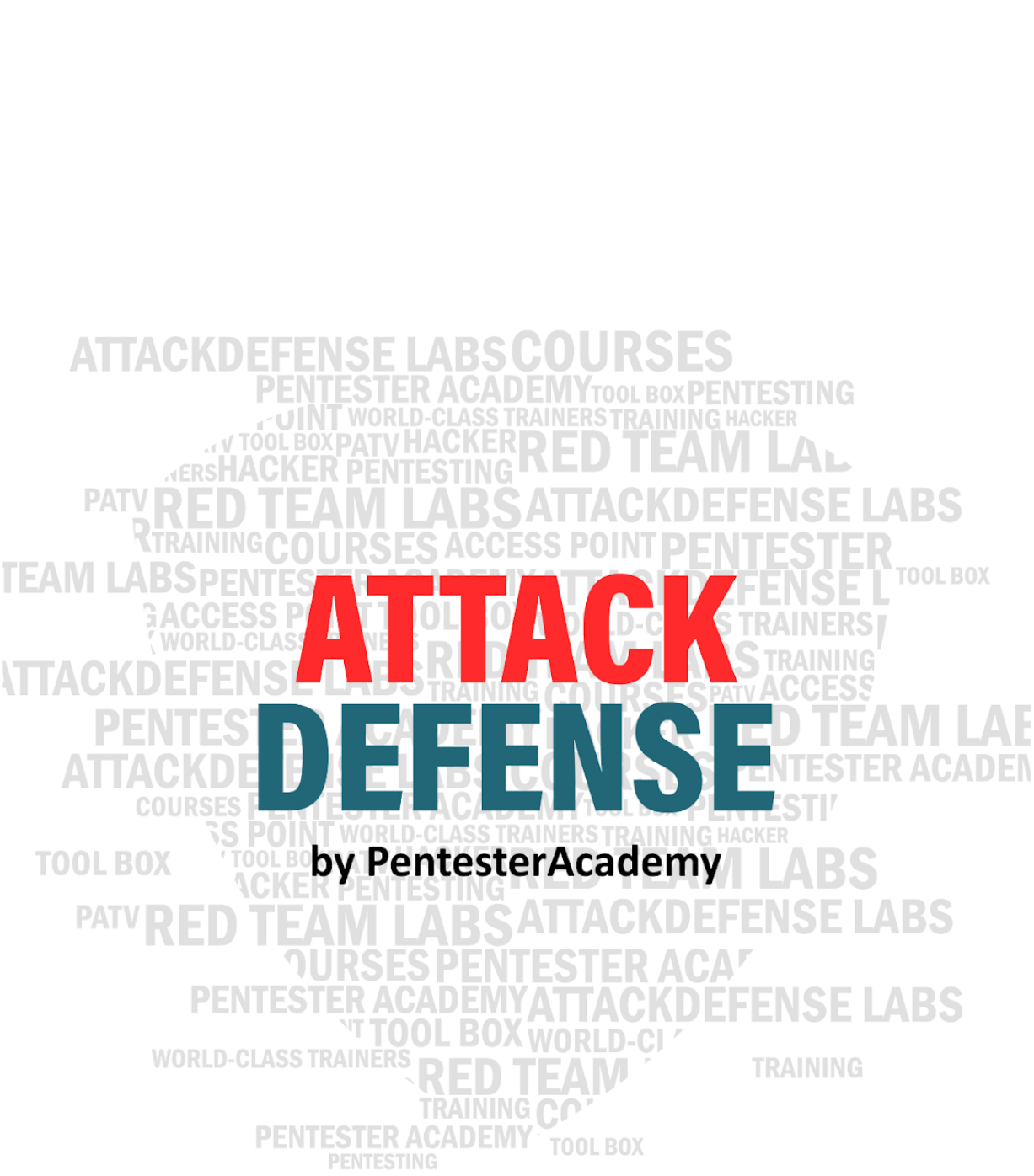
ATTACK
DEFENSE
by PentesterAcademy

Name	Tools : Socat
URL	https://attackdefense.com/challengedetails?cid=1812
Type	Beginner Skills : Linux For Pentesters

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Learn socat by doing the following activities.

1. Use socat to listen on TCP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance.
2. Use socat to listen on UDP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance.
3. Use socat to listen on TCP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance. Protect the communication using SSL/TLS.
4. Use socat to listen on port 4000 of the localhost and provide bash access to any session connecting to it. Open a new tab and use socat to connect to the listening instance.
5. Use socat to listen on a UNIX socket. Open a new tab and use socat to connect to the listening instance.
6. SSH service is running on one of the remote target machines. Use socat to create a tunnel from local port 8000 to this remote SSH service. The SSH credentials are given below:
    - Username: student
    - Password: student

Solution:

Task 1: Use socat to listen on TCP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance.

