TCPCopy

Information Security Inc.

Contents

About TCPCopy

- TCPCopy is a TCP stream replay tool to support real testing of Internet server applications

TCPCopy - A TCP Stream Replay Tool

Scenarios

- Distributed stress testing
  - Use tcpcopy to copy real-world data to stress test your server software. Bugs that only can be produced in high-stress situations can be found
- Live testing
  - Prove the new system is stable and find bugs that only occur in the real world
- Regression testing
- Performance comparison

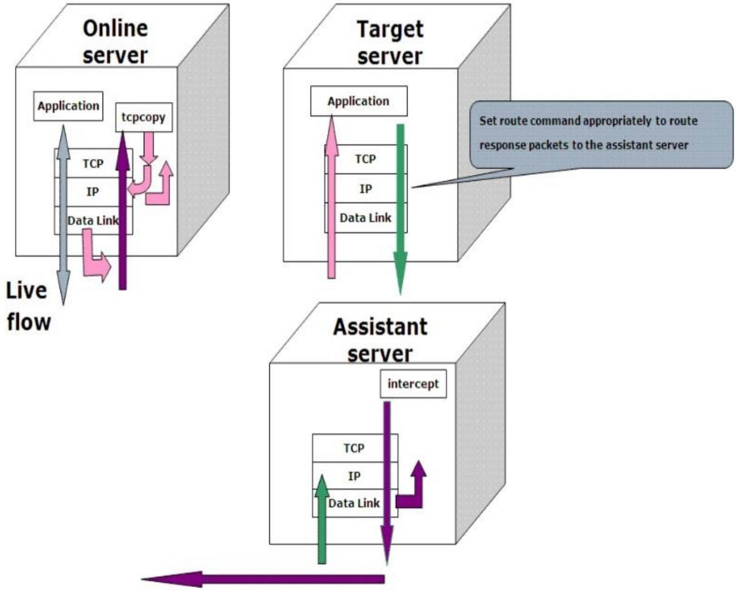
How does TCPCopy work?

- TCPCopy copies packets on the online server, modifies TCP/IP headers, and sends modified packets to the target test server
- In this way, TCP applications on the target test server will consider the packets from the online server as online requests from end-users

Architecture

- Architecture

- The TCPCopy server (intercept) runs on a separate machine instead of the test server



Demo configuration

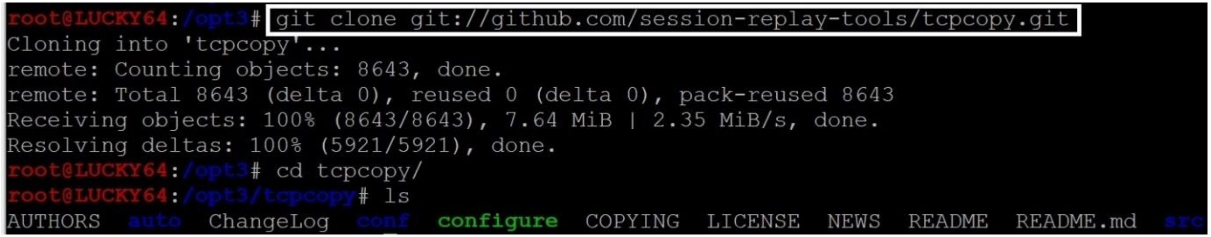
- Architecture

```
     ---------------            -------------          ----------------
     | Online     |            |  Test     |          | Assistant |
     | Server     |    ===     |  Server   |   ===    | Server    |
     | Apache     |            |  Nginx    |          |           |
     ---------------            -------------          ----------------
     Kali Linux 2017        Kali Linux 2017        Kali Linux 2017
  IP 192.168.86.86          192.168.86.88          192.168.86.87
```

Client machine (making the HTTP requests to the online server) IP > 192.168.30.85

Installing TCPCopy

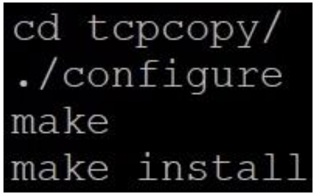
- Online server > Clone the repo



```
root@LUCKY64:/opt3# git clone git://github.com/session-replay-tools/tcpcopy.git
Cloning into 'tcpcopy'...
remote: Counting objects: 8643, done.
remote: Total 8643 (delta 0), reused 0 (delta 0), pack-reused 8643
Receiving objects: 100% (8643/8643), 7.64 MiB | 2.35 MiB/s, done.
Resolving deltas: 100% (5921/5921), done.
root@LUCKY64:/opt3# cd tcpcopy/
root@LUCKY64:/opt3/tcpcopy# ls
AUTHORS  auto  ChangeLog  conf  configure  COPYING  LICENSE  NEWS  README  README.md  src
```

