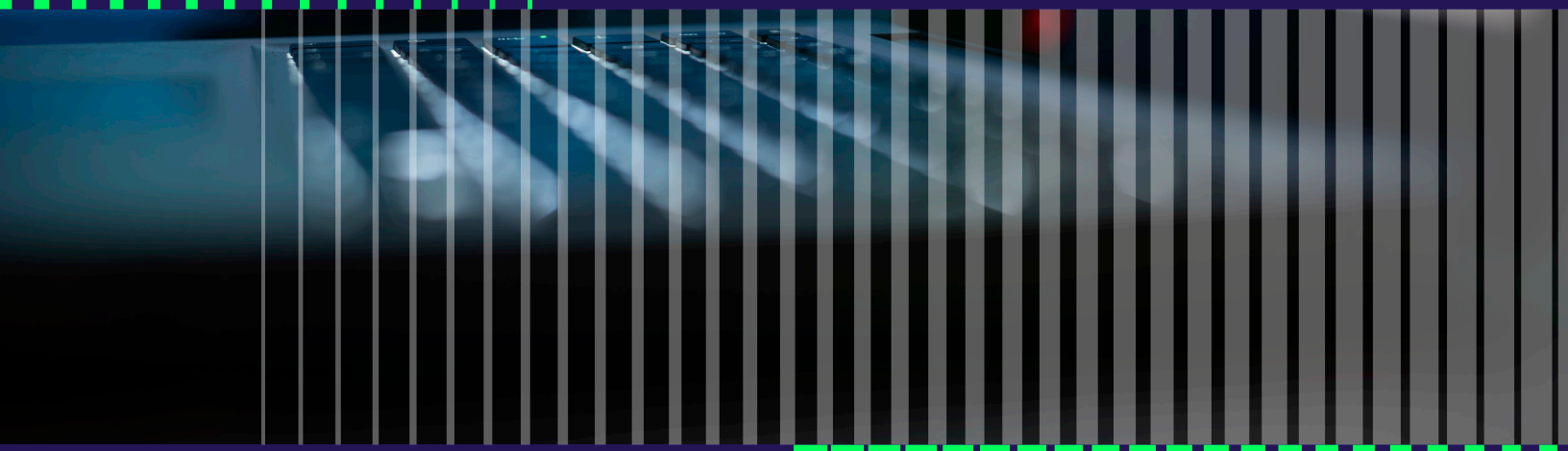
Delinea



Identify and Mitigate Risks of Privilege Escalation on Windows and Linux Endpoints



WHITEPAPER

Identify and Mitigate Risks of Privilege Escalation on Windows and Linux Endpoints

Privilege escalation is one of the most common techniques attackers use to deliver malware, discover and exfiltrate sensitive data, and shut down critical systems.

Privilege escalation is the process by which a User with limited access to IT systems can increase the scope and scale of their access permissions. For trusted Users, privilege escalation allows expanded access for a limited time to complete specific tasks. For example, Users may need access to troubleshoot a technical problem, run a quarterly financial report, or install a program.

From a hacker's perspective, however, privilege escalation is the art of increasing privileges from initial access, which is typically that of a standard User or application account, all the way up to administrator or full system access.

In this whitepaper, you'll learn how to combat unwanted privilege escalation by reverse engineering the tactics that cybercriminals use. By seeing the world through the eyes of a hacker you'll be able to identity the red flags of privileged-account attacks. And, you'll know which techniques and tools to use to mitigate them.

How privilege escalation attacks work

Privileged accounts are at the core of every business. They ensure the IT team can manage your organization's systems, infrastructure, and software, and they enable business users to access the data for critical decisions. Based on higher levels of authorization, privileged accounts enable administrative or specialized levels of access to sensitive, business-critical, and protected systems and data.

In a privileged account attack, attackers use common steps and proven techniques to identify system misconfigurations, vulnerabilities, overprivileged Users, and weak credentials. First, they explore systems to determine an attack path that won't alert the security team that anything malicious is occurring. They're looking for privileged accounts which are left unmanaged and unmonitored.

Privilege escalation is a key part of a criminal's attack path



Some of the most common Windows Operating System privileged accounts that attackers seek include:

- The king of accounts, "Domain Admin Accounts"
- The challenging and scary "Domain Service Accounts"
- The forgotten "Local Administrator Accounts"
- The help me "Emergency Accounts"
- The hidden and forever "Service Accounts"
- The elevated "Application Accounts"
- The silent but deadly "Privileged Data User Accounts"

You can learn more about these types of Windows privileged accounts in the blog: The 7 Deadly Privileged Accounts You MUST Discover, Manage, and Secure.

Some of the most common Linux privileged accounts that attackers go after include:

- The king of Linux, "Root"
- The secret keys of Linux, "the private SSH key"
- The challenging and scary "sudoers users and setuid/setgid"
- The forgotten "System Adm Accounts"
- The help me "Emergency Accounts"
- The hidden and forever "Service Accounts such as www-data"
- The elevated "Dev Accounts"
- The silent but deadly "Privileged Data User Accounts"

Stealing passwords

Let's say an attacker successfully steals a User's password or credentials and gains access to their privileged account. That password may enable certain privileges, for example, it may only unlock data stored locally on a laptop, web server, web application, or content management system.

But the attacker is hungry for more. They're looking for more sensitive data they can resell on the Dark Web. They're looking for access to business-critical systems so they can deploy ransomware, threaten shutdown, and demand financial payment.

To achieve those goals, an attacker employs a variety of strategies to escalate privileges, which apply broadly to both Windows and Linux systems.

- Vertical privilege escalation, sometimes referred to as privilege elevation, is when an attacker compromises a User account has limited permissions on a system. They then look for ways to increase their privileges using the same account.
- Horizontal privilege escalation, the more common method, is when an attacker gains access to another credential on the network with higher privileges than the initial one used to gain their foothold. With higher-level privileges, an attacker can move freely around the network without detection.

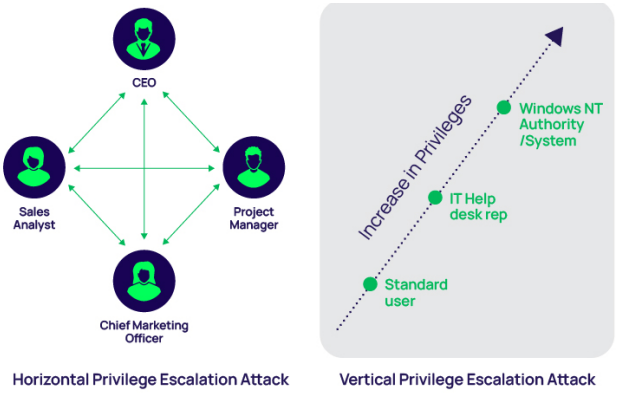


Figure 1: Difference between Windows vertical and horizontal privilege escalation

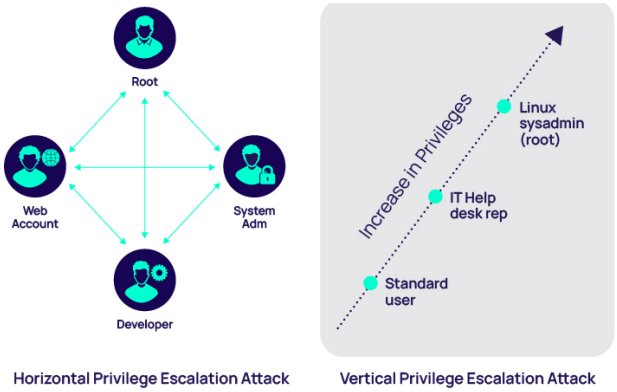
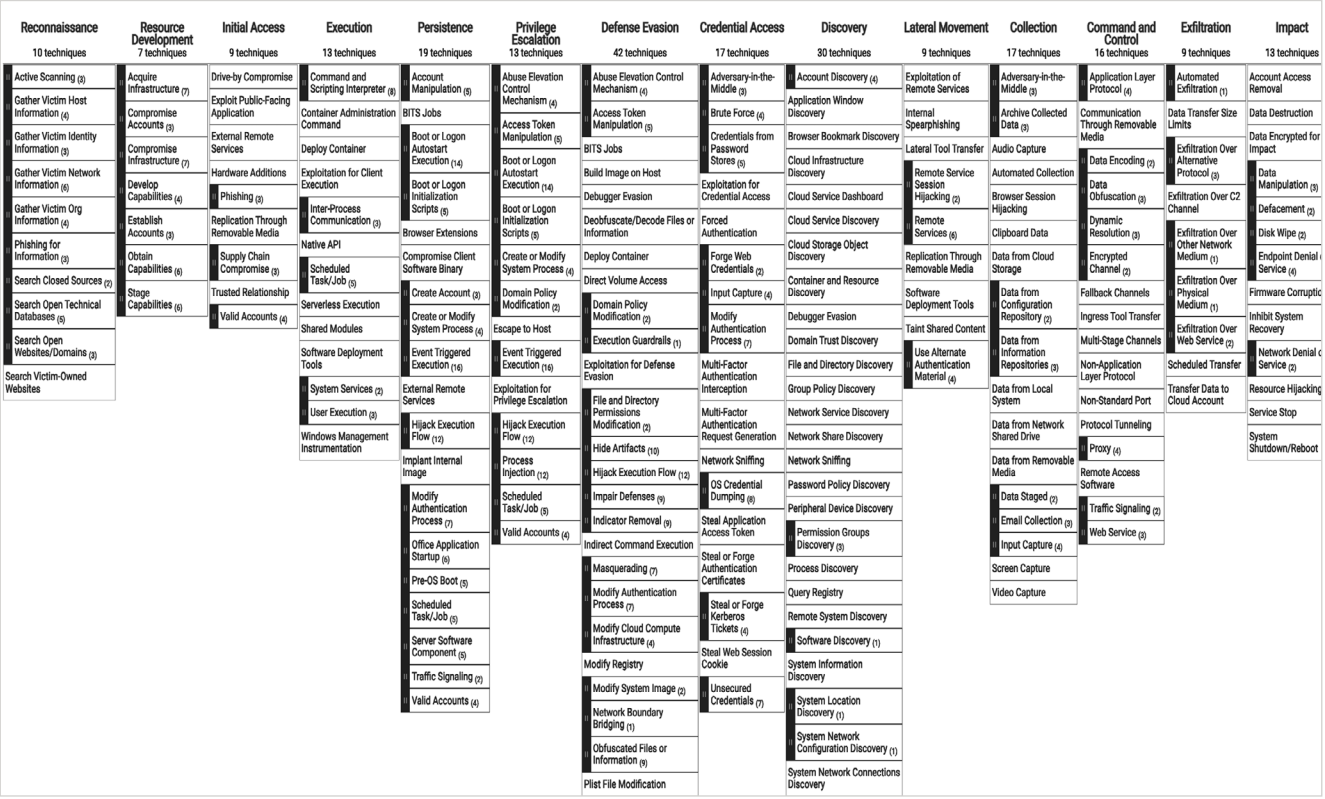


