Privilege Escalation

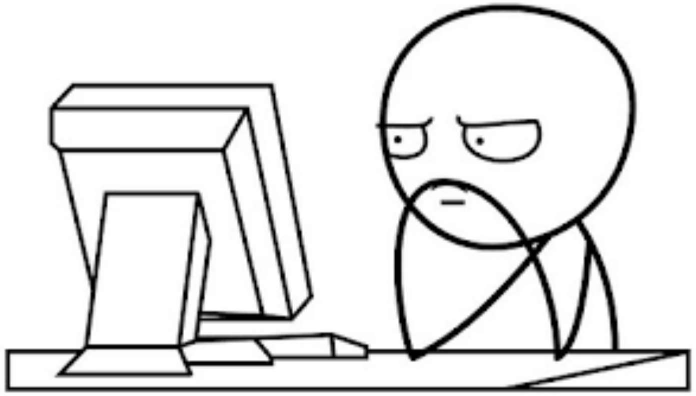
Manual privilege escalation techniques on Unix and Windows

Michal Knapkiewicz, May 2016

whoami

- Senior Consultant at Advanced Security Centre (ASC), EY

- Penetration tester, specialising in:
  - Web Application Testing
  - Thick Application Testing
  - Network & Infrastructure Testing
  - Exploit Development
  - Social Engineering



- Contact:
  - Michal.Knapkiewicz@au.ey.com
  - https://blog.knapsy.com
  - https://twitter.com/TheKnapsy (@TheKnapsy)

Disclaimer

- Always make sure you have a permission from the system owner before engaging in any "hacking" activities
- Techniques discussed here, if used without permission, are considered malicious and illegal
- DO NOT try them on University network or machines
  - There are number of vulnerable VMs available to download from https://www.vulnhub.com to practice your skills on instead

Syntax and Colour Coding

- Anything in **green** relates to Unix, e.g.

```
# whoami
root
```

- Anything in **blue** relates to Windows, e.g.

```
C:\> whoami
SYSTEM
```

Agenda

- Privilege Escalation Overview

- Privilege Escalation on Linux & Windows
  - Enumeration
  - Quick wins
  - Exploiting weak configuration
  - Exploiting vulnerable services
  - Kernel exploitation

- Post exploitation

Privilege Escalation Overview

- You have obtained remote access to a host through various methods:
  - Exploiting web application vulnerability (RCE, SQLi)
  - Exploiting vulnerable services exposed on the server
  - Password guessing attack (dictionary based, default credentials)
  - Social Engineering (phishing email)

- But the user you have compromised is a low privileged account (i.e. not **Administrator** or **root**)

- What's next…?

Privilege Escalation Overview (cont.)

- Why do we want a privileged account anyway?
  - To have <u>unrestricted access</u> to everything on the system
  - To access <u>parts of the file system</u> that we normally cannot, e.g.
    - `C:\Users\Administrator\`
    - `/root/` or `/home/root/`