In first terminal, use socat to listen on TCP port 4000

Command: socat TCP-LISTEN:4000,reuseaddr,pf=ip4,fork -

```
root@attackdefense:~# socat TCP-LISTEN:4000,reuseaddr,pf=ip4,fork -
```

Open another terminal and use socat to connect to this

Command: socat TCP:localhost:4000 -

```
root@attackdefense:~# socat TCP:localhost:4000 -
hello
whoami
```

Whatever the user types on this side, will be replayed to the other side.

```
root@attackdefense:~# socat TCP-LISTEN:4000,reuseaddr,pf=ip4,fork -
hello
whoami
```

In this manner, socat can be used to create a listener and connector.

Task 2: Use socat to listen on UDP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance.

In first terminal, use socat to listen on UDP port 4000

Command: socat UDP-LISTEN:4000,reuseaddr,pf=ip4,fork -

```
root@attackdefense:~# socat UDP-LISTEN:4000,reuseaddr,pf=ip4,fork -
```

Open another terminal and use socat to connect to this.

Command: socat UDP:localhost:4000 -

```
root@attackdefense:~# socat UDP:localhost:4000 -
hello
date
```

Whatever the user types on this side, will be replayed to the other side.

```
root@attackdefense:~# socat UDP-LISTEN:4000,reuseaddr,pf=ip4,fork -
hello
date
```

Task 3: Use socat to listen on TCP port 4000 of the localhost. Open a new tab and use socat to connect to the listening instance. Protect the communication using SSL/TLS.

First, to use SSL/TLS, the certificates are needed.

Certificate Generatation for Server

Step 1: Set the variable

Command: FILENAME="server"

```
root@attackdefense:~# FILENAME="server"
```

Step 2: Generating 2048-bit RSA key

Commands:
openssl genrsa -des3 -passout pass:x1234 -out "$FILENAME.pass.key" 2048
openssl rsa -passin pass:x1234 -in "$FILENAME.pass.key" -out "$FILENAME.key"
rm -f "$FILENAME.pass.key"

```
root@attackdefense:~# openssl genrsa -des3 -passout pass:x1234 -out "$FILENAME.pass.key" 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................................+++++
.............................+++++
e is 65537 (0x010001)
root@attackdefense:~# openssl rsa -passin pass:x1234 -in "$FILENAME.pass.key" -out "$FILENAME.key"
writing RSA key
root@attackdefense:~# rm -f "$FILENAME.pass.key"
```

Step 3: Generate a certificate signing request. In real deployment, it is advisable to fill proper information. However, for lab use, we can leave these empty.

Command: openssl req -new -key "$FILENAME.key" -out "$FILENAME.csr"

```
root@attackdefense:~# openssl req -new -key "$FILENAME.key" -out "$FILENAME.csr"
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@attackdefense:~#
```

Step 4: Generate a signed certificate

Command: openssl x509 -req -sha256 -days 365 -in "$FILENAME.csr" -signkey "$FILENAME.key" -out "$FILENAME.crt"

```
root@attackdefense:~# openssl x509 -req -sha256 -days 365 -in "$FILENAME.csr" -signkey "$FILENAME.key" -out "$FILENAME.crt"
Signature ok
subject=C = AU, ST = Some-State, O = Internet Widgits Pty Ltd
Getting Private key
root@attackdefense:~#
```

Step 5: Generate the PEM container

Command: cat "$FILENAME.key" "$FILENAME.crt" > "$FILENAME.pem"

```
root@attackdefense:~# cat "$FILENAME.key" "$FILENAME.crt" > "$FILENAME.pem"
```

Step 6: Viewing the created certificates

Command: ls -l

```
root@attackdefense:~# ls -l
total 28
-rw-r--r-- 1 root root  293 Feb  7  2019 README
-rw-r--r-- 1 root root 1123 Apr  6 19:33 server.crt
-rw-r--r-- 1 root root  956 Apr  6 19:32 server.csr
-rw------- 1 root root 1679 Apr  6 19:32 server.key
-rw-r--r-- 1 root root 2802 Apr  6 19:33 server.pem
drwxr-xr-x 1 root root 4096 May 26  2019 tools
drwxr-xr-x 2 root root 4096 Feb  7  2019 wordlists
root@attackdefense:~#
```

Certificate Generatation for Client

Step 1: Set the variable

Command: FILENAME="client"

```
root@attackdefense:~# FILENAME="client"
```

Step 2: Repeat steps 2 to 5 for client certificate (as done for server certificate above)

Step 3: Viewing the created certificates

Command: ls -l

```
root@attackdefense:~# ls -l
total 44
-rw-r--r-- 1 root root  293 Feb  7  2019 README
-rw-r--r-- 1 root root 1123 Apr  6 19:36 client.crt
-rw-r--r-- 1 root root  956 Apr  6 19:36 client.csr
-rw------- 1 root root 1675 Apr  6 19:36 client.key
-rw-r--r-- 1 root root 2798 Apr  6 19:37 client.pem
```

Socat main task

In first terminal, use socat to listen on TCP port 4000 with SSL/TLS suppport

Command: socat OPENSSL-LISTEN:4000,reuseaddr,pf=ip4,fork,cert=./server.pem,cafile=./client.pem -

```
root@attackdefense:~# socat OPENSSL-LISTEN:4000,reuseaddr,pf=ip4,fork,cert=./server.pem,cafile=./client.pem -
```

server.pem will be used by socat listerner as its own certificate whereas client.pem will be used to validate the connecting client.