Installing TCPCopy

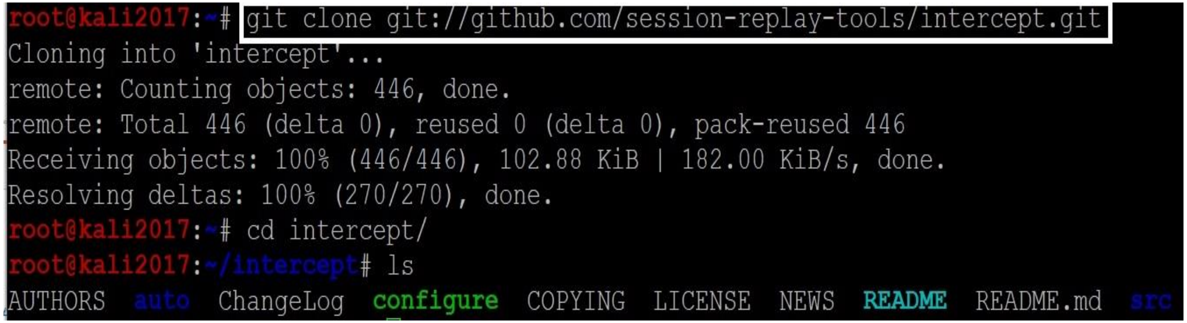
- Online Server > Configure and build TCPCopy



```
cd tcpcopy/
./configure
make
make install
```

Installing TCPCopy

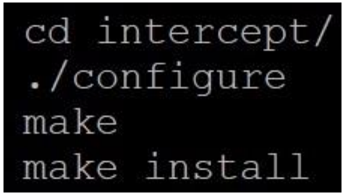
- Assistant (Intercept) server > Clone the repo



```
root@kali2017:~# git clone git://github.com/session-replay-tools/intercept.git
Cloning into 'intercept'...
remote: Counting objects: 446, done.
remote: Total 446 (delta 0), reused 0 (delta 0), pack-reused 446
Receiving objects: 100% (446/446), 102.88 KiB | 182.00 KiB/s, done.
Resolving deltas: 100% (270/270), done.
root@kali2017:~# cd intercept/
root@kali2017:~/intercept# ls
AUTHORS  auto  ChangeLog  configure  COPYING  LICENSE  NEWS  README  README.md  src
```

Installing TCPCopy

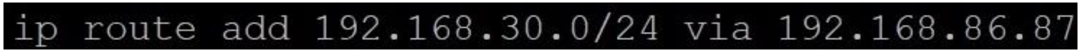
- Assistant (Intercept) server > Configure and build Intercept



```
cd intercept/
./configure
make
make install
```

Running TCPCopy

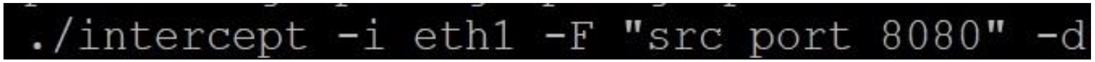
- Assume tcpcopy and intercept are both configured with "./configure“
- On the target test server which runs server applications. Set route commands appropriately to route response packets to the assistant server