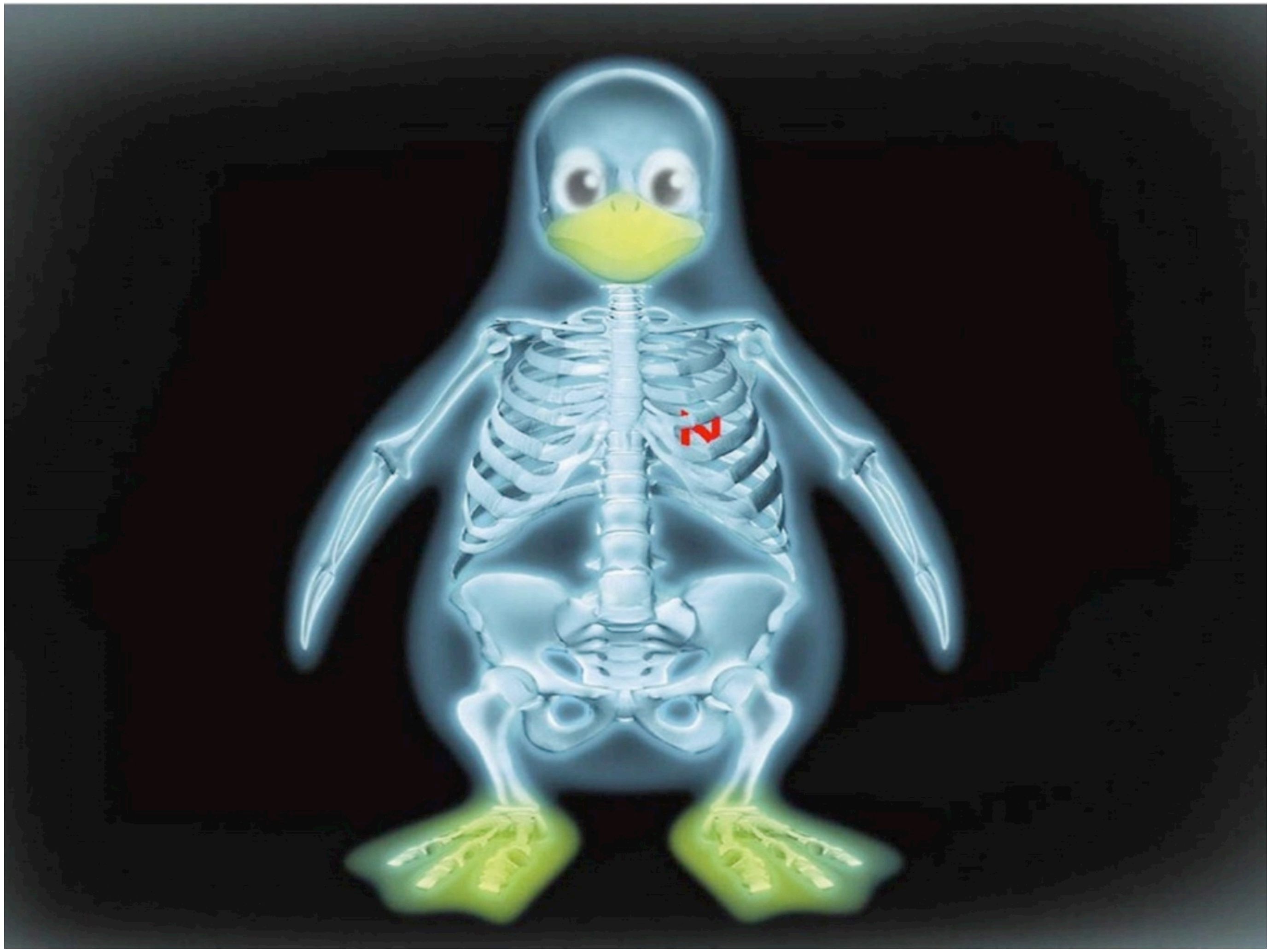
Privilege Escalation Overview (cont.)

- To access restricted <u>functionality</u>, e.g.:
  - Dumping process memory (to find clear-text passwords)
  - Network capture (to sniff traffic)

- To obtain foothold on the internal network and use the system as an <u>entry</u> and <u>pivot point</u>

- To <u>reuse credentials</u> (or hashes) on other systems on the network

Privilege Escalation - Linux



Enumeration

- The basis for a successful privilege escalation is <u>understanding the environment</u> we are up against and taking full advantage of what's on offer

- Enumeration is <u>the key</u> - take your time!
  - Finding a vulnerability takes the most time and effort; exploitation is easy!

- Often, you may be able to find a way in without firing off a single exploit
  - Leverage <u>weak configurations</u> and <u>bad habits</u> of system administrators!

Enumeration (cont.)

- Who are we?

```
whoami
id
```

- What's the operating system and kernel (32 / 64 bit?)

```
uname -a
cat /etc/issue
```

- What can we learn from environment variables?

```
env
cat /etc/profile /etc/bashrc ~/.bashrc
~/.bash_profile ~/.bash_logout
```

Enumeration (cont.)

- What services are running and with what privilege?

```
ps -ef
```

- Are there any scheduled jobs?

```
crontab -l
cat /etc/crontab
```

- What's the IP address and network interfaces?

```
ifconfig -a
cat /etc/network/interfaces
```

Enumeration (cont.)