Open another terminal and use socat to connect to this

Command: socat OPENSSL:localhost:4000,verify=0,cert=./client.pem,cafile=./server.pem -

Opposite to the listener part, client.pem will be used by socat client as its own certificate whereas server.pem will be used to validate the listerner

```
root@attackdefense:~# socat OPENSSL:localhost:4000,verify=0,cert=./client.pem,cafile=./server.pem -
hi
date
```

Whatever the user types on this side, will be replayed to the other side.

```
root@attackdefense:~# socat OPENSSL-LISTEN:4000,reuseaddr,pf=ip4,fork,cert=./server.pem,cafile=./client.pem -
hi
date
```

In this manner, socat can be used to create a listener and connector.

Task 4: Use socat to listen on port 4000 of the localhost and provide bash access to any session connecting to it. Open a new tab and use socat to connect to the listening instance.

In first terminal, use socat to listen on TCP port 4000

Command: socat TCP-LISTEN:4000 EXEC:/bin/bash

```
root@attackdefense:~# socat TCP-LISTEN:4000 EXEC:/bin/bash
```

Open another terminal and use socat to connect to this

Command: socat TCP:localhost:4000 -

```
root@attackdefense:~# socat TCP:localhost:4000 -
whoami
root
pwd
/root
```

The output for the commands will be displayed. But the user types commands and results, will not appear on the other side.

Task 5: Use socat to listen on a UNIX socket. Open a new tab and use socat to connect to the listening instance.

In first terminal, use socat to listen on a UNIX socket

Command: socat UNIX-LISTEN:./test.sock,fork -

```
root@attackdefense:~# socat UNIX-LISTEN:./test.sock,fork -
```

Open another terminal and use socat to connect to this

Command: socat UNIX-CONNECT:./test.sock -

```
root@attackdefense:~# socat UNIX-CONNECT:./test.sock -
test
```

Whatever the user types on this side, will be replayed to the other side.

```
root@attackdefense:~# socat UNIX-LISTEN:./test.sock,fork -
test
```

In this manner, socat can be used to create a listener and connector.

Task 6: SSH service is running on one of the remote target machines. Use socat to create a tunnel from local port 8000 to this remote SSH service. The SSH credentials are given below:

- Username: student
- Password: student

Check the IP address of the machine

Command: ip addr

```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
17006: eth0@if17007: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:08 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.8/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
17009: eth1@if17010: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:fc:b6:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.252.182.2/24 brd 192.252.182.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

Scan other remote machines present on the network to find the SSH service.

Command: nmap 192.252.182.3-10

```
root@attackdefense:~#  nmap 192.252.182.3-10
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-06 21:12 UTC
Nmap scan report for target-1 (192.252.182.3)
Host is up (0.000015s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 02:42:C0:FC:B6:03 (Unknown)
```

The SSH service is running on 192.252.182.3.

Create a tunnel binding local port 8000 to remote machine.

Commands: socat TCP4-LISTEN:8000,reuseaddr,fork TCP:192.252.182.3:22

```
root@attackdefense:~# socat TCP4-LISTEN:8000,reuseaddr,fork TCP:192.252.182.3:22
```

Check the listening sockets on the local machine

Commands: netstat -tpln

```
root@attackdefense:~# netstat -tpln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:45654           0.0.0.0:*               LISTEN      18/ttyd
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      26/socat
tcp        0      0 127.0.0.11:40961        0.0.0.0:*               LISTEN      -
root@attackdefense:~#
```

SSH into the remote machine using the local port 8000

Commands: ssh -p 8000 student@127.0.0.1

Provide the password “student” when prompted.

```
root@attackdefense:~# ssh -p 8000 student@127.0.0.1
The authenticity of host '[127.0.0.1]:8000 ([127.0.0.1]:8000)' can't be established.
ECDSA key fingerprint is SHA256:c4K0S8zSQIB4HSfANambo5zi+Pf4XNfFt/rckem9bWc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[127.0.0.1]:8000' (ECDSA) to the list of known hosts.
student@127.0.0.1's password:
Welcome to Ubuntu 19.04 (GNU/Linux 4.15.0-72-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

References:

1. Generating SSL certificates (https://blog.travismclarke.com/post/generating-ssl-certs/)