the target of a denial-of-service attack and is bombarded by very large responses from DNS servers all over the world to queries that it never sent. When conducted in sufficient volume, DNS amplification attacks can completely overwhelm the targeted systems, rendering them inoperable.

FIGURE 5.18 DNS amplification vulnerability



Internal IP Disclosure

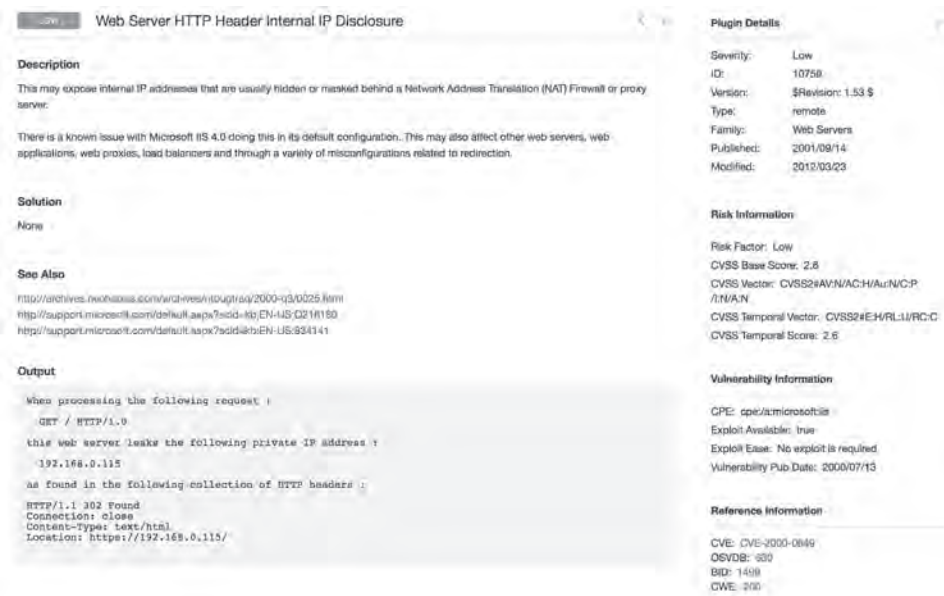
IP addresses come in two different variants: public IP addresses, which can be routed over the Internet, and private IP addresses, which can only be used on local networks. Any server that is accessible over the Internet must have a public IP address to allow that access, but the public IP address is typically managed by a firewall that uses the *Network Address Translation (NAT)* protocol to map the public address to the server's true, private IP address. Systems on the local network can use the server's private address to access it directly, but remote systems should never be aware of that address.

Servers that are not properly configured may leak their private IP addresses to remote systems. This can occur when the system includes its own IP address in the header information returned in the response to an HTTP request. The server is not aware that NAT is in use, so it uses the private address in its response. Attackers and penetration testers can use this information to learn more about the internal configuration of a firewalled network. Figure 5.19 shows an example of this type of information disclosure vulnerability.

Virtual Private Network Issues

Many organizations use *virtual private networks (VPNs)* to provide employees with secure remote access to the organization's network. As with any application protocol, administrators must ensure that the VPN services offered by the organization are fully patched to current levels. In addition, VPNs require the use of cryptographic ciphers and suffer from similar issues as SSL and TLS when they support the use of insecure ciphers.

FIGURE 5.19 Internal IP disclosure vulnerability

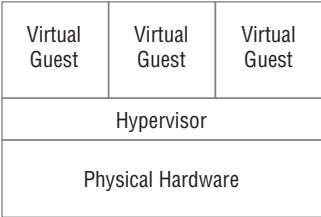


Virtualization Vulnerabilities

Most modern data centers make extensive use of *virtualization* technology to allow multiple guest systems to share the same underlying hardware. In a virtualized data center, the virtual host hardware runs a special operating system known as a *hypervisor* that mediates access to the underlying hardware resources. Virtual machines then run on top of this virtual infrastructure provided by the hypervisor, running standard operating systems such as Windows and Linux variants. The virtual machines may not be aware that they are running in a virtualized environment because the hypervisor tricks them into thinking that they have normal access to the underlying hardware when, in reality, that hardware is shared with other systems.