- Check network configuration settings

```
cat /etc/networks
cat /etc/hosts
cat /etc/resolv.conf
iptables -L
```

- Check open ports

```
netstat -antup
```

- What other users are on the system?

```
cat /etc/passwd
last
```

Enumeration (cont.)

- Check for sensitive files and directories (if you can access them as current user)

```
cat /etc/shadow
ls -al /var/mail/
ls -alR /root/
ls -alR /home/
```

- What was the user doing?

```
cat ~/.bash_history
```

- Can you find private keys?

```
ls -al ~/.ssh/
```

Quick Wins - SUDO

- Check if current user can run any commands with **sudo**, what would then execute them with root permissions

```
sudo -l
```

- What to look for?

```
User <username> may run the following commands:
    (ALL : ALL) NOPASSWD: ALL
    (ALL) NOPASSWD: /opt/scripts/*
    (ALL) NOPASSWD: /opt/admin/custom_binary
```

Quick Wins – Command History

- Some commands or poorly written scripts require users to enter their credentials as a command line parameters

- Everything that user types in is saved in the command history

- Check command history files for any sensitive data (credentials, configuration, interesting directories)

```
cat ~/.bash_history
cat ~/.ksh_history
```

Quick Wins – SSH Private Keys

- System administrators sometimes overlook the importance of keeping private keys… private, and leave them around on the servers

- Check if the current user has any SSH private keys saved on the system

```
ls -al ~/.ssh/id_rsa ~/.ssh/id_dsa
```

- Users often reuse the same key across number of different accounts, including **root**, and number of various servers

Quick Wins – Hardcoded Passwords

- You can often find hardcoded passwords to various services or user accounts in <u>scripts</u> or <u>log files</u>

- Search the <u>entire file system</u> for "password" string

```
grep –R –i "password" /
```

- See if you can access any <u>sensitive configuration files</u> or <u>logs</u>

```
cat /etc/syslog.conf
cat /etc/apache2/apache2.conf
cat /var/log/syslog
cat /var/log/apache2/access.log
```

Weak Configuration – SUID/GUID binaries

- Binaries with SUID/GUID permission bit ("sticky" bit) set will execute with permissions of the owner of this file

- Consider below example, **passwd** binary is used to change current user password on Unix system. It has SUID bit set and therefore will execute with permissions of the owner (**root**)

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 53112 Nov 19
2014 /usr/bin/passwd
```

Weak Configuration – SUID/GUID binaries (cont.)

- Find all binaries with SUID/GUID bit set. Some custom made programs or scripts may allow you to do things that may be used to escalate privileges

```
find / -type f \( -perm -4000 -o -perm -2000
\) -exec ls -l {} \;
```

- Often, such programs may contain vulnerabilities that we can easily exploit (command injection, hardcoded relative paths, buffer overflow) and obtain privileges of the file owner
  - Some reverse engineering and exploit development skills may be required

Weak Configuration – World Read/ Write directories

- Default `umask` used for file creation of either <u>0022</u> or <u>0002</u>. As a result, files that may contain sensitive information will be <u>readable by anyone</u> that has access to the system

- Files may also be <u>modified by anyone</u> on the system if they are <u>world-writable</u>

- This can lead to an attacker <u>accessing sensitive files</u> or <u>modifying files or scripts</u> used by Administrators to execute commands and, potentially, escalate privileges

Weak Configuration – World Read/ Write directories (cont.)

- To find all world writeable directories:

```
find / -perm -0002 -type d -print
```

- To find all world writeable files:

```
find / -perm -0002 -type f -print
```

- Find both files and directories (exclude symbolic links which produce false positives):

```
find / -perm -2 ! -type l -ls
```

Vulnerable Services

- Services are often configured with the minimum configuration changes needed to get them up and running
  - And then they are often left like this for a long time (no patches or configuration changes applied)

- It is not uncommon to find outdated, vulnerable services running on the server

Vulnerable Services

- Often, such services would be utilising <u>default credentials</u> and default, often <u>insecure configuration</u>
  - e.g. SQL server running with **root** permissions and utilising default admin credentials

- Some of the services may have additional plugins configured, often those <u>plugins are vulnerable</u>

Vulnerable Services (cont.)

- It all comes down to enumeration… once again!

- Find all processes running on the system:

