+

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)*

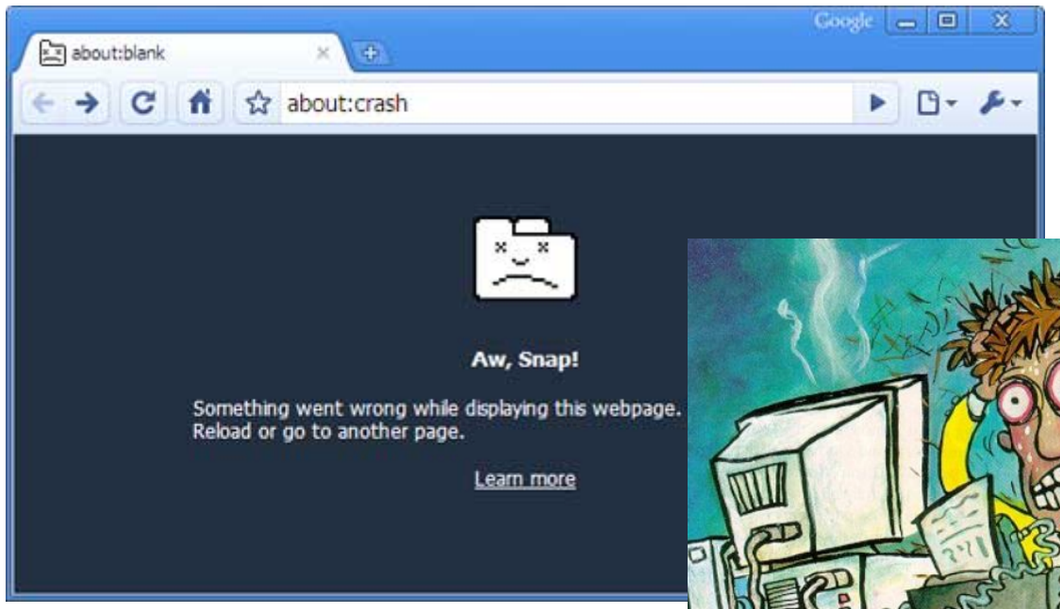
Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations
Ryan Beckett *(Princeton University)*, Ratul Mahajan *(Microsoft)*, Todd Millstein *(University of California, Los Angeles)*, Jitendra Padhye *(Microsoft)*, David Walker *(Princeton University)*

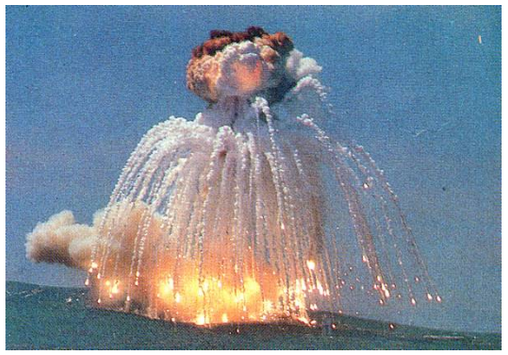
Jumpstarting BGP Security with Path-End Validation
Avichai Cohen *(Hebrew University)*, Yossi Gilad *(Boston University and MIT)*, Amir Herzberg *(Bar Ilan University)*, Michael Schapira *(Hebrew University)*

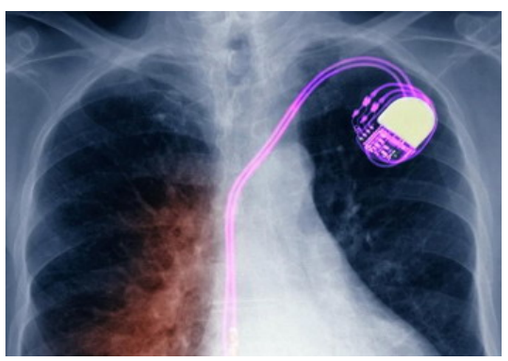
Verification Primer + Session Preview

Zach Tatlock
University of Washington

Bugs = Constant Friction



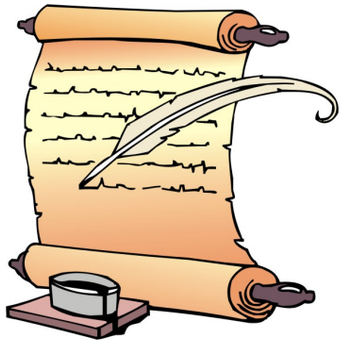




Verification: Reduce Bug Friction

Ultimately, we want to provide a principled methodology that ensures reliability.