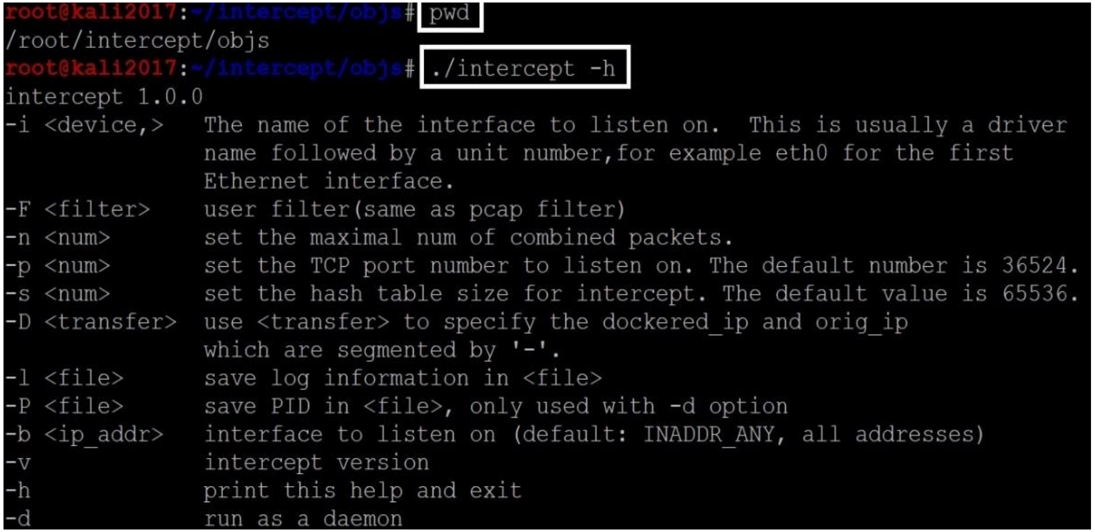
```
ip route add 192.168.30.0/24 via 192.168.86.87
```

Running TCPCopy

- On the assistant server which runs intercept



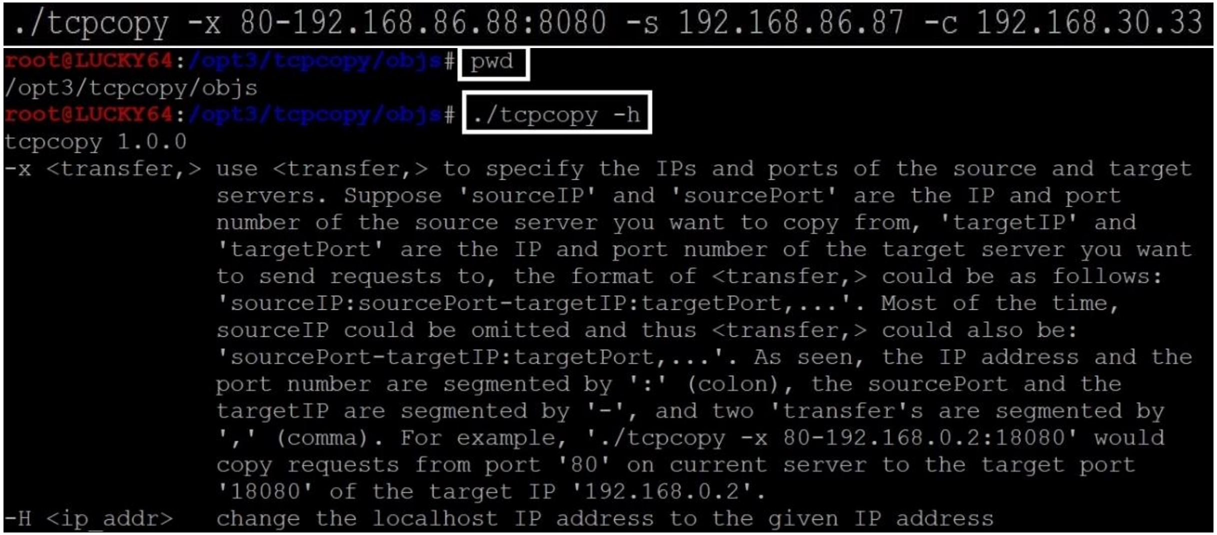
```
./intercept -i eth1 -F "src port 8080" -d
```



```
root@kali2017:~/intercept/objs# pwd
/root/intercept/objs
root@kali2017:~/intercept/objs# ./intercept -h
intercept 1.0.0
-i <device,>   The name of the interface to listen on.  This is usually a driver
               name followed by a unit number,for example eth0 for the first
               Ethernet interface.
-F <filter>    user filter(same as pcap filter)
-n <num>       set the maximal num of combined packets.
-p <num>       set the TCP port number to listen on. The default number is 36524.
-s <num>       set the hash table size for intercept. The default value is 65536.
-D <transfer>  use <transfer> to specify the dockered_ip and orig_ip
               which are segmented by '-'.
-l <file>      save log information in <file>
-P <file>      save PID in <file>, only used with -d option
-b <ip_addr>   interface to listen on (default: INADDR_ANY, all addresses)
-v             intercept version
-h             print this help and exit
-d             run as a daemon
```