Figure 2: Difference between Linux vertical and horizontal privilege escalation

Mitre ATT&CK framework

The Mitre ATT&CK framework is an awesome knowledge base of the common privilege tactics and techniques used in real-world attacks. It's best practice to go through the privilege escalation techniques within the Mitre ATT&CK framework and ensure that you have mitigation controls, such as Privileged Access Management solutions, to reduce the risk of successful attacks.



Mitre ATT&CK Framework Privilege Escalation

Where should you start?

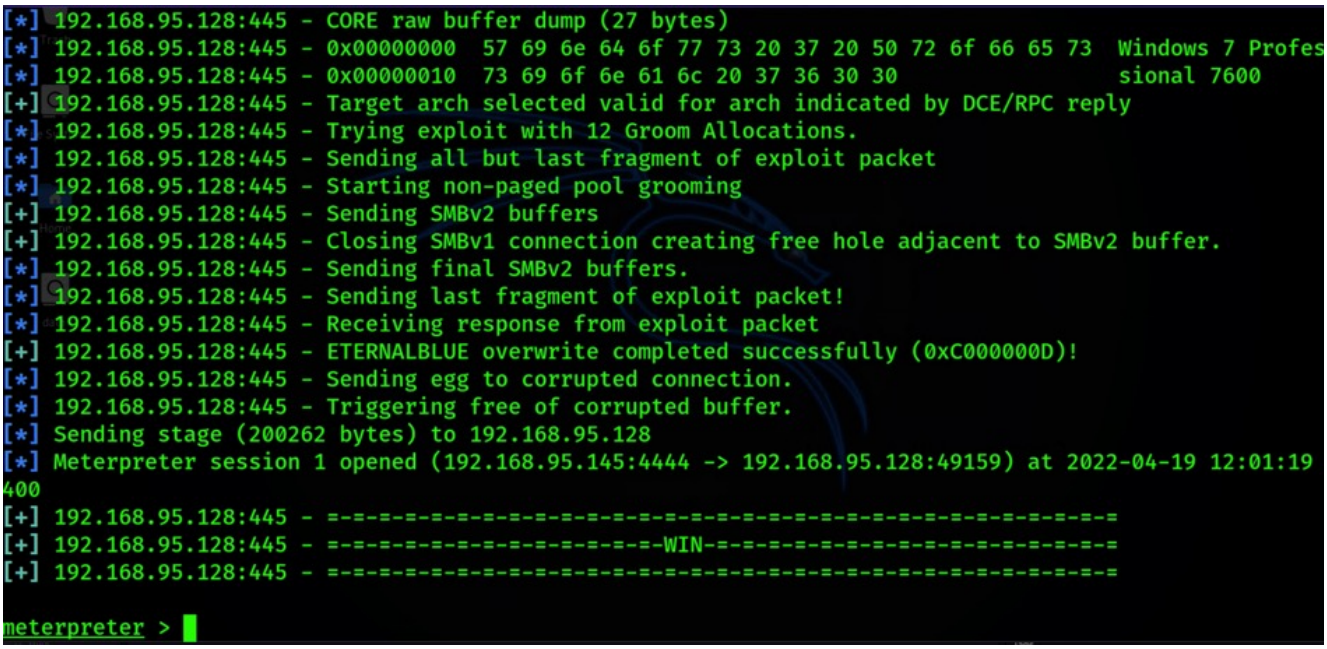
Because the way privileges are configured and managed are different for Windows and Linux, the tactics and tools cybercriminals use to escalate those privileges are also different.

Head to the section of the whitepaper that covers the systems you want to protect.

WINDOWS PRIVILEGE ESCALATION (page 5)

LINUX PRIVILEGE ESCALATION (page 15)

Privilege escalation for Windows



Example of when an attacker has pwned a system with NT Authority\System

Before we start getting into the mechanics of Windows privilege escalation attacks, we must understand what privileged accounts are used for in Windows, the different types of privileges on Windows systems, and how they work. This foundation will help you understand the strategies cybercriminals use when attacking Windows, so you know where to focus your defenses.

Types of Windows privileged accounts

A privileged account can be human or non-human (also known as machine identities). Some types of non-human high privileged accounts are application accounts used to run services requiring specific permissions.

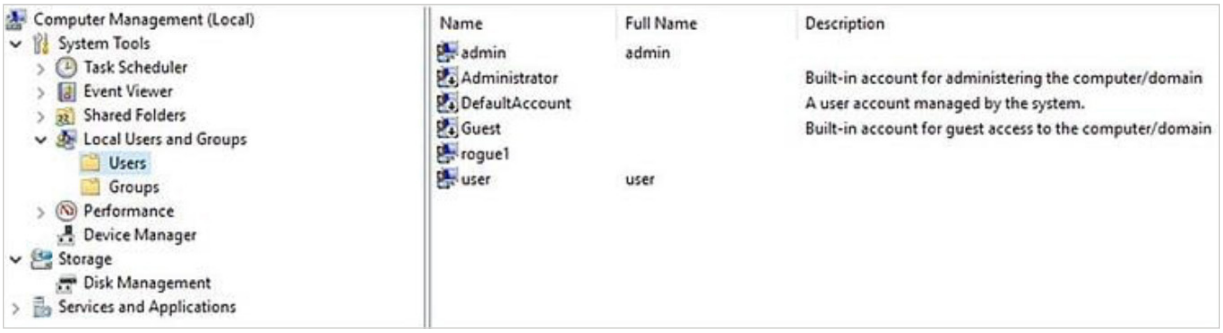
In Windows, privileged accounts are distinguished from standard User accounts that represent human identities, such as an Active Directory User account with a password to restrict access.

Privileged accounts allow Windows Users to make system and software configuration changes, execute administrative tasks, create and modify accounts, deploy software, back up data, install security patches, enable interactive logins, and of course, access privileged data. All these activities are crucial to ensure your business can function by keeping systems and software running.

Creating and delegating privileges in Windows systems

A privilege in Windows is the authorization delegated to a User account or Group that allows access to system resources, objects, and tasks. Privileges can be Local or Domain, which determines the scope of access the User account has.

On Windows systems, you can find a list of Local User Accounts under Local Users and Groups in the computer management menu. Administrator, Default, and Guest are default accounts.



Computer Management Local Users and Groups

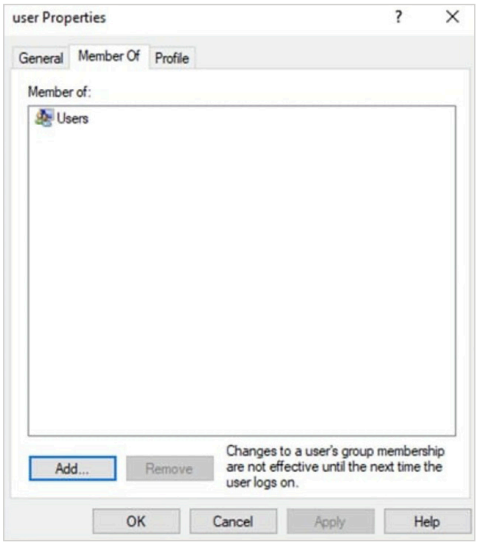
In Windows, a Local User account can be assigned as a member of a Group, which determines its privileges.

Default Groups on a Windows system depend on the operating system role and features enabled. Groups tend to be focused on roles or tasks that the User will perform, based on their job function.

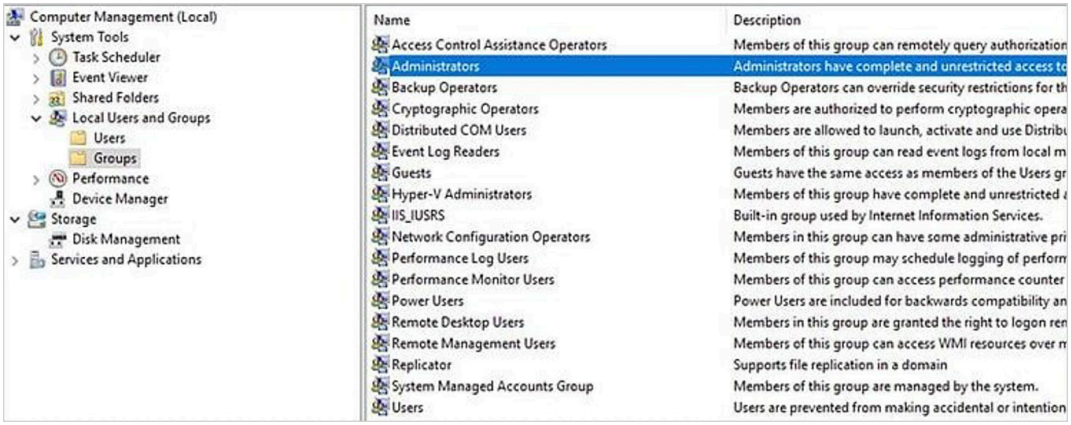
Unfortunately, many organizations rely on out-of-the-box roles. When unsure which role to assign to Users, most delegate Local Administrator privileges. This results in many overprivileged Users, which attackers search for and quickly abuse.

