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.

Skip $\dfrac{}{p \vdash \langle [],\ t \rangle \Rightarrow t}$ Accept $\dfrac{\mathrm{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \mathtt{Accept})],\ \textcircled{?} \rangle \Rightarrow \textcircled{\checkmark}}$

Drop $\dfrac{\mathrm{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \mathtt{Drop})],\ \textcircled{?} \rangle \Rightarrow \textcircled{\times}}$ Reject $\dfrac{\mathrm{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \mathtt{Reject})],\ \textcircled{?} \rangle \Rightarrow \textcircled{\times}}$

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

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

CallResult $\dfrac{\mathrm{match}_\gamma\ m\ p \qquad p \vdash \langle \Gamma\ c,\ \textcircled{?} \rangle \Rightarrow t}{p \vdash \langle [(m,\ \mathtt{Call}\ c)],\ \textcircled{?} \rangle \Rightarrow t}$

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

Log $\dfrac{\mathrm{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \mathtt{Log})],\ \textcircled{?} \rangle \Rightarrow \textcircled{?}}$ Empty $\dfrac{\mathrm{match}_\gamma\ m\ p}{p \vdash \langle [(m,\ \mathtt{Empty})],\ \textcircled{?} \rangle \Rightarrow \textcircled{?}}$

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