```
ps -ef
ps -ef | grep root
```

- Find installed applications and note their version

```
dpkg -l
rpm -qa
```

- Search for known vulnerabilities in discovered processes and services (https://exploit-db.com, Google)

Kernel Exploits

- If everything else fails, reach out for kernel exploits
- Number of various exploits for different kernel versions exist
- Note: kernel exploits may cause target system to behave in unexpected ways or cause it to crash. Use with care!
- Find out what kernel version is the system running:

```
uname -a
```

Kernel Exploits (cont.)

- Find a relevant one for the version of target kernel:
  - Full Nelson (Linux Kernel <= 2.6.37)
  - Half Nelson (Linux Kernel <= 2.6.32.2)
  - CVE-2014-4014 (Linux Kernel <= 3.13)
  - CVE-2013-2094 (Linux Kernel < 3.8.9 - x86_64)
  - And more…

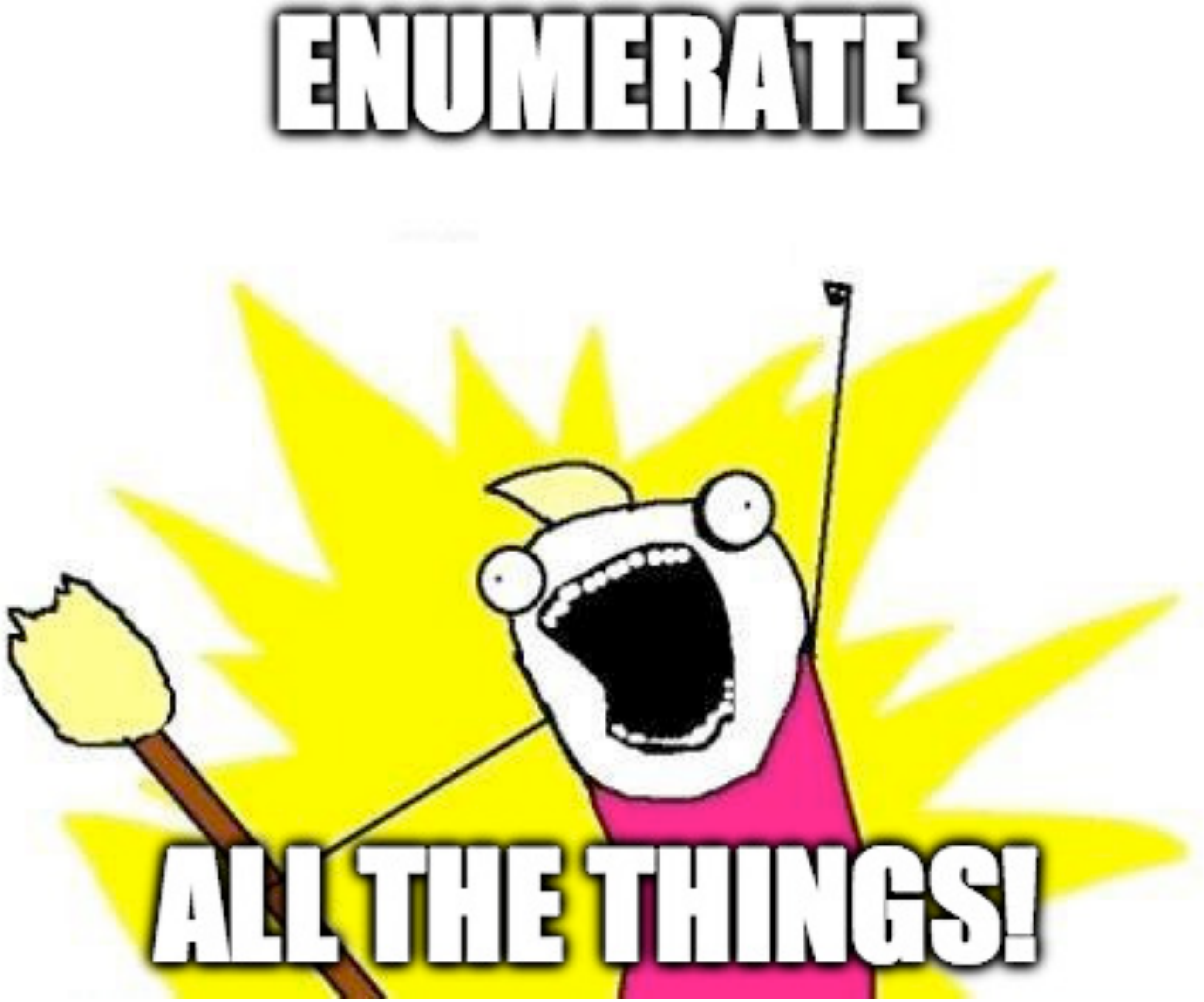
- Remember! Not all exploits will work “as is”, they often require slight modifications for your needs (e.g. change payloads, paths etc.). Also, proof-read the code first to make sure it does what it claims to be doing…

Kernel Exploits (cont.)

- Find a relevant one for the version of target kernel:
  - Full Nelson (Linux Kernel <= 2.6.37)
  - Half Nelson (Linux Kernel <= 2.6.32.2)
  - CVE-2014-4014 (Linux Kernel <= 3.13)
  - CVE-2013-2094 (Linux Kernel < 3.8.9 - x86_64)
  - And more…

- <u>Remember!</u> Not all exploits will work "as often require slight modifications for you change payloads, paths etc.). Also, proof- first to make sure it does what it claims t

Privilege Escalation - Windows


ENUMERATE
ALL THE THINGS!

Enumeration

- Find out what OS are we connected to:
  ```
  systeminfo
  ```

- Where are we and who are we:
  ```
  hostname
  echo %username%
  ```

- What are other users on the system:
  ```
  net users
  ```

- And more details about each user (permissions etc.)
  ```
  net user %username%
  ```

Enumeration (cont.)

- Find out about network interfaces & IP addresses

