VeriCon: Towards Verifying Controller Programs in Software-Defined Networks

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.

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

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

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
\[
\begin{align*}
\text{rel} & (SW, HO) \\
\text{pktIn} & (SW, pkt, prt(1)) \rightarrow \\
& \text{sw.forward}(pkt, prt(1), prt(2)) \\
& \text{tr.insert}(s, pkt, dst) \\
& \text{sw.install}(pkt, prt(1), prt(2)) \\
\text{pktIn} & (SW, pkt, prt(2)) \rightarrow \\
& \text{if} \text{tr}(SW, pkt, src) \text{ then} \\
& \text{sw.forward}(pkt, prt(2), prt(1)) \\
& \text{sw.install}(pkt, prt(2), prt(1))
\end{align*}
\]

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

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

SDN application in Core SDN + Topology constraints & invariants in first order logic

Verify conditions using the Z3 theorem prover

Guarantee invariants are satisfied

Concrete counter-example

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>

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 \(T\)
  \[\exists U, T, HO, S, SW, P, PK, \text{link}(S,PK,prt(2),U) \land \text{link}(S,PK,prt(1),T) \land P.\text{src} = U \land P.\text{dst} = T \land \text{fr}(S,P,PK,prt(2),prt(1))\]
  - For every packet sent from an untrusted host \(U\) to a trusted host \(T\) there exists a packet sent to \(U\) from \(T\)
    \(I_1 \equiv \text{fr}(S,P,PK,prt(2),prt(1)) \Rightarrow \exists P', P.\text{src} = P.\text{dst} \land P.\text{src} \land \text{fr}(S,P',PK,prt(1),prt(2))\)
  - Flow table entries only contain forwarding rules from trusted hosts
    \(I_2 \equiv \text{fr}(S,P,PK,prt(2),prt(1)) \Rightarrow \exists P': PK.\text{PK} = P.\text{PK} \land \text{fr}(S,P',PK,prt(1),prt(2))\)
  - Controller relation \(\text{tr}\) stores the correct hosts
    \(I_3 \equiv \text{tr}(S,H) \Rightarrow \exists P: PK.\text{PK} = H \land \text{fr}(S,P,PK,prt(1),prt(2))\)

Counterexample
\[
\begin{align*}
I_1 \land I_2 \land I_3 \text{ is inductive}
\end{align*}
\]

References