Securing the Software-Defined Network Control Layer

Phillip Porras, Steven Cheung, Martin Fong, Keith Skinner, and Vinod Yegneswaran

Computer Science Laboratory, SRI International
333 Ravenswood Avenue, Menlo Park, CA 94025

NDSS 2015
<table>
<thead>
<tr>
<th>Issue</th>
<th>Solution</th>
<th>but would we rather.....</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malicious Packet Stream</td>
<td>Drop</td>
<td>Auto-Redirect Malicious Source to Honeynet</td>
</tr>
<tr>
<td>Policy Violations</td>
<td>Drop</td>
<td>Redirect User to a Notification Server</td>
</tr>
<tr>
<td>Network Wide Anomaly</td>
<td>Drop</td>
<td>Selective Filtering or reprovision assets</td>
</tr>
<tr>
<td>Infected Host</td>
<td>Drop</td>
<td>Quarantine</td>
</tr>
<tr>
<td>Floods and Service Denials</td>
<td>Drop</td>
<td>Block, Migrate Mission Critical services, Redirect</td>
</tr>
<tr>
<td>Malicious Logic injection</td>
<td>Drop</td>
<td>Redirect into Sandnet</td>
</tr>
<tr>
<td>Remote Shell or C&amp;C</td>
<td>Drop</td>
<td>Redirect In and outbound flows to separate data sinks</td>
</tr>
<tr>
<td>Server Behavioral Deviations</td>
<td>Drop</td>
<td>Dynamic quota adjustment, fishbowl and reprovision new server</td>
</tr>
<tr>
<td>Network Reconnaissance</td>
<td>Drop</td>
<td>Proactively redirect probes to whitehole or honeynet</td>
</tr>
<tr>
<td>Threat Reputation</td>
<td>Drop</td>
<td>Selectively limit network privileges or apply added antifraud challenges</td>
</tr>
<tr>
<td>Stepping Stone Tunneling</td>
<td>Drop</td>
<td>Selective interruption to validate that tunnel exists</td>
</tr>
</tbody>
</table>
Security challenges: What happens when software defines your network flow policy?

• We grew up with (fairly) “static policies”: With SDNs … Traffic Engineering (TE) Apps constantly orchestrate the network flows to adapt to network conditions.

• Security must not depend on the absences of complex SDN App interactions.

• Ideally, flow policies made in response to threats should take precedence.

• The SDN Stack is itself a fair TARGET for attack.

Solving these challenges is a prerequisite for adoption by secure computing facilities, ... anywhere compliance is needed.
Security Challenge  Virtual Flow Orchestration

May 2012, A Demonstration of Inline Constraints Policy Enforcement,  6 minutes

A to B Bypass
(1) A initiates flow to C
(2) packet DST rewritten to B
   packet SRC rewritten to D
(3) forward flow D->B

B to A Bypass
(4) B initiates flow to D
(5) packet DST rewritten to A
    packet SRC rewritten to C
(6) forward flow C->A

Switch Flow Table
i. A -> B action [] (drop)
ii. B -> A action [] (drop)
iii. A-> C action [set dst -> B,
                     set src -> D,
                     forw]
iv. B -> D action [set dst -> A,
                     set src -> C,
                     forw]
Network Policy Conflict Arbitration

Application Layer

- App A1
- App A2
- App N

Control Layer

- Load Balancer
- Traffic QOS
- IPS

Data Plane

- OF Switch

Server Pool
Network Policy Conflict Arbitration

Monolithic App Design

Control Layer

Data Layer

Conflicts Arbitration

Traffic Throttling

App-Aware Flow QoS

Fault Recovery

Sharable, Composable, Design

Control Layer

Data Layer

Traffic Throttling

App-Aware Flow QoS

Fault Recovery

Recovery

We are here
What is SE-Floodlight?

An application-to-data-plane security mediation service embedded in the control layer

- Recognizes and resolve conflicts between a Candidate Flow rules and the current flow policy

- Allows the dynamism of OpenFlow applications to produce optimal flow routing decision

- Empowers OpenFlow security applications and operators to dynamically assert defensive flow policy when new threats are perceived
An OpenFlow Controller

The Floodlight Controller

- a coordination point through which traffic engineering apps
  - convey flow rules
  - submit configuration requests to the switch
  - probe the data plane for state information
  - Probe the controller state
  - configure the controller

TE modules

- Hub
- Learning Switch
- Forwarding

TE Apps

- Circuit Pusher (python)
- Rest API

OpenFlow Switch

To Data Plane

OpenFlow Protocol
Control Layer Mediation?

