CSE 508 Course Project Presentation

nftables on Android

Team Members
Subhasis Das
Gaurav Piyush
Chaitanya Saxena

Tasks Attempted

- 60 Points
  - Port nftables
  - Port the user space libraries in android
  - Port required kernel modules (apply patches)
  - Table and Input chains
  - Add / delete rules
  - Drop packets/Block traffic
- 20 Points : Redirection and Source Spoofing
  - Using NAT module
  - Create NAT table and new NAT chain
  - Add NAT rules
  - Testing
- Packet Selector
  - Matching Transport protocol , IPV4/IPV6 Headers
  - Matching TCP/UDP/UDPlite traffic
  - Matching Sender/Receiver
  - Matching packet meta information

nftables

- Common platform for iptables, etables, ip6table, arptable (aims to replace them)
- Protocol dependency in user space.
- Core common hooks in kernel space.
- Common language for rule generation and parsing.

Goldfish

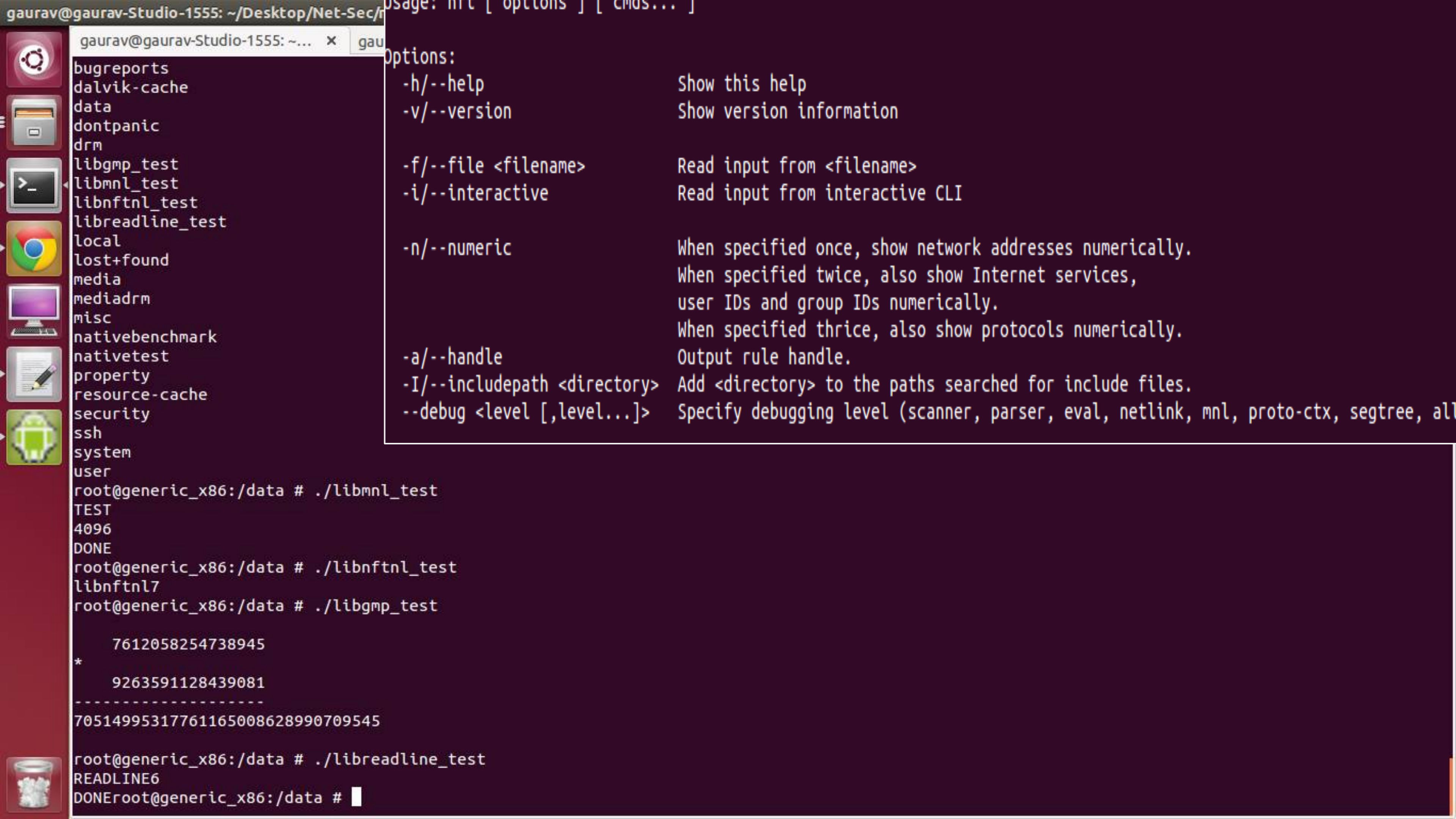
- Android OS : Built upon linux kernel 3.4
- Does not support nftables
- The Android emulator runs a virtual CPU that Google calls Goldfish. Goldfish executes ARM926T instructions and has hooks for input and output ([2])
- Kernel Version 3.10

