# PowerShell

It’s time to own….

David Kennedy (ReL1K)

Josh Kelley (Winfang)

http://www.secmaniac.com

Twitter: dave_rel1k

# About Josh

- Security Analyst with a Fortune 1000 --- Works with Dave

- Heavy experience in penetration testing, exploitation, web application security, vulnerability management, and incident response.

- Primary languages are Perl, Python…and now PowerShell ☺

# About Dave

- Director of Regional Security for a Fortune 1000

- Heavy experience in penetration testing, exploitation, web application security, wireless and physical

- Creator of the Social-Engineer Toolkit, work heavy with Back|Track and the Social-Engineer Framework.

- Heavy military background in Intelligence, deployed twice to Iraq and other middle east countries.

# Brief Intro to PowerShell

- Windows version of a bash shell in nix… Very powerful, flexible, and in some ways (don’t boo) more powerful in nature to nix.

- Installed by default on all Windows 7 and Server 2008 operating systems. Full integration for all new existing Microsoft products, including Exchange and AD integration.

- Full integration into the .NET framework and can be directly called when performing scripting.

# PowerShell Security

- Execution policies are set by default to “restricted”.
- Does not allow any scripts to be run from anywhere, except specific commands.

# Execution Policies

- Restricted – Already talked about this.
- AllSigned – This script only allows signed scripts to be executed. Has to be from a trusted publisher. This is the most restrictive policy.
- RemoteSigned – Remote scripts must be signed by a trusted publisher, things run locally don’t need to be signed.
- Unrestricted – Can run anything both remote and local.

# So why do we need to worry?

- We will be the first ones to admit the usefulness and power of PowerShell in a positive manner. The ability to perform advanced tasks on Microsoft based operating systems is a huge leap forward.

- PowerShell also gives hackers a full fledge programming and scripting language at their disposal on all operating systems by default.

# Release of Metasploit Module 1 – PowerShell Debug

- Traditionally post-exploitation phase, if you didn't have direct access to memory, traditional methods of getting a payload onto a system was through Windows debug (now removed in all newer operating systems), vbscript, TFTP, or FTP.

- These methods are now proving much more difficult with better A/V and HIPS detection (well kinda..) and TFTP and FTP blocked egress.

# DEMO – Metasploit Module

# Small Example of Conversion

- Binary is converted to hexadecimal and placed onto the filesystem.
- Convert script is created to take the hexadecimal and rewrite it back in a byte array as binary.
- Payload is now on the system for execution.

# What about that execution restriction?

- The execution restriction absolutely would have stopped this from executing.

- The payload couldn’t be converted…

# Bypassing Execution Restrictions – CreateCmd being released

- Contents of a file are concatenated, compressed, and converted to base 64 into a single string.

- A boilerplate bootstrap code created for powershell –Command or –encodedCommand args then unpack the code and then perform an Invoke-Expression

- That will execute the script contents in the current shell context with all new functions that are in the script.

# What’s this mean…

- With the most restrictive policy set on PowerShell we can still execute whatever we want…. AllSigned does not stop this attack.
- No need to disable execution restriction policies anymore.
- No registry interaction, no reboots, nothing.

# DEMO – CreateCMD

# What we can do..

- Since we have full access to both PowerShell and the .NET libraries, we can do pretty much anything we want…
- Releasing today both a bind and reverse shell programmed purely in PowerShell.
- And something fun..

# DEMO - PowerDump

# PowerDump

- Meterpreter based module, will dump the SAM database purely through powershell.

- Works on all operating systems, both x86 and 64 bit.

# What does this mean?

- PowerShell is a powerful and useful tool for administrators and security professionals.
- The full fledge programmatic language within default installations does pose significant security risk.
- Anti-Virus and HIPS aren’t picking up these types of attacks, which means it’s a safe passage for exploitation.

# Future Plans

- Process injection and code injection capabilities within PowerShell.

- P.o.C Trojan/Worm purely written in PowerShell.

# Recommendations

- Remove PowerShell if your not currently using it on your systems. This only works for Server 2008, Windows 7 it’s imbedded.

- Set the execution policy to Restricted which is the default, but doesn’t do a whole lot of good.

- That’s really about it…

SecManiac.com

Questions? ☺

Be sure to check out:

http://www.secmaniac.com

Twitter: dave_rel1k