Verification: Reduce Bug Friction

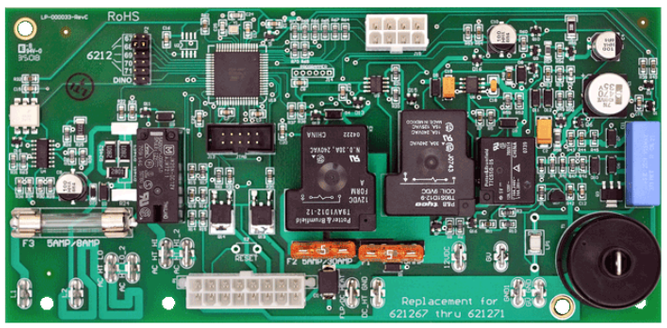


1. Specification

crisply describes ideal behavior

3. Proof

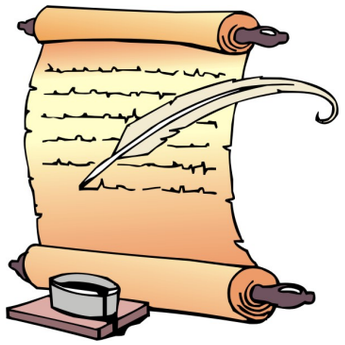
shows impl always agrees w/ spec



2. Implementation

controls actual system behavior

Verification: All About Agreement

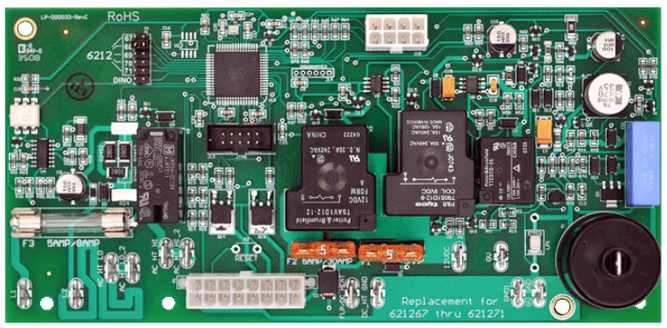


1. Specification

crisply describes ideal behavior

3. Proof

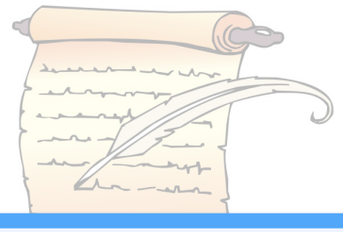
shows impl always agrees w/ spec

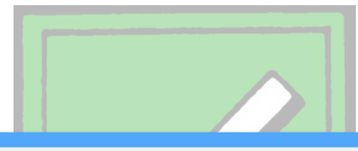


2. Implementation

controls actual system behavior

Verification: All About Agreement





1. *always*

spec

2.

controls actual system behavior

Two Interpretations:

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

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

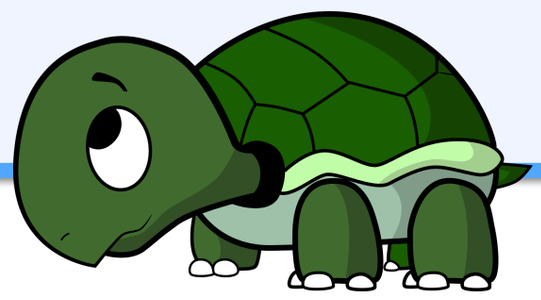
$$P(bug) = P(bug\ in\ spec) * P(bug\ in\ impl)$$

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.

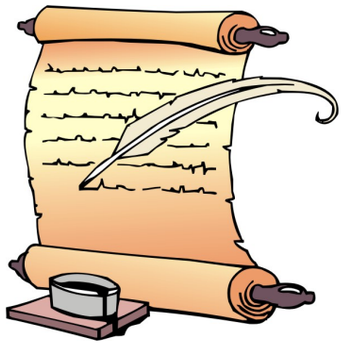
At the end of the lecture, a little old lady at the back of the room got up and said: "What you have told us is rubbish. The world is really a flat plate supported on the back of a giant tortoise." The scientist gave a superior smile before replying, "What is the tortoise standing on?”

"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?