User Space Libraries

Following user level modules were successfully ported to the kernel :

libmnl | libnftnl | libgmp | libreadline | libncurses | nftable

- Set up the proper environment path
- Check cross compilation
- Test each library by using a user program


gaurav@gaurav-Studio-1555: ~/Desktop/Net-Sec/r
gaurav@gaurav-Studio-1555: ~...
gau

```
sage: nft [ options ] [ cmds... ]

Options:
  -h/--help                      Show this help
  -v/--version                   Show version information

  -f/--file <filename>           Read input from <filename>
  -i/--interactive               Read input from interactive CLI

  -n/--numeric                   When specified once, show network addresses numerically.
                                 When specified twice, also show Internet services,
                                 user IDs and group IDs numerically.
                                 When specified thrice, also show protocols numerically.
  -a/--handle                    Output rule handle.
  -I/--includepath <directory>   Add <directory> to the paths searched for include files.
  --debug <level [,level...]>    Specify debugging level (scanner, parser, eval, netlink, mnl, proto-ctx, segtree, all
```

```
bugreports
dalvik-cache
data
dontpanic
drm
libgmp_test
libmnl_test
libnftnl_test
libreadline_test
local
lost+found
media
mediadrm
misc
nativebenchmark
nativetest
property
resource-cache
security
ssh
system
user
root@generic_x86:/data # ./libmnl_test
TEST
4096
DONE
root@generic_x86:/data # ./libnftnl_test
libnftnl7
root@generic_x86:/data # ./libgmp_test

    7612058254738945
*
    9263591128439081
--------------------
70514995317761165008628990709545

root@generic_x86:/data # ./libreadline_test
READLINE6
DONEroot@generic_x86:/data #
```

Kernel Space Patches

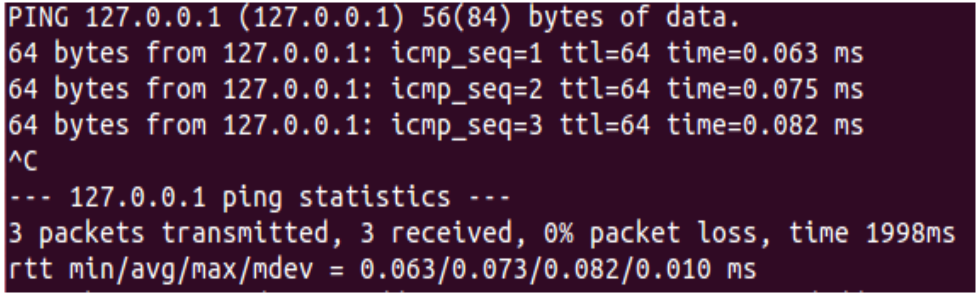
1. Patch 1 - 96518518cc417bb0a8c80b9fb736202e28acdf96
   a. Core implementation for nftables in kernel space
   b. Storage of rule list per chain - new private data pointer

2. Patch 2 - f59cb0453cd885736daa11ae2445982c5ab2fc83
   a. Creation of common module remove duplication of code for iptable and nftable (nat_decode_session, alloc_null_binding)