Figure 5.20 provides an illustration of how a hypervisor mediates access to the underlying hardware resources in a virtual host to support multiple virtual guest machines.

FIGURE 5.20 Inside a virtual host





The example described in this chapter, where the hypervisor runs directly on top of physical hardware, is known as *bare-metal virtualization*. This is the approach commonly used in data center environments and is also referred to as using a Type 1 hypervisor. There is another type of virtualization, known as *hosted virtualization*, where a host operating system sits between the hardware and the hypervisor. This is commonly used in cases where the user of an endpoint system wants to run multiple operating systems simultaneously on that device. For example, Parallels is a popular hosted virtualization platform for the Mac. Hosted virtualization is also described as using a Type 2 hypervisor.

VM Escape

Virtual machine escape vulnerabilities are the most serious issue that may exist in a virtualized environment, particularly when a virtual host runs systems with differing security levels. In an escape attack, the attacker has access to a single virtual host and then manages to leverage that access to intrude on the resources assigned to a different virtual machine. The hypervisor is supposed to prevent this type of intrusion by restricting a virtual machine's access to only those resources assigned to that machine. Escape attacks allow a process running on the virtual machine to “escape” those hypervisor restrictions.

Management Interface Access

Virtualization engineers use the management interface for a virtual infrastructure to configure the virtualization environment, set up new guest machines, and regulate access to resources. This management interface is extremely sensitive from a security perspective, and access should be tightly controlled to prevent unauthorized individuals from gaining access. In addition to using strong multifactor authentication on the management interface, cybersecurity professionals should ensure that the interface is never directly accessible from a public network. Vulnerability scans that detect the presence of an accessible management interface will report this as a security concern.

Virtual Host Patching

This chapter has already discussed the importance of promptly applying security updates to operating systems, applications, and network devices. It is equally important to ensure that virtualization platforms receive security updates that may affect the security of virtual guests or the entire platform. Patches may correct vulnerabilities that allow virtual machine escape attacks or other serious security flaws.

Virtual Guest Issues

Cybersecurity analysts should think of each guest machine running in a virtualized environment as a separate server that requires the same security attention as any other device on the network. Guest operating systems and applications running on the guest OS must be promptly patched to correct security vulnerabilities and be otherwise well maintained.

There's no difference from a security perspective between a physical server and a virtualized server.

Virtual Network Issues

As data centers become increasingly virtualized, a significant amount of network traffic never actually touches a network! Communications between virtual machines that reside on the same physical hardware can occur in memory without ever touching a physical network. For this reason, virtual networks must be maintained with the same attention to security that administrators would apply to physical networks. This includes the use of virtual firewalls to control the flow of information between systems and the isolation of systems of differing security levels on different virtual network segments.

Internet of Things (IoT)

In some environments, cybersecurity analysts may encounter the use of *supervisory control and data acquisition (SCADA)* systems, *industrial control systems (ICSs)*, and other examples of the *Internet of Things (IoT)*. These systems allow the connection of physical devices and processes to networks and provide tremendous sources of data for organizations seeking to make their business processes more efficient and effective. However, they also introduce new security concerns that may arise on vulnerability scans.

As with any other device on a network, IoT devices may have security vulnerabilities and are subject to network-based attacks. However, it is often more difficult to patch IoT devices than it is to patch their traditional server counterparts because it is difficult to obtain patches. IoT device manufacturers may not use automatic update mechanisms, and the only way that cybersecurity analysts may become aware of an update is through a vulnerability scan or by proactively subscribing to the security bulletins issued by IoT device manufacturers.

