Generation of Secure Network Configuration

Cornelius Diekmann, Johannes Naab, Lukas Schwaighofer

December 5, 2016

Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich
Problem Statement

Generation of Secure Network Configurations

Mapping to Security Goals

Security Policy

Security Policy – Manually Edited

Security Policy to Stateful Policy

Stateful Policy to Firewall

Stateful Policy to SDN Rules

From Firewall to Security Policy?

Application within the Sendate Project
Problem Statement

- Most network components can be configured for their specific purpose.
- Essential to implement a secure network.
• Most network components can be configured for their specific purpose.
• Essential to implement a secure network.

• Goal:
Most network components can be configured for their specific purpose.

Essential to implement a secure network.

Goal:

Security Requirements
Problem Statement

- Most network components can be configured for their specific purpose.
- Essential to implement a secure network.

Goal:

```
+ Security Requirements ⇒
```

```
FORWARD DROP
-A FORWARD -i tun0 -s $WebFrnt_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $WebFrnt_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $DB_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $DB_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $WebApp_ipv4 -o tun0 -d $WebFrnt_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $WebApp_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
-A FORWARD -i tun0 -s $WebApp_ipv4 -o eth0 -d $INET_ipv4 -j ACCEPT
-A FORWARD -i eth0 -s $INET_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
-I FORWARD -m state --state ESTABLISHED -i eth0 -s $INET_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
-I FORWARD -m state --state ESTABLISHED -i tun0 -s $WebFrnt_ipv4 -o eth0 -d $INET_ipv4 -j ACCEPT
```
Problem Statement

- Most network components can be configured for their specific purpose.
- Essential to implement a secure network.

- Goal:

  - Manual configuration is error prone
  - $\Rightarrow$ generate configuration automatically to avoid mistakes
Mapping to Security Goals

1. DB, Log and WebApp are internal hosts. WebFrnt must be accessible from outside.
2. Logging data must not leave the log server.
3. DB, Log contain confidential information. WebApp is trusted and allowed to declassify.
4. Only WebApp may access the DB.
1. DB, Log and WebApp are internal hosts. WebFrnt must be accessible from outside.
2. Logging data must not leave the log server.
3. DB, Log contain confidential information. WebApp is trusted and allowed to declassify.
4. Only WebApp may access the DB.

Subnets \{DB \leftrightarrow \text{internal}, \ Log \leftrightarrow \text{internal}, \ WebApp \leftrightarrow \text{internal}, \ WebFrnt \leftrightarrow \text{DMZ}\}
Mapping to Security Goals

1. DB, Log and WebApp are internal hosts. WebFrnt must be accessible from outside.
2. Logging data must not leave the log server.
3. DB, Log contain confidential information. WebApp is trusted and allowed to declassify.
4. Only WebApp may access the DB.

Subnets \{ DB \mapsto internal, \ Log \mapsto internal, \ WebApp \mapsto internal, \ WebFrnt \mapsto DMZ \}\n
Sink \{ \ Log \mapsto Sink \}
1. DB, Log and WebApp are internal hosts. WebFrnt must be accessible from outside.

2. Logging data must not leave the log server.

3. DB, Log contain confidential information. WebApp is trusted and allowed to declassify.

4. Only WebApp may access the DB.

Subnets \{DB \mapsto internal, \ Log \mapsto internal, \ WebApp \mapsto internal, \ WebFrnt \mapsto DMZ\}

Sink \{Log \mapsto Sink\}

Bell LaPadula \{DB \mapsto confidential, \ Log \mapsto confidential, \ WebApp \mapsto declassify (trusted)\}
Mapping to Security Goals

1. **DB, Log** and **WebApp** are internal hosts. **WebFrnt** must be accessible from outside.
2. Logging data must not leave the log server.
3. **DB, Log** contain confidential information. **WebApp** is trusted and allowed to declassify.
4. Only **WebApp** may access the DB.

**Subnets** \{DB → *internal*, Log → *internal*, WebApp → *internal*, WebFrnt → *DMZ*\}

**Sink** \{Log → *Sink*\}

**Bell LaPadula** \{DB → *confidential*, Log → *confidential*, WebApp → *declassify (trusted)*\}

**Comm. Partners** \{DB → *Access allowed by*: WebApp\}
Security Policy

Computing Security Policy

1. Start with allow-all policy:
   \[
   \{\text{Log, DB, WebApp, WebFrnt, INET}\} \times \\
   \{\text{Log, DB, WebApp, WebFrnt, INET}\}
   \]

2. Remove all rules which contradict the (completed) Security Goals
   
   - Sound
   - Complete: Maximum permissive policy
     (only for certain invariant templates)
• Security Policy can be edited manually
• Policy is checked against Security Goals
• Changes must not introduce violations of Security Goals
Security Policy to Stateful Policy

- In order for a TCP connection to work, a bidirectional connection is necessary.
- I.e. client (INET) sends request, response is sent from WebFront to client.
- A stateful firewall allows the reverse flow, if such a connection was established by the client.

