Bagpipe
Scaleable BGP Configuration Verification

Konstantin Weitz
Doug Woos
Michael D. Ernst
Arvind Krishnamurthy
Emina Torlak
Zachary Tatlock

University of Washington
CSNW
APLSE
The Border Gateway Protocol

http://www.iamsterdam.com/
The Border Gateway Protocol

http://www.iamsterdam.com/

I am visiting

me

ISP

ISP

ISP

ISP
The Border Gateway Protocol
The Border Gateway Protocol

http://www.iamsterdam.com/
The Border Gateway Protocol
The Border Gateway Protocol
The Border Gateway Protocol
The Border Gateway Protocol
The Border Gateway Protocol
The Border Gateway Protocol

Errors happen because:

- low-level language
- distributed system
- little static analysis

ISP
ISP
ISP
The Border Gateway Protocol

Errors happen because:

- low-level language
- distributed system
- little static analysis
The Border Gateway Protocol

Errors happen because:

- low-level language
- distributed system
- little static analysis
The Border Gateway Protocol
The Bagpipe Algorithm

- Specification
- Configuration
- Bagpipe

Policy Violation
Policy Holds
Overview

BGP Background

BGP Specification Language

Scaling Verification to > 100k LOC

Evaluation with Real Configurations
BGP Background
BGP Background
Routing Table

BGP Background
Routing Table

BGP Background
configure terminal router bgp 1
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 outC
  permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
  8.212.226.0/24 in p prefix-list outC

Routing Table

import
Routing Table

configure terminal
router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
8.212.226.0/24 to 0

BGP Background
import

configure terminal
router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit 8.212.226.0/24 le p prefix-list ou

export

BGP Background
configure terminal
router bgp 1
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-list ip prefix-list outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit 8.212.226.0/24 in
ip prefix-list outPeer permit 172.16.254.0/24 in
ip prefix-list outPeer permit 95.112.23.0/24 in

BGP Background
import

import

export

export

configure terminal
router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
8.212.226.0/24 le p prefix-list ou

BGP Background
configure terminal
router bgp 1
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 outC
   permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit 8.212.226.0/24 in p prefix-list ou

BGP Background
import

import

export

export

configure terminal
router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
8.212.226.0/24 le p
ip prefix-list outPeer permit
8.212.226.0/24 le p
ip prefix-list outC permit
import
import
export
export

configure terminal
router bgp 1
neighbor 8.212.226.20 remote-as 2
neighbor 172.16.254.1 remote-as 3
neighbor 9.51.122.23.6 remote-as 4

! prefix lists
ip prefix-list outC
  permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
  8.212.226.0/24 le p
  prefix-list ou

BGP Background
Import/Export Specifications
Import/Export Specifications

import? : A → A ∪ {drop} → bool

import?(       ,        ) = true
Import/Export Specifications

import? : A → A ∪ {drop} → bool

import?(a, a') = true

import?(a, a') :=
    if destination(a) = localhost
    then a' = ❌
    else true
**Import/Export Specifications**

\[
\text{import?} : A \rightarrow A \cup \{\text{drop}\} \rightarrow \text{bool}
\]

\[
\text{import?}(a, a') = \text{true}
\]

\[
\text{import?}(a, a') :=
\]

\[
\begin{align*}
\text{if } \text{destination}(a) &= \text{localhost} \\
\text{then } a' &= \text{false}
\end{align*}
\]

\[
\text{else } a' &= \text{true}
\]
Import/Export Specifications

import? : A → A ∪ \{\text{drop}\} → \text{bool}

import?(a, a') = true

export? : A → A ∪ \{\text{drop}\} → \text{bool}

export?(a, a') = true

import?(a, a') :=
if destination(a) = localhost
then a' = \text{x}
else true
Import/Export Specifications

import? : A → A ∪ {drop} → bool

∀ t:trace,
∀ a:received(t),
import?(a, import(a)) ∧
∀ a:selected(t),
export?(a, export(a))

export? : A → A ∪ {drop} → bool

∀ t:trace,
∀ a:received(t),
import?(a, import(a)) ∧
∀ a:selected(t),
export?(a, export(a))
Selection Specifications

import

import

import
Selection Specifications

import

import

import
Selection Specifications

\[ \leq : A \to A \to \text{bool} \]
Selection Specifications

\[ \leq : A \rightarrow A \rightarrow \text{bool} \]

\[ \leq \leq = \text{true} \]

\[ \leq \leq = \text{true} \]
Selection Specifications

\[ a \leq a' := \text{revenue}(a) \leq \text{revenue}(a') \]

\[ \leq : A \rightarrow A \rightarrow \text{bool} \]

\[ \leq \leq \text{true} = \text{true} \]
Selection Specifications

\[ \leq : A \rightarrow A \rightarrow \text{bool} \]

\[ \forall t:\text{trace}, \forall a:\text{received}(t), \forall a':\text{selected}(t), a \leq a' \]

\[ a \leq a' := \text{revenue}(a) \leq \text{revenue}(a') \]
Specifications Summary

import? : A \to A \cup \{\text{drop}\} \to \text{bool}
export? : A \to A \cup \{\text{drop}\} \to \text{bool}
\leq : A \to A \to \text{bool}

\forall t:\text{trace},
\forall a:\text{received}(t), \ \text{import?}(a, \text{import}(a)) \land
\forall a:\text{selected}(t), \ \text{export?}(a, \text{export}(a)) \land
\forall a:\text{received}(t), \ \forall a':\text{selected}(t), a \leq a'
**Bagpipe**