```
ipconfig /all
```

- Routing table information

```
route print
```

- Open ports and active network connections

```
netstat -ano
```

- Firewall state and configuration

```
netsh firewall show state
netsh firewall show config
```

Enumeration (cont.)

- Scheduled tasks

```
schtasks /query /fo LIST /v
```

- Currently running processes

```
tasklist /SVC
```

- Started Windows services

```
net start
```

- Installed hardware drivers

```
DRIVERQUERY
```

Quick Wins – Mass Rollouts

- On an environment with large number of machines, a system administrator would want to find a way to automate mass deployment
- There are configuration files left laying around that contain a lot of interesting data, often including Administrator password (either in clear-text or base64 encoded), e.g.

```
c:\sysprep.inf
c:\sysprep\sysprep.xml
%WINDIR%\Panther\Unattend\Unattended.xml
%WINDIR%\Panther\Unattended.xml
```

Quick Wins – Group Policy Preference

- Group Policy Preferences is a collection of Group Policy client-side extensions that deliver preference settings to domain-joined computers running Microsoft Windows desktop and server operating systems

- When the host you compromise is connected to a domain, it is well worth looking for the **`Groups.xml`** file from **`%LOGONSERVER%\SYSVOL`** folder, as it often contains an encrypted local administrator password in **`cpassword`** parameter
  - Note: Any authenticated domain user will have read access to this file!

Quick Wins – Group Policy Preference (cont.)

- The password in the `Groups.xml` file is encrypted with AES, however, the static key is published on the Microsoft website allowing for easy decryption of the stored value
  - Use the following script to decrypt the password: https://raw.githubusercontent.com/leonteale/pentestpackage/master/gppdecrypt.rb

- In addition to `Groups.xml` several other policy preference files can have the optional `cpassword` attribute set

```
Services\Services.xml
ScheduledTasks\ScheduledTasks.xml
Printers\Printers.xml
Drives\Drives.xml
DataSources\DataSources.xml
```

Quick Wins – AlwaysInstallElevated

- Check for registry setting **`AlwaysInstallElevated`** - if this setting is enabled it allows users of any privilege level to install **`*.msi`** files as **`NT AUTHORITY\SYSTEM`**.

- This will only work if both registry keys contain **`AlwaysInstallElevated`** with DWORD values of 1

- To find out, run the following commands:

```
reg query HKLM\SOFTWARE\Policies\Microsoft
\Windows\Installer\AlwaysInstallElevated
reg query HKCU\SOFTWARE\Policies\Microsoft
\Windows\Installer\AlwaysInstallElevated
```

Quick Wins – Hardcoded Passwords

- Search for files containing certain keywords

```
dir /s *pass* == *cred* == *vnc* == *.config*
```

- Search certain file type for keywords

```
findstr /si password *.xml *.ini *.txt
```

- Search registry for keywords

```
reg query HKLM /f password /t REG_SZ /s
reg query HKCU /f password /t REG_SZ /s
```

Exploiting Vulnerable Services

- Check access rights of Windows services. Sometimes (particularly in Windows XP SP0 and SP1), there are services which configuration can be modified by any user
- This way, a low privilege user can change configuration of the service to run a different binary during the service startup (e.g. spawning a shell for an attacker), restart the service and simply obtain a **`SYSTEM`** shell
- One particular service like this (on Windows XP SP0 and SP1) is **`upnphost`**

Exploiting Vulnerable Services (cont.)

- For checking access rights use **`accesschk.exe`** tool from Microsoft's **`Sysinternals Suite`**
  - https://technet.microsoft.com/en-us/sysinternals/bb842062.aspx

- The following steps are taken to exploit the **`upnphost`** service on Windows XP SP0 or SP1:
  1) Use **`accesschk.exe`** to find permissions to the **`upnphost`** service
  2) List upnphost configuration (informational only)
  3) Change binary loaded by the service to a reverse shell
  4) Change owner of the service to **`SYSTEM`** (what privilege the service is running as)
  5) List upnphost configuration to verify changes (information only)
  6) Restart the service and get the shell

Privilege Escalation Exploits

- If everything else fails, it’s time to reach for a heavy duty tools and look into kernel exploitation

- Number of exploits exist for numerous versions of Windows - there is pretty much an exploit for every version and every service pack.

- To find out what system are we dealing with:

```
systeminfo
```

- Remember! Proof-read the exploit code to make sure it does what it claims to be doing…

Privilege Escalation Exploits (cont.)

- Enumerate patches installed on the system

  ```
  wmic qfe get
  Caption,Description,HotFixID,InstalledOn
  ```

  Note: `wmi` comes installed by default with Windows 2000 onwards

- Look for privilege escalation exploits and look up their respective KB patch numbers, e.g.
  - KiTrap0D (KB979682)
  - MS11-011 (KB2393802)
  - MS10-059 (KB982799)
  - MS10-021 (KB979683)
  - MS11-080 (KB2592799)

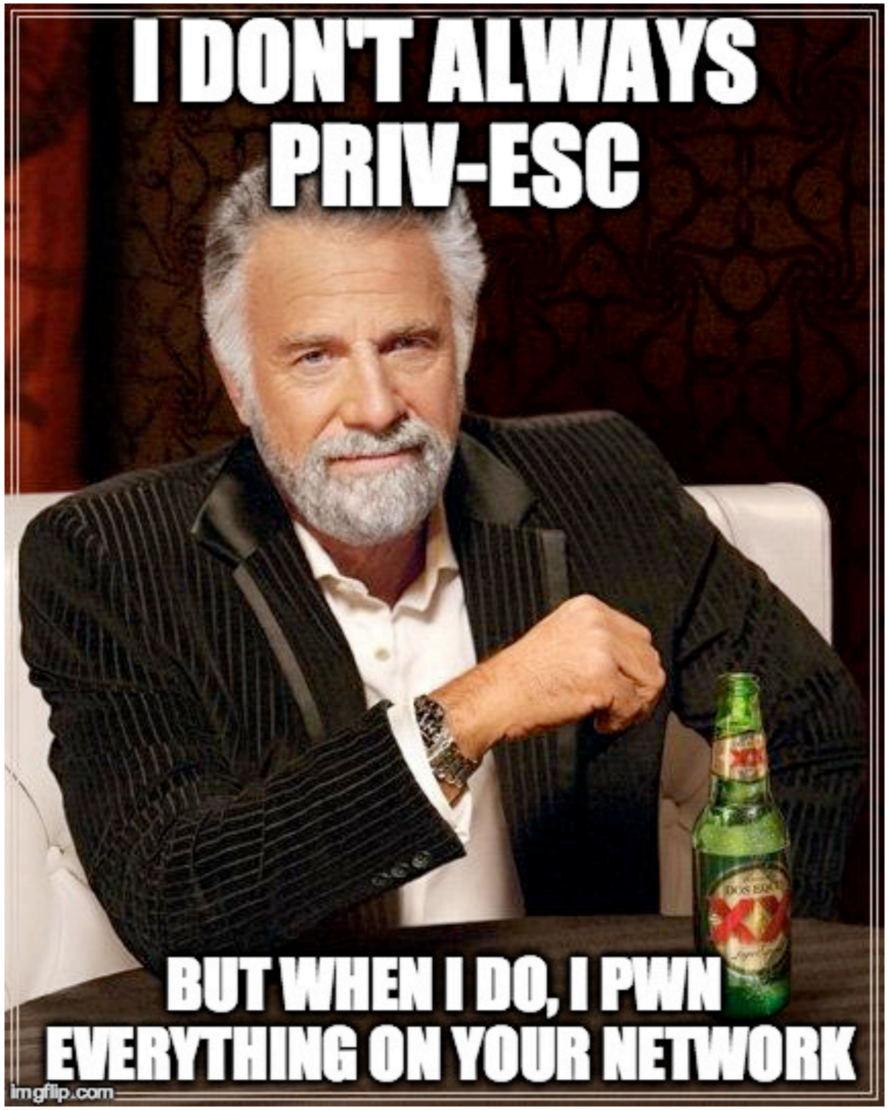
Privilege Escalation Exploits (cont.)

- Then cross-check the patches with what’s installed on the system:

```
wmic qfe get Caption,Description,HotFixID,InstalledOn | findstr /C:"KB..." /C:"KB..."
```

- Whatever doesn’t show up in the above search indicates that the patch is not installed and hence, the system is vulnerable to a privilege escalation exploit!

Post Exploitation



Post Exploitation

- So you got Administrator access, what now?

- Dump all passwords from memory
  - Using `procdump.exe` to dump `lsass.exe` process' memory
  - Extract credentials from memory image using `mimikatz` (do it offline!)

- Go through all sensitive files that you can now access, particularly ones with password hashes

```
/etc/shadow
C:\Windows\System32\config\SAM
```

Post Exploitation (cont.)

- Crack password hashes (if we didn't get clear-text credentials)
  - using **`hashcat`**, **`john`** or online rainbow tables (https://crackstation.net)

- If you can't crack the hash, use pass-the-hash technique to log-in to different hosts using only password hashes

Post Exploitation (cont.)

- Go through all logs and configs looking for sensitive information, such as:
  - Passwords
  - Private SSH keys
  - Database connection strings

- Database backups are generally left unencrypted and contain a lot of sensitive information

Post Exploitation (cont.)

- Pivot through the environment
  - Try to access other hosts on the internal network
  - Set up port-forwarding to use compromised host as a “pivot point” to carry out further attacks

- If needed, set up persistence to ensure you keep privileged access to the compromised host at all times

Summary

- Privilege escalation relies heavily on enumeration
- Various tricks exist to escalate privileges in both Linux and Windows environments
- Sometimes it may not be possible to escalate to the highest privilege level straight away. You may need to escalate number of times, overtaking different accounts, before reaching **root** or **Administrator**
- In a real-world scenarios, consider kernel exploits as a last resort – they may sometimes crash target systems

References

- https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/
- https://labs.securitycompass.com/web-applications/5-common-linux-misconfigurations/
- http://www.0daysecurity.com/penetration-testing/enumeration.html
- http://www.fuzzysecurity.com/tutorials/16.html
- http://www.fuzzysecurity.com/tutorials/18.html
- http://www.greyhathacker.net/?p=185
- http://www.greyhathacker.net/?p=738
- http://www.r00tsec.com/2012/11/howto-manual-pentest-windows-cheatsheet.html

Questions

?