Verification Primer

Zach Tatlock
University of Washington
Bugs = Constant Friction
Verification: Reduce Bug Friction

Ultimately, we want to provide a principled methodology that ensures reliability.
Verification: Reduce Bug Friction

1. Specification
   crisply describes ideal behavior

2. Implementation
   controls actual system behavior

3. Proof
   shows impl always agrees w/ spec
Verification: All About Agreement

1. Specification
   crisply describes ideal behavior

2. Implementation
   controls actual system behavior

3. Proof
   shows impl always agrees w/ spec
Verification: All About Agreement

1. Specification crisply describes ideal behavior.

2. Implementation controls actual system behavior.

3. Proof shows implementation always agrees with specification.

Two Interpretations:

1. Spec is Absolute Truth and implementation satisfies / refines it.

2. Spec and implementation are independent implementations that agree. We believe this improves reliability since

\[ P(\text{bug}) = P(\text{bug in spec}) \times P(\text{bug in impl}) \]
The Problem With Turtles

Russel once gave a public lecture on astronomy. He described how the earth orbits around the sun and how the sun, in turn, orbits around the center of a vast collection of stars called our galaxy.

At the end of the lecture, a little old lady at the back of the room got up and said: "What you have told us is rubbish. The world is really a flat plate supported on the back of a giant tortoise." The scientist gave a superior smile before replying, "What is the tortoise standing on?"

"You're very clever, young man, very clever," said the old lady. "But it's turtles all the way down!"

— Hawking, 1988
Quis custodiet ipsos custodes?

1. Specification
   crisply describes
   ideal behavior

2. Implementation
   controls actual
   system behavior

3. Proof
   shows impl always
   agrees w/ spec
   assuming

   TCB
   basic assumptions
   we must trust
Approaching Verification Papers

1. What is specification?

2. How is it implemented?

3. Why do they agree?

4. Should we believe?
Verification Primer + Session Preview
## Session Preview

### 1:30pm - 3:10pm Session 6 - Verification

**Session Chair:** Ramesh Govindan *(University of Southern California)*  
**Room:** Diamante

<table>
<thead>
<tr>
<th>Title</th>
<th>Speakers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>, Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>, Aditya Akella <em>(University of Wisconsin-Madison)</em>, Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoinescu <em>(University Politehnica of Bucharest)</em>, Matei Popovici <em>(University Politehnica of Bucharest)</em>, Lorina Negreanu <em>(University Politehnica of Bucharest)</em>, Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don’t Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>, Ratul Mahajan <em>(Microsoft)</em>, Todd Millstein <em>(University of California, Los Angeles)</em>, Jitendra Padhye <em>(Microsoft)</em>, David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>, Yossi Gilad <em>(Boston University and MIT)</em>, Amir Herzberg <em>(Bar Ilan University)</em>, Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
Abstraction for Control Planes (ARC)

Control plane config selects and installs routes

Super difficult to configure correctly:
- may fail to block unwanted traffic
- may not tolerate enough link failure
- may not route through target waypoints
Abstraction for Control Planes (ARC)

ARCS abstract configs to weighted digraph

Key property: shortest paths in ARC correspond to network behavior!

Reduces verification to standard graph queries.
Abstraction for Control Planes (ARC)

- Reachability, isolation, fault-tolerance
- Control plane configurations
- Existence of certain paths in ARC
- Reduction to ARCs, graph algos
Questions:

1. Shortest path is a global property. How do you handle node-local control plane decisions?

2. Which control plane behaviors cannot be modeled as shortest path constraints?

3. How can you extend ARCs to account for preference in BGP?
# Session Preview

**1:30pm - 3:10pm Session 6 - Verification**

**Session Chair: Ramesh Govindan (University of Southern California)**  
**Room: Diamante**

<table>
<thead>
<tr>
<th>Title</th>
<th>Speakers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson (University of Wisconsin-Madison), Raajay Viswanathan (University of Wisconsin-Madison), Aditya Akella (University of Wisconsin-Madison), Ratul Mahajan (Microsoft Research)</td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoenescu (University Politehnica of Bucharest), Matei Popovici (University Politehnica of Bucharest), Lorina Negreanu (University Politehnica of Bucharest), Costin Raiciu (University Politehnica of Bucharest)</td>
</tr>
<tr>
<td><strong>Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett (Princeton University), Ratul Mahajan (Microsoft), Todd Millstein (University of California, Los Angeles), Jitendra Padhye (Microsoft), David Walker (Princeton University)</td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen (Hebrew University), Yossi Gilad (Boston University and MIT), Amir Herzberg (Bar Ilan University), Michael Schapira (Hebrew University)</td>
</tr>
</tbody>
</table>
Session Preview

