Certifying Spoofing-Protection

Does your firewall feature spoofing protection? Check with our algorithm.

- **Formally Verified**: Machine-verifiably proven sound.
- **Real-World**: Supports the largest subset of iptables features compared to any other firewall analysis systems.
- **Tested**: Discovered errors on the largest publicly-available firewall ever analyzed in academia.
- **Fast**: Processes thousands of rules in less than a minute.

```
-i eth0 --src 192.168.0.0/24 -j DROP
-j ACCEPT
```

Spoofing protection

```
-p tcp -m recent --hitcount 41 -j LOGDROP
-i eth0 --src 192.168.0.0/24 DROP
-m future_feature -j ACCEPT
```

Spoofing protection

```
-m future_feature -j ACCEPT
-i eth0 --src 192.168.0.0/24 -j DROP
-j ACCEPT
```

Probably no spoofing protection

Formal Verification

Both the algorithm and ruleset preprocessing are machine-verifiably proven sound with the Isabelle proof assistant.

1. Semantics-preserving rewriting and abstracting over unknown features.
   - Computes a ruleset which accepts at least all the packets the original ruleset would accept.
2. Sound spoofing protection.
   - Verifies whether this more permissive ruleset blocks all potentially spoofed packets.

Using Isabelle’s code generation feature, a stand-alone Haskell tool is derived from the theory.

Understanding Real-World Firewall Rulesets

The semantics features matching with arbitrary oracles. The definition is not executable.

```
Skip
\rho \triangleright \{ [], \tau \} \Rightarrow \tau

match, m, p
\rho \triangleright \{ [[(m, \text{Accept})], \emptyset] \} \Rightarrow \emptyset

Drop
\rho \triangleright \{ [[(m, \text{Drop})], \emptyset] \} \Rightarrow \emptyset

-match, m, p
\rho \triangleright \{ [[(m, \text{Drop})], \emptyset] \} \Rightarrow \emptyset

Reject
\rho \triangleright \{ [[(m, \text{Reject})], \emptyset] \} \Rightarrow \emptyset

NoMatch
\rho \triangleright \{ [[(m, \text{Drop})], \emptyset] \} \Rightarrow \emptyset

Seq
\rho \triangleright \{ \tau_1, \emptyset \} \Rightarrow \tau
\rho \triangleright \{ \tau_2, \tau \} \Rightarrow \tau'

CallResult
\rho \triangleright \{ \Gamma, c, \emptyset \} \Rightarrow \tau
\rho \triangleright \{ [[(m, \text{Call} c)], \emptyset] \} \Rightarrow \tau

CallReturn
\rho \triangleright \{ [(m, \text{Call} c)], \emptyset \} \Rightarrow \emptyset
\rho \triangleright \{ [(m, \text{Call} c)], \emptyset \} \Rightarrow \emptyset

Log
\rho \triangleright \{ [[(m, \text{Log})], \emptyset] \} \Rightarrow \emptyset

Empty
\rho \triangleright \{ [[(m, \text{Empty})], \emptyset] \} \Rightarrow \emptyset
```

for any primitive matcher \( \gamma \) and any well-formed ruleset \( \Gamma \)

Easy to Use

adm@fw# iptables-save | ./check ipasmt.txt
preprocessing ruleset
sanity checking ipasmt
checking spoofing protection:
eth1.96 True
eth1.109 False
...  

[time] real 0m38.439s

Open Source

Iptables firewall ruleset collection:  
https://github.com/diekmann/net-network
Isabelle formalization and tool:  
https://github.com/diekmann/Iptables_Semantics