Scaling Verification

- Specification
- Configuration

Policy Violation

Policy Holds
∀ t:trace( ),
  ∀ a:received(t),
    import? (a, import (a)) ∧
  ∀ a:selected(t),
    export? (a, export (a)) ∧
  ∀ a:received(t),
    ∀ a’:selected(t),
    a ≤ a’
Considering All Traces

∀ t:trace, ∀ a, import?(a, import(a)) ∧ ∀ a, export?(a, export(a)) ∧ ∀ a:received(t), ∀ a':selected(t), a ≤ a'
Considering All Traces

∀ t:trace, 
∀ a, import?(a, import(a)) \land 
∀ a, export?(a, export(a)) \land 
∀ a:received(t), 
∀ a':selected(t), a \leq a'
Considering All Traces

∀ t:trace, ∀ a, import?(a, import(a)) ∧ 
∀ a, export?(a, export(a)) ∧ 
∀ a:received(t), 
∀ a’:selected(t), a ≤ a’
Considering All Traces

∀ t:trace,
∀ a, import?(a, import(a)) \∧
∀ a, export?(a, export(a)) \∧
∀ a:received(t),
∀ a':selected(t), a ≤ a'}
Initial Network Reduction

Check announcements independently in the empty initial network
∀ t: initTrace,
∀ a, import?(a, import(a)) \∧
∀ a, export?(a, export(a)) \∧
∀ a: received(t),
∀ a': selected(t), a ≤ a'
∀ t: initTrace,
∀ a, import?(a, import(a)) \∧
∀ a, export?(a, export(a)) \∧
∀ a: received(t),
∀ a’: selected(t), a ≤ a’
∀ t:initTrace,
∀ a, import?(a, import(a)) \∧
∀ a, export?(a, export(a)) \∧
∀ a:received(t),
∀ a':selected(t), a \leq a'
Initial Network Reduction

∀ t: initTrace,
∀ a, import?(a, import(a)) ∧
∀ a, export?(a, export(a)) ∧
∀ a: received(t),
∀ a’: selected(t), a ≤ a’
Initial Network Reduction

∀ t:initTrace,
∀ a, import?(a, import(a)) \land
∀ a, export?(a, export(a)) \land
∀ a:received(t),
∀ a':selected(t), a \leq a'

finite
∀ t:trace,
∀ a, import?(a, import(a)) \∧
∀ a, export?(a, export(a)) \∧
∀ a:received(t),
∀ a’:selected(t), a ≤ a’
Problem: BGP is Stateful

Some announcements may only be forwarded in the non-empty network
Selection is Stateful

Are some announcements only selected in the non-empty network?
Selection is Stateful

Are some announcements only selected in the non-empty network?
Selection is Stateful

Are some announcements only selected in the non-empty network?
Selection is Stateful

If ever selected, announcement is selected in the empty network!

Are some announcements only selected in the non-empty network?
Are some announcements only exported in the non-empty network?
Export is Stateful

Are some announcements only exported in the non-empty network?
Export is Stateful

Are some announcements only exported in the non-empty network?
Export is Stateful

If ever exported, announcement is exported in the empty network!

Are some announcements only exported in the non-empty network?
BGP has **maximal forwarding behavior** in the empty initial network!
configure terminal

router bgp 1

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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit 8.212.226.0/24 le
∀ t:trace(*),
    ∀ a:received(t),
        import? (a, import_ (a)) ∧
    ∀ a:selected(t),
        export? (a, export_ (a)) ∧
    ∀ a:received(t),
        ∀ a’:selected(t),
        a ≤ a’
configure terminal router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit 8.212.226.0/24 le
∀ t:initTrace(
∀ a:received(t),
   import? (a, import(a)) ∧
∀ a:selected(t),
   export? (a, export(a)) ∧
∀ a:received(t),
   ∀ a’:selected(t),
   a ≤ a’
)
Policy Violation

Configuration

Specification

Parallelization

Policy Holds

Policy Violation

Bagpipe Algorithm
configure terminal
router bgp 1
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 outC
permit 0.0.0.0/0 le 32
ip prefix-list outPeer permit
8.212.226.0/24 le prefix-list outPeer permit
evaluate policy violation

Bagpipe

Policy Holds

Policy Violation
Evaluation

- 10 Juniper Scenarios
- No Martian
- Block To External
- Gao & Rexford

- Internet2 >100K
- BelWü >200K
- Selfnet >50

Bagpipe

14 Policies
Hold

19 Policy Violations,
No False Positives
Bagpipe Specification Evaluation

Internet2

Policy Holds

Policy Violation
Bagpipe

Found 5 Leaks

Block To External

Internet2

5 leaks 8h
Found 14 Problems

Gao & Rexford

Internet2

14 problems
3 days

Bagpipe
Related Work

• Propane: high-level BGP configuration language
  Beckett et.al. SIGCOMM’16

• Batfish: BGP verification with given traces
  Fogel et.al. NSDI’15

• NetKAT: high-level SDN configuration language
  Anderson et.al. POPL’14

• Header Space Analysis: packet forwarding verification
  Kazemian et.al. NSDI’12
Contributions

BGP Specification Language treats BGP networks as programs

Scaling Verification to $> 100k$ LOC with initial network reduction, symbolic execution, parallelization

Evaluation with Real Configurations verified correctness, found violations

http://bagpipe.uwplse.org