IoT devices also often have unique characteristics compared to other devices attached to the networks. They often exist as *embedded systems*, where there is an operating system and computer running in the device that may not be obvious or accessible to the outside world. For example, large multifunction copier/printer units found in office environments often have an entire Windows or Linux operating system running internally that may act as a file and print server. IoT devices also often run *real-time operating systems (RTOS)*. These are either special purpose operating systems or variants of standard operating systems designed to process data rapidly as it arrives from sensors or other IoT components.

IoT Uprising

On October 21, 2016, a widespread distributed denial of service (DDoS) attack shut down large portions of the Internet, affecting services run by Amazon, *The New York Times*, Twitter, Box, and other providers. The attack came in waves over the course of the day and initially mystified technologists seeking to bring systems back online.

Investigation later revealed that the outages occurred when Dyn, a global provider of DNS services, suffered a debilitating attack that prevented it from answering DNS queries. Dyn received massive amounts of traffic that overwhelmed its servers.

The source of all of that traffic? Attackers used an IoT botnet named Mirai to leverage the bandwidth available to baby monitors, DVRs, security cameras, and other IoT devices in the homes of normal people. Those botnetted devices received instructions from a yet-unknown attacker to simultaneously bombard Dyn with requests, knocking it (and a good part of the Internet!) offline.

Web Application Vulnerabilities

Web applications are complex environments that often rely not only on web servers but also on backend databases, authentication servers, and other components to provide services to end users. These web applications may also contain security holes that allow attackers to gain a foothold on a network, and modern vulnerability scanners are able to probe web applications for these vulnerabilities.

Injection Attacks

Injection attacks occur when an attacker is able to send commands through a web server to a backend system, bypassing normal security controls and fooling the backend system into believing that the request came from the web server. The most common form of this attack is the *SQL injection attack*, which exploits web applications to send unauthorized commands to a backend database server.

Web applications often receive input from users and use it to compose a database query that provides results that are sent back to a user. For example, consider the search function on an e-commerce site. If a user enters **orange tiger pillows** into the search box, the web server needs to know what products in the catalog might match this search term. It might send a request to the backend database server that looks something like this:

```
SELECT ItemName, ItemDescription, ItemPrice
FROM Products
WHERE ItemName LIKE '%orange%' AND
ItemName LIKE '%tiger%' AND
ItemName LIKE '%pillow%'
```

This command retrieves a list of items that can be included in the results returned to the end user. In a SQL injection attack, the attacker might send a very unusual-looking request to the web server, perhaps searching for

```
orange tiger pillow'; SELECT CustomerName, CreditCardNumber FROM Orders; --
```

If the web server simply passes this request along to the database server, it would do this (with a little reformatting for ease of viewing):

```
SELECT ItemName, ItemDescription, ItemPrice
FROM Products
```

```

WHERE ItemName LIKE '%orange%' AND
ItemName LIKE '%tiger%' AND
ItemName LIKE '%pillow';
SELECT CustomerName, CreditCardNumber
FROM Orders;
--%'

```

This command, if successful, would run two SQL queries (separated by the semicolon). The first would retrieve the product information, and the second would retrieve a listing of customer names and credit card numbers.

The two best ways to protect against SQL injection attacks are input validation and the enforcement of least privilege restrictions on database access. Input validation ensures that users don't provide unexpected text to the web server. It would block the use of the apostrophe that is needed to "break out" of the original SQL query. Least privilege restricts the tables that may be accessed by a web server and can prevent the retrieval of credit card information by a process designed to handle catalog information requests.

Vulnerability scanners can detect injection vulnerabilities, such as the one shown in Figure 5.21. When cybersecurity analysts notice a potential injection vulnerability, they should work closely with developers to validate that the vulnerability exists and fix the affected code.

