A Formal Framework for Secure Routing Protocols

Chen Chen, Limin Jia, Hao Xu, Cheng Luo, Wenchao Zhou and Boon Thau Loo
University of Pennsylvania, Carnegie Mellon University, Georgetown University
What makes Internet Routing vulnerable today?

◆ BGP: De-facto Internet routing protocol

Prefix hijacking

{IP prefix(foo),<AS2>}

{IP prefix(foo),<AS5,AS6,AS7>}

{IP prefix(foo),<AS6,AS7>}

{IP prefix(foo),<AS7>}

www.foo.com
What makes Internet Routing vulnerable today?

- BGP: De-facto Internet routing protocol

![Diagram showing BGP routing and forged announcement]
Solutions

◆ Secure BGP extension (through cryptography):

Lack of formal proofs in terms of security guarantees

◆ Clean slate (new infrastructure):
Our solution

- SeNDlog specification
- Property specification

- Executable code generator
- Verification condition generator

- Distributed implementation
- Proof obligations

- Simulator (emulator)
- Theorem prover
Outline

• Syntax of SeNDLog
• Operational Semantics
• Program Logic
• Case study
• Conclusion & future
Secure Network Datalog

- Network Datalog: Distributed declarative language
  - Network protocols
- User-defined cryptographic function
  - Secure network protocols

R1 reachable(@S, D) :- link(@S, D).
R2 reachable(@Z, D) :- link(@S, Z), reachable(@S, D).
Outline

• Syntax of SeNDLog
• Operational Semantics
• Program Logic
• Case study
• Conclusion & future
Distributed Execution Model

- Small-step operational semantics
  - State: node state & network queue

R1 : reachable(@S, D) ← link(@S, D)
R2 : reachable(@Z, D) ← link(@S, Z), reachable(@S, D)

\[
Q = \{\}
\]
\[
S_A = \{\text{link}(\@A, \@B)\}
\]
\[
S_B = \{\text{link}(\@B, \@A), \text{link}(\@B, \@C)\}
\]
\[
S_C = \{\text{link}(\@C, \@B)\}
\]
Distributed Execution Model

- Small-step operational semantics
  - State: node state & network queue
  - Transition: Invocation of rules change node states, thus changing network states.

R1 : reachable(@S, D) ← link(@S, D)

R2 : reachable(@Z, D) ← link(@S, Z), reachable(@S, D)

\[ Q = \{ \} \]

\[ S_A = \{ \text{link(@A, B), reachable(@A, B)} \} \]

\[ S_B = \{ \text{link(@B, A), link(@B, C), reachable(@B, A), reachable(@B, C)} \} \]

\[ S_C = \{ \text{link(@C, B), reachable(@C, B)} \} \]
Distributed Execution Model

- Small-step operational semantics
  - State: node state & network queue
  - Transition: Invocation of rules change node states, thus changing network states.

R1 : reachable(@S, D) ← link(@S, D)

R2 : reachable(@Z, D) ← link(@S, Z), reachable(@S, D)

Q = \{reachable(@A, C)\}

\[ S_A = \{\text{link}(\@A, \@B), \text{reachable}(\@A, \@B)\} \]

\[ S_B = \{\text{link}(\@B, \@A), \text{link}(\@B, \@C), \text{reachable}(\@B, \@A), \text{reachable}(\@B, \@C)\} \]

\[ S_C = \{\text{link}(\@C, \@B), \text{reachable}(\@C, \@B)\} \]
Outline

• Syntax of SeNDLog
• Operational Semantics
• Program Logic
• Case study
• Conclusion & future
Program Logic

• Syntax
  – Atoms:
    • $\text{tuple}(a_1,a_2...a_n)@(loc, t)$; $\text{send}(loc, \text{tuple})@t$; $\text{receive}(loc, \text{tuple}())@t$;
      $\text{honest}(loc, \text{prog}, t)$
  – Inequality

• Formula $\varphi$:
  – Atoms $| \varphi \lor \varphi | \varphi \land \varphi | \varphi \Rightarrow \varphi | \neg \varphi | \forall x, \varphi | \exists x, \varphi$

• Variable context $\Sigma$ & Logical context $\Gamma$

• Inference rule
  – Standard first-order logic rule
  – rules based on semantics
    • Invariant rule (INV)
    • Honest rule (HONEST)
Invariant Rule (INV)

\[ [\text{prog}]_i : \{i, ts, te\} : \text{property of SeNDLog program } [\text{prog}]_i \text{ running from } ts \text{ till } te(\varphi(i, ts, te)) \]

