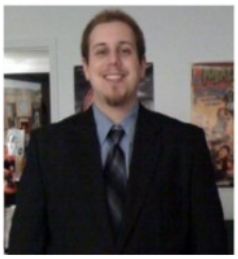

sqrrl
Threat Hunting for Lateral Movement
Corgi Edition
Presented by:
Ryan Nolette – Security Technologist

$whoami

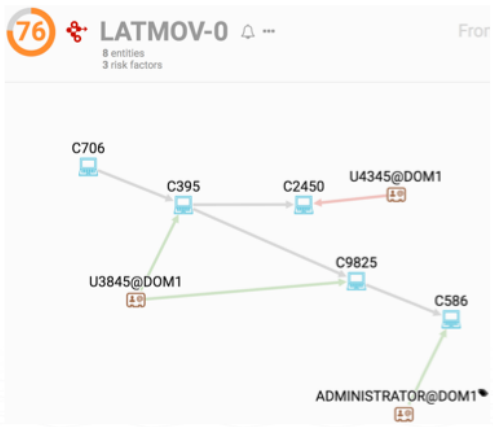


- I am currently the Security Technologist for Sqrrl
- 10+ year veteran of IT, Security Operations, Threat Hunting, Incident Response, Threat Research, and Forensics
- GitHub
  - https://github.com/sonofagl1tch
- Career highlight
  - Time's person of the year 2006

Agenda

- Lateral Movement Overview
  - What is it?
  - Common Techniques
- The Lateral Movement Process
  - Compromise
  - Reconnaissance
  - Credential Theft
  - The Lateral Movement event
- Lateral Movement Threat Hunting
- Q&A

What am I referring to when I say Lateral Movement?

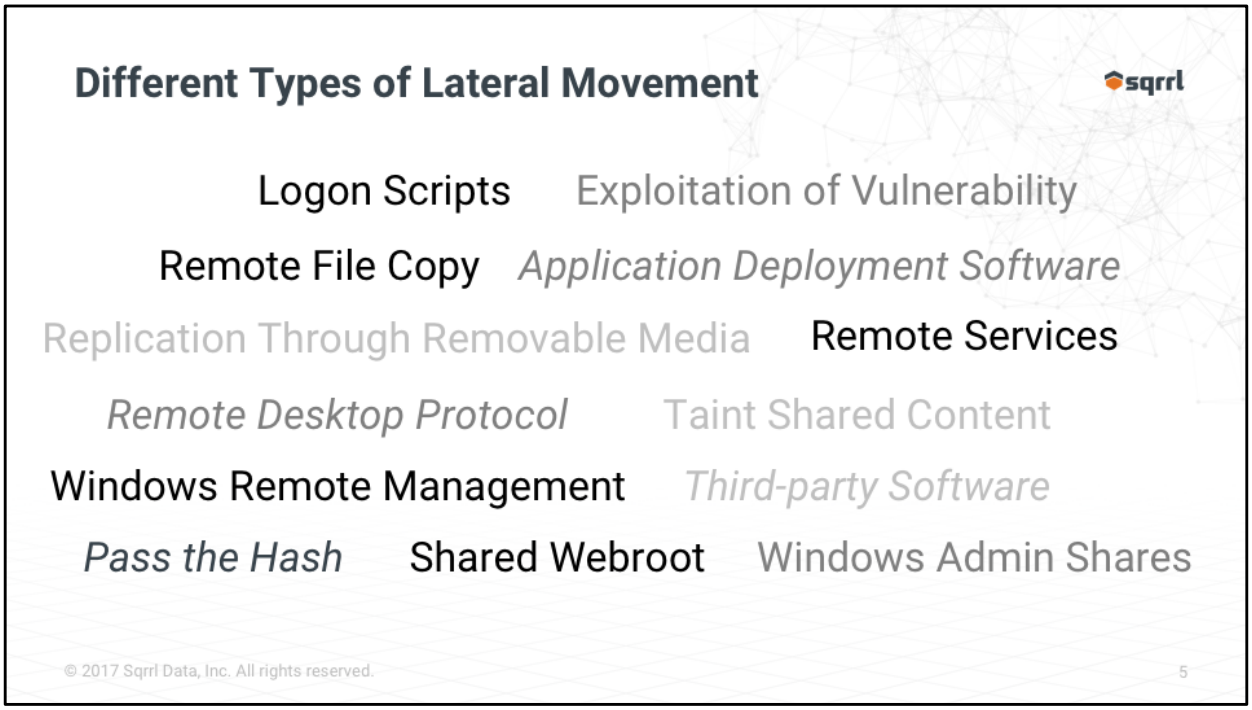


- Techniques that enable attackers to access and control systems within your network
- Leveraged for:
  - Access to specific information or files
  - Remote execution of tools
  - Pivoting to additional systems
  - Access to additional credentials
- Movement across a network from one system to another may be necessary to achieve goals
- Often key to an attacker's capabilities and a piece of a larger set of dependencies

Before we go into detection techniques, let's start by building a foundation of knowledge on Lateral movement. The definition of Lateral movement that I am using is a "term encompassing techniques and tools that enable an attacker to access and/or control systems within your environment."

Something commonly overlooked about lateral movement is that this activity is not the end goal of an attacker, but is instead just a piece of the attack and is often a requirement or dependency of the attacker to achieve their ultimate goal.

The ability to remotely execute scripts or code can be a cornerstone of an attack, but adversaries also attempt to reduce their footprint in environments by abusing legitimate credentials combined with native network and OS functionality to remotely access systems.

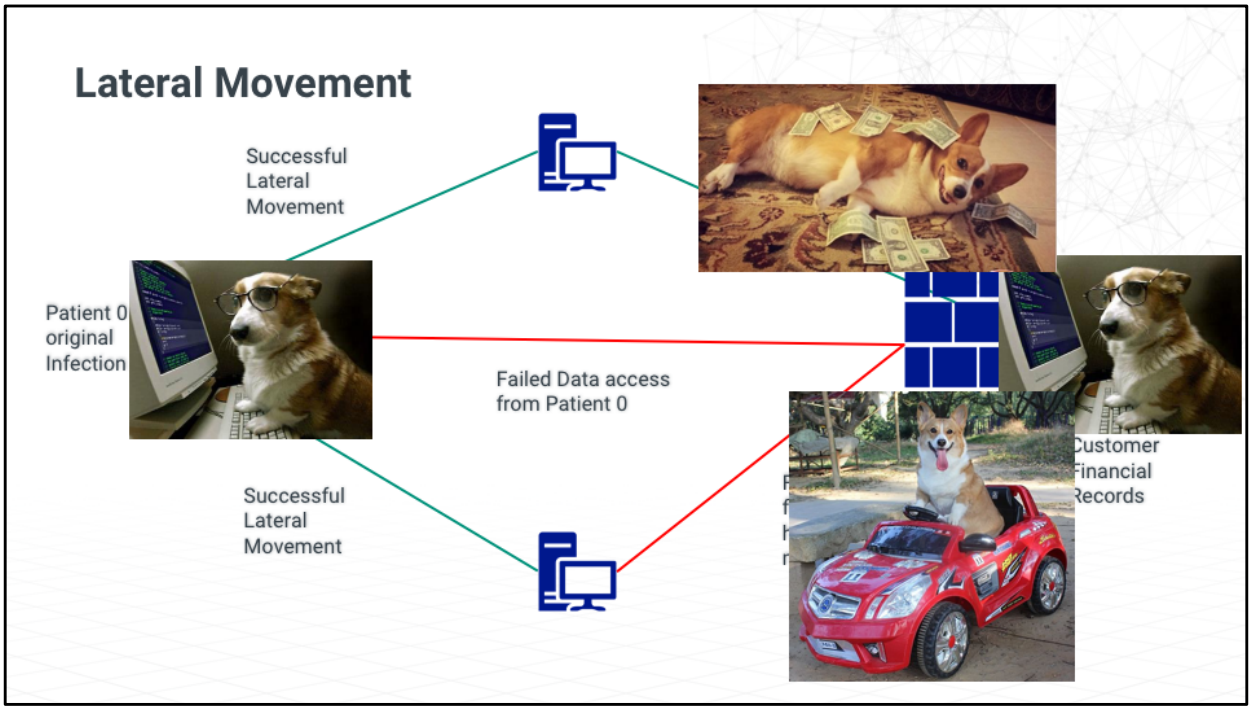


Breaking down the attack further, there are dozens of methods to achieve lateral movement in an environment. Because of this, the attacker has a wide variety of tricks at their disposal. A few of the most common techniques I have seen in the wild are:

Pass the hash (PTH):
A method of authenticating as a user without having access to the user's cleartext password.