FIGURE 5.21 SQL injection vulnerability

The screenshot displays a Nessus vulnerability scan result for the plugin 'CGI Generic SQL Injection (blind, time based)'. The interface is divided into several sections:

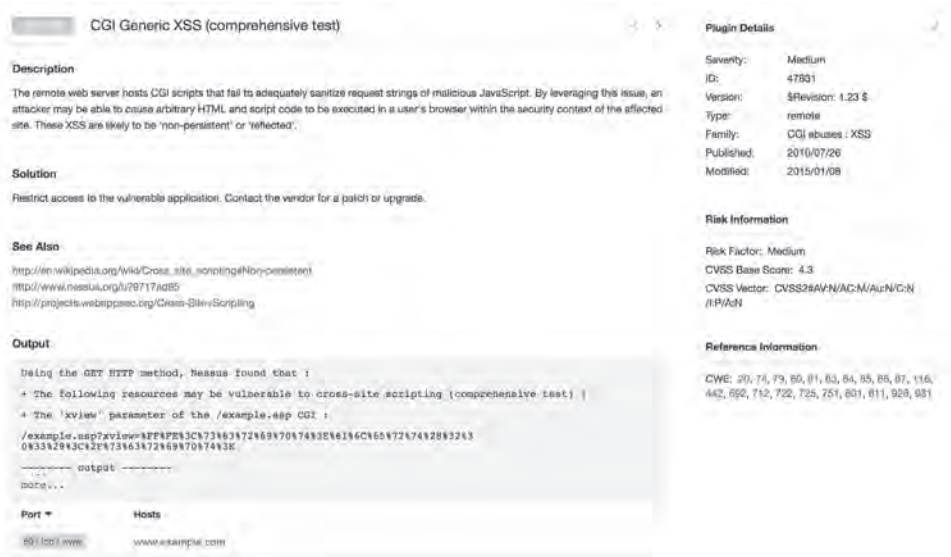
- Description:** Explains that by sending specially crafted parameters to CGI scripts, an attacker can bypass authentication, read confidential data, or even take control of the remote operating system. It includes a note that the script is experimental and may have false positives.
- Solution:** Advises modifying affected CGI scripts to properly escape arguments.
- See Also:** Provides links to external resources for further information on SQL injection.
- Output:** Shows the results of a GET HTTP method request, indicating that the application is vulnerable to blind SQL injection (time based) by exploiting the 'company' parameter of the '/experience-company.asp' CGI script. The output includes a sample HTTP response showing headers like 'Content-Type: text/html' and 'Content-Length: 100'.
- Plugin Details:** A sidebar on the right provides metadata about the plugin, including its severity (High), ID (43160), version (1.15), type (remote), family (CGI abuses), and publication/modification dates.
- Risk Information:** Displays the risk factor (High), CVSS Base Score (7.5), and CVSS Vector (CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P).
- Reference Information:** Lists CVE identifiers (20, 77, 88, 713, 723, 727, 751, 801, 810, 826, 829) associated with this vulnerability.

Cross-Site Scripting

In a *cross-site scripting* (XSS) attack, an attacker embeds scripting commands on a website that will later be executed by an unsuspecting visitor accessing the site. The idea is to trick a user visiting a trusted site into executing malicious code placed there by an untrusted third party.

Figure 5.22 shows an example of an XSS vulnerability detected during a Nessus vulnerability scan.

FIGURE 5.22 Cross-site scripting vulnerability



Cybersecurity analysts discovering potential XSS vulnerabilities during a scan should work with developers to assess the validity of the result and implement appropriate controls to prevent this type of attack, such as implementing input validation.

Summary

Vulnerability scanners produce a significant amount of information that can inform penetration tests. Penetration testers must be familiar with the interpretation of vulnerability scan results and the prioritization of vulnerabilities as attack targets to improve the efficiency and effectiveness of their testing efforts.

Vulnerability scanners usually rank detected issues using the Common Vulnerability Scoring System (CVSS). CVSS provides six different measures of each vulnerability: the

access vector metric, the access complexity metric, the authentication metric, the confidentiality metric, the integrity metric, and the availability metric. Together, these metrics provide a look at the potential that a vulnerability will be successfully exploited and the impact it could have on the organization.

