Windows Privilege Escalation Part 1: Local Administrator Privileges

The process of stealing another Windows user's identity may seem like black magic to some people, but in reality any user who understands how Windows works can pull it off. This is the first of two blog entries giving an overview of privilege escalation techniques that prove that fact. Part 1 (this entry) discusses obtaining local SYSTEM and administrative privileges from an unprivileged user account, and Part 2 will focus on obtaining domain administrative privileges from local administrator or domain user accounts.

Escalation Techniques

Weak configurations and missing patches often lead to local user and service account access. Sometimes these accounts can be used to access sensitive information directly, but usually access to the affected systems and connected networks doesn't stop there. Using the 10 escalation vectors listed below, penetration testers can often gain unauthorized access to databases, network devices, and other systems on the network.

1. ***Clear Text Passwords Stored in Files*** Never underestimate the *efficiency* of IT administrators. In their quest to play more Halo, most administrators have automated a large number of their processes and made their systems as homogeneous as possible. Although this sounds like a dream come true, the red team isn't the only one in the crosshairs. The scripts used to automate processes and connect to databases often include clear text user names and passwords that can be used to gain unauthorized access to systems and applications. Also, because most of their systems are built from the same image, finding a security issue on one system is like finding it on all of them. So, while doing penetration tests, take the time to pick the low hanging fruit. Checking local files for sensitive credentials can be one of the easiest ways to escalate privileges. Personally, I recommend using Spider 2008 for the task. It is a great open-source tool created by the folks at Cornell University that will accept custom regex and can be used to search for sensitive information on systems, shares, and websites.
2. ***Clear Text Passwords Stored in the Registry:*** The registry is a hidden treasure chest of passwords and network paths. Application developers and system administrators leverage it for all kinds of interesting things, but they don't always use encryption when they should. So spend some time browsing for passwords and network paths that could lead to secret stashes of sensitive data. I encourage people to write their own scripts or use an automated tool to search the registry for sensitive information, but feel free to use the find function in regedit if your fingers need a little exercise.
3. ***Write Access to the System32 Directory:*** By default, unprivileged users do not have write access to the system32 directory in Windows operating systems. However, many older applications and misguided system administrators change the permissions to avoid access errors and ensure smooth operations. Unfortunately, the result of their good intentions also allows penetration testers to avoid access errors and ensure smooth privilege escalation to local SYSTEM.Specifically, accessibility programs such system32sethc.exe (Sticky Keys), and system32utilman.exe (Windows Utility Manager) can be used to gain SYSTEM level access by replacing their executables with cmd.exe. This works because Windows doesn't perform any file

integrity checks on those files prior to login. As a result, users can create a local administrator account and disable anti-virus, among many other creative possibilities. This is one of my favorites. I hope you enjoy it as much as I have.

4. ***Write Access to the All Users Startup Folder*** Every user on a Windows operating system has his or her own Windows startup folder that is used to start programs just for them when they log on. As luck would have it, there is also a Windows startup folder that contains programs that run for (you guessed it) ALL users when they log on. If unprivileged users have the ability to write to that directory, they can escalate their privileges on the local system and potentially the network by placing an evil executable or script in the directory and tricking a trusting user into logging into their machine. If your penetration test allows for a little bit of social engineering, I recommend calling up the help desk and asking them to sign into your system in order to help with a random computer issue. The help desk team usually has the privileges to add, edit, and remove local and domain users.
5. ***Insecurely Registered Executables:*** When a program is installed as a Windows service or logon application, it is required to register with Windows by supplying a path to the program's executable. If it is not registered securely, penetration testers may be able to escalate their privileges on the system by running commands under the context of the service user. Specifically, if a registered path contains spaces and is not enclosed in quotes, penetration testers may be able to execute their own program based on the registered executable path.

   Insecurely registered executables can be an easy, low-impact way to gain SYSTEM level access on a system. Although insecurely registered executables can be identified manually, I recommend using some of the automated tools available for free on the internet.
6. ***Windows Services Running as SYSTEM*** Lots of services run as SYSTEM, but not all of them protect their files and registry keys with strong access controls. In some cases SYSTEM-level access can be obtained by overwriting files and registry keys related to SYSTEM services. For example, overwrite an executable used by a SYSTEM service with cmd.exe. If the overwrite is successful, then the next time the service calls the executable, a SYSTEM console will open. Modifying configuration files and performing DLL injection have also proven to be effective techniques. No super-secret tools are need for this vector of attack; Windows Explorer or a command console should do.
7. ***Weak Application Configurations:*** I've found weak application configurations during every penetration test I've ever done, and in many cases they can be leveraged to gain SYSTEM-level access. Even in an age of application hardening guides and industry compliance requirements, I regularly find applications like IIS and MSSQL running as SYSTEM instead of a least privilege service account.

   I recommend doing some research on the applications installed on your target systems to determine how best to leverage their configurations.
8. ***Windows At Command:*** I know this is an oldie, but it's still worth mentioning, because oddly enough there are some environments out there running unpatched versions of WinNT and Windows 2000. So, for those who have not used this technique before, it may still be useful. The "AT" command is a tool that is used to schedule tasks in Windows. By default, in earlier versions of the Windows operating system the "AT" command was run as SYSTEM. As a result, users can gain access to a console with SYSTEM access when they schedule the cmd.exe as a task.
9. ***Install a User-Defined Service:*** In some cases Windows users may have excessive rights that allow them to create services on the system using the Instrsrv.exe and Srvany.exe tools that come

with the Windows NT resource kit. Instsrv.exe is used to install a program as a service, and Srvany.exe is used to run the program as a service. By default, the services installed with instsrv.exe should be configured to automatically start at boot time. However, you can also use the sc.exe command to ensure that it is configured how you want it. Either way, a restart may be required depending on your existing privileges. I've personally seen users in the "Power Users" group with the ability to install their own services, but it may be a coincidence. I recommend creating a Metasploit binary that adds a local administrator to the system, and using it to create the service. It's easy, and the new administrator account can be used to sign in via remote desktop. However, if that's not your cup of tea, feel free to use your favorite payloads.

10. ***Local and Remote Exploits:*** Managing and distributing patches for third-party software seem to be among the greatest challenges facing IT today. This is bad news for IT, but good news for penetration testers, because it leaves vectors of attack open. Exploiting local and remote vulnerabilities can provide SYSTEM-level access with very little effort, and—with a little luck—domain administrator access. However, they can have some negative impacts on sensitive systems, like crashing them. If that is not a problem in the environment you're working in, then more power to you. However, if you're working with people who don't like their systems to crash unexpectedly, then take the time to understand the exploits and avoid using the ones that have a history of negative effects. There are a number of great open-source and commercial toolsets available for exploiting known software vulnerabilities. However, the frontrunners seem to be Metasploit, Canvas, and Core Impact. They all have a variety of options and exploits, but I recommend using the one that fits your immediate needs and budget.

Conclusion

Usually, it doesn't require super hack tools or a degree in wizardry to perform local privilege escalation as an unprivileged user. What it does require is enough understanding of how Windows works to use it against itself. Until next time, keep shining light into the dark places. – SPS

Tool and Command References

- http://support.microsoft.com/kb/137890
- http://support.microsoft.com/kb/137890
- http://support.microsoft.com/kb/313289
- http://www2.cit.cornell.edu/security/tools/
- http://www.metasploit.org/
- http://www.sqlsecurity.com/Tools/FreeTools/tabid/65/Default.aspx
- http://www.foundstone.com/us/resources/proddesc/diredetectinginsecurelyregisteredexecutables.htm