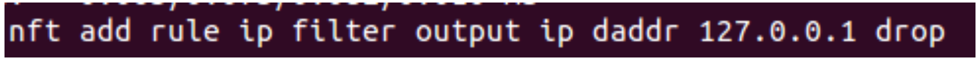
3. Patch 3 - 795aa6ef6a1aba99050735eadd0c2341b789b53b
   a. The user space nftable utility communicates to the kernel space through hooks. This patch creates a generic hook function consisting of common hook functions

4. Patch 4 - 20a69341f2d00cd042e81c82289fba8a13c05a25
   a. Defines nftable sets, different from rule sets.
   b. Defines operations like creation, deletion, lookup etc. on sets.
   c. Defines lockless operation on sets if defined as a constant (not allowed to change when a rule is linked).

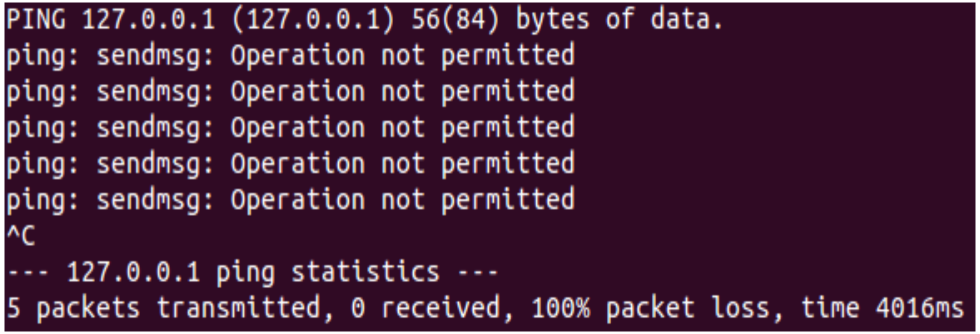
NFT rule to drop packet



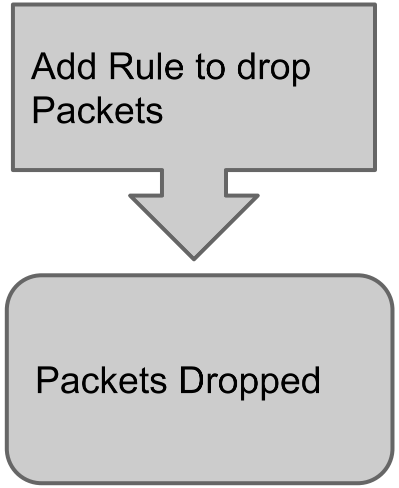
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.063 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.075 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.082 ms
^C
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.063/0.073/0.082/0.010 ms
```



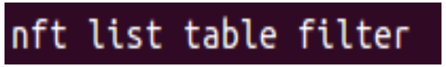
```
nft add rule ip filter output ip daddr 127.0.0.1 drop
```



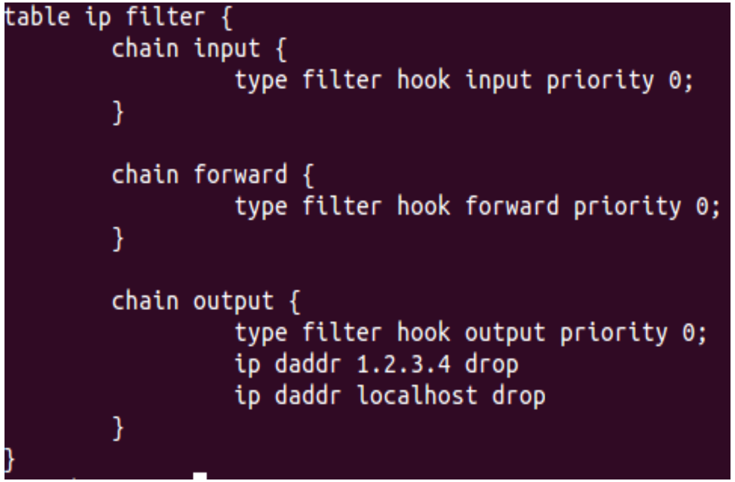
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 127.0.0.1 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4016ms
```