1:30pm - 3:10pm Session 6 - Verification
Session Chair: Ramesh Govindan *(University of Southern California)*
Room: Diamante

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>, Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>, Aditya Akella <em>(University of Wisconsin-Madison)</em>, Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoenescu <em>(University Politehnica of Bucharest)</em>, Matei Popovici <em>(University Politehnica of Bucharest)</em>, Lorina Negreanu <em>(University Politehnica of Bucharest)</em>, Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don’t Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>, Ratul Mahajan <em>(Microsoft)</em>, Todd Millstein <em>(University of California, Los Angeles)</em>, Jitendra Pachye <em>(Microsoft)</em>, David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>, Yossi Gilad <em>(Boston University and MIT)</em>, Amir Herzberg <em>(Bar Ilan University)</em>, Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
Symbolic Execution
Symbolic Execution

\[
x = y \times z; \\
\text{if}(x > 0) \\
\quad z = 10; \\
\text{else} \\
\quad z = -10; \\
y = z \times x;
\]
Symbolic Execution

\[
x = y \times z; \\
\text{if}(x > 0) \\
\quad z = 10; \\
\text{else} \\
\quad z = -10; \\
y = z \times x;
\]

\[
x = x0 \\
y = y0 \\
z = z0
\]
Symbolic Execution

\[ x = y \times z; \]
\[ \text{if}(x > 0) \]
\[ \quad z = 10; \]
\[ \text{else} \]
\[ \quad z = -10; \]
\[ y = z \times x; \]

- \[ x = y_0 \times z_0 \]
- \[ y = y_0 \]
- \[ z = z_0 \]
Symbolic Execution

\[ x = y \times z; \]
\[ \text{if}(x > 0) \]
\[ \quad z = 10; \]
\[ \text{else} \]
\[ \quad z = -10; \]
\[ y = z \times x; \]

\[ x = y_0 \times z_0 \]
\[ y = y_0 \]
\[ z = 10 \]
Symbolic Execution

\[
x = y \times z; \\
\text{if}(x > 0) \\
\quad z = 10; \\
\text{else} \\
\quad z = -10; \\
y = z \times x;
\]

- \(x = y_0 \times z_0\)
- \(y = y_0\)
- \(z = 10\)

- \(x = y_0 \times z_0\)
- \(y = y_0\)
- \(z = -10\)
\( x = y \times z; \)

\begin{align*}
\text{if}(x > 0) & \quad z = 10; \\
\text{else} & \quad z = -10;
\end{align*}

\( y = z \times x; \)

\( x = y_0 \times z_0 \)

\( y = y_0 \)

\( z = y_0 \times z_0 > 0 \ ? \ 10 : -10 \)
Symbolic Execution

\[
x = y \times z; \\
\text{if}(x > 0) \\
\quad z = 10; \\
\text{else} \\
\quad z = -10; \\
y = z \times x;
\]

\[
x = y_0 \times z_0 \\
y = (y_0 \times z_0 > 0 ? 10 : -10) \times y_0 \times z_0 \\
z = y_0 \times z_0 > 0 ? 10 : -10
\]
Symbolic Execution

- Symbolic state provides pure mathematical representation of program
- Simplifies verification, e.g. "Is y always positive?"
- For net: "Is this tunnel correctly encapsulated / decapsulated?"

```
x = y * z;
if(x > 0)
    z = 10;
else
    z = -10;
y = z * x;
```

```
x = y0 * z0
y = (y0 * z0 > 0 ? 10 : -10) * y0 * z0
z = y0 * z0 > 0 ? 10 : -10
```
Symbolic Exec for Networks

Spec

Current techniques do not scale

Check Sym State

Switch Impl in C
Scaling Symbolic Exec for Networks

Spec → Check Sym State

Model → Testing Fidelity

Switch Impl in C
Scaling Symbolic Exec for Networks

Reachability, tunneling, encryption

SEFL model of network

Check formula over symbolic state

Model fidelity, SEFL evaluator
Scaling Symbolic Exec for Networks

Questions:

1. Could router implementations be synthesized from SEFL models?

2. How could we add support for unbounded loops in SEFL?

3. Can we incrementalize evaluation to speed up checking when net changes?
### Session Preview

**1:30pm - 3:10pm Session 6 - Verification**

**Session Chair:** Ramesh Govindan *(University of Southern California)*  
**Room:** Diamante

<table>
<thead>
<tr>
<th>Session Title</th>
<th>Speakers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>, Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>, Aditya Akella <em>(University of Wisconsin-Madison)</em>, Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoinescu <em>(University Politehnica of Bucharest)</em>, Matei Popovici <em>(University Politehnica of Bucharest)</em>, Lorina Negreanu <em>(University Politehnica of Bucharest)</em>, Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>, Ratul Mahajan <em>(Microsoft)</em>, Todd Millstein <em>(University of California, Los Angeles)</em>, Jitendra Padhye <em>(Microsoft)</em>, David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>, Yossi Gilad <em>(Boston University and MIT)</em>, Amir Herzberg <em>(Bar Ilan University)</em>, Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
## Session Preview

1:30pm - 3:10pm Session 6 - Verification

**Session Chair:** Ramesh Govindan *(University of Southern California)*  
**Room:** Diamante

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>, Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>, Aditya Akella <em>(University of Wisconsin-Madison)</em>, Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoinescu <em>(University Politehnica of Bucharest)</em>, Matei Popovici <em>(University Politehnica of Bucharest)</em>, Lorina Negreanu <em>(University Politehnica of Bucharest)</em>, Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>, Ratul Mahajan <em>(Microsoft)</em>, Todd Millstein <em>(University of California, Los Angeles)</em>, Jitendra Pachyae <em>(Microsoft)</em>, David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>, Yossi Gilad <em>(Boston University and MIT)</em>, Amir Herzberg <em>(Bar Ilan University)</em>, Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
The Border Gateway Protocol
Configuring BGP is HARD

Errors happen because:
- distributed system
- low-level language
- little static analysis
Configuring BGP Is HARD
Compiling Net Policies to BGP

Propane Spec

Generate Impl from Spec!

BGP Router Config

Synthesis
Compiling Net Policies to BGP

Fault-tolerance, preference, filtering

BGP router configurations

Configs \textit{synthesized} from spec!

Correctness of Propane compiler
Compiling Net Policies to BGP

Questions:

1. Could we build routers that just implement Propane?

2. Can we build a Propane to OSPF compiler? How hard would it be to verify the Propane compiler?

3. What if we want to tweak output for features Propane doesn’t consider (e.g. timeouts)?
### Session Preview

1:30pm - 3:10pm Session 6 - Verification

Session Chair: Ramesh Govindan *(University of Southern California)*
Room: Diamante

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>, Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>, Aditya Akella <em>(University of Wisconsin-Madison)</em>, Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoenescu <em>(University Politehnica of Bucharest)</em>, Matei Popovici <em>(University Politehnica of Bucharest)</em>, Lorina Negreanu <em>(University Politehnica of Bucharest)</em>, Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don’t Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>, Ratul Mahajan <em>(Microsoft)</em>, Todd Millstein <em>(University of California, Los Angeles)</em>, Jitendra Padhye <em>(Microsoft)</em>, David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>, Yossi Gilad <em>(Boston University and MIT)</em>, Amir Herzberg <em>(Bar Ilan University)</em>, Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
### Session Preview

1:30pm - 3:10pm Session 6 - Verification

**Session Chair:** Ramesh Govindan *(University of Southern California)*  
**Room:** Diamante

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fast Control Plane Analysis Using an Abstract Representation</strong></td>
<td>Aaron Gember-Jacobson <em>(University of Wisconsin-Madison)</em>,</td>
</tr>
<tr>
<td></td>
<td>Raajay Viswanathan <em>(University of Wisconsin-Madison)</em>,</td>
</tr>
<tr>
<td></td>
<td>Aditya Akella <em>(University of Wisconsin-Madison)</em>,</td>
</tr>
<tr>
<td></td>
<td>Ratul Mahajan <em>(Microsoft Research)</em></td>
</tr>
<tr>
<td><strong>Symnet: scalable symbolic execution for modern networks</strong></td>
<td>Radu Stoenescu <em>(University Politehnica of Bucharest)</em>,</td>
</tr>
<tr>
<td></td>
<td>Matei Popovici <em>(University Politehnica of Bucharest)</em>,</td>
</tr>
<tr>
<td></td>
<td>Lorina Negreanu <em>(University Politehnica of Bucharest)</em>,</td>
</tr>
<tr>
<td></td>
<td>Costin Raiciu <em>(University Politehnica of Bucharest)</em></td>
</tr>
<tr>
<td><strong>Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations</strong></td>
<td>Ryan Beckett <em>(Princeton University)</em>,</td>
</tr>
<tr>
<td></td>
<td>Ratul Mahajan <em>(Microsoft)</em>,</td>
</tr>
<tr>
<td></td>
<td>Todd Millstein <em>(University of California, Los Angeles)</em>,</td>
</tr>
<tr>
<td></td>
<td>Jitendra Pachye <em>(Microsoft)</em>,</td>
</tr>
<tr>
<td></td>
<td>David Walker <em>(Princeton University)</em></td>
</tr>
<tr>
<td><strong>Jumpstarting BGP Security with Path-End Validation</strong></td>
<td>Avichai Cohen <em>(Hebrew University)</em>,</td>
</tr>
<tr>
<td></td>
<td>Yossi Gilad <em>(Boston University and MIT)</em>,</td>
</tr>
<tr>
<td></td>
<td>Amir Herzberg <em>(Bar Ilan University)</em>,</td>
</tr>
<tr>
<td></td>
<td>Michael Schapira <em>(Hebrew University)</em></td>
</tr>
</tbody>
</table>
BGP Insecurity