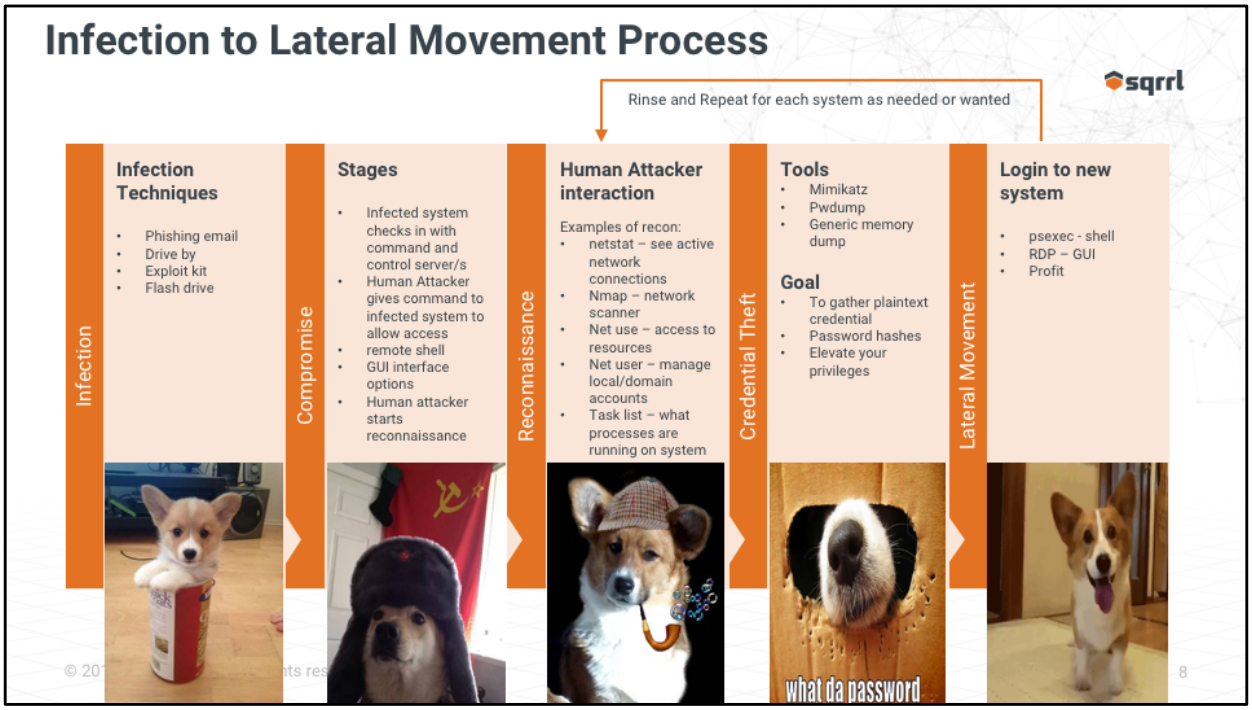
Remote Services:
Where an adversary may use valid credentials to log into a service specifically designed to accept remote connections, such as PsExec, RDP, telnet, SSH, or VNC.

Taint Shared Content:
Content stored on network drives may be tainted by adding applications, scripts, or exploits to otherwise legitimate files.



To demonstrate a typical scenario, we'll look at an example where the attacker will attempt to move from the patient 0 compromised system to gather the company's financial records. Next, the attacker discovers that they cannot directly access the files from the infected host. They then attempt to move laterally to another system they can see on the network. When this system also does not have access to the data the attacker wants, the attacker will attempt to move laterally again and again until finally finding a system with the access they want.


sqrrl
Personal
Attacker Life cycle
© 2017 Sqrrl Data, Inc. All rights reserved.
7



We can take the previous scenario and dissect it into stages.

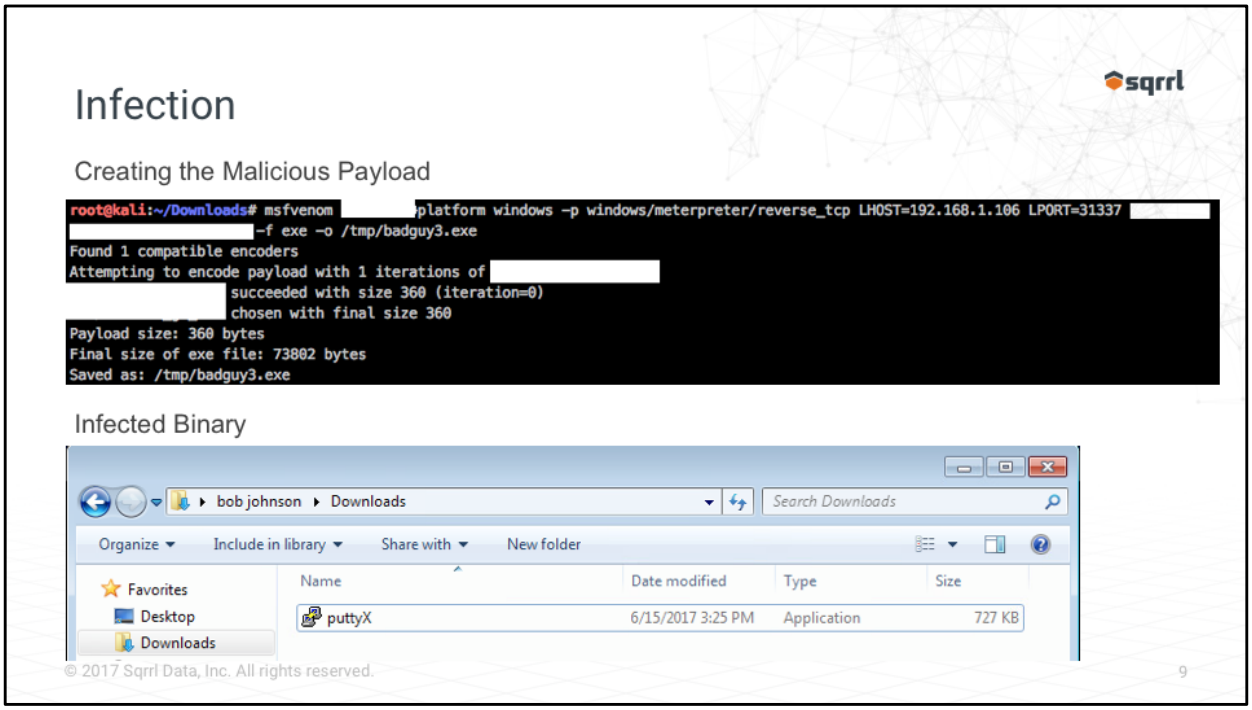
First, we had the initial infection, this occurred by any ordinary means, phishing email, exploit kit, whatever.

Next, we have the compromise stage. This phase differs from the infection stage because we are defining compromise as the attacker has direct access to the system where the infection is defined as automated malware infecting the system. To expand on that, infection is something like getting a trojan on your computer, but compromise does not occur until the malware opens up a connection on the system for an attacker to get direct CLI access to the system.

The third stage is reconnaissance which contains the attacker collecting data about the system they are on and what other systems the attacker can see from the compromised host.

The next stage is credential theft. This stage is vital for the attacker because, without credentials, their ability to move laterally will be extremely limited. Credential theft comes in many forms, and we will dig into that in a bit.

Lastly, comes the actual lateral movement stage where the attacker combines the information they have gathered with the credentials they have gathered and attempt to authenticate to other systems in the environment. These three stages of recon, credential theft, and lateral movement, will be repeated on every system the attacker successfully authenticates.



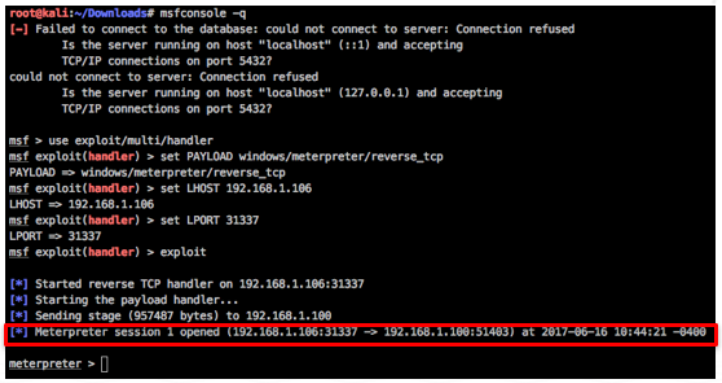
Now that we have a rough idea of the progression of this attack lifecycle let's dig into the stages a bit more.

First I am going to craft the malicious version of a legitimate binary. Here I am using a legitimate copy of putty and injection a malicious reverse_tcp payload.