Rule Chain



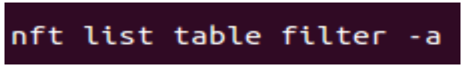
```
nft list table filter
```



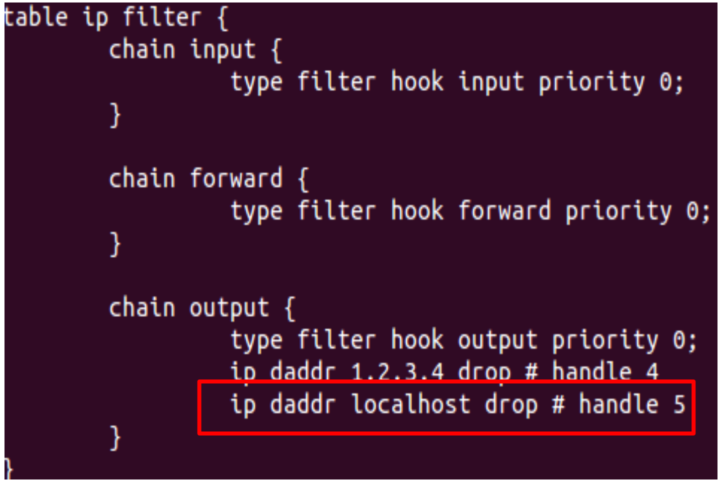
```
table ip filter {
        chain input {
                type filter hook input priority 0;
        }

        chain forward {
                type filter hook forward priority 0;
        }

        chain output {
                type filter hook output priority 0;
                ip daddr 1.2.3.4 drop
                ip daddr localhost drop
        }
}
```



```
nft list table filter -a
```

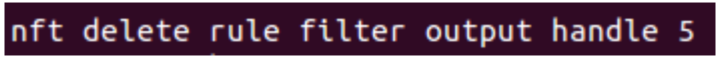


```
table ip filter {
        chain input {
                type filter hook input priority 0;
        }

        chain forward {
                type filter hook forward priority 0;
        }

        chain output {
                type filter hook output priority 0;
                ip daddr 1.2.3.4 drop # handle 4
                ip daddr localhost drop # handle 5
        }
}
```

Next : Delete Rule



```
nft delete rule filter output handle 5
```

NFT more rules to drop packets

- Add rule to a network :

  ```
  nft add rule ip filter output ip daddr 172.24.241.0/24 counter
  ```

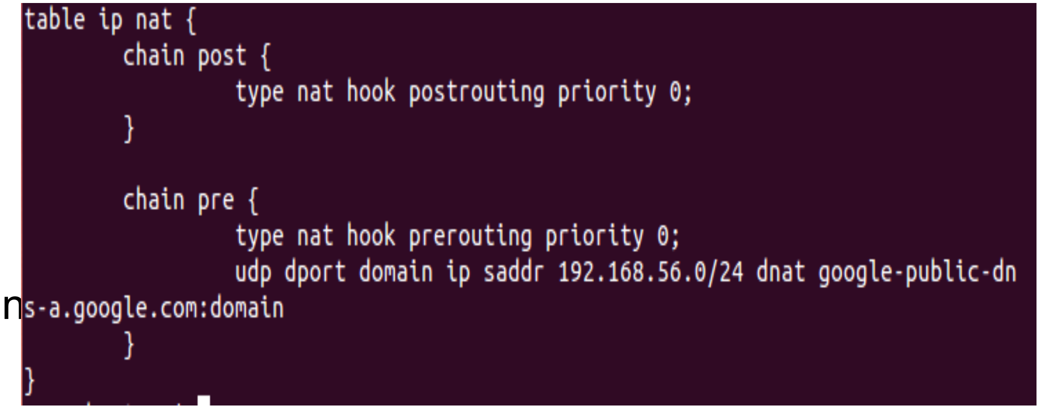
- Add rule to a port 80

  ```
  nft add rule ip filter input tcp dport 80 drop
  ```