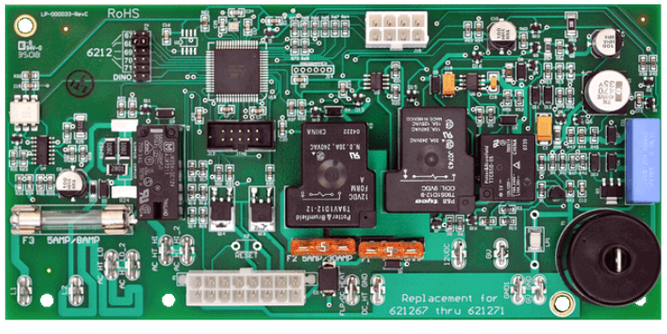
1. Specification

crisply describes ideal behavior

3. Proof

shows impl always agrees w/ spec

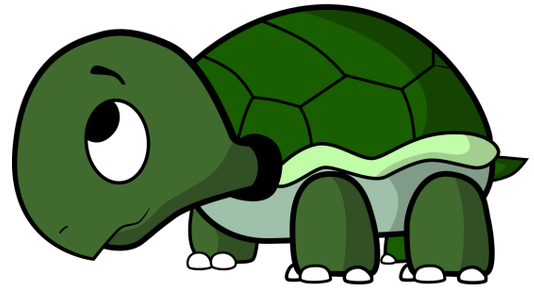
assuming



2. Implementation

controls actual system behavior

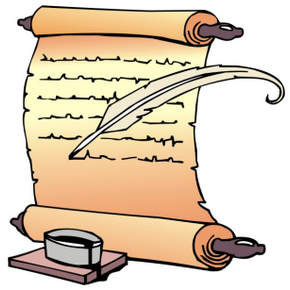
TCB



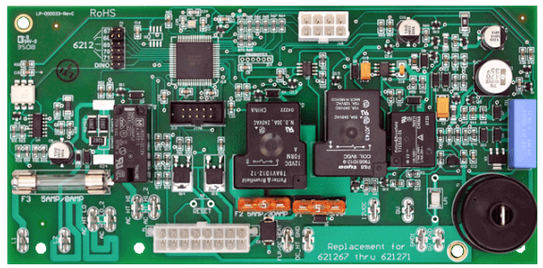
basic assumptions we must trust

Approaching Verification Papers

1. What is specification?

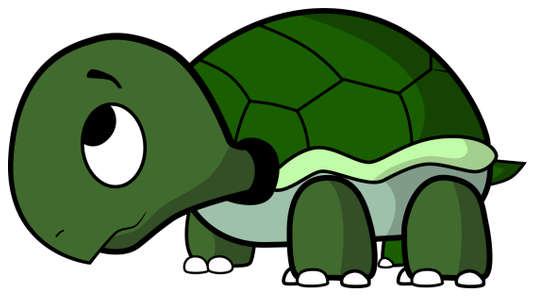


2. How is it implemented?



3. Why do they agree?

4. Should we believe?



Verification Primer

+

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)*

Session Preview

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)*

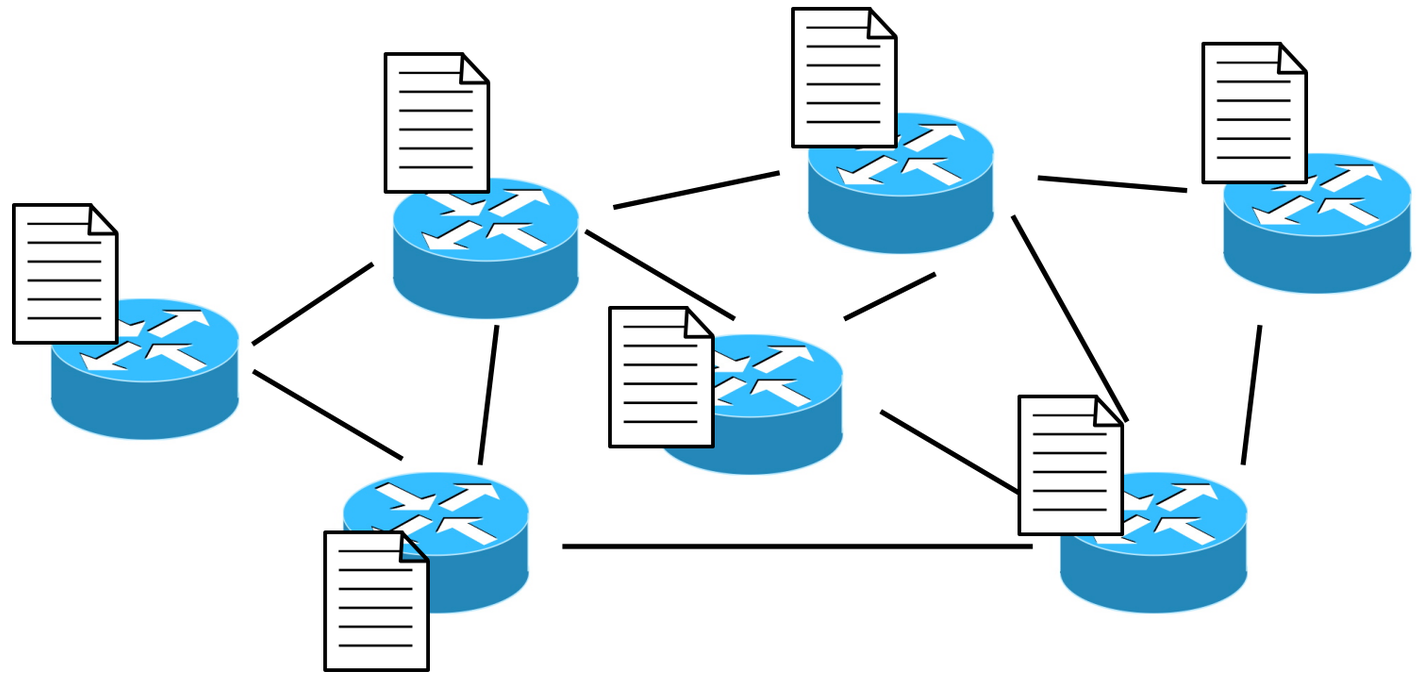
Symnet: scalable symbolic execution for modern networks
Radu Stoenescu *(University Politehnica of Bucharest)*, Matei Popovici *(University Politehnica of Bucharest)*, Lorina Negreanu *(University Politehnica of Bucharest)*, Costin Raiciu *(University Politehnica of Bucharest)*

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)*

Abstraction for Control Planes (ARC)

Control plane config selects and installs routes

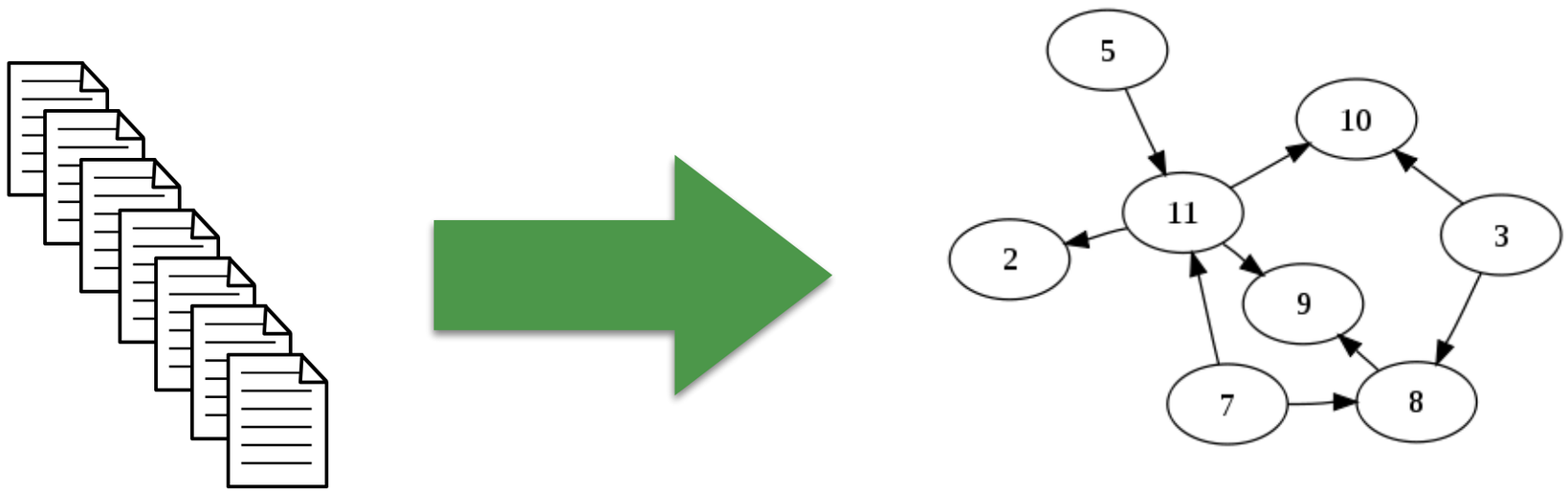


Super difficult to configure correctly:

- may fail to block unwanted traffic
- may not tolerate enough link failure
- may not route through target waypoints

Abstraction for Control Planes (ARC)

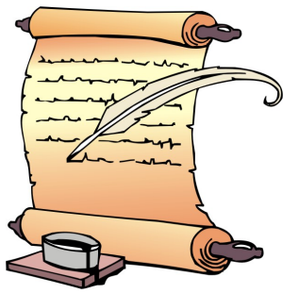
ARCS abstract configs to weighted digraph



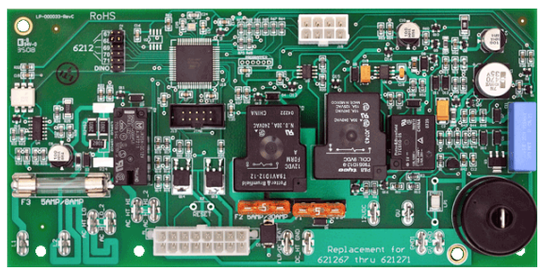
Key property: shortest paths in ARC correspond to network behavior!