I have also renamed the file from badguy3.exe to puttyX.exe so that the end user is not aware of the change in the file. Since I am skipping the infection stage in this write up, I will not be going into how I made this binary or how I tricked the user into running it. #PleaseDontSueMe

Compromise – Meterpreter Session

sqrrl



- Communication with the compromised systems and C&C (command and control) servers is established
- Threat actors need to sustain persistent access across the network
- They move laterally within the network and gain higher privileges through the use of different tools

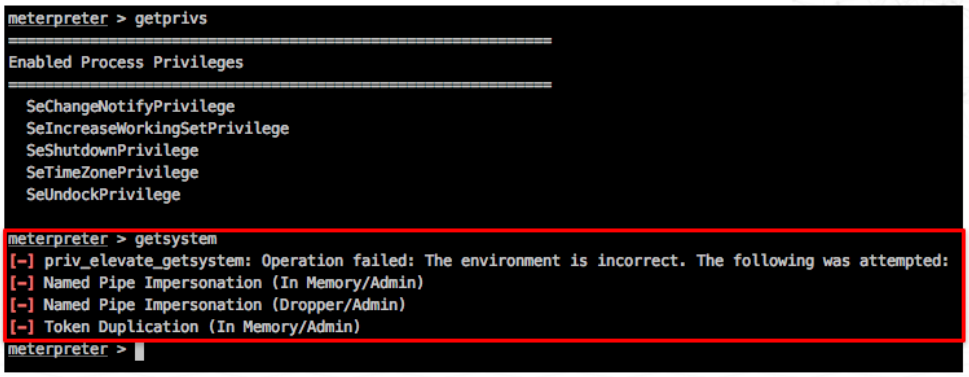
The first stage that matters is the compromise stage. What you see here is a popular penetration testing suite, called Kali, that is being used to connect to the reverse shell generated by the malware on the target host. You can see that with this shell, I as the attacker, have the ability to run admin-level commands to perform recon of the system and network it is on.

Communication with the compromised systems and C&C (command and control) servers is established
Threat actors need to sustain persistent access across the network
They move laterally within the network and gain higher privileges through the use of different tools

After sending the user the malicious binary I start a metasploit console that is going to wait and listen for a connection request from the user. Below you can see the session being started and initiated by the victim system.

You can also see that the last line in the image reads “meterpreter”. This means I now have a direct shell on the victim system from my attacker system. From this shell I can either run plugins, scripts, payloads, or start a local shell session against the victim.

Compromise – discovering privileges



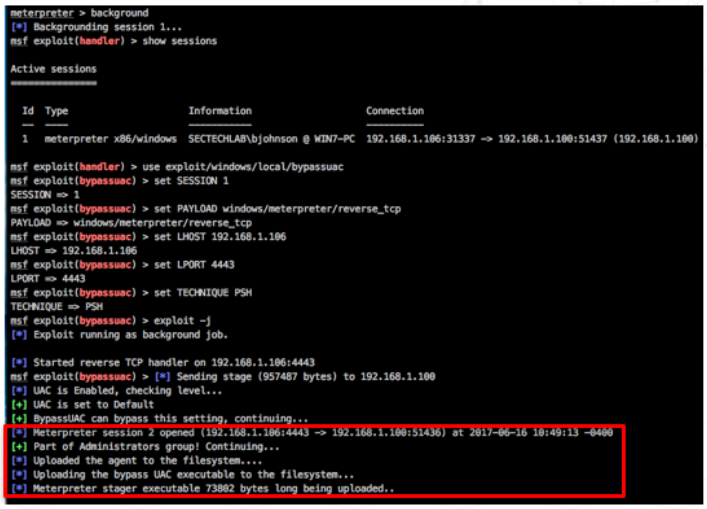
First thing I am going to do with my session is see what privileges I have.

Discovering privileges of user who executed the infected binary and compromised the system

Here I can see that the user I have tricked into running my malware is not a local admin and i have very restricted privileges on the system. Good for SecOps but bad for me as an attacker. Luckily, I am not so easily blocked.

I am going to background my meterpreter session so i can load more attacks to use against the host. In this image we can see that the user I am running as is “bjohnson” in the domain “sectechlab” on the host “win7-pc”. I can also see that I am currently running as x86 windows and my victim IP is 192.168.1.100.

Compromise – elevate privileges



Now that I have the session in the background, I am going to load up a generic UAC bypass exploit (this is patched in current windows versions) and run it against the victim system.

Well what do you know? This enterprise isn't keeping up to date with their patches. Thanks to the successful UAC bypass, I know have a second session started on the victim and this session I can start my evil. How can I do this when the sessions look identical? I can do this because I just bypassed windows user access control. To prove i now have more access, I will rerun the getprivs command on the second session to see if I know have admin privs.

Bam! Money privs.

Now that I have admin privs, i can just simply give myself system level access and then start hiding myself by starting a new x64 process and migrating into that new process. Here I am choosing to use notepad but in reality this will popup and application on the victim system that the user can then quit and I will lose my session. What i really would do is run a process without a Graphical User Interface (gui) but I don't think I should show that here. #PleaseDontSueMe

Now that I have system level access and have hidden myself in a new process. Let’s start the recon!

Reconnaissance – User accounts

sqrrl

- To move laterally within a breached network and maintain persistence, attackers obtain information like network hierarchy, services used in the servers and operating systems
- Check the host naming conventions to easily identify specific assets to target
- Utilize this info to map the network and acquire intelligence about their next move

Recon Local Accounts

Recon Domain Accounts

Now that we have direct CLI access to the compromised host, we need to enumerate the users on that host, the network that it is on, as well as gather generic system information like running processes and such for possible usage for persistence.

Waste not, want not.

The images that you see are of me using the net user command to find local users on the host as well as domain users who have previously logged in. What I am looking for are admin and power user level users that I can reuse elsewhere in the environment. The first thing I am looking for is the local admin accounts on the system. This account is commonly used as an IT backdoor to get into systems that are having AD issues and is commonly the same username and password on all systems in the environment because it is part of the build process. I beg you not to do this in your environments. There is even a GPO setting that randomizes the local admin password on your systems. There is no excuse for this anymore.

First I want see who I am and where I am. I will run some basic sysinfo, whoami, and hostname commands. But what I really want to know is who has been on this system and where this system is in the network.

Reconnaissance – Network

sqrrl

Network settings

```
C:\Windows\system32>ipconfig /all
ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : win7-pc
   Primary Dns Suffix  . . . . . . . : sectechlab.net
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : sectechlab.net

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . : sectechlab.net
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection
   Physical Address. . . . . . . . . : 00-0C-29-6A-BB-C8
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.1.100(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Thursday, June 15, 2017 4:19:27 PM
   Lease Expires . . . . . . . . . . : Saturday, June 24, 2017 10:42:21 AM
   Default Gateway . . . . . . . . . : 192.168.1.1
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DNS Servers . . . . . . . . . . . : 192.168.1.1
   Primary WINS Server . . . . . . . : 192.168.1.1
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Mounted Drives

```
C:\Windows\system32>net use
net use
New connections will be remembered.

There are no entries in the list.
```

Check ARP table

```
C:\Windows\system32>ARP -a
ARP -a

Interface: 192.168.1.100 --- 0xb
  Internet Address      Physical Address      Type
  192.168.1.1           00-0c-29-34-42-0a     dynamic
  192.168.1.4           00-0c-29-ea-27-03     dynamic
  192.168.1.106         00-0c-29-3a-2b-9f     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