It's also common to have shared privileged access used by a department or group of Users to access applications or systems. This practice makes it difficult to identity which privileged account may have been compromised and track unique privileged behavior.

Each Windows system has its own Security Account Manager database, known as the SAM file, which stores User accounts and security descriptors on the local computer. When a User logs onto the system, they access a token that contains privileges. When they perform any action on the system, it checks if those permissions permit the action.

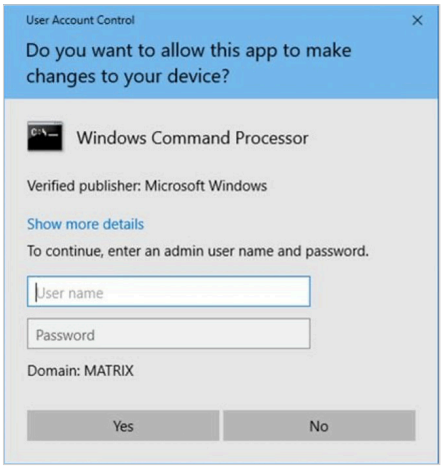


User account properties showing it is a member of the Users Group



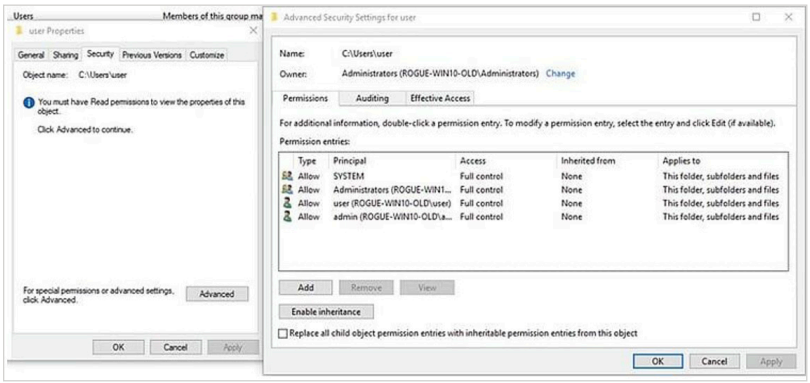
A few default Groups that determine the privileges of users

Typically, when a User doesn't have permissions to an object within Windows, they'll be prompted to enter a different User account with the necessary privileges. This is commonly known as User Account Control (UAC) and enables a User to run most tasks as a non-Administrator.

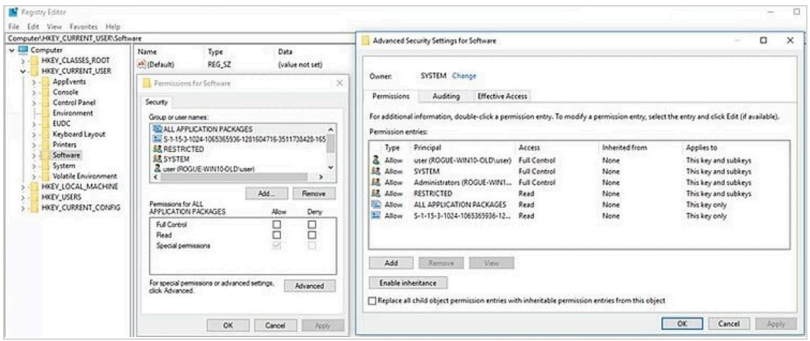


User Account Control requiring elevated privileges to run cmd application

Once the Users and Groups have been assigned and configured, security settings are determined and privileges are assigned to each Object, such as file systems, registries, services, and system resources. In a hierarchy, each Object can inherit permissions and privileges from its parent.



Common Security Settings aka ACL for an Object in Windows

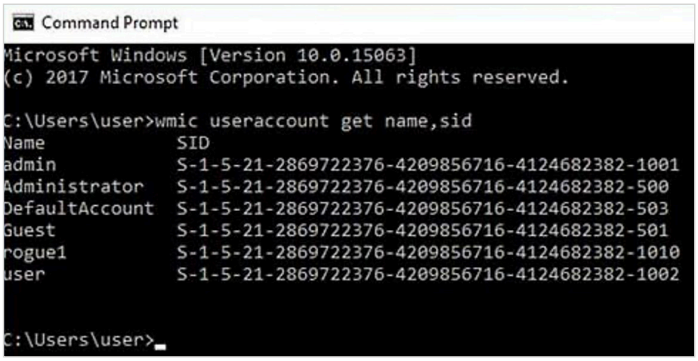


Example of permissions aka ACL within the registry

Windows Security Identifiers (SID)

In Windows, the SID is how the operating system refers to accounts and processes, instead of using an account or process name.

Each account, Group, and process, is assigned a unique SID (Security Identifier) to represent the security context it's running under. When a User logs onto a system or executes a process, the SID is assigned an access token that contains everything the system needs to determine the security context, permissions, privileges, Groups, and access.



An example of User Account SIDs

To check the User account's SID, you can use the WMIC (Windows Management Instrumentation Command-Line Utility). Use the command prompt: wmic useraccount get name,sid.

To learn more about SIDs, read Microsoft's full documentation: Microsoft Security Identifiers Documentation.

Did you catch the security concerns?

We've now covered the ways privileges are created and assigned in Windows. That's the "happy path," in which everything works out according to plan. But, even if you follow all the steps outlined above to manage privileges, you're leaving yourself open to a privilege escalation attack.

Next, we'll cover the "unhappy path." That's when a cyber attacker targets your Windows privileged accounts and successfully exploits the security holes.

Windows privilege escalation attacks and exploit techniques

Let's assume the attacker has gained an initial foothold on a Windows system. The initial foothold could mean different things, such as a reverse shell (aka without creds), access to an application running on the system, or creds (credentials) of an account with limited privileges, such as a standard user account. Each type of foothold allows the attacker to begin their privilege escalation attack path.

In a privilege escalation exploit, the attacker commonly seeks to discover as much as possible about an IT environment to determine their attack path. They do this through reconnaissance and enumeration of the compromised systems.

They perform some type of system enumeration using commands like the ones below:

Enumeration Commands	Description
ver	Windows Version
whoami	Current User
hostname	System Hostname
systeminfo	System Information
sc query state=all	Display Services
tasklist / svc	Display Processes and Services
ipconfig	Show IP Configuration
net users	Show Local Users
net localgroup	Show Local Groups
netsh advfirewall show allprofiles	Display Firewall Profiles and State

Below is an example of these enumeration commands in action:

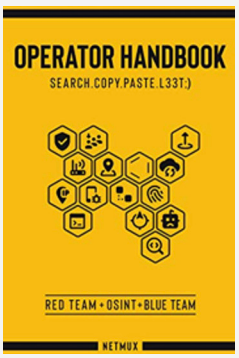
```
Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Windows\system32>net users

User accounts for \\ROGUE-WIN10-OLD

-------------------------------------------------------------------------------
admin                    Administrator            DefaultAccount
Guest                    rogue1                   user
The command completed successfully.


C:\Windows\system32>net localgroup
```

An example of manual enumeration

When performing enumeration, attackers are looking for security vulnerabilities that allow for privilege escalation exploits, such as:

1 | Insecure service permissions

This occurs when a service that's running under SYSTEM privileges, but the User has permissions to change the executable binpath to one which could create a reverse shell.



Operator Handbook by NETMUX is a great book that includes many enumeration commands and tips.
This book has an awesome collection of steps to enumerate a system.

2 | Unquoted service paths

Surprisingly, while this is a known technique used for many years, it's still common to find many services with unquoted service paths. When combined with weak folder permissions, this allows an attacker to place an executable in a parent folder, where Windows will look for the executable first to execute. For example, you might have a service path as "C:\Program Files\Vendor\binary.exe". When the path is unquoted and the User has permissions to place objects in the "C:\Program Files\" path, Windows will first try to execute program.exe. If the attacker can place a binary called program.exe in the path, they can then elevate privileges to the account which that service is running.

3 | Weak registry permissions

Like the insecure service permissions example, if an attacker can modify the registry configuration of a service, they can then change the path in service configuration to execute a binary they choose. This could create a reverse shell or elevate privileges on the system.

4 | Insecure service executables

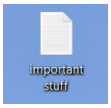
If an attacker can simply replace the original executable with their own, they can then gain privilege escalation of the account which that service is running under.

5 | Passwords

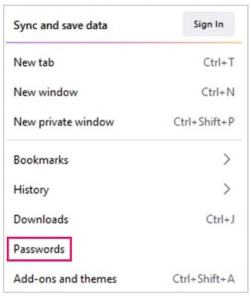
Many people create weak, crackable passwords, reuse them, and share them. It's still common for people to store passwords in easy-to-find places, like a text file on a desktop, in a browser, or in a configuration file.

Attackers can quickly search for stored passwords using the common techniques, such as:

- Searching the registry using the query: reg query HKLM /f password /t REG_SZ /s.
- Searching the file system within xml, ini and txt files for the string password: findstr /si password *.xml *.ini *.txt.
- Finding a text file on the desktop labeled something like "passwords," or "important stuff."



