Henry Post

ITMS 443

Penetration Testing Report

Contents

[Executive Summary 2](#_Toc531786177)

[Testing 3](#_Toc531786178)

[Preamble 3](#_Toc531786179)

[Web Forms 3](#_Toc531786180)

[wget and grep 3](#_Toc531786181)

[Directory Traversal 4](#_Toc531786182)

[dirsearch 4](#_Toc531786183)

[Port Scanning 5](#_Toc531786184)

[nmap 5](#_Toc531786185)

[Vulnerability Analysis 6](#_Toc531786186)

[High 7](#_Toc531786187)

[NVT: MS15-034 HTTP.sys Remote Code Execution Vulnerability 7](#_Toc531786188)

[Medium 8](#_Toc531786189)

[NVT: DCE/RPC and MSRPC Services Enumeration Reporting 8](#_Toc531786190)

[NVT: Cleartext Transmission of Sensitive Information via HTTP 8](#_Toc531786191)

[NVT: SSL/TLS: Report Weak Cipher Suites 8](#_Toc531786192)

[NVT: SSL/TLS: Certificate Signed Using A Weak Signature Algorithm 9](#_Toc531786193)

[NVT: SSL/TLS: Diffie-Hellman Key Exchange Insufficient DH Group Strength Vulnerability 9](#_Toc531786194)

[Low 10](#_Toc531786195)

[NVT: TCP timestamps 10](#_Toc531786196)

[Penetration of Target 11](#_Toc531786197)

[Summary 12](#_Toc531786198)

# Executive Summary

This penetration test will attempt to gain full access to the target’s machine.

As many known services as possible will be tested, to maximize the chance that penetration is successful.

I will test things such as directories, web servers, and exploiting common vulnerabilities described by lists of software weaknesses such as CWE, CVSS, and CVE.

# Testing

## Preamble

As with most vulnerability scans, we start by identifying an attack surface that we can explore.

Initially, we are only given two URLs:

www.goodshopping.com, and

www.moviescope.com.

They both operate on the same IP, **172.43.35.10**. (Given by instructor.)

The most obvious attack surface, and arguably the simplest to explore, are their web interfaces!

## Web Forms

Web forms generally use HTTP verbs like POST, GET, PUT. Sometimes, they use JavaScript as well.

We can attempt to specially craft URL parameters to cause the website to:

* Execute SQL queries,
* Cause errors, which (if unhandled) can give us valuable information
* Expose other information (API that mishandles raw user input)
* Input malformed information if the web server does not scrub user input

### wget and grep

Below is an excerpt from wget and grep.

out/wget/www.goodshopping.com/index.html: <form method="post" action="./" id="form1">

out/wget/www.goodshopping.com/index.html:<input type="hidden" name="\_\_VIEWSTATE" id="\_\_VIEWSTATE" value="/wEPDwULLTE2NDI3NTE5MThkZKgJ3U6JI6YLa+wDmM9Iynokt8lUslL4EsvCvGUHz6kF" />

out/wget/www.goodshopping.com/index.html:<input type="hidden" name="\_\_VIEWSTATEGENERATOR" id="\_\_VIEWSTATEGENERATOR" value="C2EE9ABB" />

out/wget/www.moviescope.com/index.html: <td><input name="txtusername" type="text" id="txtusername" class="form-text" /></td>

out/wget/www.moviescope.com/index.html: <td style="padding-bottom:9px;"><input name="txtpwd" type="password" id="txtpwd" class="form-text" /></td>

As you can see, we already have some good information from *just visiting the site*.

First, we know they use ASP.NET!

\_\_VIEWSTATE is something that only ASP.NET forms have.

Second, we know that we use HTTP POST with the parameters of txtusername and txtpwd.

## Directory Traversal

This is an older and simpler attack than most others, and relies on the target not denying access to sensitive files or folders inside the web server’s content root.

Tools like “dirbuster” or any URL fuzzing tool can generate and traverse large URLs quite fast, storing ones which work and ignoring ones that don’t.

Even 403 pages are useful, because they say “This exists, but you can’t see it.”

### dirsearch

Using a directory traversal script, I was able to get about 18 pages that I’m not allowed to see, with various status codes.

This is useful because the directory structure can tell us quite a lot of information about the target, such as:

* Web technologies used
* Versions of web technologies
* Serverside scripting languages
* Hidden files/directories

403 312B http://www.goodshopping.com:80/%2e%2e//google.com

400 3KB http://www.goodshopping.com:80/%3f/

301 154B http://www.goodshopping.com:80/db

301 155B http://www.goodshopping.com:80/pdf

302 128B http://www.goodshopping.com:80/contactus.aspx

301 155B http://www.goodshopping.com:80/css

301 154B http://www.goodshopping.com:80/DB

403 1KB http://www.goodshopping.com:80/db/

301 158B http://www.goodshopping.com:80/Images

301 158B http://www.goodshopping.com:80/images

302 128B http://www.goodshopping.com:80/index.aspx

301 154B http://www.goodshopping.com:80/js

200 13KB http://www.goodshopping.com:80/Login.aspx

200 13KB http://www.goodshopping.com:80/login.aspx

301 155B http://www.goodshopping.com:80/TMP

403 1KB http://www.goodshopping.com:80/tmp/

301 155B http://www.goodshopping.com:80/tmp

403 2KB http://www.goodshopping.com:80/Trace.axd

I have omitted the www.moviescope.com entries as they are the same.