- A combined rule (filters ICMP and drops O/P to destination)

  ```
  nft add rule ip filter output ip protocol icmp  ip daddr 127.0.0.1 counter drop
  ```

Redirection



```
table ip nat {
        chain post {
                type nat hook postrouting priority 0;
        }

        chain pre {
                type nat hook prerouting priority 0;
                udp dport domain ip saddr 192.168.56.0/24 dnat google-public-dn
s-a.google.com:domain
        }
}
```

- Port nat module
- Step 1: Make kernel aware of NAT

  modprobe nft_chain_nat_ipv4

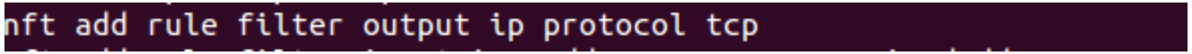
- Step 2: Create NAT dedicated chain
  - sudo nft add table nat
  - sudo nft add chain nat post \{ type nat hook postrouting priority 0 \; \}
  - sudo nft add chain nat pre \{ type nat hook prerouting priority 0 \; \}
- Step 3 : Add some nat rules

  nft add rule nat pre udp dport 53 ip saddr 192.168.56.0/24 dnat 8.8.8.8:53

  (Redirects all DNS trafic from 192.168.56.0/24 to the 8.8.8.8(Google Public DNS))

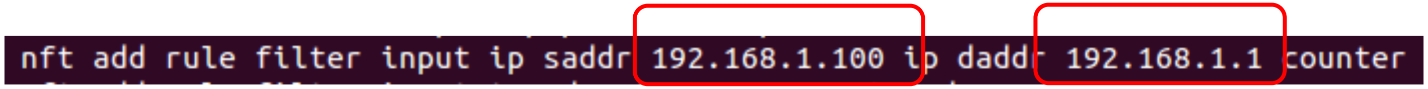
Packet Selectors and actions

- Matching Transport protocol



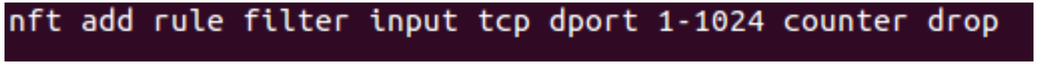
```
nft add rule filter output ip protocol tcp
```

- Matching IPV4 heading : Sender and Receiver



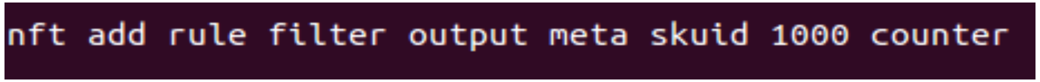
```
nft add rule filter input ip saddr 192.168.1.100 ip daddr 192.168.1.1 counter
```

- Matching TCP traffic : matches and drops all tcp traffic for low TCP ports (1-1024)