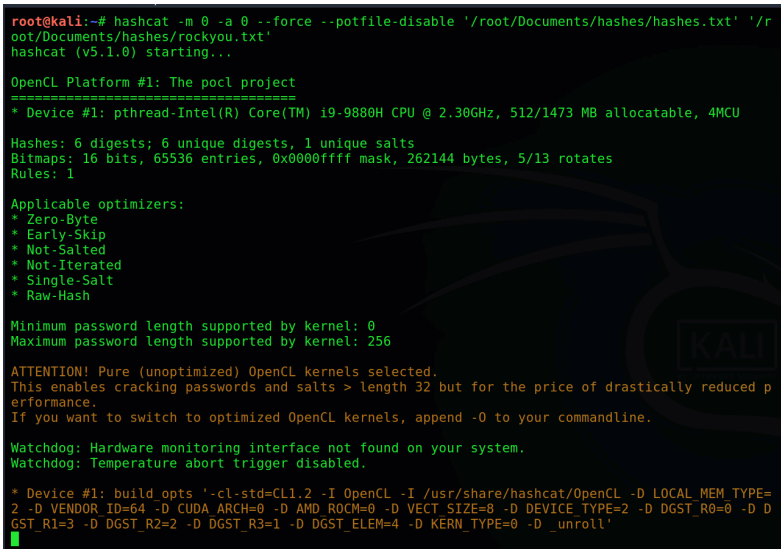
- identifying stored passwords in an internet browser.



6 | Overprivileged Users

If standard business Users have Local Administrator rights on their personal workstations, attackers can leverage these Local Administrator rights to escalate privileges up to Full Domain using tools such as mimikatz and changes in the OS configuration. You can see how this privilege escalation strategy progresses in the video below.



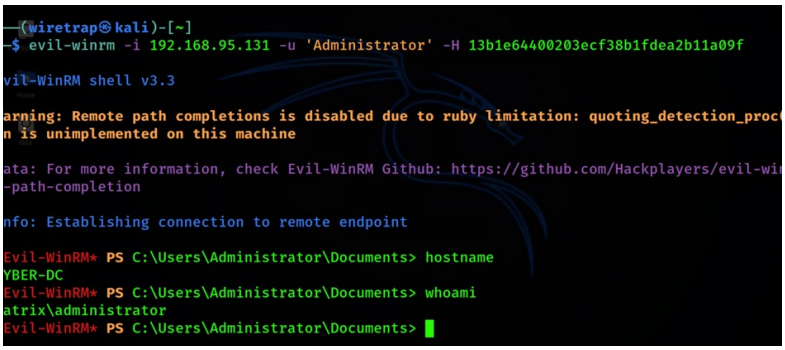


An example using Hashcat to crack hashes

7 | Security Account Manager (SAM)

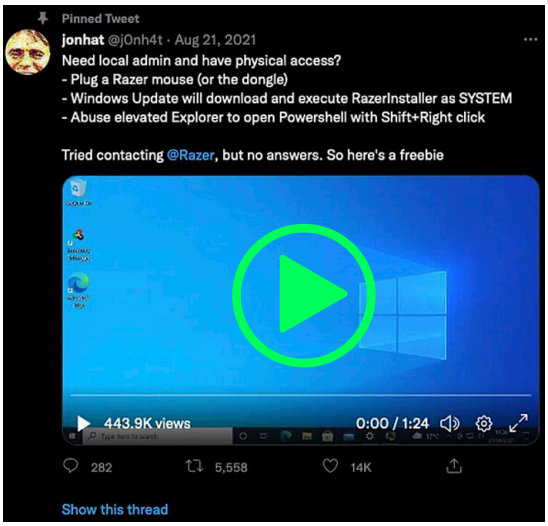
Windows stores credentials within the Security Account Management (SAM) database in an encrypted form called a hash. This is a one-directional cryptographic algorithm, which means you need to know the original password to recreate the hash. The SAM and SYSTEM files are locked when Windows is running but an attacker might find a backup copy under the Windows repair directory.

Getting access to both the SYSTEM and SAM database could allow an attacker to extract the hashes. If the original password was a weak, reused password, it's very likely the attacker can crack it. Here's an example of cracking a hash using Hashcat.



8 | Pass-the-hash

If an attacker can't crack the password from the hash, they may still be able to move around the network using only the hash. This technique is called pass-the-hash. An example of privilege escalation using pass-the-hash for lateral movement is to the left.



9 | Insecure GUI apps

For example, a recent vulnerability in a Razer Mouse software enabled a User who plugs in a mouse to escalate privileges to a Windows 10 Administrator. Once the mouse was plugged in, a Windows update would launch, downloading the Razer Installer as SYSTEM. A User could then use Explorer to launch an elevated PowerShell window.

An example of privilege escalation using insecure GUI Apps is to the left.

```
------------------------------------------------------------------------------
admin                    Administrator            DefaultAccount
Guest                    rogue1                   user
The command completed successfully.


C:\Users\neo\Desktop\enum>powershell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\neo\Desktop\enum> Import-Module .\CVE-2021-34527.ps1
PS C:\Users\neo\Desktop\enum> Invoke-Nightmare
[+] using default new user: adm1n
[+] using default new password: P@ssw0rd
[+] created payload at C:\Users\neo\AppData\Local\Temp\nightmare.dll
[+] using pDriverPath = "C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_8c30031d8d0a
[+] added user  as local administrator
[+] deleting payload from C:\Users\neo\AppData\Local\Temp\nightmare.dll
PS C:\Users\neo\Desktop\enum> net users

User accounts for \\ROGUE-WIN10-OLD

------------------------------------------------------------------------------
adm1n                    admin                    Administrator
DefaultAccount           Guest                    rogue1
user
The command completed successfully.

PS C:\Users\neo\Desktop\enum>
```

10 | OS vulnerabilities or kernel exploits

While these types of escalation attacks are less common, sometimes attackers will wait for months with access to a low privileged User, waiting for the moment a vulnerability is disclosed. A recent example known as Print Nightmare (CVE 2021 34527) is a vulnerability in the Print Spooler that enabled an attacker to perform remote code execution and privilege escalation.

An example of privilege escalation using Print Nightmare to create an Admin User on the system is to the left:

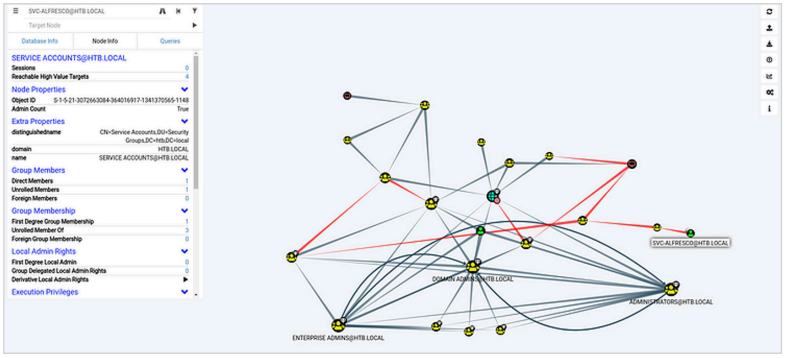
How cybercriminals automate privilege escalation in Windows

To avoid detection, a hacker wants to escalate privileges as quickly as possible. Therefore, they may use tools to automate the privilege escalation process, such as the examples below:

BloodHound

BloodHound is a single-page Javascript web application, built on top of Linkurious, compiled with Electron, with a Neo4j database fed by a C# data collector. It uses graph theory to reveal hidden and often unintended relationships within an Active Directory environment.

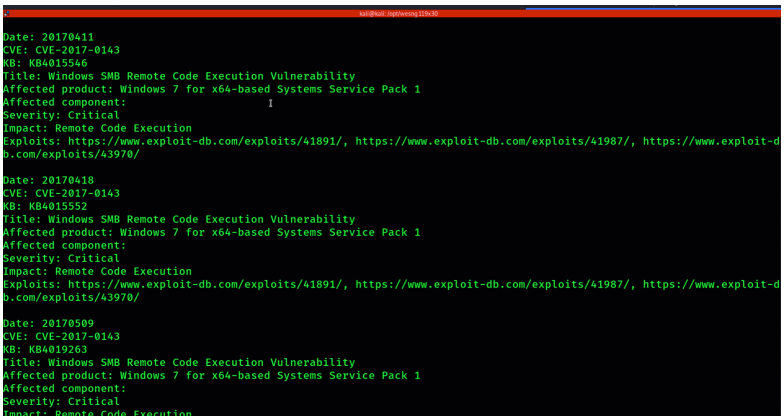
Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to find. Defenders can use BloodHound to identify and eliminate those same paths. Both blue and red teams can use BloodHound to easily gain a deeper understanding of privilege relationships in an Active Directory environment. An example of an Active Directory privilege escalation attack is below, using Bloodhound to find the attack path to elevation of privileges.