As penetration testers interpret scan results, they should be careful to watch for common issues. False positive reports occur when the scanner erroneously reports a vulnerability that does not actually exist. These may present false leads that waste testing time, in the best case, or alert administrators to penetration testing activity, in the worst case.

To successfully interpret vulnerability reports, penetration testers must be familiar with the vulnerabilities that commonly occur. Common server and endpoint vulnerabilities include missing patches, unsupported operating systems and applications, buffer overflows, privilege escalation, arbitrary code execution, insecure protocol usage, and the presence of debugging modes. Common network vulnerabilities include missing firmware updates, SSL/TLS issues, DNS misconfigurations, internal IP disclosures, and VPN issues. Virtualization vulnerabilities include virtual machine escape vulnerabilities, management interface access, missing patches on virtual hosts, and security misconfigurations on virtual guests and virtual networks.

Exam Essentials

Vulnerability scan reports provide critical information to penetration testers. In addition to providing details about the vulnerabilities present on a system, vulnerability scan reports also offer crucial severity and exploitation information. The report typically includes the request and response that triggered a vulnerability report as well as a suggested solution to the problem.

The Common Vulnerability Scoring System (CVSS) provides a consistent standard for scoring vulnerability severity. The CVSS base score computes a standard measure on a 10-point scale that incorporates information about the access vector required to exploit a vulnerability, the complexity of the exploit, and the authentication required to execute an attack. The base score also considers the impact of the vulnerability on the confidentiality, integrity, and availability of the affected system.

Servers and endpoint devices are common sources of vulnerability. Missing patches and outdated operating systems are two of the most common vulnerability sources and are easily corrected by proactive device maintenance. Buffer overflow, privilege escalation, and arbitrary code execution attacks typically exploit application flaws. Devices supporting insecure protocols are also a common source of vulnerabilities.

Network devices also suffer from frequent vulnerabilities. Network administrators should ensure that network devices receive regular firmware updates to patch security issues. Improper implementations of SSL and TLS encryption also cause vulnerabilities when they use outdated protocols, insecure ciphers, or invalid certificates.

Virtualized infrastructures add another layer of potential vulnerability. Administrators responsible for virtualized infrastructure must take extra care to ensure that the hypervisor is patched and protected against virtual machine escape attacks. Additionally, administrators should carefully restrict access to the virtual infrastructure's management interface to prevent unauthorized access attempts.

Lab Exercises

Activity 5.1: Interpreting a Vulnerability Scan

In Activity 4.2, you ran a vulnerability scan of a network under your control. In this lab, you will interpret the results of that vulnerability scan.

Review the scan results carefully and develop a plan for the next phase of your penetration test. What vulnerabilities that you discovered seem the most promising targets for exploitation? Why? How would you approach exploiting those vulnerabilities?

Activity 5.2: Analyzing a CVSS Vector

In this lab, you will interpret the CVSS vectors found in a vulnerability scan report to assess the severity and impact of two vulnerabilities.

Review the vulnerability reports in Figures 5.23 and 5.24.

FIGURE 5.23 First vulnerability report

The screenshot displays a vulnerability scan report for the title "Internet Key Exchange (IKE) Aggressive Mode with Pre-Shared Key". The report is divided into several sections:

- Description:** The remote Internet Key Exchange (IKE) version 1 service seems to support Aggressive Mode with Pre-Shared key (PSK) authentication. Such a configuration could allow an attacker to capture and crack the PSK of a VPN gateway and gain unauthorized access to private networks.
- Solution:**
 - Disable Aggressive Mode if supported.
 - Do not use Pre-Shared key for authentication if it's possible.
 - If using Pre-Shared key cannot be avoided, use very strong keys.
 - If possible, do not allow VPN connections from any IP addresses.