Reduces verification to standard graph queries.

Abstraction for Control Planes (ARC)

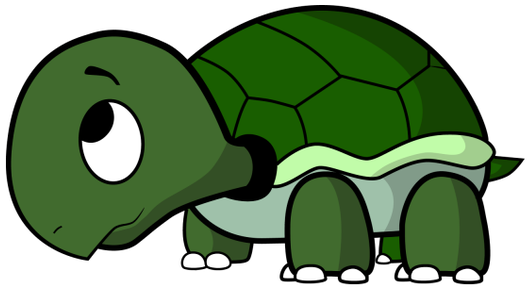


Reachability, isolation, fault-tolerance



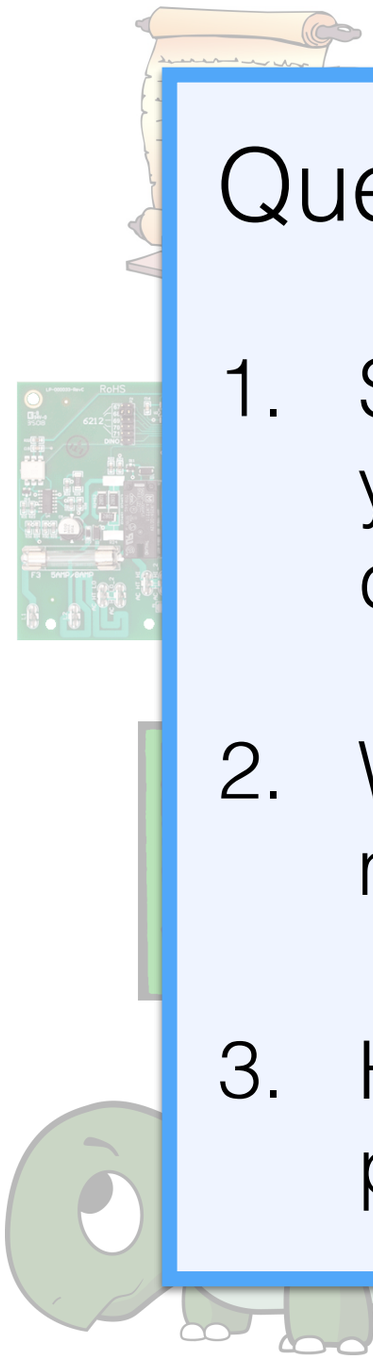
Control plane configurations

Existence of certain paths in ARC



Reduction to ARCs, graph algos

Abstraction for Control Planes (ARC)



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

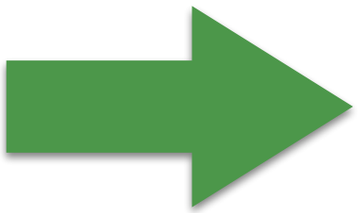
Fast Control Plane Analysis Using an Abstract Representation
Aaron Gember-Jacobson *(University of Wisconsin-Madison)*, Raajay Viswanathan *(University of Wisconsin-Madison)*, Aditya Akella *(University of Wisconsin-Madison)*, Ratul Mahajan *(Microsoft Research)*

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)*

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)*

Symbolic Execution

Symbolic Execution

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

Symbolic Execution

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

x = x0
y = y0
z = z0

Symbolic Execution

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

x = y0 * z0
y = y0
z = z0

Symbolic Execution

```
x = y * z;

if(x > 0)

   z = 10;

else

   z = -10;

y = z * x;
```