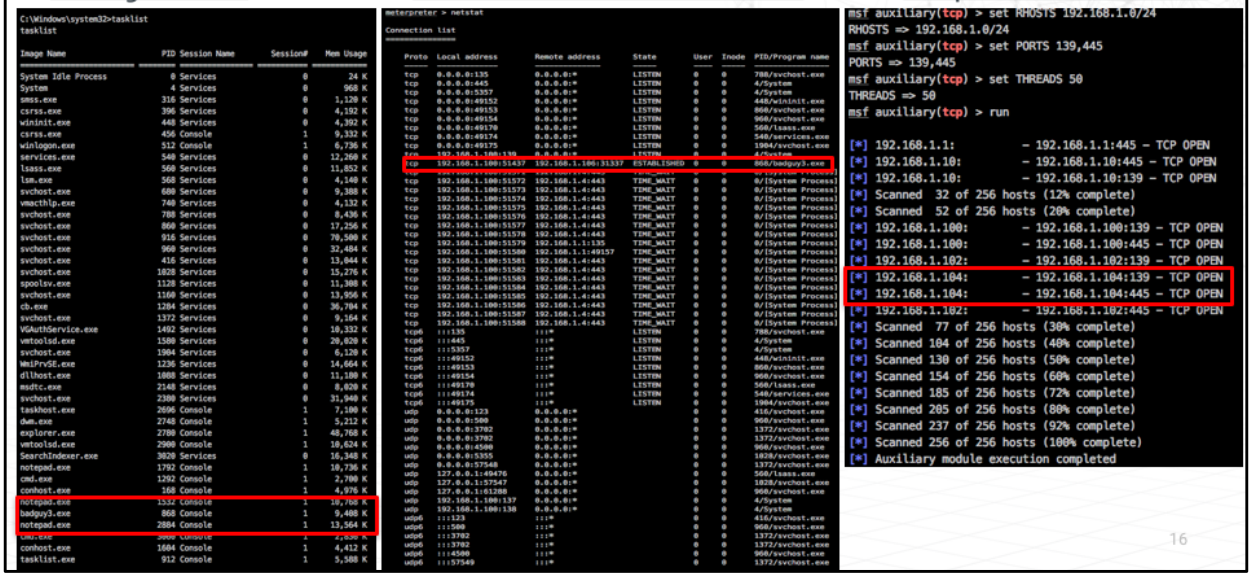
Reconnaissance – Processes

sqrrl

Running Processes

Processes with Network Connections

Nmap network the host is on



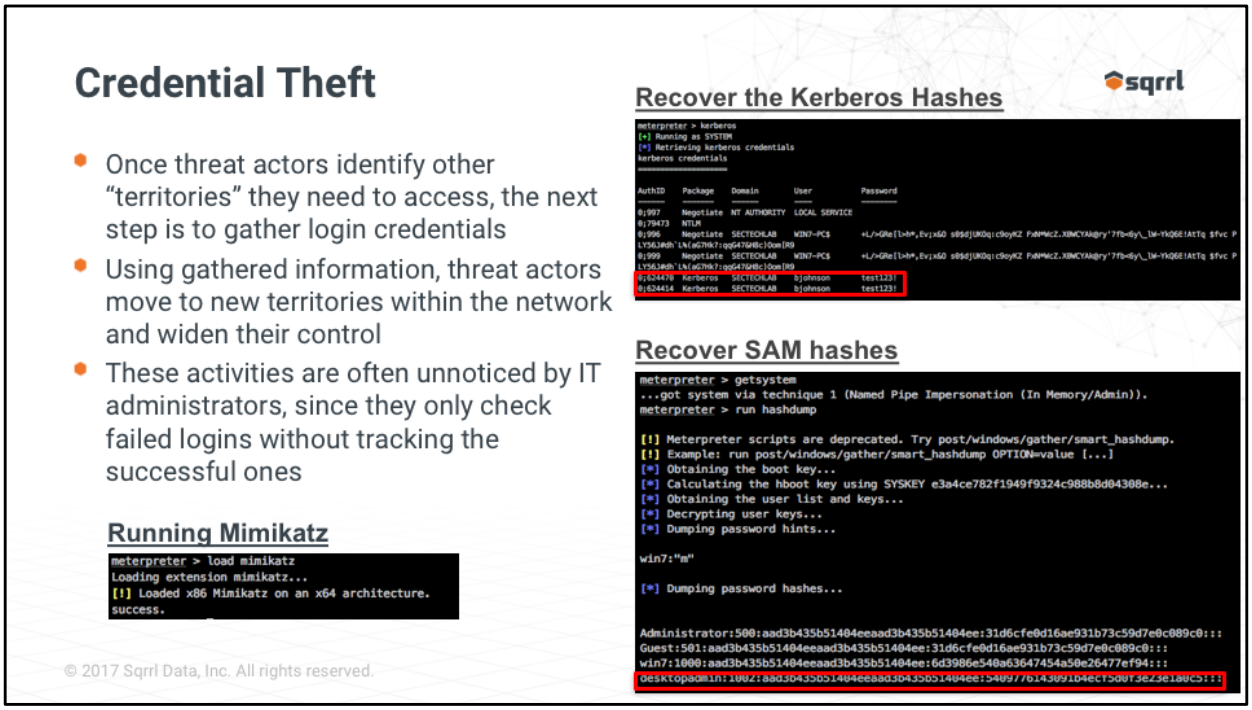
```
C:\Windows\system32>tasklist
tasklist

Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
System Idle Process              0 Services                   0         24 K
System                           4 Services                   0        968 K
smss.exe                       316 Services                   0      1,120 K
csrss.exe                      396 Services                   0      4,192 K
wininit.exe                    448 Services                   0      4,392 K
csrss.exe                      456 Console                    1      9,332 K
winlogon.exe                   512 Console                    1      6,736 K
services.exe                   548 Services                   0     12,260 K
lsass.exe                      560 Services                   0     11,852 K
lsm.exe                        568 Services                   0      4,140 K
svchost.exe                    680 Services                   0      9,388 K
vmacthlp.exe                   740 Services                   0      4,132 K
svchost.exe                    788 Services                   0      8,436 K
svchost.exe                    860 Services                   0     17,256 K
svchost.exe                    916 Services                   0     70,500 K
svchost.exe                    960 Services                   0     32,484 K
svchost.exe                    416 Services                   0     13,044 K
svchost.exe                   1028 Services                   0     15,276 K
spoolsv.exe                   1128 Services                   0     11,308 K
svchost.exe                   1160 Services                   0     13,956 K
cb.exe                        1284 Services                   0     36,794 K
svchost.exe                   1372 Services                   0      9,164 K
VGAuthService.exe             1492 Services                   0     10,332 K
vmtoolsd.exe                  1580 Services                   0     20,020 K
svchost.exe                   1904 Services                   0      6,120 K
WmiPrvSE.exe                  1236 Services                   0     14,664 K
dllhost.exe                   1808 Services                   0     11,180 K
msdtc.exe                     2148 Services                   0      8,020 K
svchost.exe                   2380 Services                   0     31,940 K
taskhost.exe                  2696 Console                    1      7,100 K
dwm.exe                       2748 Console                    1      5,212 K
explorer.exe                  2780 Console                    1     48,768 K
vmtoolsd.exe                  2900 Console                    1     18,624 K
SearchIndexer.exe             3020 Services                   0     16,348 K
notepad.exe                   1792 Console                    1     10,736 K
cmd.exe                       1292 Console                    1      2,700 K
conhost.exe                    168 Console                    1      4,976 K
notepad.exe                   1532 Console                    1     10,760 K
badguy3.exe                    868 Console                    1      9,488 K
notepad.exe                   2884 Console                    1     13,564 K
[illegible]                   3004 Console                    1      2,636 K
conhost.exe                   1604 Console                    1      4,412 K
tasklist.exe                   912 Console                    1      5,588 K
```

```
meterpreter > netstat

Connection list
===============

    Proto  Local address          Remote address        State        User  Inode  PID/Program name
    -----  -------------          --------------        -----        ----  -----  ----------------
    tcp    0.0.0.0:135            0.0.0.0:*             LISTEN       0     0      788/svchost.exe
    tcp    0.0.0.0:445            0.0.0.0:*             LISTEN       0     0      4/System
    tcp    0.0.0.0:5357           0.0.0.0:*             LISTEN       0     0      4/System
    tcp    0.0.0.0:49152          0.0.0.0:*             LISTEN       0     0      448/wininit.exe
    tcp    0.0.0.0:49153          0.0.0.0:*             LISTEN       0     0      860/svchost.exe
    tcp    0.0.0.0:49154          0.0.0.0:*             LISTEN       0     0      960/svchost.exe
    tcp    0.0.0.0:49170          0.0.0.0:*             LISTEN       0     0      560/lsass.exe
    tcp    0.0.0.0:49174          0.0.0.0:*             LISTEN       0     0      548/services.exe
    tcp    0.0.0.0:49175          0.0.0.0:*             LISTEN       0     0      1904/svchost.exe
    tcp    192.168.1.100:139      0.0.0.0:*             LISTEN       0     0      4/System
    tcp    192.168.1.100:51437    192.168.1.100:31337   ESTABLISHED  0     0      868/badguy3.exe
    tcp    192.168.1.100:51572    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51573    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51574    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51575    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51576    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51577    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51578    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51579    192.168.1.1:135       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51580    192.168.1.1:49157     TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51581    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51582    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51583    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51584    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51585    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51586    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51587    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp    192.168.1.100:51588    192.168.1.4:443       TIME_WAIT    0     0      0/[System Process]
    tcp6   :::135                 :::*                  LISTEN       0     0      788/svchost.exe
    tcp6   :::445                 :::*                  LISTEN       0     0      4/System
    tcp6   :::5357                :::*                  LISTEN       0     0      4/System
    tcp6   :::49152               :::*                  LISTEN       0     0      448/wininit.exe
    tcp6   :::49153               :::*                  LISTEN       0     0      860/svchost.exe
    tcp6   :::49154               :::*                  LISTEN       0     0      960/svchost.exe
    tcp6   :::49170               :::*                  LISTEN       0     0      560/lsass.exe
    tcp6   :::49174               :::*                  LISTEN       0     0      548/services.exe
    tcp6   :::49175               :::*                  LISTEN       0     0      1904/svchost.exe
    udp    0.0.0.0:123            0.0.0.0:*                          0     0      416/svchost.exe
    udp    0.0.0.0:500            0.0.0.0:*                          0     0      960/svchost.exe
    udp    0.0.0.0:3702           0.0.0.0:*                          0     0      1372/svchost.exe
    udp    0.0.0.0:3702           0.0.0.0:*                          0     0      1372/svchost.exe
    udp    0.0.0.0:4500           0.0.0.0:*                          0     0      960/svchost.exe
    udp    0.0.0.0:5355           0.0.0.0:*                          0     0      1028/svchost.exe
    udp    0.0.0.0:57548          0.0.0.0:*                          0     0      1372/svchost.exe
    udp    127.0.0.1:49476        0.0.0.0:*                          0     0      560/lsass.exe
    udp    127.0.0.1:57547        0.0.0.0:*                          0     0      1028/svchost.exe
    udp    127.0.0.1:61288        0.0.0.0:*                          0     0      960/svchost.exe
    udp    192.168.1.100:137      0.0.0.0:*                          0     0      4/System
    udp    192.168.1.100:138      0.0.0.0:*                          0     0      4/System
    udp6   :::123                 :::*                               0     0      416/svchost.exe
    udp6   :::500                 :::*                               0     0      960/svchost.exe
    udp6   :::3702                :::*                               0     0      1372/svchost.exe
    udp6   :::3702                :::*                               0     0      1372/svchost.exe
    udp6   :::4500                :::*                               0     0      960/svchost.exe
    udp6   :::57549               :::*                               0     0      1372/svchost.exe
```