Running TCPCopy

- On the online source server

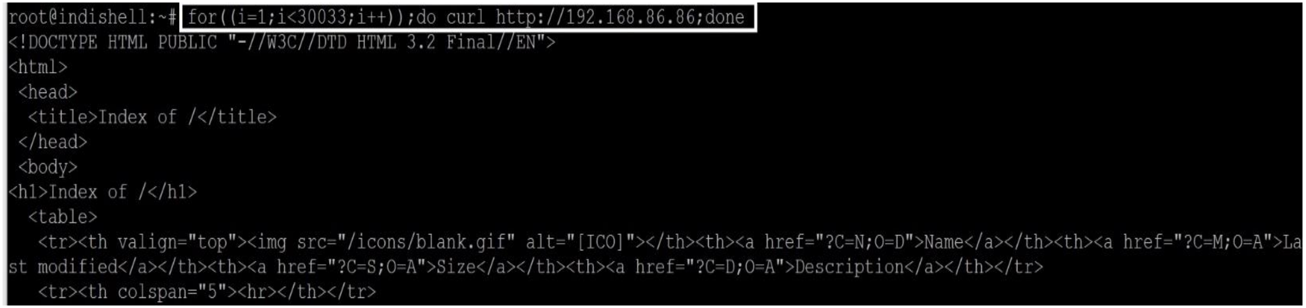


```
./tcpcopy -x 80-192.168.86.88:8080 -s 192.168.86.87 -c 192.168.30.33
```

```
root@LUCKY64:/opt3/tcpcopy/objs# pwd
/opt3/tcpcopy/objs
root@LUCKY64:/opt3/tcpcopy/objs# ./tcpcopy -h
tcpcopy 1.0.0
-x <transfer,> use <transfer,> to specify the IPs and ports of the source and target
               servers. Suppose 'sourceIP' and 'sourcePort' are the IP and port
               number of the source server you want to copy from, 'targetIP' and
               'targetPort' are the IP and port number of the target server you want
               to send requests to, the format of <transfer,> could be as follows:
               'sourceIP:sourcePort-targetIP:targetPort,...'. Most of the time,
               sourceIP could be omitted and thus <transfer,> could also be:
               'sourcePort-targetIP:targetPort,...'. As seen, the IP address and the
               port number are segmented by ':' (colon), the sourcePort and the
               targetIP are segmented by '-', and two 'transfer's are segmented by
               ',' (comma). For example, './tcpcopy -x 80-192.168.0.2:18080' would
               copy requests from port '80' on current server to the target port
               '18080' of the target IP '192.168.0.2'.
-H <ip_addr>   change the localhost IP address to the given IP address
```

Running TCPCopy

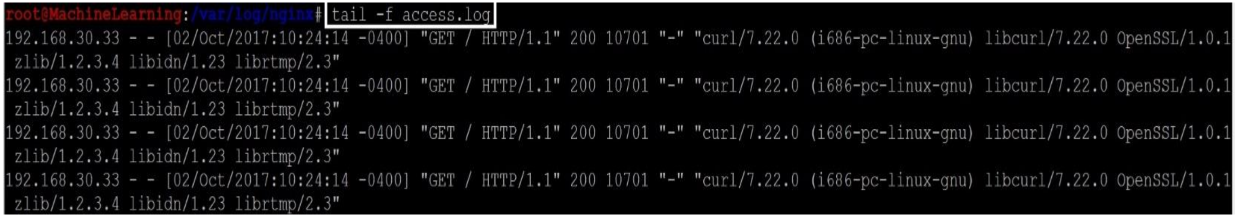
- Generating traffic from the client to the online Apache server



```
root@indishell:~# for((i=1;i<30033;i++));do curl http://192.168.86.86;done
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /</title>
 </head>
 <body>
<h1>Index of /</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">La
st modified</a></th><th><a href="?C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
```