x = y0 * z0
y = y0
z = 10

Symbolic Execution

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

x = y0 * z0
y = y0
z = 10

x = y0 * z0
y = y0
z = -10

Symbolic Execution

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

x = y0 * z0
y = y0
z = y0 * z0 > 0 ? 10 : -10

Symbolic Execution

```
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

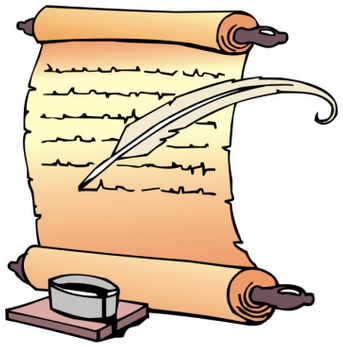
Symbolic Execution

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

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

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

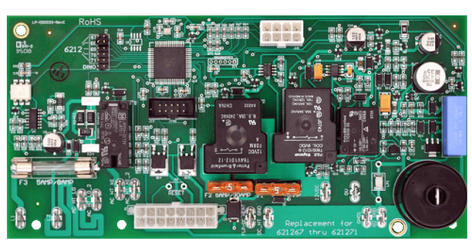
Symbolic Exec for Networks



Spec

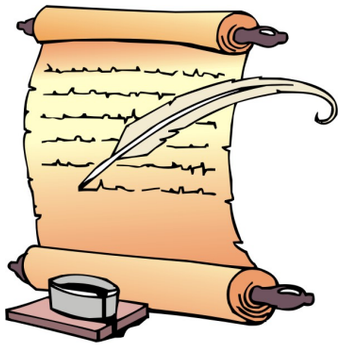
Check Sym State

Current techniques do not scale

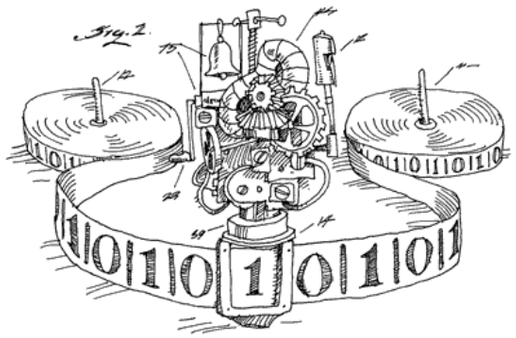


Switch Impl in C

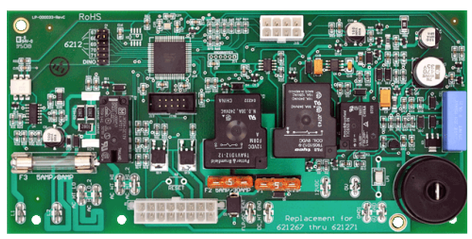
Scaling Symbolic Exec for Networks



Spec

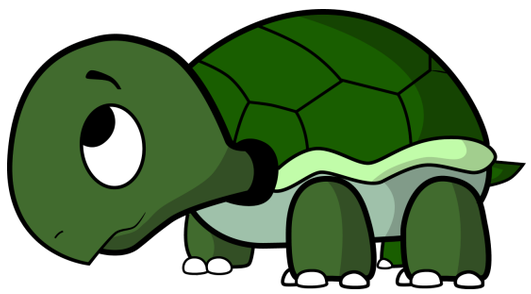


Model



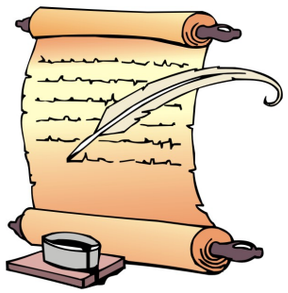
Switch Impl in C

Check Sym State

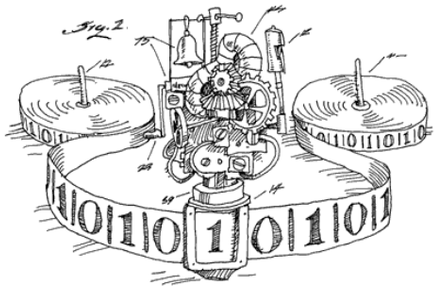


Testing Fidelity

Scaling Symbolic Exec for Networks

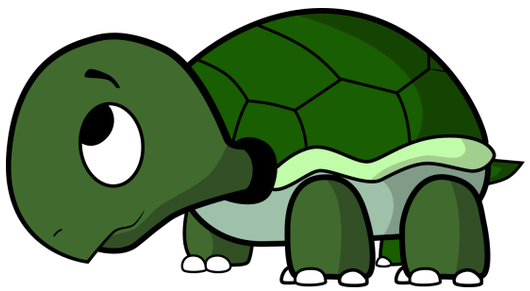


Reachability, tunneling, encryption



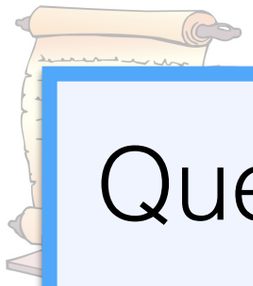
SEFL model of network