Note that this plugin does not run over IPv6.
- See Also:**
 - <http://www.nessus.org/10711252b>
 - <https://www.exploit-db.com/exploits/1411/>
 - <http://www.vpnc.org/ietf-ipsec09/ipsec/mag01411.html>
 - <http://www.securityfocus.com/bid/7423>
- Output:** No output recorded.
- Plugin Details:**
 - Severity: Medium
 - ID: 62694
 - Version: \$Revision: 1.7 \$
 - Type: remote
 - Family: General
 - Published: 2012/10/24
 - Modified: 2015/11/10
- Risk Information:**
 - Risk Factor: Medium
 - CVSS Base Score: 5.0
 - CVSS Vector: CVSS2#AV:N/AC:L/AU:N/C:P/I:N/A/N
 - CVSS Temporal Vector: CVSS2#E:F/RL:W/RC:C
 - CVSS Temporal Score: 4.5
- Vulnerability Information:**
 - Exploit Available: true
 - Exploit Ease: Exploits are available
 - Vulnerability Pub Date: 2002/09/03

FIGURE 5.24 Second vulnerability report

SSLv3 Padding Oracle On Downgraded Legacy Encryption (POODLE)

Description

The remote host is affected by a man-in-the-middle (MitM) information disclosure vulnerability known as POODLE. The vulnerability is due to the way SSL 3.0 handles padding bytes when decrypting messages encrypted using block ciphers in cipher block chaining (CBC) mode. MitM attackers can decrypt a selected byte of a cipher text in as few as 256 tries if they are able to force a victim application to repeatedly send the same data over newly created SSL 3.0 connections.

As long as a client and service both support SSLv3, a connection can be 'rolled back' to SSLv3, even if TLSv1 or newer is supported by the client and service.

The TLS Fallback SCSV mechanism prevents 'version rollback' attacks without impacting legacy clients; however, it can only protect connections when the client and service support the mechanism. Sites that cannot disable SSLv3 immediately should enable this mechanism.

This is a vulnerability in the SSLv3 specification, not in any particular SSL implementation. Disabling SSLv3 is the only way to completely mitigate the vulnerability.

Solution

Disable SSLv3.

Services that must support SSLv3 should enable the TLS Fallback SCSV mechanism until SSLv3 can be disabled.

See Also

<https://www.exploit-db.org/exploits/14/poodle.html>
<https://www.exploit-db.org/exploits/14/poodle.pdf>
<https://tools.eff.org/ssl-rtls-downgrade-sslv3-cv>

Output

```

nmap determined that the remote server supports SSLv3 with at least one CBC
cipher suite, indicating that this server is vulnerable.

It appears that TLSv1 or newer is supported on the server. However, the
Fallback SCSV mechanism is not supported, allowing connections to be "rolled
back" to SSLv3.

```

Plugin Details

Severity: Medium
ID: 78479
Version: 1.14
Type: remote
Family: General
Published: 2014/10/15
Modified: 2016/01/20

Risk Information

Risk Factor: Medium
CVSS Base Score: 4.2
CVSS Vector: CVSS2#AV:N/AC:M/Au:N/C:P/R:N/N:N
CVSS Temporal Vector: CVSS2#E:END/RL:OF/RC:C
CVSS Temporal Score: 3.7

Vulnerability Information

Exploit Available: true
Exploit Ease: Exploits are available
Vulnerability Pub Date: 2014/10/14
In the news: true

Reference Information

CVE: CVE-2014-3566
OSVDB: 113251
BID: 70874
CERT: 577103

Explain the components of the CVSS vector for each of these vulnerabilities. Which vulnerability is more serious? Why?

Activity 5.3: Developing a Penetration Testing Plan

In the scenario at the beginning of this chapter, you read about three vulnerabilities discovered in a web server operated by MCDS, LLC. In this lab, you will develop a penetration testing plan that exploits those vulnerabilities.

1. Review each of the three vulnerabilities identified in the scenario.
2. Assess the significance of each vulnerability for use during a penetration test.
3. Identify how you might exploit each vulnerability and what you might hope to achieve by exploiting the vulnerability.

Review Questions

You can find the answers in the Appendix.