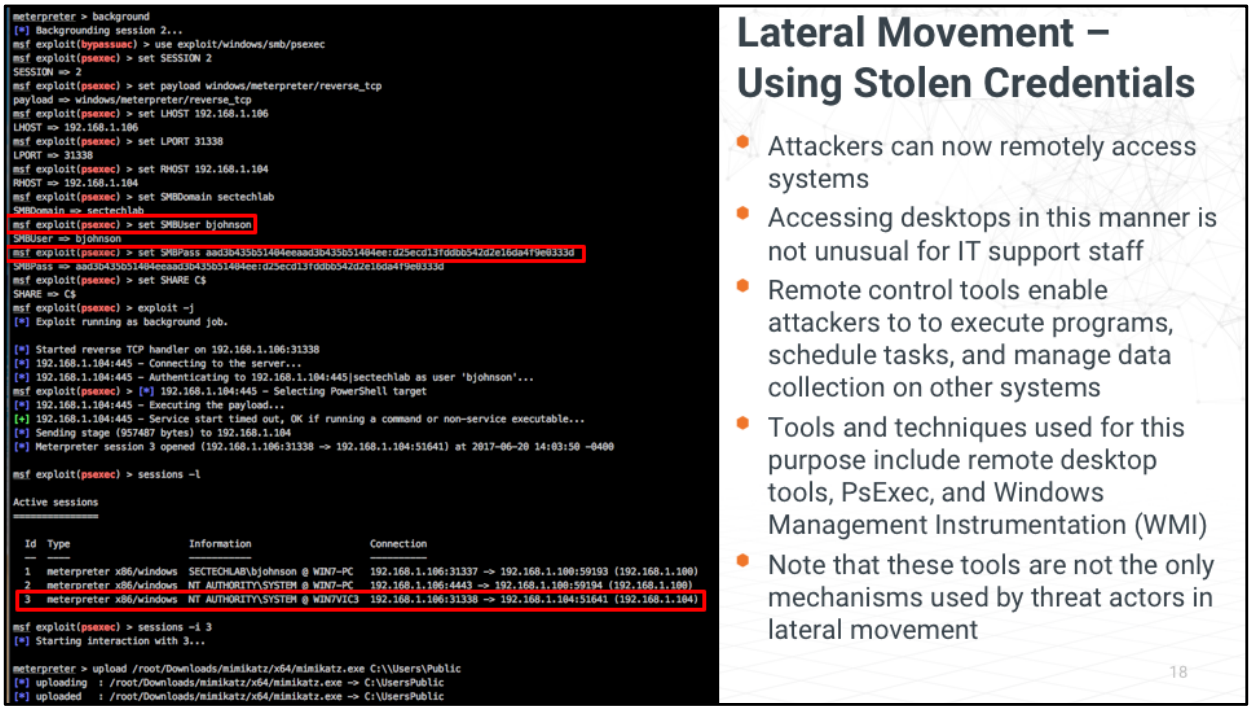
```
msf auxiliary(tcp) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(tcp) > set PORTS 139,445
PORTS => 139,445
msf auxiliary(tcp) > set THREADS 50
THREADS => 50
msf auxiliary(tcp) > run

[*] 192.168.1.1:          - 192.168.1.1:445 - TCP OPEN
[*] 192.168.1.10:         - 192.168.1.10:445 - TCP OPEN
[*] 192.168.1.10:         - 192.168.1.10:139 - TCP OPEN
[*] Scanned  32 of 256 hosts (12% complete)
[*] Scanned  52 of 256 hosts (20% complete)
[*] 192.168.1.100:        - 192.168.1.100:139 - TCP OPEN
[*] 192.168.1.100:        - 192.168.1.100:445 - TCP OPEN
[*] 192.168.1.102:        - 192.168.1.102:139 - TCP OPEN
[*] 192.168.1.104:        - 192.168.1.104:139 - TCP OPEN
[*] 192.168.1.104:        - 192.168.1.104:445 - TCP OPEN
[*] 192.168.1.102:        - 192.168.1.102:445 - TCP OPEN
[*] Scanned  77 of 256 hosts (30% complete)
[*] Scanned 104 of 256 hosts (40% complete)
[*] Scanned 130 of 256 hosts (50% complete)
[*] Scanned 154 of 256 hosts (60% complete)
[*] Scanned 185 of 256 hosts (72% complete)
[*] Scanned 205 of 256 hosts (80% complete)
[*] Scanned 237 of 256 hosts (92% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

16

SMB ports
Port 139 is ‘NBT over IP’ NETBIOS
Port 445 is ‘SMB over IP’ PSEXEC usage



Next up we have the credential theft stage. What you see here is me running a mimikatz metasploit module. I used this because mimikatz will take credentials out of memory and crack the hashes for me. This allows me to harvest credentials without having to put an executable on the system. These activities are often unnoticed by IT administrators, since they only check failed logins without tracking the successful ones.



Finally, we get to the stage where the rubber hits the road. I am going to use the network data I enumerated from the victim as well as the credentials I just took to PsExec into another system on the network. PsExec is a Sysinternals tool that is signed by Microsoft and commonly exists in enterprise environments for legitimate administrative work. This tool gives me full CLI access to a target system so that I can use that remote system while the authorized user is using it without them being the wiser. Once on that system, I will try to reach my goal of stealing information or I will start the attack lifecycle again at the recon stage and repeat it on more systems in the environment until I achieve my goals.

I can now remotely access desktops

Accessing desktops in this manner is not unusual for IT support staff

Remote access will therefore not be readily associated with an ongoing attack

Attackers may also gather domain credentials to log into systems, servers, and switches

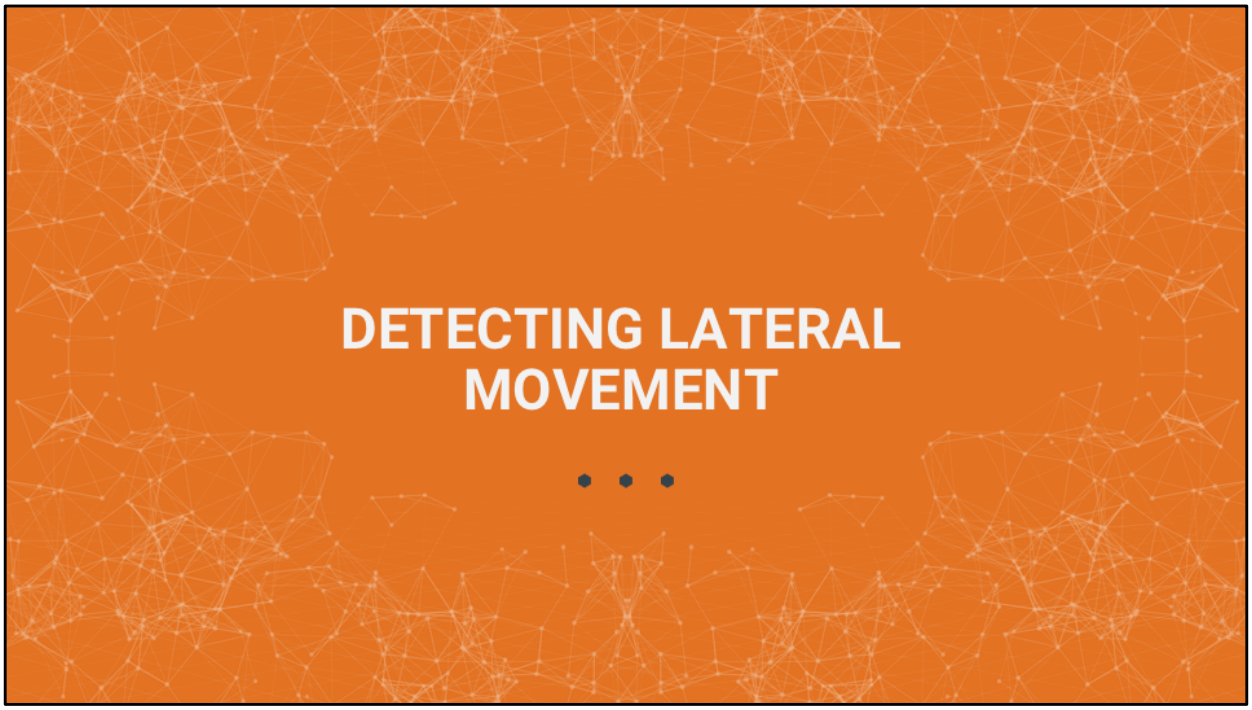
Because of password reuse by users

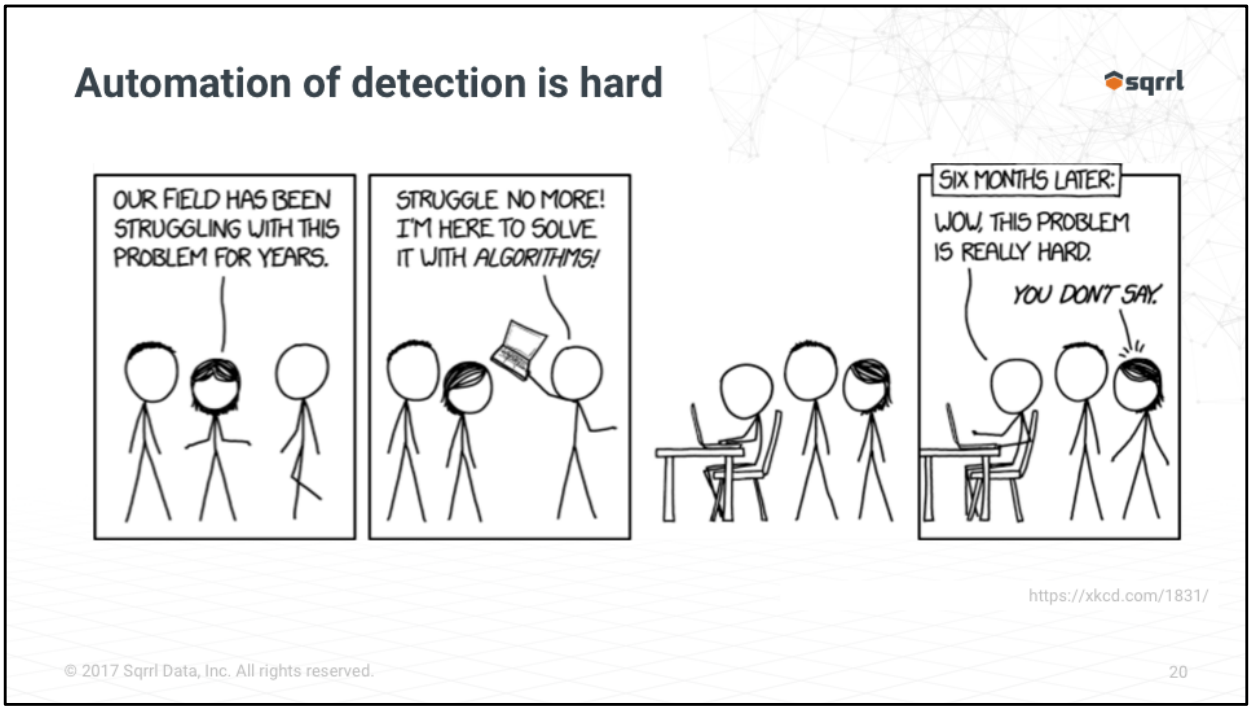