Check formula over symbolic state



Model fidelity, SEFL evaluator

Scaling Symbolic Exec for Networks



otion

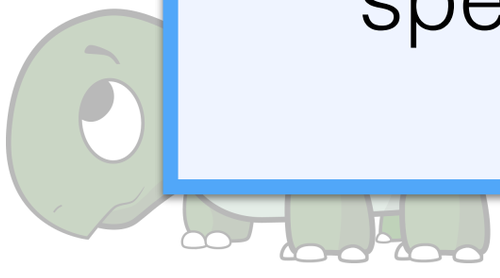
state

Questions:

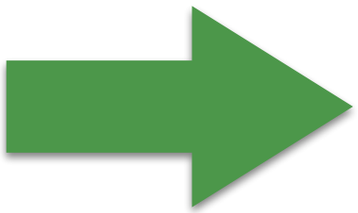
1. Could router implementations be synthesized from SEFL models?

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

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)*

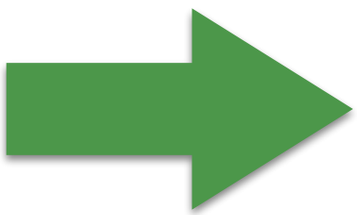
Session Preview

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)*

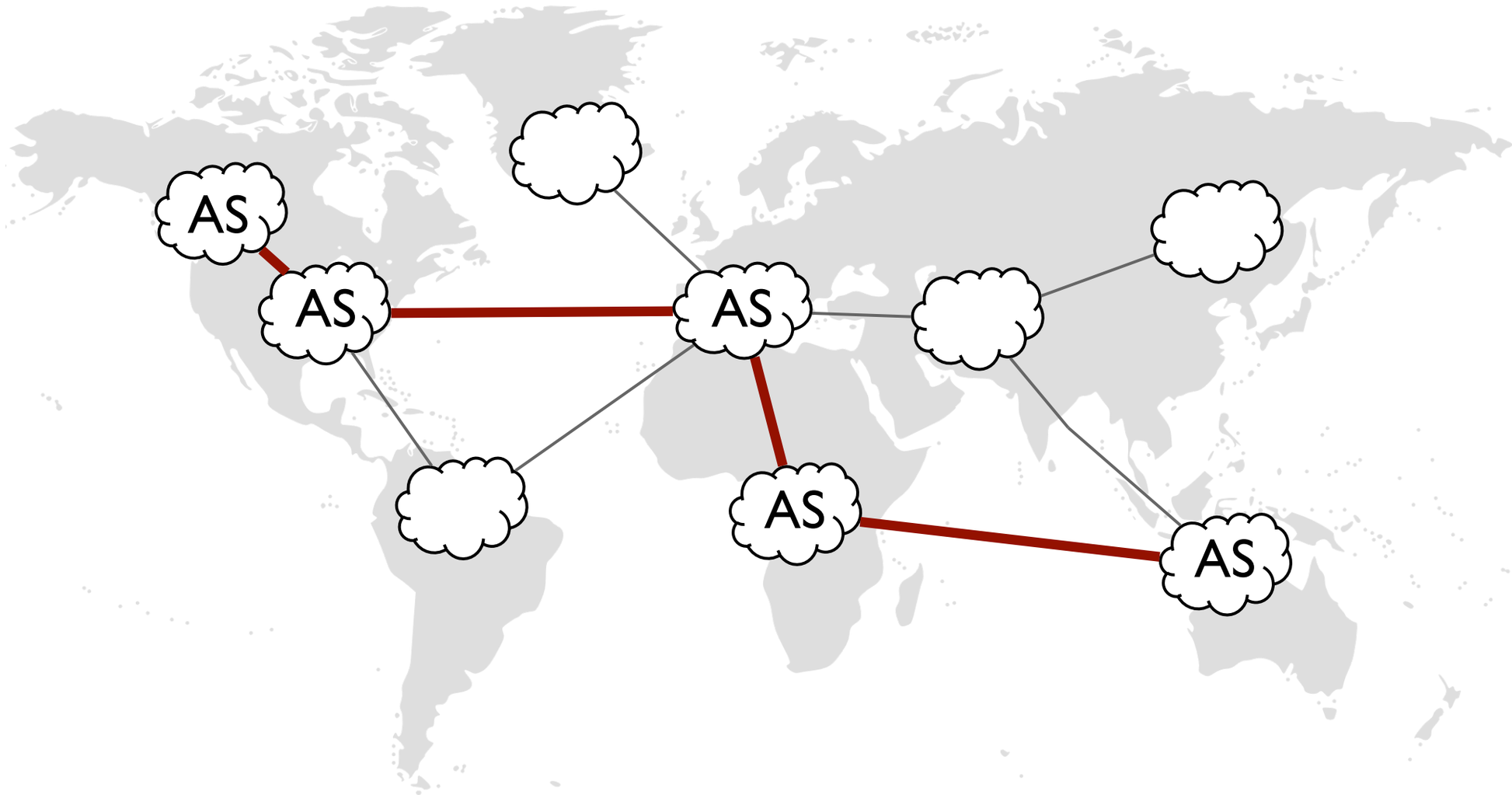
Symnet: scalable symbolic execution for modern networks
Radu Stoenescu *(University Politehnica of Bucharest)*, Matei Popovici *(University Politehnica of Bucharest)*, Lorina Negreanu *(University Politehnica of Bucharest)*, Costin Raiciu *(University Politehnica of Bucharest)*



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)*

The Border Gateway Protocol



Configuring BGP Is HARD

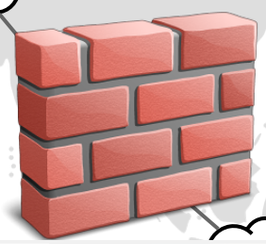
Errors happen because:
- distributed system
- low-level language
- little static analysis

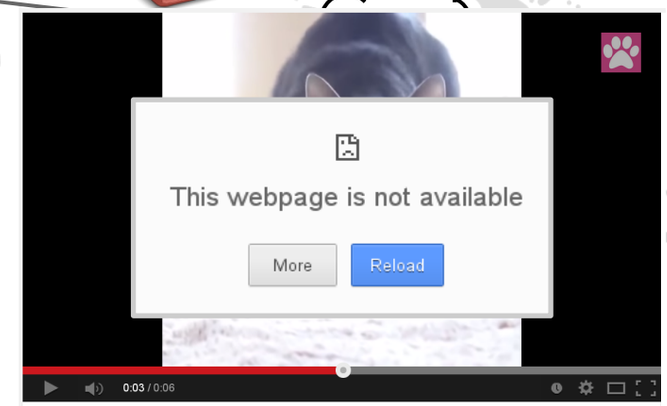
```
configure terminal router
neighbor 8.212.226.20 remote-as 2
neighbor 172.16.254.1 remote-as 3
neighbor 95.112.23.51 remote-as 4

