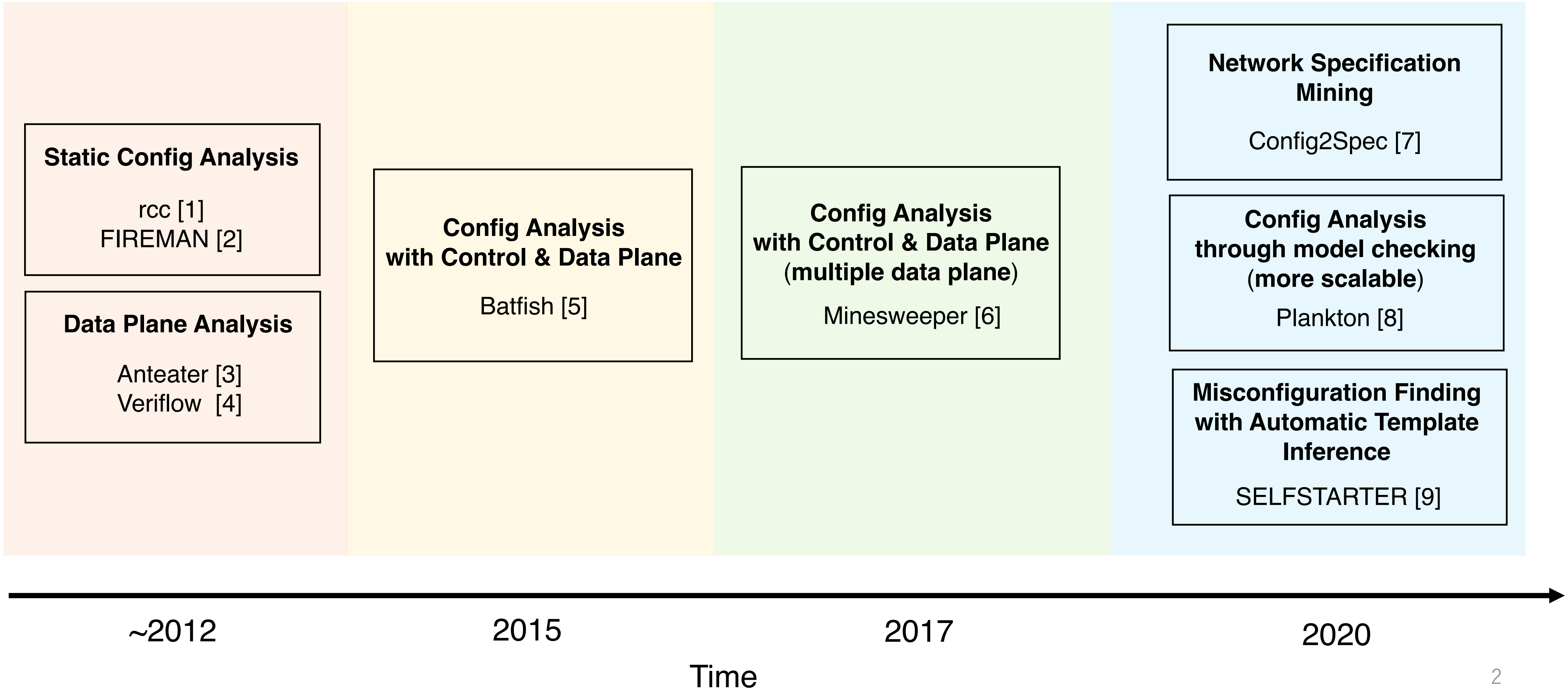
Survey v2 in Network Config Analytics

siiba

Recap of the survey in last week



Static Config Analysis

rcc [1]
FIREMAN [2]

Data Plane Analysis

Anteater [3]
Veriflow [4]

Config Analysis with Control & Data Plane

Batfish [5]

Config Analysis with Control & Data Plane (multiple data plane)

Minesweeper [6]

Network Specification Mining

Config2Spec [7]

Config Analysis through model checking (more scalable)

Plankton [8]

Misconfiguration Finding with Automatic Template Inference

SELFSTARTER [9]

Today’s theme

1. Grasp **recent** research efforts in the config analytics

2. Discuss about the research directions in the future

Papers in this presentation

Mining network specifications in the configuration

- Config2Spec

Incremental Network Configuration Verification [10]

- RealConfig

Proactive verification of DNS configurations [11]

- GROOT

Papers in this presentation

Mining network specifications in the configuration

- Config2Spec

Incremental Network Configuration Verification [10]