- SDN-IP
- Business Partnership Logic
- Traffic Engineering
- Security Actuator
- Fault Recovery

App-to-Controller Authentication
App Permission Mediation
Application Role or an Authority level
Conflict Detection and Resolution
Consistent Policy Implementation

Distributed Data Plane
Identifying Authentication

**TE modules**

- Pre-inspection JNI, classes with reserved packages, custom ClassLoaders, etc.
- Loadable TE Module
- Protected Factory Adds credential
- Northbound Client Proxy
- Northbound Proxy Server
- Protected Factory Adds credential
- OK

**Floodlight Controller**

- SSL + Alternate Northbound API
- To Data Plane
- OpenFlow Protocol
- OpenFlow Switch
- Controller Admin-provided App Credentials

**Runtime Credential**
admin generates runtime credential signed manifest, module and classes, SE-Fi credentials

**Class Validation Function**
Validates integrity of module and manifest

**Essentially,** We add the credential as an opaque object provided to every client request

**5. Legacy modules**
objects without the protected opaque credential inherit app credential
App Credentials: Hierarchical Authorization Roles

Administrator Applications – scripts and console apps

Security Applications – dynamic filtering and redirection in response to perceived threats or vulnerabilities

Applications - Traffic Engineering applications

flowmod

Conflict resolution

Priorities

Permissions

 ADMIN

SEC

APP

p1, p2, p3,....pn

p1, p2, p3,....pn

p1, p2, p3,....pn
We Introduce an app permission model for OpenFlow

<table>
<thead>
<tr>
<th>Flow Direction</th>
<th>Data Exchange Operation</th>
<th>Mediation Policy</th>
<th>(default) Minimum Authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>01: A to D</td>
<td>Flow rule mod</td>
<td>ARR (Section 1.5)</td>
<td>APP</td>
</tr>
<tr>
<td>02: D to A</td>
<td>Flow removal messages</td>
<td>Public</td>
<td>APP</td>
</tr>
<tr>
<td>03: D to A</td>
<td>Flow error reply</td>
<td>Public</td>
<td>APP</td>
</tr>
<tr>
<td>04: A to D</td>
<td>Barrier requests</td>
<td>Permissions</td>
<td>APP</td>
</tr>
<tr>
<td>05: D to A</td>
<td>Barrier replies</td>
<td>upon request</td>
<td>APP</td>
</tr>
<tr>
<td>06: D to A</td>
<td>Packet-In return</td>
<td>upon request</td>
<td>APP</td>
</tr>
<tr>
<td>07: A to D</td>
<td>Packet-Out</td>
<td>Permissions</td>
<td>SEC</td>
</tr>
<tr>
<td>08: A to D</td>
<td>Switch port mod</td>
<td>Permissions</td>
<td>ADMIN</td>
</tr>
<tr>
<td>09: D to A</td>
<td>Switch port status</td>
<td>upon request</td>
<td>ADMIN</td>
</tr>
<tr>
<td>10: A to D</td>
<td>Switch set config</td>
<td>Permissions</td>
<td>ADMIN</td>
</tr>
<tr>
<td>11: A to D</td>
<td>Switch get config</td>
<td>Permissions</td>
<td>APP</td>
</tr>
<tr>
<td>12: D to A</td>
<td>Switch config reply</td>
<td>upon request</td>
<td>APP</td>
</tr>
<tr>
<td>13: A to D</td>
<td>Switch stats request</td>
<td>Permissions</td>
<td>APP</td>
</tr>
<tr>
<td>14: D to A</td>
<td>Switch stats report</td>
<td>upon request</td>
<td>APP</td>
</tr>
<tr>
<td>15: A to D</td>
<td>Echo requests</td>
<td>Permission</td>
<td>APP</td>
</tr>
<tr>
<td>16: D to A</td>
<td>Echo replies</td>
<td>upon request</td>
<td>APP</td>
</tr>
<tr>
<td>17: D to A</td>
<td>Vendor features</td>
<td>Permission</td>
<td>ADMIN</td>
</tr>
<tr>
<td>18: A to D</td>
<td>Vendor actions</td>
<td>Permissions</td>
<td>ADMIN</td>
</tr>
</tbody>
</table>