! prefix lists ip prefix-list
outCustomer permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
8.212.226.0/24 le p prefix-list
outProvider permit
```

AS

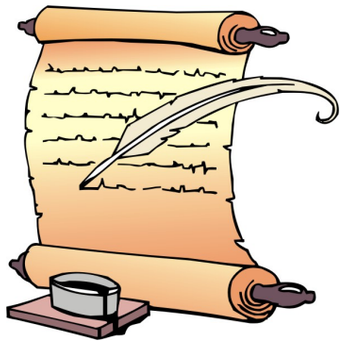
AS

AS

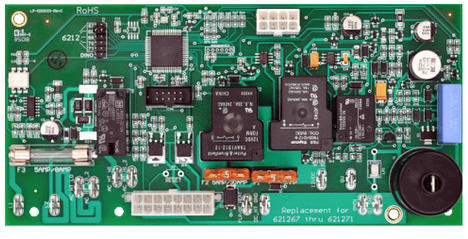




Configuring BGP Is HARD



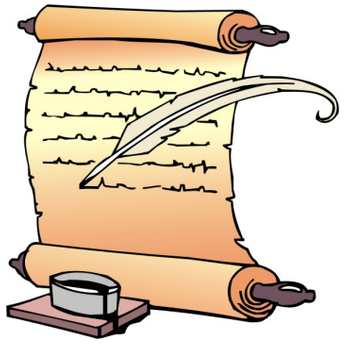
Spec



BGP Router Config

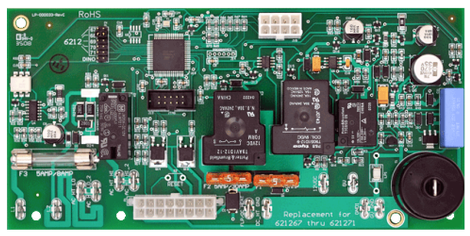
✗ Expensive or Impossible

Compiling Net Policies to BGP



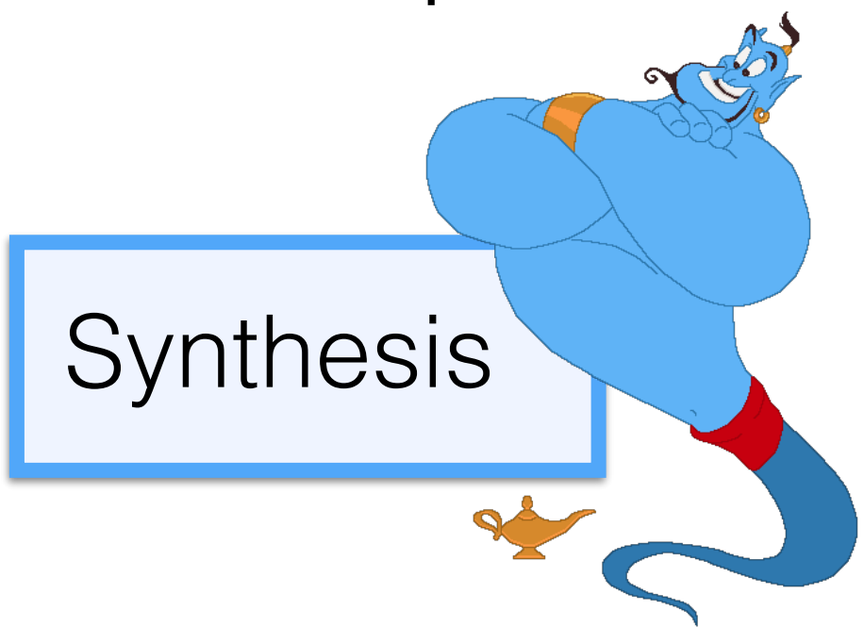
Propane Spec

Generate Impl from Spec!

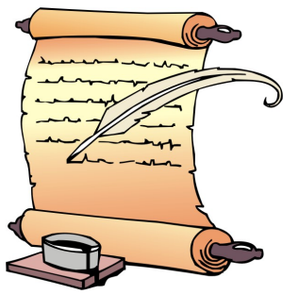


BGP Router Config

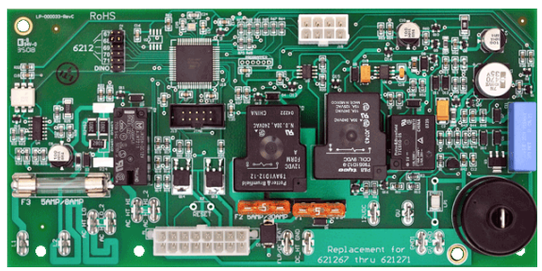
Synthesis



Compiling Net Policies to BGP

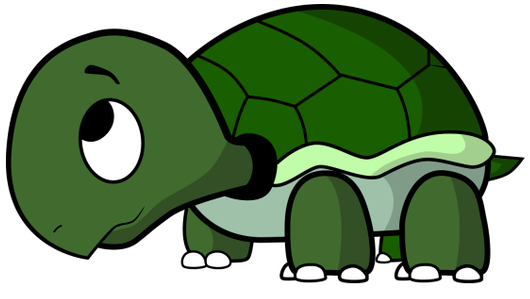


Fault-tolerance, preference, filtering



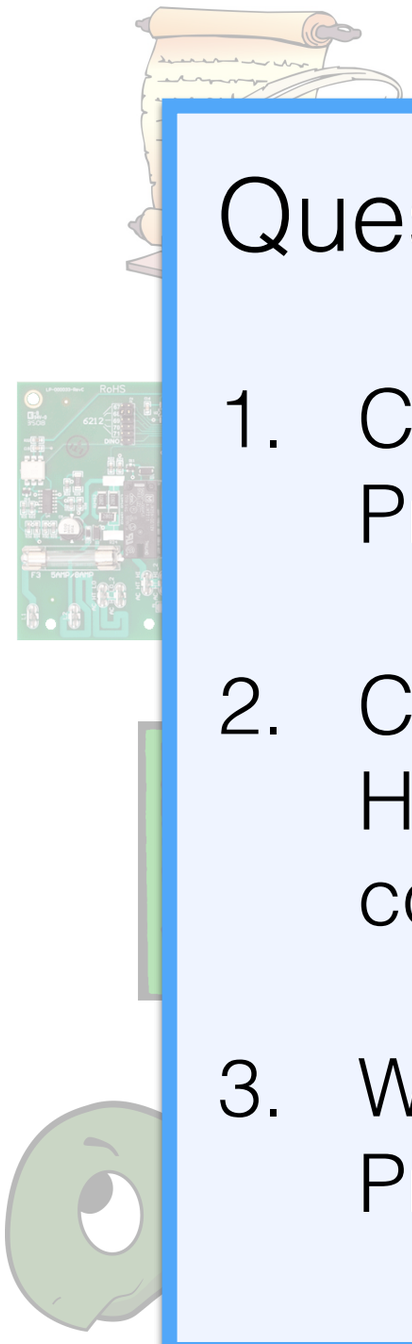
BGP router configurations

Configs *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)?

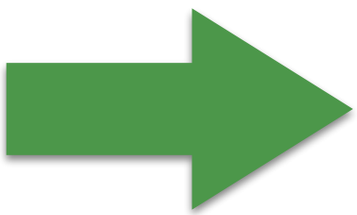
Session Preview

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

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

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)*

Session Preview

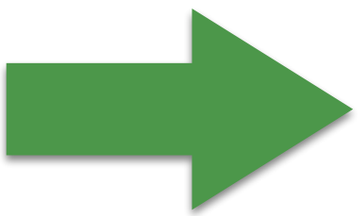
1:30pm - 3:10pm Session 6 - Verification

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)*

BGP Insecurity

Traffic Hijacking via Bogus Announcements

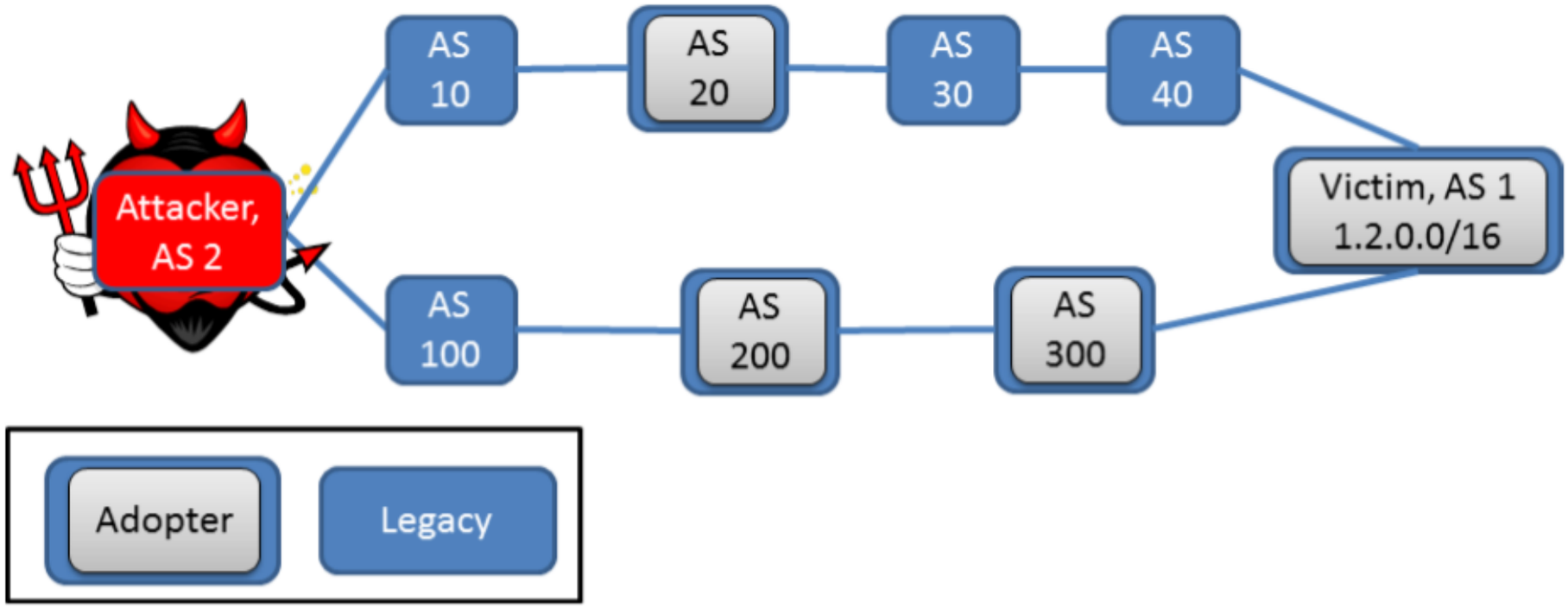
Bogus Origin:
- announce you can directly route target prefix
- RPKI prevents by registering who owns what

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

Jump Starting BGP Security

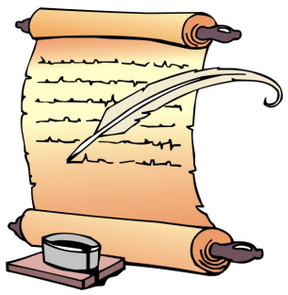
Key Idea:

extend RPKI to additionally register last hop

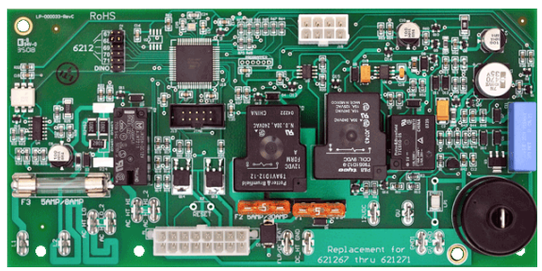


Real incentive for incremental deployment!

Jump Starting BGP Security

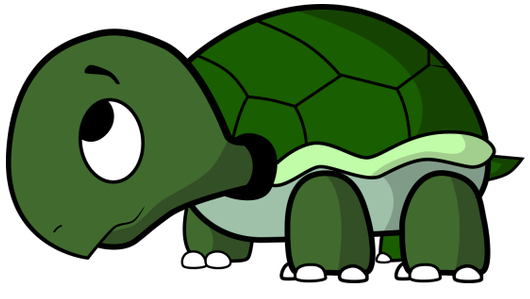


Next-to-last hop is legitimate



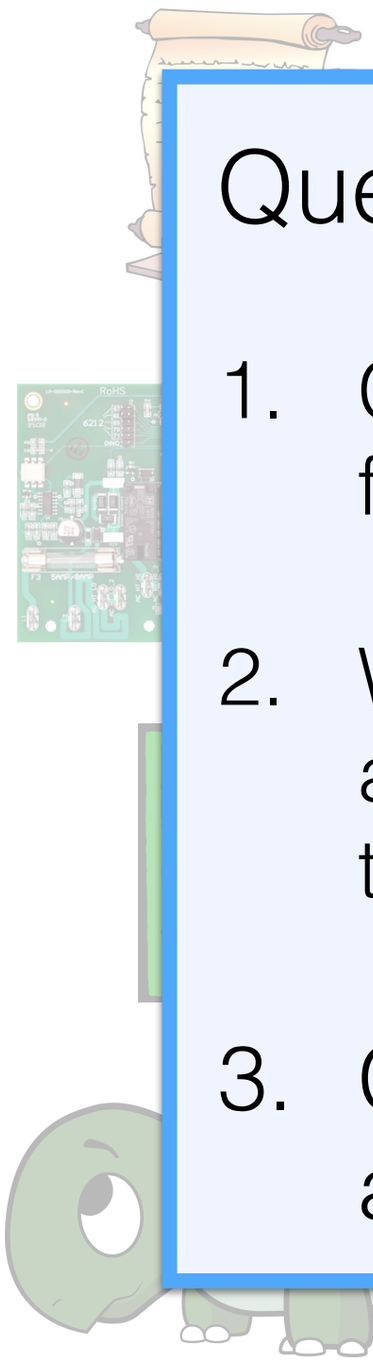
BGP networks

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

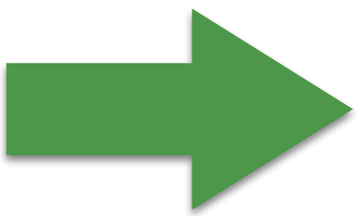
1:30pm - 3:10pm Session 6 - Verification

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

+

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)*

Session Preview

Question Cheat Sheet

If you hear _____	then ask about:
Symbolic Evaluation	Loop Handling
Model Checking	Model Fidelity, Check Bounds
Automation	Manual Annotations
Manual Proofs / Annotations	Automation
Reals Modeled as Floats	Rounding Error
Computing X from Y and Z	Inferring Z from X and Y

Question Cheat Sheet (cont.)

If you hear _____	then ask about:
Object Oriented Type System	Subtyping
Distributed System	Byzantine Faults
Fully Formal Verification	TCB, Performance
Fully Automated Verification	Limits of Expressiveness
New Verified Implementation	Compatibility
Verifying Existing Code	Scalability, TCB