Remote control tools enable attackers to access other desktops in the network and perform actions like executing programs, scheduling tasks, and managing data

collection on other systems

Tools and techniques used for this purpose include remote desktop tools, PsExec, and Windows Management Instrumentation (WMI).

Note that these tools are not the only mechanisms used by threat actors in lateral movement.


DETECTING LATERAL
MOVEMENT

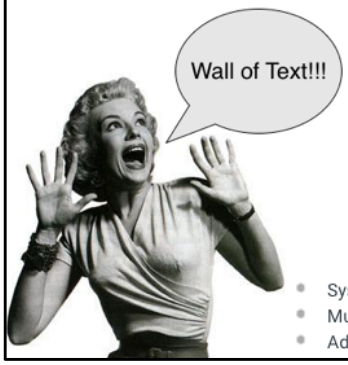


The type of dataset that you use to hunt for lateral movement depends on what you are hunting for and, by extension, what your hypothesis is. Attempting to automate this is difficult.

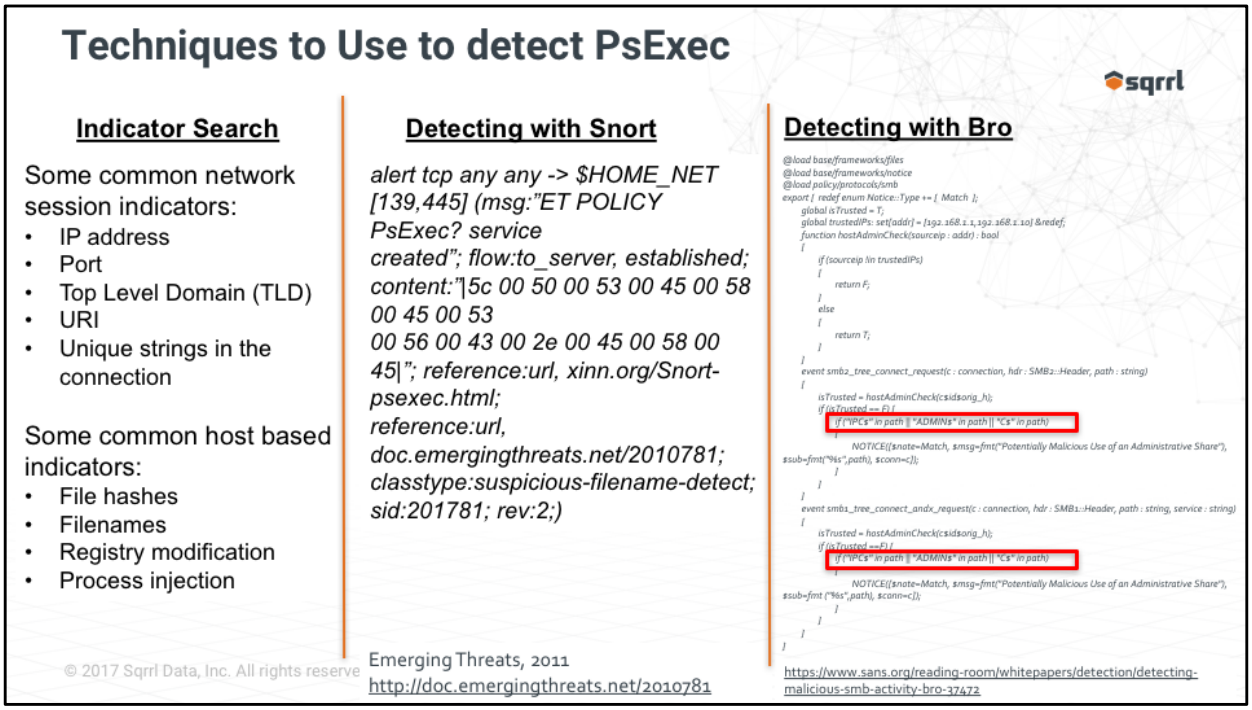
Datasets suggested for detection of lateral movement

sqrrl

- For identifying use of remote access protocols, you will want to focus primarily on network session metadata, including:
  - Netflow ("flow" data in general)
  - Firewall logs (should log allowed / accepted packets)
  - Bro Conn log
- For identifying User Access Control (UAC) events, you will want to focus on authentication logs, including:
  - Active Directory logs/Windows Security Event logs
    - EventID
      - 528 or 4624 is indicative of a successful logon
      - 529 or 4625 is a failed logon
      - 552 and 4648 are indicative of an attacker attempting to use the runas command or authenticate against a remote host as an alternative user, **#privilegeEscalation**.
      - 602 and 4698 are indicative of a scheduled task creation
      - 601 and 4697 are indicative of a service creation
    - Account Features
      - Service account
      - Interactive login
  - System Security Event logs
  - Multi-Factor Authentication (MFA) logs
  - Additional UAC applications if exists



21



After having developed the right hypotheses and chosen the necessary datasets, a hunter must still know what techniques to use to investigate a hypothesis. Here we will survey 3 types of techniques that you can use to investigate the above.

Indicator search

As in all cases of using indicators in hunting, the value of this approach will be impacted by the value of the indicator. Locally sourced indicators will generally provide a high value because they tend to be timely and relevant to the network or systems you might be trying to protect. These types of indicators can be gathered from previous incidents or by internal threat intelligence teams.

It's also important to remember that it is relatively easy for attackers to change the remote infrastructure that they use to conduct attacks; if you are using indicators to hunt, then you should be aware that the indicators may no longer be relevant to a particular attacker or attack tool.

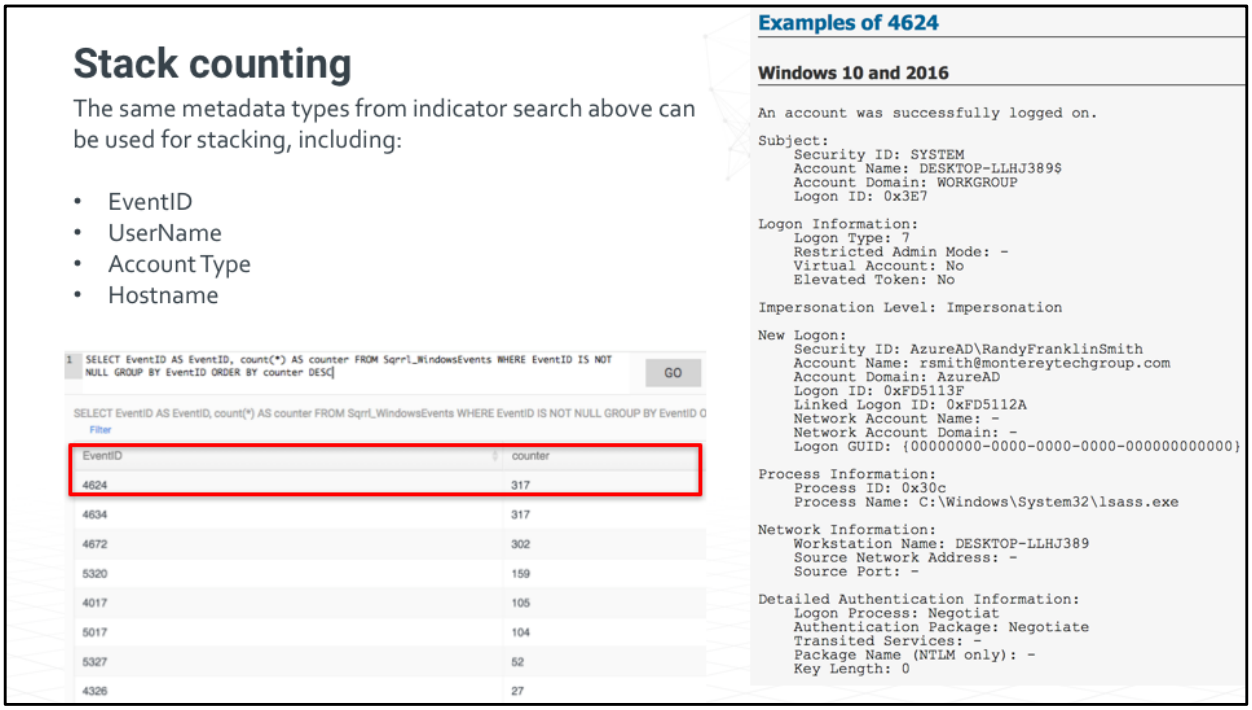
snort

Snort can be used to detect malicious SMB activity. Snort is an open-source intrusion detection and and prevention system, and designed to detect attacks via a pattern-matching signature.