Traffic Hijacking via Bogus Announcements

Bogus Origin:
- announce you can directly route target prefix
- RPKI prevents by registering who owns what

Bogus Path:
- modify path to entice traffic for target
- BGPsec prevents in principle, hard to adopt
Jump Starting BGP Security

Key Idea:
extend RPKI to additionally register last hop

Real incentive for incremental deployment!
Jump Starting BGP Security

Next-to-last hop is legitimate

BGP networks

Check AS path against RPKI++

ASes do not collude, registry secure
Jump Starting BGP Security

Questions:

1. Can the list of valid last-hops be inferred from configurations?

2. What should an AS do when it detects attempted hijacks? Drop all routes through the attacking AS, inform neighbors?

3. Can insights from this technique be applied to other protocols?
Session Preview

1:30pm - 3:10pm Session 6 - Verification
Session Chair: Ramesh Govindan (University of Southern California)
Room: Diamante

- **Fast Control Plane Analysis Using an Abstract Representation**
  Aaron Gember-Jacobson (University of Wisconsin-Madison), Raajay Viswanathan (University of Wisconsin-Madison), Aditya Akella (University of Wisconsin-Madison), Ratul Mahajan (Microsoft Research)

- **Symnet: scalable symbolic execution for modern networks**
  Radu Stoenescu (University Politehnica of Bucharest), Matei Popovici (University Politehnica of Bucharest), Lorina Negreanu (University Politehnica of Bucharest), Costin Raiciu (University Politehnica of Bucharest)

- **Don’t Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations**
  Ryan Beckett (Princeton University), Ratul Mahajan (Microsoft), Todd Millstein (University of California, Los Angeles), Jitendra Padhye (Microsoft), David Walker (Princeton University)

- **Jumpstarting BGP Security with Path-End Validation**
  Avichai Cohen (Hebrew University), Yossi Gilad (Boston University and MIT), Amir Herzberg (Bar Ilan University), Michael Schapira (Hebrew University)
Verification Primer + Session Preview
<table>
<thead>
<tr>
<th>If you hear ___________</th>
<th>then ask about:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbolic Evaluation</td>
<td>Loop Handling</td>
</tr>
<tr>
<td>Model Checking</td>
<td>Model Fidelity, Check Bounds</td>
</tr>
<tr>
<td>Automation</td>
<td>Manual Annotations</td>
</tr>
<tr>
<td>Manual Proofs / Annotations</td>
<td>Automation</td>
</tr>
<tr>
<td>Reals Modeled as Floats</td>
<td>Rounding Error</td>
</tr>
<tr>
<td>Computing X from Y and Z</td>
<td>Inferring Z from X and Y</td>
</tr>
<tr>
<td>If you hear</td>
<td>then ask about:</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Object Oriented Type System</td>
<td>Subtyping</td>
</tr>
<tr>
<td>Distributed System</td>
<td>Byzantine Faults</td>
</tr>
<tr>
<td>Fully Formal Verification</td>
<td>TCB, Performance</td>
</tr>
<tr>
<td>Fully Automated Verification</td>
<td>Limits of Expressiveness</td>
</tr>
<tr>
<td>New Verified Implementation</td>
<td>Compatibility</td>
</tr>
<tr>
<td>Verifying Existing Code</td>
<td>Scalability, TCB</td>
</tr>
</tbody>
</table>