**Apps**: Insert Flow Policies  
**Sec**: Adds the ability to use **PacketOut**  
**Admin**: manipulate switch configuration  
...or select your own model
Application Mediation Service

4 main functions

- **State Manager** Maintains aggregate flow logic representation
- **RCA** Performs inline conflict detection between candidate rule and existing rules
- **Resolution** enables authorization rules of rule produces to resolve conflicts
- **Permission Mediator** enforces Module credential permissions
There are four output disposition categories (1) output to port, $O_p$ (which may include broadcasts); (2) output to table, $O_T$; output to controller, $O_C$; and (4) no output (or Drop).
**RCA Rule-Chain Conflict Analysis**

**Candidate flowmod**

- **Direct Conflict Testing**
  - Yes

**Conflict resolution**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC is lower</td>
<td>Reject RC</td>
</tr>
<tr>
<td>RC is higher</td>
<td>Delete conflicting Rs and insert RC</td>
</tr>
<tr>
<td>RC is equal</td>
<td>FIFO (reject RC)</td>
</tr>
<tr>
<td></td>
<td>LIFO (expunge R, accept RC)</td>
</tr>
</tbody>
</table>

**Direct Conflict**
arises when RC alters a flow disposition that is currently defined by existing flow rules.
Candidate flowmod

Direct Conflict Testing

No chain

Chaining Analysis

$R^* + R_C$

$R_C + R^*$

$R^* + R_C + R^*$

Insert $R_C$
**RCA**

Candidate **flowmod**

- **RC$_1$**
- **RC$_n$**

Direct Conflict Testing

Conflict resolution

| RC is lower | Reject RC |
| RC is higher | Delete conflicting Rs and insert RC |
| RC is equal | FIFO (reject RC) | LIFO (expunge R, accept RC) |

**Rule Criteria Matching Alg**

- **Yes**
- **No**

- **R* + RC**
- **RC + R**
- **R* + RC + R**

Chaining Analysis

- **No**
- **Yes**

- **tail**
- **head**
- **middle**

- **RC**

Insert **RC**

- **No chain**
- **RC**

- **Yes**

- **Fixes ARR overfitting**
- **Handles multi-table**
- **Multi-OF Switch**
Security Audit

SDN Network Auditing

*NetSight* packet-level flow traversal

*ndb* post-card-based route flow route mapping

*OFRewind* audits and plays back SDN Control Plane traffic

**Security audit subsystem**
- Flow rule insertions
- Packet_In Events
- All mediation results
- Switch flow table management
- Authentication events
- REST API events
SE-Floodlight

**Floodlight controller with security extensions**

- Java Class OF-App
- Northbound Proxy Server
- App Credential Management
- Security Audit Subsystem

**Security Enforcement Kernel**

- Role-based Source Auth
- ARR – Conflict Analyzer
- State Table Manager
- Switch Callback Tracking
- Permission Mediator

**Aggregate Flow Table**

- Administrator flow rules
- Security Service flow rules
- Application flow rules

**Role-based Segregation**

**Inline flow rule conflict detection**

**Role-based Authorization (conflict resolution)**

**Digital Authentication of FlowRule Source**

**Privilege Separation (OF Apps)**

**Security Audit**

**Application Permission Model**

SE-Floodlight beta1.0 OpenFlow Controller

SE-Floodlight www.openflowsec.org
The Security Actuator Package

SDN Security Mediation Actuator
- SRI CL SE-Northbound API

SE-Floodlight V1.0 OpenFlow Controller
- (beta)

IDS
- Vulnerability Scanner / Redirector
- Security Proxy
- SDN-enabled Honeynet
- SandNet

Security Actuator implements high-level Security directives
- BLOCK
- QUARANTINE
- REDIRECT
- NETMAP
- INFO
- DENY
- UNPLUG
- ALLOW
- UNDO

3rd-parties apps can extend to perform other remediation concepts.
Thank You

More Information

www.openflowsec.org
www.sdnsecurity.org

Acknowledgements

DARPA
MRC

NSF
ACI EAGER Grant