Installation Guide

Server:

- Make sure you have python3
- Make sure “server_pattern.txt” and “server.py” in the same folder
- Then go to server folder and run: python3 server.py -p [your port]

Fuzzer:

- You need a VM running Ubuntu 18.x and running as root.
- Install scapy using “pip3 install scapy”
- Set up your iptables:
  iptables -A OUTPUT -p tcp --sport [choose your port] --tcp-flags RST RST -j DROP
- Go to “fuzzer” folder, make sure you have all these file in the same folder:

__init__.py
Fuzzer.py
TcpSession.py
IPFuzzer.py
TCPFuzzer.py
APPFuzzer.py
default_payload.txt
IP_samples.txt
TCP_samples.txt
APP_samples.txt

User Guide

Server:

- Go to the server folder
- Run: python3 server.py -p [your port]
- Press CTRL-C to stop it.
- “python3 server.py —help” to print help information
- You can specify the pattern in server_pattern.txt, the pattern should be written in one single line, and the format is “AA BB CC”

Fuzzer:

- Go to fuzzer folder
- To run the fuzzer:
  python3 Fuzzer.py --src {source ip} --dst {target ip} --sport {source port} --dport {target port} -l {choose your layer}
  These 5 arguments are required, you should specify ip and port here, and choose one layer to fuzz. See other optional arguments in the following session.

- **IP Fuzzing:**
  Fuzzed fields are {‘version', 'ihl', 'tos', 'len', 'id', 'flags', 'frag', 'ttl', ‘proto’}.

  (1)command to start a IP fuzzing :
  python3 Fuzzer.py --src {source ip} --dst {target ip} --sport {source port} --dport {target port} -l IP

  (2)Optional arguments:
  -t {field} : this specify a target field in default fuzzing, without this argument, the fuzzer will fuzz every field.
  -n {number}: this specify a number of tests to run in default fuzzing, when this is specified, the fuzzer will randomly generate value for field(s), otherwise it will try every possible value.
  -f {filename}: this specify the file that stores a series of tests. If this is given, the fuzzer will run fuzzing from this file.
  Examples:
  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l IP : this runs default fuzzing for every field and every possible value

  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l IP -t version: this runs default fuzzing for version field, using every possible value.

  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l IP - t version -n 10: this runs default fuzzing for version field and generates 10 random value to fuzz.

  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l IP -f IP_samples.txt: this runs tests from IP_samples.txt file.

  (3)Sample File:
  One single line in the sample file is the field:value pairs for one packet, the format should be version:04, ihl:05, tos:ff, len:ff, id:ff, flags:e, frag:f, ttl:fe, proto:fe

- **TCP fuzzing**
  Fuzzed fields are {'seq', 'ack', 'flags', 'window', 'urgptr', 'dataofs', ‘reserved'}.

  (1)command to start a TCP fuzzing:
  python3 Fuzzer.py --src {source ip} --dst {target ip} --sport {source port} --dport {target port} -l TCP

  (2) optional arguments:
  -t {field} : this specify a target field in default fuzzing, without this argument, the fuzzer will fuzz every field.
  -n {number}: this specify a number of tests to run in default fuzzing, when this is specified, the fuzzer will randomly generate value for field(s), otherwise it will try every possible value. (however, seq and ack is 32 bit long, generate every possible value is slow, so it will use a

large step in the for loop to generate less number of values)
-f {filename}: this specify the file that stores a series of tests. If this is given, the fuzzer will run fuzzing from this file.

Examples:

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l TCP : this runs default fuzzing for every field and every possible value

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l TCP -t flags: this runs default fuzzing for flags field, using every possible value.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l TCP - t flags -n 10: this runs default fuzzing for flags field and generates 10 random value to fuzz.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l TCP -n 5: this generates 5 random value for each field and then run the default fuzzing.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l TCP -f TCP_samples.txt: this runs tests from TCP_samples.txt file.

(3)Sample File:

One single line in the sample file is the field:value pairs for one packet, the format should be seq:02, ack:ff, flags:ff, window:fefe, urgptr:fefe, dataofs:0f, reserved:0f

- **Application fuzzing**

  (1)command to start application layer fuzzing:

  python3 Fuzzer.py --src {source ip} --dst {target ip} --sport {source port} --dport {target port} -l APP

  (2) optional arguments:

  -n {number}: this specifies how many payloads to fuzz, if not specified, defaulted to 10
  -s {size}: this specifies the size of each payload. It can be a single integer(like, 5) means the payload size is fixed. It can be a range(like, 5-10), means the payload size is variable and will be in that range. If not specified, defaulted to 10.
  -f {filename}: this specify the file that stores a series of tests. If this is given, the fuzzer will run fuzzing from this file.

  Examples:

  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l APP : this runs default fuzzing, fuzz 10 payload, each payload size is 10 bytes, each byte is generated randomly.

  python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l APP -n 5: this runs default fuzzing. 5 payloads will be fuzzed, each payload is 10 bytes, each byte is generated randomly.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l APP -n 5 -s 3: This runs default fuzzing. 5 payloads will be fuzzed, each payload is 3 bytes, each byte is generated randomly.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l APP -s 3-10: This runs default fuzzing. 10 payloads will be fuzzed, each payload size is between [3, 10] each byte is generated randomly.

python3 Fuzzer.py --src 1.2.3.4 --dst 4.3.2.1 —sport 7890 --dport 9999 -l APP -f APP_samples.txt: this runs fuzzing from APP_samples.txt

After finishing fuzzing, it will show how many payloads are fuzzed, and the valid counts and invalid counts of fuzzed payloads.

(3)Sample File:

One single line in the sample file is one payload in hex form, the format should be
AA AA AA BB BB CC CC
Considering the max length of a packet, the size of one payload should not be larger than 1000 bytes.