

Automated Firewall Repair With Example-Based Synthesis



William Hallahan

Ruzica Piskac, Avi Silberschatz, Arvind Swaminathan, and Ennan Zhai
Yale University

1. Motivation

- Firewalls are critical for network security and management
- Modern firewalls are becoming increasingly large and complex
- If a firewall does contain some error, and needs to be repaired, it can be difficult to tell if the repair has unwanted side effects

Our Goal:

Given a faulty firewall script, and examples of packets and the actions that the firewall should perform on them, automatically repair the firewall without side effects.

iptables script

```
iptables -N myFw
iptables -F myFw

iptables -A myFw -s 199.83.128.227 -j DROP
iptables -A myFw -p tcp -dport 22 -j ACCEPT
```

specification language

```
myFw = (source_ip = 199.83.128.227 => DROP,
        protocol = tcp AND destination_port = 22
        => ACCEPT)
```

example

```
source_ip = 199.83.127.227 AND
destination_port = 22 AND
(protocol = TCP OR protocol = UDP)
=> ACCEPT
```

Synthesizer

2. Synthesis Process

- Automatically convert the firewall script to a specification language
- Allows for easier reasoning for multiple firewall scripting languages
- Take examples of packets that the firewall's behavior should be adjusted on
- Output of synthesis is specification language, which can be automatically converted back to a firewall script

Repaired specification language

```
myFw = (protocol = tcp AND destination_port = 22 => ACCEPT,
        source_ip = 199.83.128.227 AND protocol = udp
        AND destination_port = 22 => ACCEPT,
        source_ip = 199.83.128.227 => DROP)
```

Repaired iptables script

```
iptables -N myFw
iptables -F myFw

iptables -A myFw -p tcp -dport 22 -j ACCEPT
iptables -A myFw -s 199.83.128.227 -p udp -dport 22 -j ACCEPT
iptables -A myFw -s 199.83.128.227 -j DROP
```

Challenge: Packet Timing and Limits

- Firewalls can have rules that limit the number of times they are matched in a given length of time

```
iptables -A INPUT -p tcp -m limit
--limit 5/second --limit-burst 15 -j ACCEPT
```

- Use token bucket algorithms, which allow for bursts of a large number of packets or continuous streams of a lower number of packets
- We can describe the timing in our model using integer arithmetic
- Given packets with relative (to each other) arrival times, we attempt to synthesize solutions that use the minimal number of limits

```
protocol = 17 AND (time = 0 OR time = 10) => ACCEPT
Protocol = 17 AND time = 20 => DROP
```

```
-p 17 -m limit --limit 2/min --limit-burst 2 -j ACCEPT
-p 17 -j DROP
```

Future Expansions

- Network Address Translation - mapping multiple devices on a private network to a single external IP address

```
iptables -p 16 -j SNAT --to-source 1.2.3.4
```

- Decisions based on previously received packets IP address and ports
 - Represent sets of IP addresses and ports in our SMT model using arrays or uninterpreted functions
 - How do we determine when to synthesize a solution with limits versus IP addresses?

```
source_ip = 1.2.3.4 => DROP,
destination_ip = 1.2.3.4 => ACCEPT,
source_ip = 1.2.3.4 => ACCEPT
```

This work was funded in part by NSF grant CCF-1553168.