Bloodhound example from Hackthebox

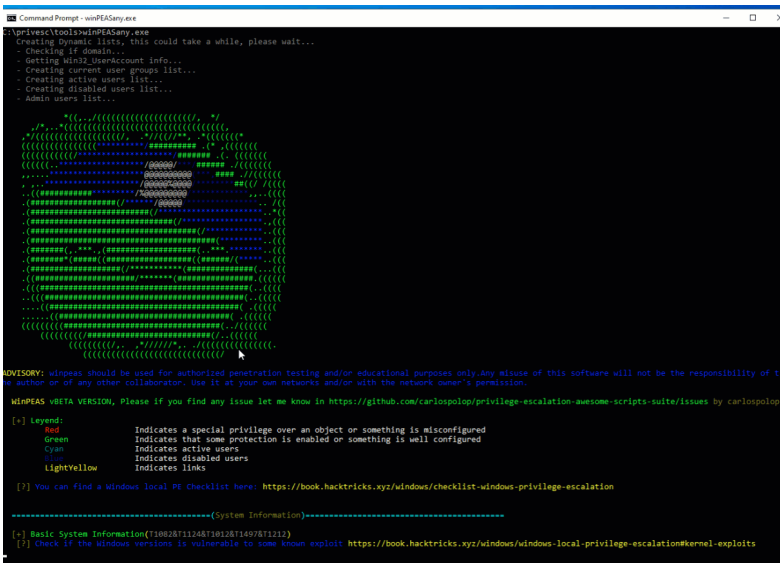
Windows Exploit Suggester—Next Generation (WES-NG)

WES-NG is a tool based on the output of Windows' systeminfo utility. It provides the list of vulnerabilities the OS is vulnerable to, including any exploits for these vulnerabilities. Every Windows OS between Windows XP and Windows 10, including their Windows Server counterparts, is supported.



Windows Privilege Escalation Awesome Scripts

Using winPEAS can help Administrators look for ways to prevent privilege escalation attacks by assessing systems for possible misconfigurations or vulnerabilities. WinPEAS is a compilation of local Windows privilege escalation scripts to check for cached credentials, user accounts, access controls, interesting files, registry permissions, service accounts, patch levels, and more. WinPEAS is helpful because it includes hints on where you should focus your attention.

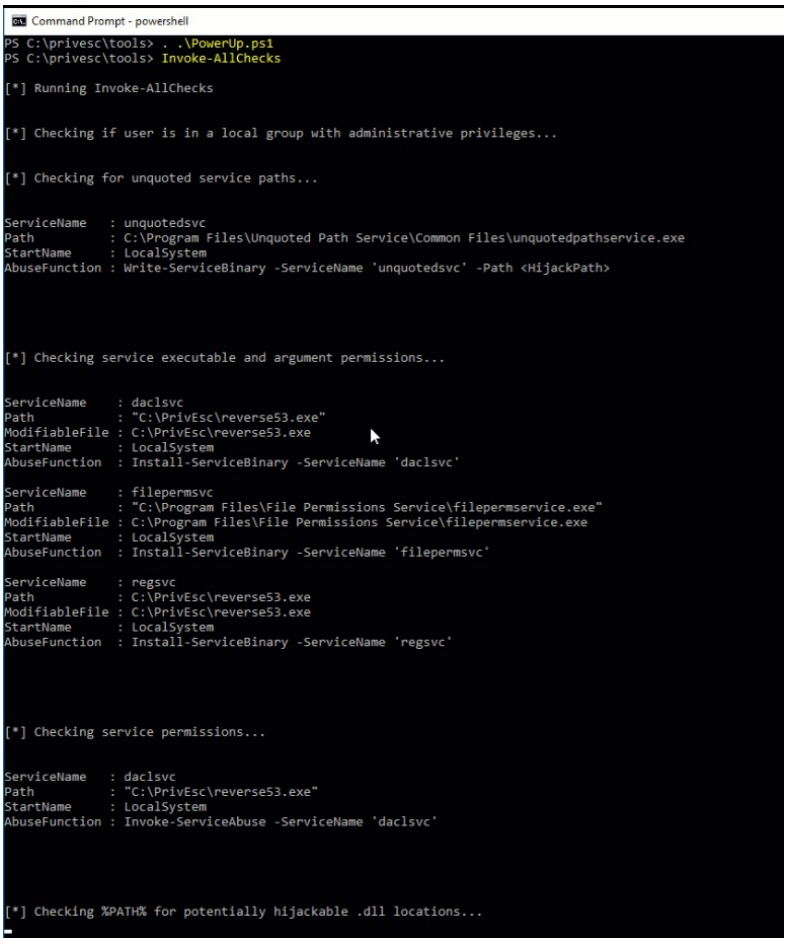


WinPEAS running on Windows 10 endpoint

Carlos Polop, author of winPEAS and Hacktricks.xyz, discussed winPEAS and privilege escalation on the 401 Access Denied podcast. Make sure to check out the podcast and Carlos's Hacktricks book, as it goes into full detail on many of the techniques used in this blog.

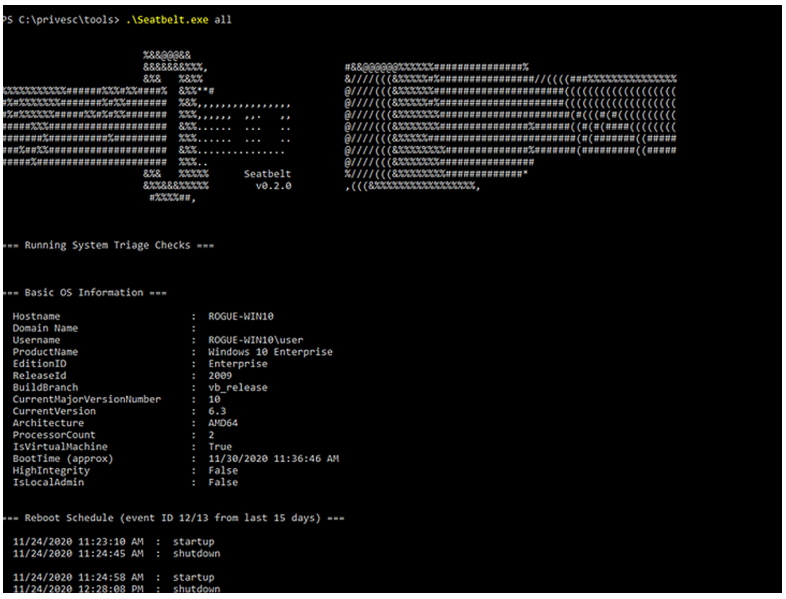
PowerUp

PowerUp is a collection of PowerShell scripts for finding common Windows privilege escalation vectors that rely on misconfigurations. Running Invoke-All checks will look for common misconfigurations on Windows endpoints.



Seatbelt

Seatbelt is a C# project that performs security-oriented host-survey “safety checks” relevant from both offensive and defensive security perspectives. Simply, Seatbelt is an enumeration tool to show information that could lead to privilege escalation.



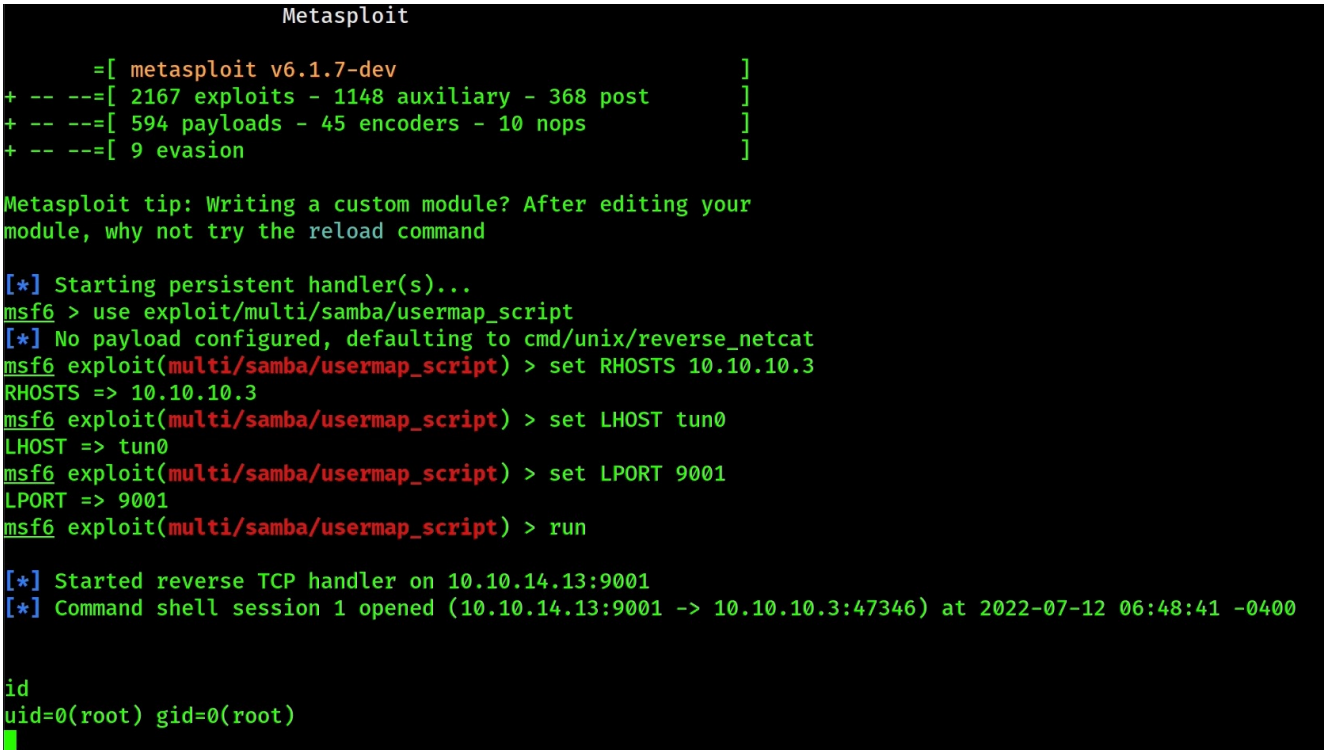
Seatbelt running on Windows endpoint

How to reduce risks from privilege escalation attacks

Now that we've covered the common attack strategies cybercriminals use to escalate privileges on Windows, let's dig into the defense strategies you can use to protect those systems.

Skip to page 5 to see how you can reduce risks from privileged escalation attacks on Windows.

