NAME

netfuzz.rules — Network Traffic Fuzzing Rules File

DESCRIPTION

The netfuzz(4) Netfuzz modifies packets according to rules specified in **netfuzz.rules**.

The rules are traversed one by one until both the packet filter and the probability matches. When a rule match, the packet is modified according to the rule and written to the network. No more rules are traversed after a match, so a single rule modifies the packet before it is written to the network.

GRAMMAR

Syntax for **netfuzz.rules** in EBNF:

```
<ruleset>       ::= <ruleset> '\n'
                  | <ruleset> <include> '\n'
                  | <ruleset> <varset> '\n'
                  | <ruleset> <drop> '\n'
                  | <ruleset> <dup> '\n'
                  | <ruleset> <tcp-kill> '\n'
                  | <ruleset> <fuzz> '\n'
<include>       ::= 'include' <file-path>
<varset>        ::= <identifier> '=' <value>
<fuzz>          ::= 'fuzz' <interface-name> <filter> <probability> <offset> <fuzz-
<drop>          ::= 'drop' <interface-name> <filter> <probability>
<dup>           ::= 'dup' <interface-name> <filter> <probability>
<tcp-kill>      ::= 'tcp-kill' <interface-name> <filter> <probability>
<filter>        ::= 'filter' <bpf-filter-string>
<probability>   ::= 'probability' <unsigned-number>
<offset>        ::= 'offset-start' <offval> 'offset-end' <offval>
<offval>        ::= <unsigned-number>
                  | 'ip-header'
                  | 'ip-payload'
                  | 'tcp-header'
                  | 'udp-header'
                  | 'payload'
                  | 'packet-end'
<fuzz-rule>     ::= 'rule' 'bitflip' <min-max>
                  | 'rule' 'bytewrite' <min-max> 'value' <hex-byte>
                  | 'rule' 'bytereplace' <min-max> 'old' <hex-byte> 'new' <hex-byte>
<min-max>       ::= 'min' <unsigned-number> 'max' <unsigned-number>
```

EXAMPLE RULES FILE

```
# Netfuzz rules.
# The first rule that matches probability modifies the
# outgoing packet and the rest of the rules
# are ignored. Note that you can have multiple
# rules with the same packet filter and change
# the probability to spread the rule used.
#

# include a file with rules and/or macros
include "/dev/null"
```

```
# Set up some macros, in this case
# some packet filter strings.
finger_client = "\"tcp dst port 79\""
ftp_client = "\"tcp dst port 21\""

# Kill TCP connection by sending RST in both directions
# Avoid handshake packets
tcp-kill all filter \
       "tcp port 79 and not (tcp[tcpflags] & (tcp-syn) != 0)" \
       probability 10000


# Write a maximum of 3 A's to the TCP payload
# on approximately every fifth packet that matches the filter
fuzz em1 filter $finger_client \
        probability 5 \
        offset-start payload \
        offset-end packet-end \
        rule bytewrite min 1 max 3 value 0x41

# Replace a maximum of 10 'A's with 'B's
# on approximately every eleventh packet that matches the filter
fuzz em1 filter $ftp_client \
        probability 11 \
        offset-start payload \
        offset-end packet-end  \
        rule bytereplace min 1 max 10 old 0x41 new 0x42

# Fuzz 1 to 3 bits in approximately every 100 IPv6 header
# on all interfaces
fuzz all filter "ip6" probability 100 \
        offset-start ip-header \
        offset-end ip-payload \
        rule bitflip min 1 max 3

# Drop icmp echo request packet
drop all filter "icmp[icmptype] = icmp-echo" probability 5

drop all filter "icmp6 and ip6[40]=128" probability 5

# Duplicate icmp echo request packet
dup all filter "icmp[icmptype] = icmp-echo" probability 5

dup all filter "icmp6 and ip6[40]=128" probability 5
```

FILES

```
/etc/hosts        Host name database.
```

```
/etc/netfuzz.rules
                Default location of the ruleset file.
/etc/protocols  Protocol name database.
/etc/services   Service name database.
```

SEE ALSO

netfuzz(4), netfuzzctl(8),

HISTORY

This is the second generation of the network fuzzing suite of programs that was first implemented as a kernel patch on OpenBSD

AUTHORS