\( \varphi(i, t) \): property of SeNDLog program at time \( t \)

\[ \varphi(i, ts, te) \leftrightarrow \forall t, ts \leq t \leq te \land \varphi(i, t) \]

Each invocation of rules at time \( t \) maintains \( \varphi(i, t) \)

R1 : reachable(@S, D) \( \leftarrow \) link(@S, D)

R2 : reachable(@Z, D) \( \leftarrow \) link(@S, Z), reachable(@S, D)

\( \varphi(i, t) \): \( \forall d, \text{reachable}(i, d)@\( i, t \) \rightarrow \text{link}(i, d)@\( i, t \) \lor \exists m, \text{link}(m, i)@\( m, t \).

\( \varphi(i, ts, te) \): \( \forall t, ts \leq t \leq te \land \varphi(i, t) \)
Verification Condition Generator

Each invocation of rules at time \( t \) maintains \( \varphi(i,t) \)

\[
\varphi(i,t_s,t_e) : \forall d,t, t_s \leq t \leq t_e \land (\text{reachable}(i,d)@i(t) \rightarrow \text{link}(i,d)@i(t) \lor \exists m, \text{link}(m,i)@m(t)).
\]

R1 : reachable(\( @S, D \)) \( \leftarrow \) link(\( @S, D \))

R2 : reachable(\( @Z, D \)) \( \leftarrow \) link(\( @S, Z \)), reachable(\( @S, D \))

Lemma1 : \( \varphi_{\text{reachable}} \) \( \leftarrow \) link(\( s, d \))@\( s, t \).

Lemma2 : \( \varphi_{\text{reachable}} \) \( \leftarrow \) link(\( s, z \))@\( s, t \), reachable(\( s, d \))@\( s, t \) .

Automation: Verification condition generator
Honest Rule (HONEST)

\[
[\text{prog}]_i : \{i,t_s,t_e\} . \varphi(i,t_s,t_e)
\]
Honest(n, [prog]_i, t_n)

\[
\forall t', t' > t_n \Rightarrow \varphi(n,t_n,t')
\]

Honest(n, [prog]_i, t):
- n is honest:
  - ✓ n does not leak its private key
  - ✓ n follows the behavior specified by SeNDLog program
- n runs the program in question: [prog]_i
- n runs the program at time t.

\[
[\text{prog}]_i : \{i,t_s,t_e\} . \varphi(i,t_s,t_e): \forall d,t, t_s \leq t \leq t_e \land (\text{reachable}(i,d)@(i,t) \Rightarrow \text{link}(i,d)@(i,t) \lor \exists m, \text{link}(m,i)@(m,t)).
\]
Honest(n, [prog]_i, t_n)

\[
\forall t', t' > t_n \Rightarrow \\
\forall d,t, t_n \leq t \leq t' \land (\text{reachable}(n,d)@(n,t'') \Rightarrow \text{link}(n,d)@(n,t'') \lor \exists m, \text{link}(m,n)@(m,t'')).
\]
Outline

- Syntax of SeNDLog
- Operational Semantics
- Program Logic
- Case study
- Conclusion & future
# Protocols

<table>
<thead>
<tr>
<th>S-BGP</th>
<th>SCION</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Secure extension to BGP protocol</td>
<td>• Clean-slate design of Internet routing.</td>
</tr>
<tr>
<td>• Layered signature</td>
<td>• Group autonomous systems (AD) into trusted domains (TD)</td>
</tr>
<tr>
<td></td>
<td>• Layered signature</td>
</tr>
</tbody>
</table>

## Control-plane path authenticity:

\[ \forall n\ m\ d\ p\ sl\ t,\ \text{Honest}(n) \land \text{advertisement}(m,n,d,p,sl)@(n,t) \Rightarrow \text{goodPath}(t,p,d) \]

<table>
<thead>
<tr>
<th>S-BGP</th>
<th>SCION</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

## Data-plane path authenticity:

Each node authenticates neighboring links in the specific path during data forwarding (To be done)

<table>
<thead>
<tr>
<th>S-BGP</th>
<th>SCION</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>True</td>
</tr>
</tbody>
</table>
Conclusion & Future work

• Conclusion:
  - A unified framework for verification and implementation of Internet routing protocol.
  - Operational semantics for SeNDLog
  - Program logic for proving security properties
  - Verification condition generator

• Future work:
  - Automation of invariant generation?
  - Automation of proof?
Thank you

Questions?