Privilege escalation for Linux



Example of a Linux attack exploiting a known vulnerability in Samba

For authorized Users on Linux, privilege escalation allows elevated access to complete a specific task or make system configuration modifications. For example, system administrators may need access to troubleshoot a technical problem, add a User, make configuration changes to an application, or install a program.

On Linux, this is typically done via the sudo (Super User DO) command that enables condition-based privilege elevation for User accounts. To use sudo privilege elevation, you simply precede the command with sudo, which will then execute the command as a super-user.

An example of using sudo to execute ifconfig as a super-user is below:

```
└─$ sudo ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:0c:29:d9:9d:21  txqueuelen 1000  (Ethernet)
        RX packets 112667  bytes 154200278 (147.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16325  bytes 1393613 (1.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Another common method for authorized Users to elevate privileges is via the setuid or setgid, a special permission bit set on an executable that will run with elevated privileges such as root when executed.

Basics of Linux privilege escalation

Before we explain how to mitigate unwanted privilege escalation, it's important to have a basic understanding of how access controls work on Linux systems.

User account management is one of the critical steps to hardening and securing Linux systems. Unmanaged User accounts leave a door open for attackers to exploit. Unused User accounts or accounts with easily cracked or guessable passwords are the most targeted by attackers.

A big difference between Windows and Linux is that Windows was originally planned as a single-user system with administrator privileges. In contrast, Linux was designed as a multi-user operating system. With many Users accessing the same system, you can see why managing directory and file access is critical.

How privileges are created and delegated in Linux systems

One of the most important files on the Linux system is the passwd file, located at /etc/passwd. This file lists all the Users known to the system which could also be included in directory services.

If we look inside the passwd file using the "cat" command, we find something like the following:

Each line represents a User on the Linux system.

Each field is separated using the colon ":" character in which the fields represent the following passwd file format:

1. Username
2. Password Placeholder (x indicates encrypted password is stored in the /etc/shadow file)
3. User ID (UID)

```
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
_apt:x:104:65534::/nonexistent:/bin/false
usbmux:x:105:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
rtkit:x:106:110:RealtimeKit,,,:/proc:/bin/false
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/bin/false
messagebus:x:108:111::/var/run/dbus:/bin/false
geoclue:x:109:115::/var/lib/geoclue:/bin/false
avahi:x:110:117:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
colord:x:111:118:colord colour management daemon,,,:/var/lib/colord:/bin/false
saned:x:112:119::/var/lib/saned:/bin/false
speech-dispatcher:x:113:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
pulse:x:114:120:PulseAudio daemon,,,:/var/run/pulse:/bin/false
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
Debian-gdm:x:116:122:Gnome Display Manager:/var/lib/gdm3:/bin/false
sshd:x:117:65534::/run/sshd:/usr/sbin/nologin
james:x:1000:1000:james:/home/james/:/bin/bash
mindy:x:1001:1001:mindy:/home/mindy:/bin/rbash
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$
```

Example of a Linux attack exploiting a known vulnerability in Samba

4. Group ID (GID)
5. Personal Information (separated by comma's) – can contain full name, department, etc.
6. Home Directory
7. Shell – absolute path to the command shell used (if /sbin/nologon then logon isn't permitted, and the connection gets closed)

Any User with a user ID above 1000 (this can slightly vary on certain Linux/Unix operating systems) isn't a default User and has been added to the system. "UID 0" is reserved for the root account, however it can also be used for other Users, giving them super-user privileges. "Effective ID 0" and "1-99" are for predefined accounts and "100-999" are for system administration. If you find + or – in the field, this likely represents an account being used with a directory service, for example LDAP.

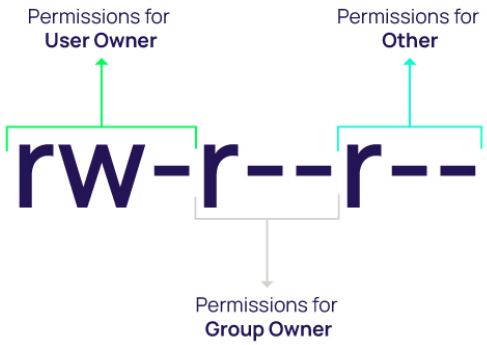
In older Linux systems the password of the User was stored in the /etc/passwd file, however this was deemed insecure and was replaced using the /etc/shadow file. That's why the password placeholder containing an "x" indicates the password is stored in the /etc/shadow file.

The passwd file is typically readable by all Users. For example, if you run the command "ls -la /etc/passwd" you will get output like the following:

```
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$ ls -la /etc/passwd
-rw-r--r-- 1 root root 2107 Aug 22  2017 /etc/passwd
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$
```

Using the ls -la command to view file permissions

The permissions in the "ls -la" output for the /etc/passwd file show the following format:



The first character "-" is reserved as a special permission that can vary. Another special permission that can be used here is known as SUID, GUID or the Sticky Bit. We will cover the special permission a little bit later.

Permission Groups are defined as follows:

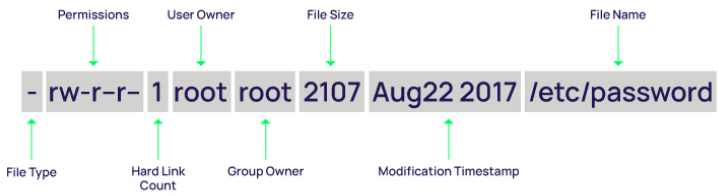
1. Owner
2. Group
3. All Users

Permission Types are defined as follows:

1. Read = 4
2. Write = 2
3. Execute = 1
4. – (No permissions set) = 0

So, let's explain the output from the /etc/passwd file above. It has the following access permissions:

1. File Type
2. Owner (root) has read and write permissions
3. Group (root) has read permissions
4. All Users has read permissions
5. Number represents hard links to the file
6. Owner
7. Group



Special Permissions (first bit in permissions) has the following options:



1. _ - no special permissions set
2. d – directory
3. l – file has symbolic links
4. s – setuid or setgid is set
5. t – sticky bit set

Next, let's cover Access Control Lists or File Attributes on Linux. These can apply to both directories and files. You can use the "getfacl" command to get the file access control lists for each file. It will display the access permissions defined by the traditional file mode permission bits.

```
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$ getfacl user.txt
# file: user.txt
# owner: mindy
# group: mindy
user::rw-
group::---
other::---

${debian_chroot:+($debian_chroot)}mindy@solidstate:~$
```

Using the getfacl to view the file access control lists

Some common commands used to view or manage access control lists in Linux are:

- getfacl – get file access control list
- setfacl – set file access control list
- chmod – change file mode bits
- acl – Access Control Lists
- chown – change file owner and group

Users can also be a member of a group on Linux systems, and this is defined in the /etc/group file in which an example is shown to the right:

Group format is as follows:

1. Group Name
2. Password Placeholder
3. Group ID
4. Members of the group

```
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:james
floppy:x:25:james
tape:x:26:
sudo:x:27:
audio:x:29:pulse,james
dip:x:30:james
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:james
sasl:x:45:
plugdev:x:46:james
staff:x:50:
games:x:60:
users:x:100:
nogroup:x:65534:
```

Let's talk about the /etc/shadow file, along with the format and how it's used.

The shadow file is one of the most protected files on a Linux system as it contains the encrypted password used by all known Users to log on to the system. It stores the User account information along with details on password settings. If the shadow file can be accessed by an unauthorized User, then attackers can attempt to crack the hash to find the clear text password used. If the password is human-created, this method is often successful for an attacker.

```
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$ ls -la /etc/shadow
-rw-r----- 1 root shadow 1375 Aug 22  2017 /etc/shadow
${debian_chroot:+($debian_chroot)}mindy@solidstate:~$
```

Root is the only user who can read and write to the shadow file

```
id
uid=0(root) gid=0(root) groups=0(root)
cat /etc/shadow
root:$6$iQr/I1zE$4WMMLNiLVeP0bWApzkPgIORQpVmjkV1FeTiI0usV1fFsZu9nemKXtp31G3cDZJV6k76G8oSimA29F9vGl375/:17400:0:99999:7:::
daemon:*:17336:0:99999:7:::
bin:*:17336:0:99999:7:::
sys:*:17336:0:99999:7:::
sync:*:17336:0:99999:7:::
games:*:17336:0:99999:7:::
man:*:17336:0:99999:7:::
lp:*:17336:0:99999:7:::
mail:*:17336:0:99999:7:::
news:*:17336:0:99999:7:::
uucp:*:17336:0:99999:7:::
proxy:*:17336:0:99999:7:::
www-data:*:17336:0:99999:7:::
backup:*:17336:0:99999:7:::
```

Like the passwd file, each field is separated by a colon ":" and the format of the shadow file is the following:

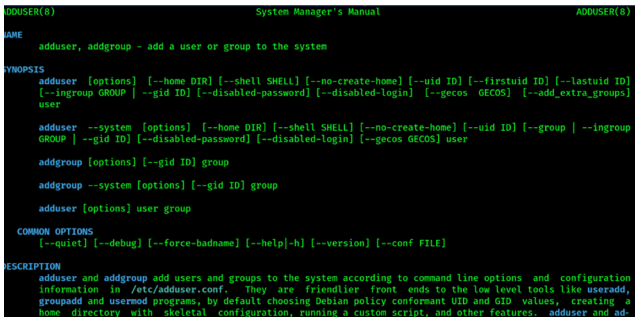
1. Username
2. Password (typically encrypted in a one-way hash format) such as:
   a. 1 is MD5
   b. $2a$ is Blowfish
   c. 5 is SHA-256
   d. 6 is SHA-512
3. Last password change
4. Minimum password age
5. Maximum password age
6. Warn period
7. Inactivity period
8. Expiration date
9. Unused field

Common commands used to manage users on Linux systems are the following:

- adduser – add a User to the system
- addgroup – add a group to the system
- deluser – delete a User from the system
- usermod – modify a User account
- passwd – used to change a User password

If you want to read more details on how to use each of the commands noted so far, including a full description along with command options, refer to the Linux man command, which is used to access the Linux man pages.

An example is below for the command "man adduser":



The output when using "man adduser" command

Below is an example of using "mkpasswd" to create an encrypted password with a SHA-512 hash. This can sometimes be helpful when you want to add a User with password directly in the passwd file or if you have permissions to modify the shadow file. As you can see, the password starts with 6, which indicates that this password is encrypted with SHA-512 followed by the hash salt. Then the password hash is after the 3rd "$" sign.