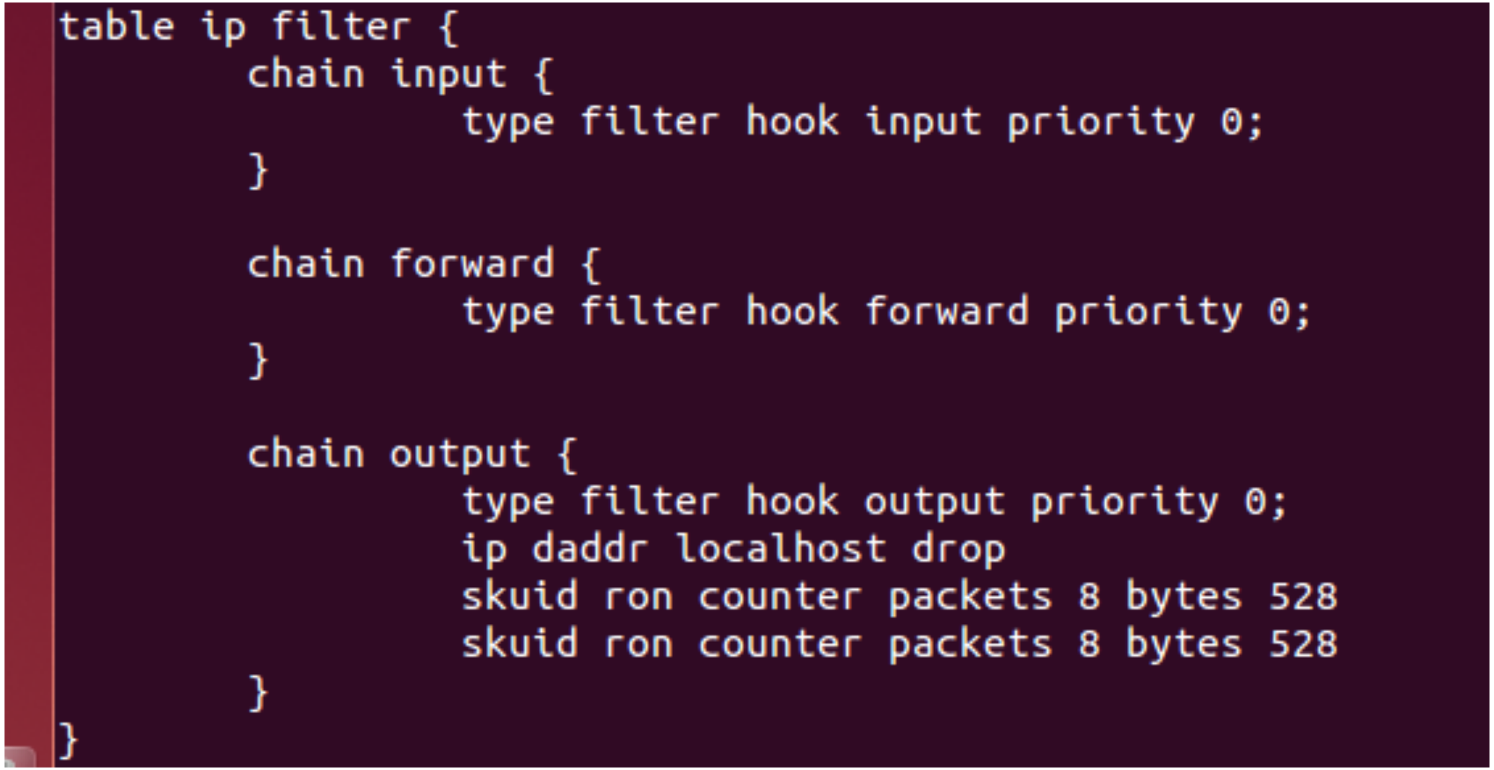
```
nft add rule filter input tcp dport 1-1024 counter drop
```

- Matching traffic based on user name



```
nft add rule filter output meta skuid 1000 counter
```

Packet Selectors and Action



```
table ip filter {
        chain input {
                type filter hook input priority 0;
        }

        chain forward {
                type filter hook forward priority 0;
        }

        chain output {
                type filter hook output priority 0;
                ip daddr localhost drop
                skuid ron counter packets 8 bytes 528
                skuid ron counter packets 8 bytes 528
        }
}
```

Challenges Faced

- No proper documentation available about nftable porting
- Cross compilation issues
- Locating and adding kernel patches
- Running the emulator
- Running internet on emulator

References

1. http://en.wikipedia.org/wiki/Nftables : Netfilter Introdcution
2. https://groups.google.com/forum/#!topic/android-kernel/M4SjXulUeUo : Goldfish
3. https://wiki.archlinux.org/index.php/Nftables : A good basic documentation on usage and design of nftables.
4. https://github.com/sam8dec/NetSec : We have referred to this excellent write up by Samudra for our initial setup.
5. http://en.wikipedia.org/wiki/GNU_Multiple_Precision_Arithmetic_Library#Example : Excellent reference for writing test cases
6. Source for nfnetlink_compat.h : https://git.netfilter.org/libnetfilter_acct/tree/include/linux/netfilter.
7. http://kernelnewbies.org/nftables_examples : Excellent examples of NAT rule handling
8. Links to patches: [Patch 1] [Patch 2] [Patch 3] [Patch 4]