Running TCPCopy

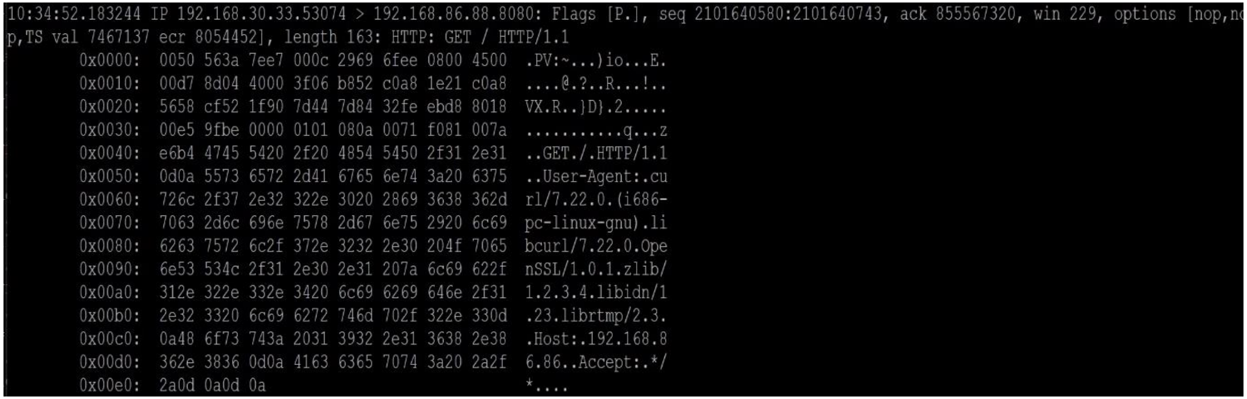
- While Apache is running online, the TCP flows are copied from Apache to Nginx. The TCP flows are just forwarded to Nginx



```
root@MachineLearning:/var/log/nginx# tail -f access.log
192.168.30.33 - - [02/Oct/2017:10:24:14 -0400] "GET / HTTP/1.1" 200 10701 "-" "curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1
 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"
192.168.30.33 - - [02/Oct/2017:10:24:14 -0400] "GET / HTTP/1.1" 200 10701 "-" "curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1
 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"
192.168.30.33 - - [02/Oct/2017:10:24:14 -0400] "GET / HTTP/1.1" 200 10701 "-" "curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1
 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"
192.168.30.33 - - [02/Oct/2017:10:24:14 -0400] "GET / HTTP/1.1" 200 10701 "-" "curl/7.22.0 (i686-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1
 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"
```

Running TCPCopy

- While Apache is running online, the TCP flows are copied from Apache to Nginx. The TCP flows are just forwarded to Nginx



```
10:34:52.183244 IP 192.168.30.33.53074 > 192.168.86.88.8080: Flags [P.], seq 2101640580:2101640743, ack 855567320, win 229, options [nop,no
p,TS val 7467137 ecr 8054452], length 163: HTTP: GET / HTTP/1.1
        0x0000:  0050 563a 7ee7 000c 2969 6fee 0800 4500  .PV:~...)io...E.
        0x0010:  00d7 8d04 4000 3f06 b852 c0a8 1e21 c0a8  ....@.?..R...!..
        0x0020:  5658 cf52 1f90 7d44 7d84 32fe ebd8 8018  VX.R..}D}.2.....
        0x0030:  00e5 9fbe 0000 0101 080a 0071 f081 007a  ...........q...z
        0x0040:  e6b4 4745 5420 2f20 4854 5450 2f31 2e31  ..GET./.HTTP/1.1
        0x0050:  0d0a 5573 6572 2d41 6765 6e74 3a20 6375  ..User-Agent:.cu
        0x0060:  726c 2f37 2e32 322e 3020 2869 3638 362d  rl/7.22.0.(i686-
        0x0070:  7063 2d6c 696e 7578 2d67 6e75 2920 6c69  pc-linux-gnu).li
        0x0080:  6263 7572 6c2f 372e 3232 2e30 204f 7065  bcurl/7.22.0.Ope
        0x0090:  6e53 534c 2f31 2e30 2e31 207a 6c69 622f  nSSL/1.0.1.zlib/
        0x00a0:  312e 322e 332e 3420 6c69 6269 646e 2f31  1.2.3.4.libidn/1
        0x00b0:  2e32 3320 6c69 6272 746d 702f 322e 330d  .23.librtmp/2.3.
        0x00c0:  0a48 6f73 743a 2031 3932 2e31 3638 2e38  .Host:.192.168.8
        0x00d0:  362e 3836 0d0a 4163 6365 7074 3a20 2a2f  6.86..Accept:.*/
        0x00e0:  2a0d 0a0d 0a                             *....
```

References

- Kitploit
http://www.kitploit.com/2017/09/tcpcopy-tcp-stream-replay-tool.html

- Kali Linux
https://www.kali.org/downloads/