```
─$ mkpasswd -m sha-512 newpassword
$6$gtPUno/KmNzoHQ6Y$iyHJlH4rPpr1s9rjF1/OCxv9dR4LVZjZHQhHMB90fnC8MiNls/3OMo0rjiLazHrCVhccyY94.UMcIFBlAEpSs0
```

Now that you have a basic understanding of Linux access controls and permissions, let's move on to common Linux privilege escalation techniques used by cybercriminals.

Linux privilege escalation attacks and exploit techniques

In a Linux privilege escalation attack, an attacker gains initial access to a limited or full interactive shell of a basic User or system account with limited privileges. They perform enumeration to discover the path to elevate access to the root user, the default super-user account on all Linux- based systems. Once they gain root user access, they have ultimate control of an entire Linux system.

For example, they might add the compromised account to sudoers file so they can execute commands as the super-user account or use the special permission bit set by setuid and setgid to execute an executable as a privileged user.

To target privileged accounts, cyber attackers use common steps and proven techniques to identify system misconfigurations, vulnerabilities, overprivileged Users, and weak credentials.

- Kernel exploits
- Application vulnerabilities
- Misconfigurations such as weak file permissions
- Abuse of sudo
- Abuse of setuid and setgid
- Cron jobs
- Poor passwords

Attackers explore Linux systems to determine an attack path that won't alert the security team. They seek privileged accounts which are left unmanaged and unmonitored. The attackers prefer to stay stealthy, hiding for long periods of time to find the path to the root user on Linux. Kernel exploits tend to be the last step attackers will take, as sometimes they can be noisier and alert the security team.

In a privilege escalation exploit, the attacker commonly seeks to discover as much as possible about an IT environment to determine their attack path. They do this through reconnaissance and enumeration of the compromised systems.

During the enumeration phase of privilege escalation, it's common for attackers to search for all the possible ways to elevate privileges by checking out the detailed version and configuration of the Linux system.

Some common commands used during the enumeration phase are the following:

Enumeration Commands	Description
id	print real and effective user and group IDs
whoami	current user
hostname	show or set the system's host name
uname	print system information
ps -ef	report a snapshot of the current processes
echo $PATH	print environment PATH variable
ifconfig	configure a network interface
cat /etc/passwd	show passwd file contents
sudo -l	list commands allowed using sudo
find / -type f -a \(-perm -u+s -o -perm -g+s \) -exec ls -l {} \; 2> /dev/null	Find all files suid and sgid files

If you're doing something like a Capture the Flag gamification event (a safe way to try your skills using hackthebox.com or tryhackme.com), then you can try out some of the automated ways to perform enumeration. However, these are typically very noisy methods and will create alerts. It's common in live environments for attackers to use manual methods, but it's important to familiar yourself with these automated Linux privilege escalation enumeration tools as well.

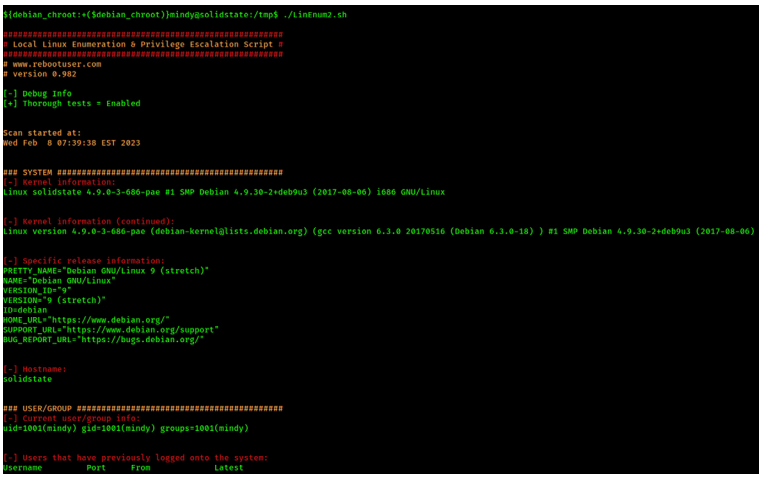
LinEnum – Linux Enumeration Script

This is an awesome Linux enumeration script. It's run on the target host and searches for many of the common privilege escalation methods or misconfigurations.

Some of the enumeration information collected includes:

Kernel and distribution release details

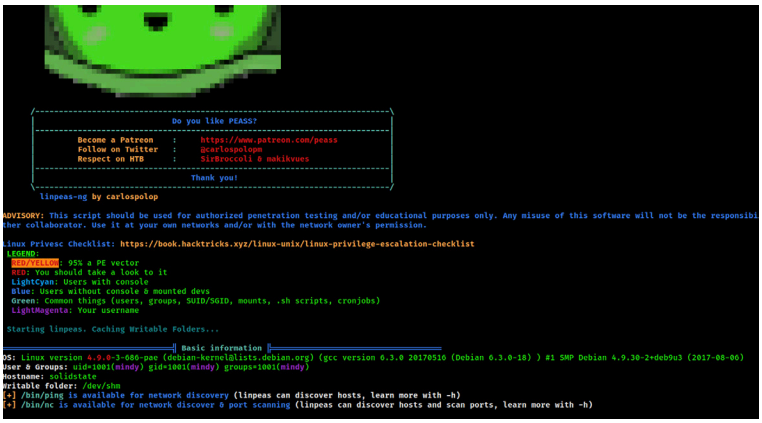
- System information
- User information
- Privileged access
- Environmental information



LinPEAS - Linux Privilege Escalation Awesome Script

LinPEAS is a script that searches for possible paths to escalate privileges on Linux/Unix*/ MacOS hosts. The checks are explained on book.hacktricks.xyz.

Check the Local Linux Privilege Escalation Checklist from book.hacktricks.xyz.



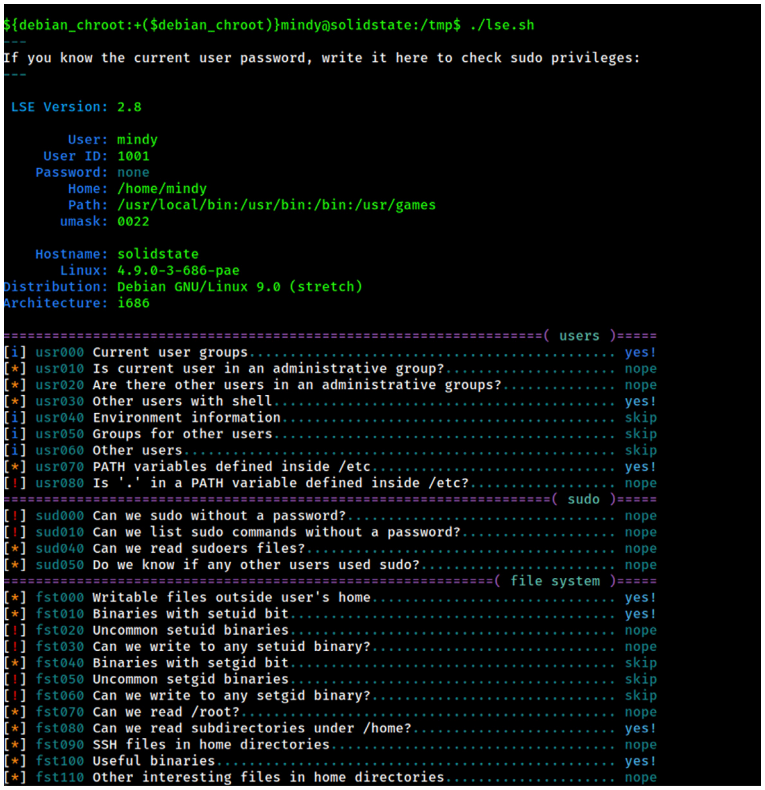
We recently had the awesome Carlos Polop, author of linPEAS and Hacktricks.xyz, on the 401 Access Denied podcast to discuss winPEAS, linPEAS and privilege escalation. Make sure to check out the podcast and Carlos's Hacktricks book, as it goes into full detail on many of the techniques used in this blog.

Linux Smart Enumeration

These Linux enumeration tools are designed for penetration testing and CTFs.

This project was inspired by https://github.com/rebootuser/LinEnum and uses many of its tests.

Unlike LinEnum, lse tries to gradually expose the information depending on its importance from a privesc point of view.



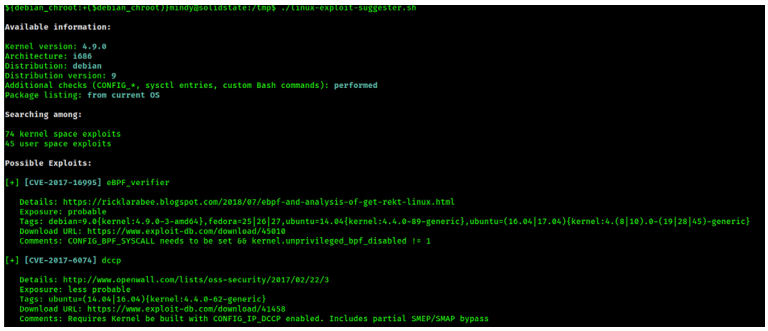
Linux Exploit Suggester 2

This next-generation exploit suggester is based on Linux_Exploit_Suggester.

Key improvements include:

- More exploits!
- Option to download exploit code directly from Exploit DB
- Accurate wildcard matching. This expands the scope of searchable exploits.
- Output colorization for easy viewing.
- And more to come!

This script is extremely useful for quickly finding privilege escalation vulnerabilities both in on-site and exam environments.



Linux privilege elevation techniques

Now let's look at some of the methods hackers may use to elevate Linux privileges to compromise the root super-user. Performing enumeration provides ideas on possible privilege escalation paths. The next step is to validate whether those will apply to this Linux system and configuration. It's always important to double-check everything such as operating system version, architecture, and configuration along with binary file version used as all of those could have an impact on the results.

1 | Kernel exploits

Kernel exploits tend to be tricky as they can cause system crashes or system instability, so handle with care. These methods tend to be the last resort to get local privilege escalation.

Dirty c0w is a vulnerability that impacted many Linux Kernels prior to 2018. It exploited a race condition affecting the copy-on-write mechanism within the memory management system, allowing an unprivileged user for example to overwrite the /etc/passwd file. Quite a few different variations of the Dirty c0w exploit exist, all performing slightly different elevation techniques using the same vulnerability. It's also important in this case to check the exact system architecture on the target system.

The following is an example of using the Dirty c0w vulnerability:

```
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fi1IpG9ta02N.:0:0:pwned:/root:/bin/bash

mmap: 7ffe8ec67000
ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'password123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
user@debian:~$ madvise 0

Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'password123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd

user@debian:~$ su firefart
Password:
firefart@debian:/home/user# id
uid=0(firefart) gid=0(root) groups=0(root)
firefart@debian:/home/user# _
```

Another Kernal exploit is known as the Nelson (Full Nelson or Half Nelson). This name is attributed to the researcher who discovered the vulnerabilities. It's a much older, local privilege escalation technique and another way to get to root. Many unpatched systems are still vulnerable to this technique.

2 | Programs or services running as root

If you discover that a program or service running on the system is configured to run as the root user, you might be able to find paths to escalate privileges if those programs allow you to execute a command via a script, web interface via a web application, or mssql. It's common that many applications allow Users to execute code or scripts, and if they run under the root user then an unauthorized User could use those techniques to gain access.

3 | Scripts running as root user

Many system administrators of Linux systems tend to configure programs or scripts to run under the root user since they might require certain system permissions to function correctly. If these programs or scripts are misconfigured and allow other Users to modify them, an attacker could execute a command under the root privileges.

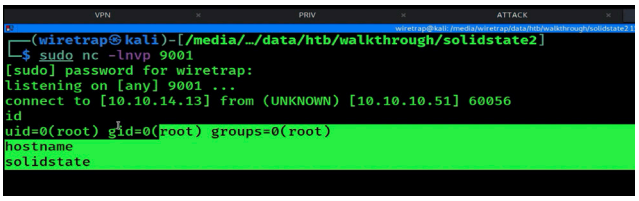
An example of this is a script that cleans up a /tmp folder that's running under the user root. It's misconfigured to be a world-writable file allowing an attacker to make changes and gain access to the root super-user.

Running LinEnum, the Linux enumeration script discovers a world-writable file owned by the root user.

```
[-] World-writable files (excluding /proc and /sys):
-rwxrwxrwx 1 root root 105 Aug 22  2017 /opt/tmp.py
```

When looking at the content it's a simple Python script that cleans up a temp directory. Because we can write to this file we can edit and get a reverse shell under the root user.

An example of using this technique is below:



Example of Linux PrivEsc using Hackthebox platform SolidState machine

4 | Installed software

If a User has installed software on the Linux system and it hasn't been patched, it might include a vulnerability that allows an unauthorized user to gain privilege escalation. Using searchsploit in kali or searching exploitdb could discover that the Linux system is running some unpatched software.

5 | Weak/reused/plaintext passwords

If passwords are created by Users, it's common to find weak or commonly reused passwords across Linux systems and applications. If you find a password being used for a web application, for example, you might also find that it's the same password the User is using for the User account – or even worse, for the root user. Searching common configuration files such a config.php or looking in a User's command history could lead to finding some interesting ways to elevate privileges.

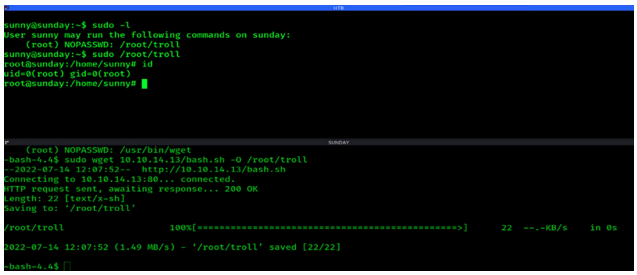
6 | SUID and SGID misconfiguration

If an executable has been configured with the special permission bit using the setuid or setgid and is owned by root, then another User could abuse this to gain access to the root user. Many of the executables with this permission bit set can easily be exploited to gain access to the root user.

7 | Abusing sudo-rights

If a User is part of the sudoers file and is permitted to execute certain binaries or executables using sudo, then in many situations it might be possible to abuse the sudo permissions and gain access to root. Gtfobins is an excellent resource to look for ways to bypass security restrictions on misconfigured systems.

Below is an example using hackthebox platform Sunday Machine. It shows how a User with sudo permissions to execute wget can overwrite a file and then abuse sudo permissions to elevate to the root super-user.



Busing sudo permissions

Other common methods used to escalate privileges on Linux include:

8. World writable scripts invoked by root
9. Bad path configuration
10. Cronjobs
11. Unmounted filesystems

Tips to protect Linux from privilege escalation

Now that we've covered the common attack strategies cybercriminals use to elevate privileges on Linux systems, let's dive into the security strategies you can use to protect your organization.

Final six top tips to make privilege escalation difficult for attackers

First, let's address the word "prevent." It's dangerous to assume anyone can definitively prevent privilege escalation attacks. In fact, assuming that you can prevent can lead to a false sense of security. Rather, we can reduce risks, we can make it more difficult for attackers to be successful, and we can increase visibility to stop the progression of exploits.

Therefore, it's important to harden your systems as much as possible, forcing attackers to use more noisy techniques that raise alarms. The earlier you get the warning signs, the better chance you can prevent a catastrophe.

Here are some best practices to make privilege escalation exploits harder for attackers.

1 | Move passwords into the background using an enterprise password manager or Privileged Access Management (PAM) solution

Help people avoid making poor security decisions by helping them move passwords into secure vaults so they aren't easily found by attackers during the enumeration phase. A strong password manager or PAM solution will help prevent passwords being left on the desktop in clear text, hidden in configuration files, or stored in insecure browsers.

With a PAM solution, you can ensure all services have a provisioned account with the correct security controls, including complex passwords, rotated frequently. Throw universal keys and standing privileges away and move to on-demand, just-in-time privileges.

Learn more in the whitepaper, Invisible PAM: Balance Productivity and Security Seamlessly.

2 | Remove local administrator privileges and practice the Principle of Least Privilege

You've likely heard about Zero Trust. And you've likely wondered how you put those strategies into practice. The best place to start is making the shift to non-persistent, least standing privileges, which means a User or account should have just enough privileges to perform the task or action they are assigned to do. This strategy reduces the possibility of exposing several of the privilege escalation paths described in this paper, such as insecure services, registry, and directory paths.

3 | Implement MFA everywhere

Passwords are one of the most common targets so let's not rely on them as the only security control to protect critical systems, applications, or privileges.

Rather, make sure to add security controls such as Multi-Factor Authentication, access workflows, and approvals, to ensure that only authorized Users can use privileges even if an attacker compromises the password. MFA should be required not only at system log-in, but also at the point of horizontal and vertical privilege elevation.

4 | Implement application control

Limit the ability to run applications and scripts that can be used to enumerate or exploit privileges. Approval lists control which applications and scripts can execute. Deny lists control which known malicious applications and scripts are blocked or require additional auditing.

5 | Patch and update systems and applications

It's not 100% protection, but it certainly makes the attacker's goal of privilege attack and escalation more difficult.

6 | Audit and log all privilege access usage

Make sure to monitor privilege usage for abuse and suspicious activity. Set alerts to ensure you have visibility and can investigate when something suspicious occurs. Should a privilege escalation exploit occur, you'll be able to conduct incident response and identify the root cause of the attack to prevent it from happening again.

Looking for more advice on defending against privilege escalation attacks?

For Windows, privilege escalation defense does get more complicated when we start looking into Active Directory, cloud environments, and Single-Sign-On. Check out the whitepaper, Active Directory Security and Hardening: An Ethical Hacker's Guide to Reducing AD Risks to learn more.

For Linux, you can learn more via Tib3rius, an excellent Linux privilege escalation course on Udemy. Also, check out hackthebox, the hacking gamification and education platform, and a vulnerable Debian virtual machine from the lpeworkshop by sagishahar.

Delinea

Defining the boundaries of access

Delinea is a leading provider of Privileged Access Management (PAM) solutions for the modern, hybrid enterprise. The Delinea Platform seamlessly extends PAM by providing authorization for all identities, controlling access to an organization's most critical hybrid cloud infrastructure and sensitive data to help reduce risk, ensure compliance, and simplify security. Delinea removes complexity and defines the boundaries of access for thousands of customers worldwide. Our customers range from small businesses to the world's largest financial institutions, intelligence agencies, and critical infrastructure companies. **delinea.com**

© Delinea