VeriCon: Towards Verifying Controller Programs in Software-Defined Networks

Thomas Ball*, Nikolaj Bjørner*, Aaron Gember‡, Shachar Itzhaky§, Aleksandr Karbyshev*, Mooyi Sagiv", Michael Schapira* and Asaf Valadarsky

*Microsoft Research, ‡University of Wisconsin, §Tel Aviv University, †Technische Universität München, *Hebrew University

Motivation
Software defined networking (SDN) aims to simplify network management by removing the control plane from switches and running custom control applications at a logically central controller. Unfortunately, writing control applications that always maintain a set of network invariants (e.g., the network does not contain forwarding loops or blackholes) is a challenging task.

Prior work [1, 2] uses finite-state model checking and network snapshots to identify bugs in control applications. They can find errors, but they cannot guarantee the absence of errors.

Overview
VeriCon verifies network-wide invariants for any event sequence and all admissible topologies.

Types of Invariants
- **Topology**: define admissible topologies \{ assumed to hold initially \}
- **Safety**: define the required consistency of network-wide states \{ checked initially \ after each event \}
- **Transition**: define the effect of executing event handlers

Core SDN (CSDN)
- Define and initialize relations: \( rel(r) \) \( init \ r = () \)
  - Topology relations: \( \text{link}(S,O,H) \) \( \text{path}(S,O,H) \)
  - Forwarding relations: \( \text{fr}(S,P,I,O) \) \( \text{fr}(S,P,I,O) \)
- Write packet-in event handlers: \( \text{pktIn}(S,P,I) \)
  - Update defined relations: \( r.\text{insert}() \) \( r.\text{remove}() \)
  - Install rules (\( fr.\text{insert} \)): \( S.\text{install}(P,I,O) \)
  - Forward packet (\( fr.\text{insert} \)): \( S.\text{forward}(P,I,O) \)
  - Conditionals: \( \text{if } \text{Cond} \text{ then } \text{Cmd}^* \text{ else } \text{Cmd}^* \)

Learn More!
http://agember.com/go/vericon

Verification Time

<table>
<thead>
<tr>
<th>Program</th>
<th>LOCs</th>
<th>Topo</th>
<th>Safety + Trans Inv.</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firewall</td>
<td>8</td>
<td>1</td>
<td>3 + 0</td>
<td>0.12</td>
</tr>
<tr>
<td>Stateless Firewall</td>
<td>4</td>
<td>1</td>
<td>2 + 0</td>
<td>0.06</td>
</tr>
<tr>
<td>Firewall + Host Migration</td>
<td>9</td>
<td>0</td>
<td>3 + 0</td>
<td>0.16</td>
</tr>
<tr>
<td>Learning Switch</td>
<td>8</td>
<td>1</td>
<td>4 + 2</td>
<td>0.16</td>
</tr>
<tr>
<td>Learning Switch + Auth</td>
<td>15</td>
<td>2</td>
<td>5 + 3</td>
<td>0.21</td>
</tr>
<tr>
<td>Resonance (simplified)</td>
<td>93</td>
<td>6</td>
<td>5 + 2</td>
<td>0.21</td>
</tr>
<tr>
<td>Stratos (simplified)</td>
<td>29</td>
<td>12</td>
<td>3 + 0</td>
<td>0.09</td>
</tr>
</tbody>
</table>

Example: Stateful Firewall

- Always forward from trusted to untrusted hosts
- Only forward from untrusted to trusted hosts if a trusted host previously sent a packet to the untrusted host

Application in Core SDN
\[ \text{rel } \text{tr}(\text{SW, HO}) \]
\[ \text{pktIn}(\text{sw.pkt prt}(1)) \rightarrow \]
\[ \text{sw.forward}(\text{pkt.prt}(1),\text{prt}(2)) \]
\[ \text{tr.insert(s.pkt, dst)} \]
\[ \text{sw.install(pkt.prt}(1),\text{prt}(2)) \]
\[ \text{if } \text{tr(s.pkt.src, then} \]
\[ \text{sw.forward(pkt.prt}(2),\text{prt}(1)) \]
\[ \text{sw.install(pkt.prt}(2),\text{prt}(1)) \]

Invariants
- At least one switch with ports \( \text{prt}(1) \) & \( \text{prt}(2) \); a packet \( P \) is forwarded from an untrusted host \( U \) to a trusted host \( T \)

\[ \exists U, T, HO, S, SW, P, PK, \]
\[ \text{link}(S,P,PK(U)) \land \text{link}(S,\text{prt}(1),T) \land \]
\[ P\text{.src} = U \land P\text{.dst} = T \land P\text{.fr}(S,P,\text{prt}(2),\text{prt}(1)) \]

- For every packet sent from an untrusted host \( U \) to a trusted host \( T \) there exists a packet sent to \( U \) from \( T \)

\[ I_1 = \text{fr}(S,P,\text{prt}(2),\text{prt}(1)) \Rightarrow \exists P', P\text{.src} = P\text{.src} \land P\text{.fr}(S,P',\text{prt}(1),\text{prt}(2)) \]

- Flow table entries only contain forwarding rules from trusted hosts

\[ I_2 = \text{fr}(S,P,\text{prt}(2),\text{prt}(1)) \Rightarrow \exists P', P\text{.PK}\text{.dst} = P\text{.PK}\text{.src} \land P\text{.fr}(S,P',\text{prt}(1),\text{prt}(2)) \]

- Controller relation \( tr \) stores the correct hosts

\[ I_3 = \text{tr}(S,H) \Rightarrow \exists P, P\text{.PK}\text{.dst} = H \land P\text{.fr}(S,P,\text{prt}(1),\text{prt}(2)) \]

Counterexample

Reference