Bro

One technique to detect PsExec activity with Bro is by using custom Bro scripts looking for PsExec's use of the C$, ADMIN$, and/or IPC$ shares. These shares added notice messages of "Potentially Malicious Use of an Administrative Share" in the Bro Notice log. The use of PsExec creates an executable named PSEXESVC.exe on the target system.

Why does this detect the usage of PsExec? Metasploit contains a modified version of PsExec.exe that doesn't have a source file hash as it is run from the attacker system via meterpreter. To detect this PsExec usage, we depend on the Bro notice log, which verifies detection of the Metasploit PsExec module's use of C$, ADMIN$, and IPC$ shares.

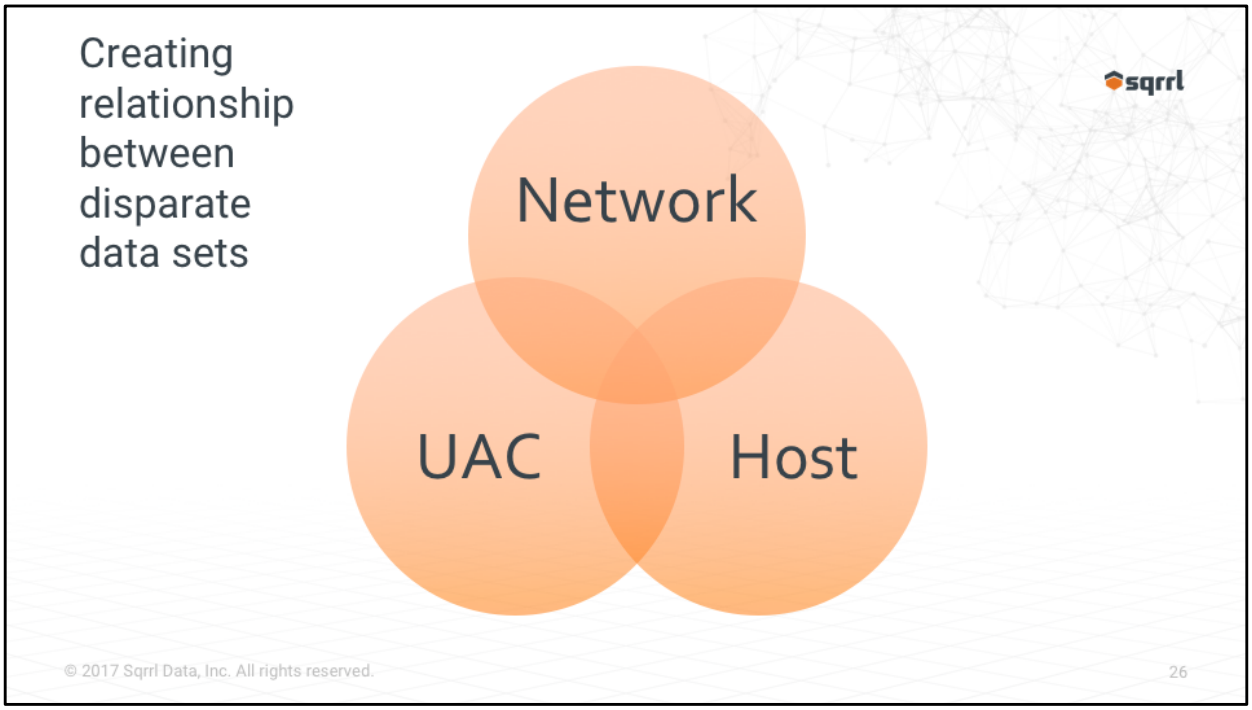


Stacking is a technique commonly used in many different kinds of hunts. In the case of hunting for command and control activity, a hunter will want to stack for anomalous instances of inbound or outbound traffic.

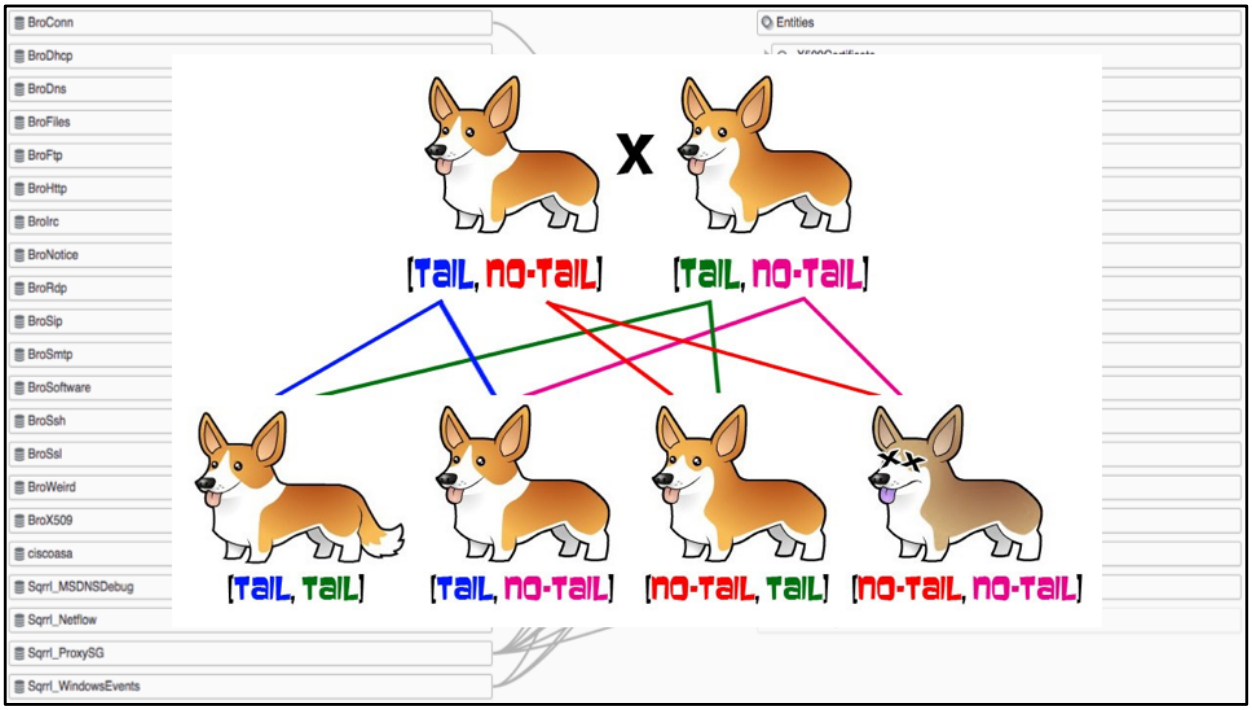
To find lateral movement, you will want to focus on either bidirectional or external to inbound connection flows. The effectiveness of using stacking is dependent on having a finely tuned input. Too little won't reveal enough and too much will flood your ability to tease out meaningful deviations. If a given result set is too large, then consider further filtering of the input data set (e.g., isolate your focus to specific internal subnets). Alternatively, change the metadata that is being stacked (e.g., change from stacking hostnames to stacking EventID and username).


REAL WORLD
THREAT HUNTING FOR
LATERAL MOVEMENT

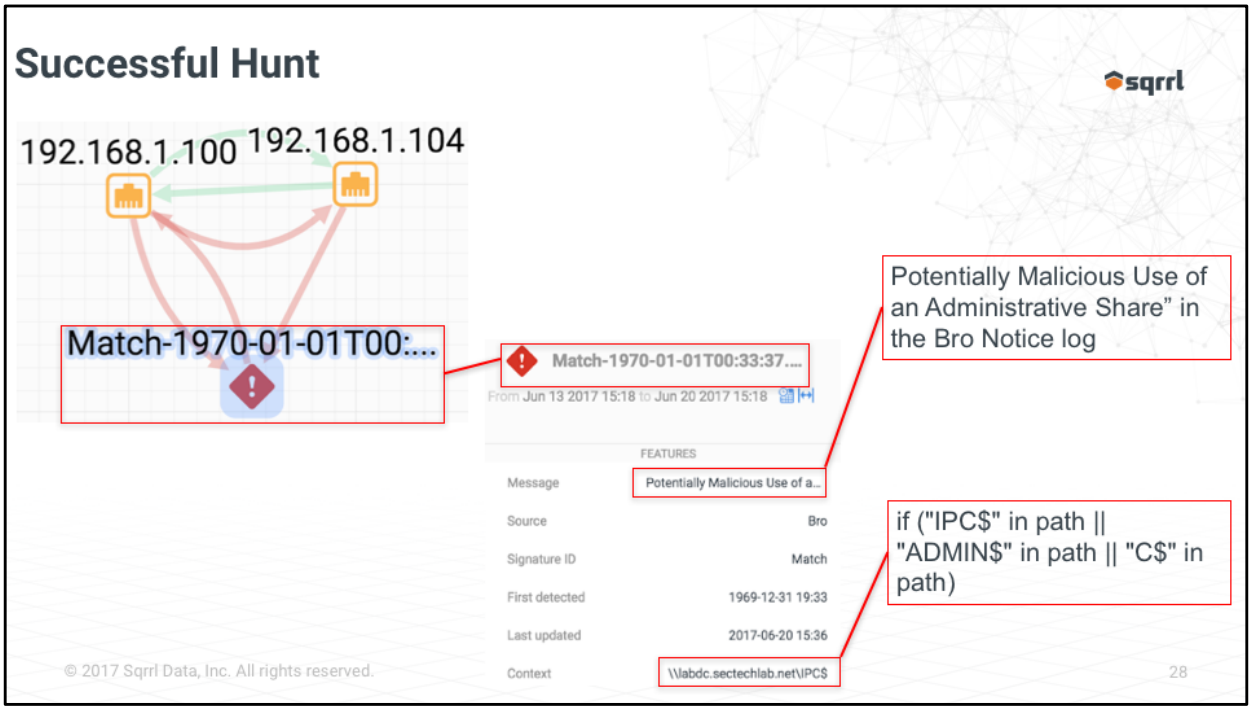
Stages	Lateral Movement (Windows Environment)
What are you looking for? (Hypothesis)	**Hypothesis:** Attackers may be attempting to move laterally in my Windows environment by leveraging PsExec. **Look for:** Anomalies in host to host traffic leveraging the PsExec binary, service, and/or network traffic. "C$\|ADMIN$|IPC$" shares being used in network traffic.
Investigation (Data)	**Datasets:** For identifying use of PsExec, you will want to focus primarily on application protocol metadata, including: • Netflow ("flow" data in general) • Active Directory logs • Windows Security Event logs • Multi-Factor Authentication (MFA) logs (if windows hosts leverage MFA) • Additional UAC applications logs (if exists) • EDR tool logs (if exists)
Uncover Patterns and IOCs (Techniques)	1. Use a search to identify "Potentially Malicious Use of an Administrative Share" messages in your bro_notice log. 2. Take the output of step 1 and remove hosts as you confirm they are legitimately connecting to a destination over SMB. This should leave only unexplained SMB connections that need further analysis. 3. Take the results of step 2 and stack the data for what is useful to investigating your hypothesis 1. For example: destination IP, port used, connection duration/length, etc.