Consistency:
1. No information flow violation must occur
2. No access control side effects must be introduced
Stateful Policy to Firewall

Assumptions

**Structure**  Enforced network connectivity structure = policy.
Links: confidential and integrity protected.

**Authenticity**  Policy’s entities must match their network representation (e.g. IP/MAC addresses).

**State**  The stateful connection handling must match the stateful policy’s semantics.

Term rewriting

FORWARD DROP
- A FORWARD -i tun0 -s $WebFrnt_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $WebFrnt_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $DB_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $DB_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $WebApp_ipv4 -o tun0 -d $WebFrnt_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $WebApp_ipv4 -o tun0 -d $DB_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $WebApp_ipv4 -o tun0 -d $Log_ipv4 -j ACCEPT
- A FORWARD -i tun0 -s $WebApp_ipv4 -o eth0 -d $INET_ipv4 -j ACCEPT
- A FORWARD -i eth0 -s $INET_ipv4 -o tun0 -d $WebFrnt_ipv4 -j ACCEPT
- I FORWARD -m state --state ESTABLISHED -i eth0 -s $INET_ipv4 -o tun0 -d $WebApp_ipv4 -j ACCEPT
- I FORWARD -m state --state ESTABLISHED -i tun0 -s $WebFrnt_ipv4 -o eth0 -d $INET_ipv4 -j ACCEPT
Stateful Policy to SDN Rules

Term rewriting

- Only as single network security device is considered
- Stateful firewall handling is not provided by SDN switch
- Could be introduced by iptables firewall or Open vSwitch >= 2.5.0

```
# ARP Request
in_port=$port_src dl_src=$mac_src dl_dst=ff:ff:ff:ff:ff:ff  # $port_dst
arp arp_sha=$mac_src arp_spa=$ip4_src arp_tpa=$ip4_dst  # $port_dst
priority=40000 action=mod_dl_dst:$mac_dst,output:$port_dst

# ARP Reply
dl_src=$mac_dst dl_dst=$mac_src arp arp_sha=$mac_dst arp_spa=$ip4_dst  # $port_dst
arp_tpa=$ip4_src priority=40000 action=output:$port_src

# IPv4 one-way
in_port=$port_src dl_src=$mac_src ip nw_src=$ip4_src nw_dst=$ip4_dst  # $port_dst
priority=40000 action=mod_dl_dst:$mac_dst,output:$port_dst

# if src (resp. dst) is INET, replace $ip4_src (resp. $ip4_dst) with *
# and decrease the priority

ovs-vsctl set-fail-mode $switch secure && ovs-ofctl add-flows
```
From Firewall to Security Policy?

- Generating a configuration requires an existing security policy
- A lot of firewalls are managed manually and encode implicit knowledge about the security goals
From Firewall to Security Policy?

- Generating a configuration requires an existing security policy
- A lot of firewalls are managed manually and encode implicit knowledge about the security goals
From Firewall to Security Policy?

- Generating a configuration requires an existing security policy
- A lot of firewalls are managed manually and encode implicit knowledge about the security goals
From Firewall to Security Policy?

- Validate that security policy matches with our expectations
- Detect hidden bugs hidden within the firewall configuration
- “Visualize” existing firewalls
  - What did the previous administrator configure?
  - Are there security violations embedded within the firewall?

Automated checking of the firewall configuration before the deployment can help to avoid problems:

- Are the security devices and switches only reachable from the controller or management network?
  - I.e. no unauthorized access is possible
- Are the devices accessible by the controller or management network?
  - Even if there is an error, are the devices still reachable to change the configuration
  - ⇒ Allow in-band management of devices
  - ⇒ Protect from (obvious) configuration mistake
From Firewall to Security Policy?

- Validate that security policy matches with our expectations
- Detect hidden bugs hidden within the firewall configuration
- “Visualize” existing firewalls
  - What did the previous administrator configure?
  - Are there security violations embedded within the firewall?

Automated checking of the firewall configuration before the deployment can help to avoid problems:
From Firewall to Security Policy?

- Validate that security policy matches with our expectations
- Detect hidden bugs hidden within the firewall configuration
- “Visualize” existing firewalls
  - What did the previous administrator configure?
  - Are there security violations embedded within the firewall?

Automated checking of the firewall configuration before the deployment can help to avoid problems:

- Are the security devices and switches only reachable from the controller or management network?
  - I.e. no unauthorized access is possible
From Firewall to Security Policy?

- Validate that security policy matches with our expectations
- Detect hidden bugs hidden within the firewall configuration
- “Visualize” existing firewalls
  - What did the previous administrator configure?
  - Are there security violations embedded within the firewall?

Automated checking of the firewall configuration before the deployment can help to avoid problems:

- Are the security devices and switches only reachable from the controller or management network?
  - I.e. no unauthorized access is possible
- Are the devices accessible by the controller or management network?
  - Even if there is an error, are the devices still reachable to change the configuration
  - ⇒ Allow in-band management of devices
  - ⇒ Protect from (obvious) configuration mistake
Application within the Sendate Project

- Goals to be checked in Data Center Networks
  - Separation of tenants/slices
  - Even if the slices provide their own configuration
  - Accessibility (both positive and negative) of management interfaces

- Validation of configuration must be integrated within the management
- Each and every validation must be checked, for maximum benefit before deploying to the devices
- Configuration must be centralized
- No manual configuration/change to the firewall
- Integration with configuration and change management tools
  - Ansible, Puppet, Salt

- Performance Measurements
  - Impact of rule sets on performance of network devices
Application within the Sendate Project

- Goals to be checked in Data Center Networks
  - Separation of tenants/slices
  - Even if the slices provide their own configuration
  - Accessibility (both positive and negative) of management interfaces
- Validation of configuration must be integrated within the management
  - Each and every validation must be checked, for maximum benefit before deploying to the devices
- Configuration must be centralized
  - No manual configuration/change to the firewall
- Integration with configuration and change management tools
  - Ansible, Puppet, Salt
- Performance Measurements
  - Impact of rule sets on performance of network devices
Bibliography

Semantics-Preserving Simplification of Real-World Firewall Rule Sets.

Demonstrating topoS: Theorem-Prover-Based Synthesis of Secure Network Configurations.