Netfilter & Packet Dropping

- Netfilter provides a set of hooks in several points of the kernel network stack.
- The hooks can be exploited to define custom functions for manipulating IP packets
  - Dropping
  - Manipulation of header fields
  - Etc.
- The hooks are triggered by the kernel after the execution of the functions that implement the network procedures
An incoming IP packet travels in the kernel following a path.
Netfilter Architecture

- Kernel path for incoming packets
  1. Sanity checks (i.e., not truncated, IP checksum OK, etc)
  2. Routing decision (it decides whether the packet is destined for another interface, or a local process)
     - Local process: the netfilter framework is called again for the NF_IP_LOCAL_IN hook
     - Another interface: the netfilter framework is called for the NF_IP_FORWARD hook
  3. Final step (the packet passes a final netfilter hook, the NF_IP_POST_ROUTING hook)
Netfilter Architecture

- When a hook is triggered, a customized function can manipulate the packet content.
- Kernel modules can register to listen at any of the hooks described in the previous slide.
- After manipulating a packet, the module returns a code to the calling function:
  - NF_ACCEPT: continue traversal as normal
  - NF_DROP: drop the packet; don't continue traversal
  - NF_STOLEN: stole the packet from the path
  - NF_QUEUE: queue the packet (for userspace handling)
  - NF_REPEAT: call this hook again
• The iptables tool has been developed over the netfilter framework
• Kernel modules can register a new table, and ask for a packet to traverse a given table
• Hooks registered with netfilter

<table>
<thead>
<tr>
<th>Prerouting</th>
<th>Forwarding</th>
<th>Postrouting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conntrack</td>
<td>Mangle</td>
<td>Mangle</td>
</tr>
<tr>
<td>Mangle</td>
<td></td>
<td>Src NAT</td>
</tr>
<tr>
<td>Dst NAT</td>
<td>Filter</td>
<td>Conntrack</td>
</tr>
<tr>
<td>QDisc</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Registration of filtering functions

- Structure containing the function handle:
  ```c
  static struct nf_hook_ops netfilter_ops_pre;
  ```
- Customized attributes of the hook:
  ```c
  netfilter_ops_pre.hook = hook_pre_routing;
  netfilter_ops_pre.pf = PF_INET;
  netfilter_ops_pre.hooknum = NF_INET_PRE_ROUTING;
  netfilter_ops_pre.priority = NF_IP_PRI_FIRST;
  ```

Registration of the hook
```
ret = nf_register_hook(&netfilter_ops_pre);
```
Registration of filtering functions

- hook_pre_routing is the function implementing the packet filtering
- PF_INET: Internet Protocol Family
- NF_INET_PRE_ROUTING: the function is triggered before the routing decision
- NF_IP_PRI_FIRST: the registered function has the highest priority of execution
- The unregistration is performed using the following function:
  
  ```c
  nf_unregister_hook(&netfilter_ops_pre);
  ```
The kernel module implementing the filtering function needs to be cross-compiled for the um architecture.

The Makefile is very similar to the Makefile used to compile kernel modules.

In addition to indicating the directory which contains the headers and the objects of the Netkit kernel, it is necessary to define the architectures of the host and target machines:

- ARCH=um
- SUBARCH=i386
Makefile for the pkt_drop module

obj-m += pkt_drop.o

KERNELPATH="path/to/kernel/src"

all:

    make -C $(KERNELPATH) M=$(shell pwd) ARCH=um SUBARCH=i386 modules

clean:

    make -C $(KERNELPATH) M=$(shell pwd) ARCH=um SUBARCH=i386 clean
Packet dropping:

- A simple Linux kernel module which defines a function that drops data packets before performing the routing decision.
- The function is registered as a PREROUTING hook.
- Note that some of the auxiliary functions defined by the kernel to access the header fields may NOT work.

See the code pkt_drop.c
Example

- The command **insmod** is usually used to load kernel modules
  
  ```bash
  insmod pkt_drop.ko drop_deg=5
  (5 out of 10 ICMP echo reqs will be discarded)
  ```

- The command **rmmod** is usually used to unload kernel modules
  
  ```bash
  rmmod pkt_drop
  ```

- modprobe is an alternative command to load and unload modules
  
  ```bash
  modprobe -i pkt_drop.ko drop_deg=5
  modprobe -r pkt_drop.ko
  ```
Netfilter Queue Subsystem

- Netfilter system provides a special target NFQUEUE used to queue packets to user-space programs
- Netfilter provides up to $2^{16}$ queues
- An user space programs can bind to one or more queues using the libnetfilter_queue library which provides the features below:
  - receiving queued packets from the kernel nfnetlink_queue subsystem
  - issuing verdicts and/or reinjecting altered packets to the kernel nfnetlink_queue subsystem
NFQueue Example

- See nfqueue lab (nfqueue.tar.gz)
- An iptable rule send all the incoming ICMP packets to the queue 0:
  ```
  iptables -t mangle -A PREROUTING -i eth0 -p icmp -j NFQUEUE --queue-num 0
  ```
- The application nfql_test get the packets from queue 0 and print some info about them:
  ```
  hw_protocol=0x0800 hook=0 id=1
  hw_src_addr=aa:a4:c2:90:2d:bd indev=3
  payload_len=84
  ```