Inform and Enrich Analytics (Takeaways)	The destination IP addresses, path, and ports involved in the Lateral Movement activity you have discovered can be taken as IOCs and added to an indicator database in order to expand automated detection systems. You can also create packet-level signatures to trigger alerts for cases where the admin share connections you have discovered may appear again.



First I need to define relationships between different data sources. For example, my network data and authentication event data both contain the IP address field. This means I can create a relationship between disparate data sets as shown below.

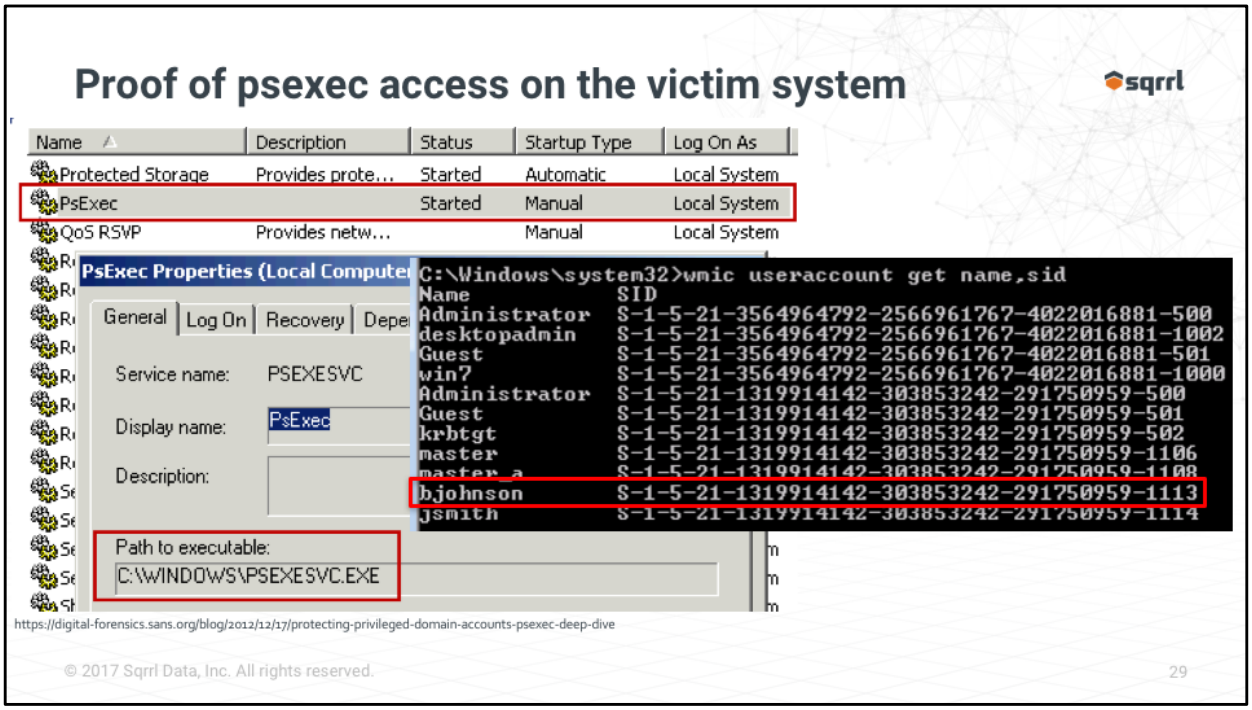


The above visualization expands on the high level visualization and illustrates all of the low level field mappings between each data source being ingested and each entity defined. This is where the actual mapping of the IP address field between authentication data and network data.



What you see here are confirmed network connections between 2 hosts as well as having the bro alert created earlier in this presentation fire and also be connected to both hosts. This proves that psexec was attempted between both hosts. Upon further investigation the psexec

Finally, with all the hard work done, we are able to visualize the event of psexec being used to move from 192.168.1.100 to 192.168.1.104 as well as incorporate a bro alert for psexec to add further validation. While this image only shows the exact activity I am describing for ease of reading, this is what I expect an end result of a hunt to look like. All additional data and possible connections have been investigated and excluded from the original large data set until you are only left with the anomalous/suspicious/malicious event.



Notice the service executable resides in C:\WINDOWS.

Once the user has cleanly logged off (exited) *PsExec*, the service is removed and PSEXESVC.EXE is deleted. Although *PsExec* is forensic investigation will show the file and its metadata.

Also notice that the owner of this file is the user who connected remotely to run *PsExec*. A profile is created for this user at the time of first login, if it didn't already exist. This is regardless of whether an interactive logon occurred or not. This user profile and SID now exist on the victim host for a user account that has no reason to have logged into this system.

I consider this a successful hunt. I would also have considered it a success if I had found nothing at all because the point of a hunt isn't to a true positive malicious event every time, but instead it is to validate a hypothesis, to answer a question with a definitive yes or no. Good luck to all and happy hunting.

And as always, remember my motto, Flag it, Tag it, and Bag it.

Questions?