## Port Scanning

Scanning ports on a machine is an excellent way to identify an attack surface.

Tools like “nmap” are very useful to get a list of open ports.

### nmap

Below is an excerpt from a port scan.

**PORT STATE SERVICE**

21/tcp open ftp

80/tcp open http

135/tcp open msrpc

139/tcp open netbios-ssn

445/tcp open microsoft-ds

1801/tcp open msmq

2103/tcp open zephyr-clt

2105/tcp open eklogin

2107/tcp open msmq-mgmt

3389/tcp open ms-wbt-server

5985/tcp open wsman

16450/tcp open unknown

16451/tcp open unknown

16452/tcp open unknown

16453/tcp open unknown

17001/tcp open unknown

47001/tcp open winrm

49152/tcp open unknown

49153/tcp open unknown

49154/tcp open unknown

49155/tcp open unknown

49156/tcp open unknown

49157/tcp open unknown

49215/tcp open unknown

49216/tcp open unknown

52710/tcp open unknown

# Vulnerability Analysis

I used OpenVAS as well as simple probing and some automated scripts (found in final\scripts) to determine vulnerabilities in the target.

There are also PDFs in final\data.

Below is a summary, organized by severity, of the vulnerabilities I have found.

## High

### NVT: MS15-034 HTTP.sys Remote Code Execution Vulnerability

#### Details

This vulnerability is a remote code execution exploit that just involves sending a malicious HTTP request.

It exists because of flawed handling of HTTP requests in MS IIS’s HTTP.sys module.

It’s ranked this high as it’s trivial to execute the BSOD variant, given you can send this using a single line in any linux terminal:

wget -O /dev/null --header="Range: 20-18446744073709551615" hxxp://some.site.com/my\_cute\_cat.jpg

That large number is .

Depending on the number(s) and other factors, you can:

* Induce a BSOD in an IIS server
* Get kernel memory before reboot
* Likely many other things. It *does* say ‘remote code execution’ on their site.

More info [here](https://isc.sans.edu/forums/diary/MS15034+HTTPsys+IIS+DoS+And+Possible+Remote+Code+Execution+PATCH+NOW/19583).

#### Scope

Anything using HTTP.sys is vulnerable.

#### Mitigation

This vulnerability can be mitigated by:

* [Updating your Windows software!](https://support.microsoft.com/en-us/help/3042553/ms15-034-vulnerability-in-http-sys-could-allow-remote-code-execution-a)
* Disabling IIS kernel caching
* Not using IIS

## Medium

### NVT: DCE/RPC and MSRPC Services Enumeration Reporting

#### Details

This vulnerability is one that allows for RPC services to be enumerated and queried.

This is more of an ‘information disclosure’ vulnerability than one which can modify the state of a server.

#### Scope

All RPC or MSRPC services running on the server can be queried.

#### Mitigation

Filter incoming traffic to prevent outside hosts from asking about RPC services.

### NVT: Cleartext Transmission of Sensitive Information via HTTP

#### Details

This web application communicates over unencrypted HTTP.

This means that a MITM attack can be easily performed by anyone listening to communications between the server and someone else.

Passwords and usernames entered by users are especially affected.

#### Scope

All unencrypted HTTP transmissions.

#### Mitigation

Enforce SSL and/or HTTPS over all connections.

### NVT: SSL/TLS: Report Weak Cipher Suites

#### Details

The TCP connection on port 3389 accepts old and weak cipher suites (MD5, SHA128) that can be easily cracked.

#### Mitigation

Use up-to-date cipher suites and do not accept old ones.

#### Scope

All TLS traffic.

### NVT: SSL/TLS: Certificate Signed Using A Weak Signature Algorithm

#### Details

The target uses a certificate in the certificate chain signed with sha1 & RSA, which is considered insecure now.

#### Mitigation

Generate a new certificate to replace the old one signed with a sufficiently strong signature algorithm.

#### Scope

All applications or services that rely on certificates to ensure validity of data.

Spoofing or hijacking attacks could be performed.

### NVT: SSL/TLS: Diffie-Hellman Key Exchange Insufficient DH Group Strength Vulnerability

#### Details

The target’s SSL/TLS service uses weak Diffie-Hellman groups.

The key size used is 1024, and should be >= 2048.

Someone listening to TLS communication could decrypt it offline, given enough time and computing power.

#### Mitigation

Use Elliptic-Curve Diffie-Hellman for TLS, some other sufficiently strong encryption, or a stronger DH group.

#### Scope

All TLS traffic.

## Low

### NVT: TCP timestamps

#### Details

The target implements TCP timestamps and helps attackers try to compute how long TCP packets have been up for.

#### Mitigation

Disable TCP timestamps.

# Penetration of Target

Penetration of target was not achieved.

# Summary

Most of the vulnerabilities I was able to found were rather simple, requiring updates to components of the server.

There was one high-impact item, and that one is the one that just requires a piece of software to be updated.

There were five medium-impact items, and mostly had to do with weak or no cryptography used.

There is also the issue of not preventing malicious actors from repeatedly trying usernames and passwords which is an issue that shows up frequently in the server.