1. Tom is reviewing a vulnerability scan report and finds that one of the servers on his network suffers from an internal IP address disclosure vulnerability. What protocol is likely in use on this network that resulted in this vulnerability?
 - A. TLS
 - B. NAT
 - C. SSH
 - D. VPN
2. Which one of the CVSS metrics would contain information about the number of times an attacker must successfully authenticate to execute an attack?
 - A. AV
 - B. C
 - C. Au
 - D. AC
3. Which one of the following values for the CVSS access complexity metric would indicate that the specified attack is simplest to exploit?
 - A. High
 - B. Medium
 - C. Low
 - D. Severe
4. Which one of the following values for the confidentiality, integrity, or availability CVSS metric would indicate the potential for total compromise of a system?
 - A. N
 - B. A
 - C. P
 - D. C
5. What is the most recent version of CVSS that is currently available?
 - A. 1.0
 - B. 2.0
 - C. 2.5
 - D. 3.0

6. Which one of the following metrics is not included in the calculation of the CVSS exploitability score?
 - A. Access vector
 - B. Vulnerability age
 - C. Access complexity
 - D. Authentication
7. Kevin recently identified a new security vulnerability and computed its CVSSv2 base score as 6.5. Which risk category would this vulnerability fall into?
 - A. Low
 - B. Medium
 - C. High
 - D. Critical
8. Tara recently analyzed the results of a vulnerability scan report and found that a vulnerability reported by the scanner did not exist because the system was actually patched as specified. What type of error occurred?
 - A. False positive
 - B. False negative
 - C. True positive
 - D. True negative
9. Which one of the following is not a common source of information that may be correlated with vulnerability scan results?
 - A. Logs
 - B. Database tables
 - C. SIEM
 - D. Configuration management system
10. Which one of the following operating systems should be avoided on production networks?
 - A. Windows Server 2003
 - B. Red Hat Enterprise Linux 7
 - C. CentOS 7
 - D. Ubuntu 16
11. In what type of attack does the attacker place more information in a memory location than is allocated for that use?
 - A. SQL injection
 - B. LDAP injection
 - C. Cross-site scripting
 - D. Buffer overflow

12. The Dirty COW attack is an example of what type of vulnerability?
 - A. Malicious code
 - B. Privilege escalation
 - C. Buffer overflow
 - D. LDAP injection
13. Which one of the following protocols should never be used on a public network?
 - A. SSH
 - B. HTTPS
 - C. SFTP
 - D. Telnet
14. Betty is selecting a transport encryption protocol for use in a new public website she is creating. Which protocol would be the best choice?
 - A. SSL 2.0
 - B. SSL 3.0
 - C. TLS 1.0
 - D. TLS 1.1
15. Which one of the following conditions would not result in a certificate warning during a vulnerability scan of a web server?
 - A. Use of an untrusted CA
 - B. Inclusion of a public encryption key
 - C. Expiration of the certificate
 - D. Mismatch in certificate name
16. What software component is responsible for enforcing the separation of guest systems in a virtualized infrastructure?
 - A. Guest operating system
 - B. Host operating system
 - C. Memory controller
 - D. Hypervisor
17. In what type of attack does the attacker seek to gain access to resources assigned to a different virtual machine?
 - A. VM escape
 - B. Management interface brute force
 - C. LDAP injection
 - D. DNS amplification

- 18.** Which one of the following terms is not typically used to describe the connection of physical devices to a network?
- A.** IoT
 - B.** IDS
 - C.** ICS
 - D.** SCADA
- 19.** Monica discovers that an attacker posted a message attacking users who visit a web forum that she manages. Which one of the following attack types is most likely to have occurred?
- A.** SQL injection
 - B.** Malware injection
 - C.** LDAP injection
 - D.** Cross-site scripting
- 20.** Alan is reviewing web server logs after an attack and finds many records that contain semicolons and apostrophes in queries from end users. What type of attack should he suspect?
- A.** SQL injection
 - B.** LDAP injection
 - C.** Cross-site scripting
 - D.** Buffer overflow