Firewall Analysis

Enforcing BCP 38

Technische Universität München

Chair for Network Architectures and Services

C. Diekmann, L. Schwaighofer, and G. Carle
{diekmann | schwaighofer | carle}@net.in.tum.de

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
```

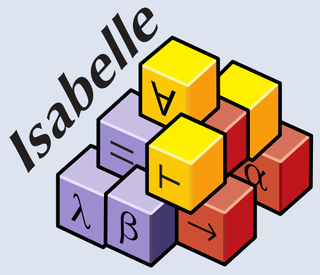
Spoofing protection

```
-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.

$$\text{Skip}\ \frac{}{p \vdash \langle [],\ t\rangle \Rightarrow t} \qquad \text{Accept}\ \frac{\text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \texttt{Accept})],\ ?\rangle \Rightarrow \checkmark}$$

$$\text{Drop}\ \frac{\text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \texttt{Drop})],\ ?\rangle \Rightarrow \times} \qquad \text{Reject}\ \frac{\text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \texttt{Reject})],\ ?\rangle \Rightarrow \times}$$

$$\text{NoMatch}\ \frac{\neg\ \text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ a)],\ ?\rangle \Rightarrow ?} \qquad \text{Decision}\ \frac{t \neq ?}{p \vdash \langle rs,\ t\rangle \Rightarrow t}$$

$$\text{Seq}\ \frac{p \vdash \langle rs_1,\ ?\rangle \Rightarrow t \qquad p \vdash \langle rs_2,\ t\rangle \Rightarrow t'}{p \vdash \langle rs_1 :: rs_2,\ ?\rangle \Rightarrow t'}$$

$$\text{CallResult}\ \frac{\text{match}_\gamma\ m\ p \qquad p \vdash \langle \Gamma\ c,\ ?\rangle \Rightarrow t}{p \vdash \langle [(m,\ \texttt{Call}\ c)],\ ?\rangle \Rightarrow t}$$

$$\text{CallReturn}\ \frac{\text{match}_\gamma\ m\ p \qquad \Gamma\ c = rs_1 :: (m',\ \texttt{Return}) :: rs_2 \qquad \text{match}_\gamma\ m'\ p \qquad p \vdash \langle rs_1,\ ?\rangle \Rightarrow ?}{p \vdash \langle [(m,\ \texttt{Call}\ c)],\ ?\rangle \Rightarrow ?}$$

$$\text{Log}\ \frac{\text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \texttt{Log})],\ ?\rangle \Rightarrow ?} \qquad \text{Empty}\ \frac{\text{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \texttt{Empty})],\ ?\rangle \Rightarrow ?}$$

for any primitive matcher γ and any well-formed ruleset Γ

Easy to Use

```
adm@fw# iptables-save | ./check ipassmt.txt
preprocessing ruleset
sanity checking ipassmt
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

Conference on Network and Service Management, Barcelona, Spain, November 2015