- RealConfig

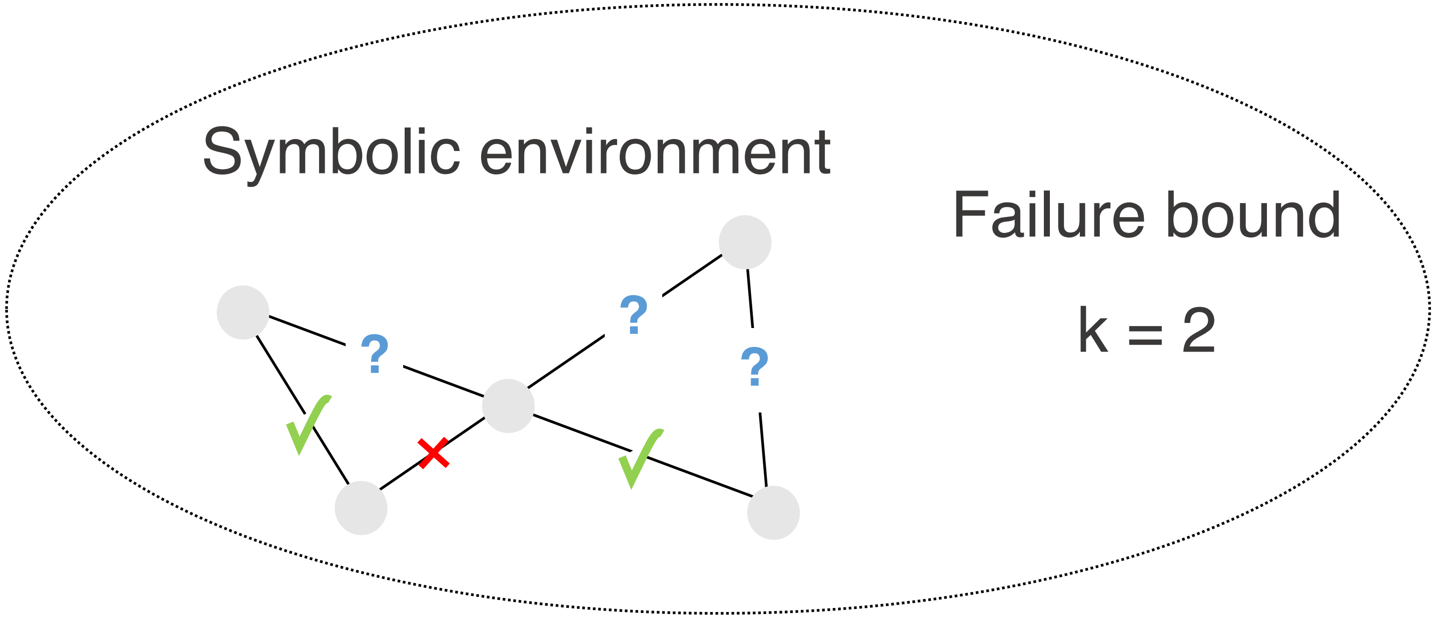
Proactive verification of DNS configurations [11]

- GROOT

Definition

The specification of a network

→ set of all policies that hold under a given failure model



Failure Model

Set of policies

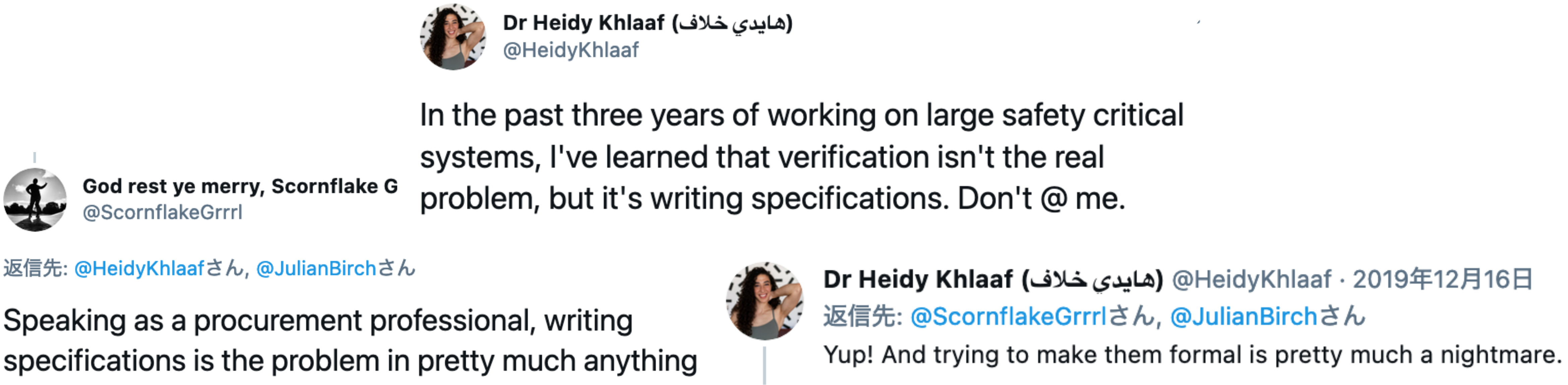
reachability(s1,d1)
reachability(s2,d1)
waypoint(s3,r1,d2)
…
loadbalancing(r3,d1)

Background and Motivation

Network verification is important to make networks more reliable and secure

on the other hand ...

Writing formal and precise specifications is hard



Dr Heidy Khlaaf (هايدي خلاف)
@HeidyKhlaaf

In the past three years of working on large safety critical systems, I've learned that verification isn't the real problem, but it's writing specifications. Don't @ me.

God rest ye merry, Scornflake G
@ScornflakeGrrrl

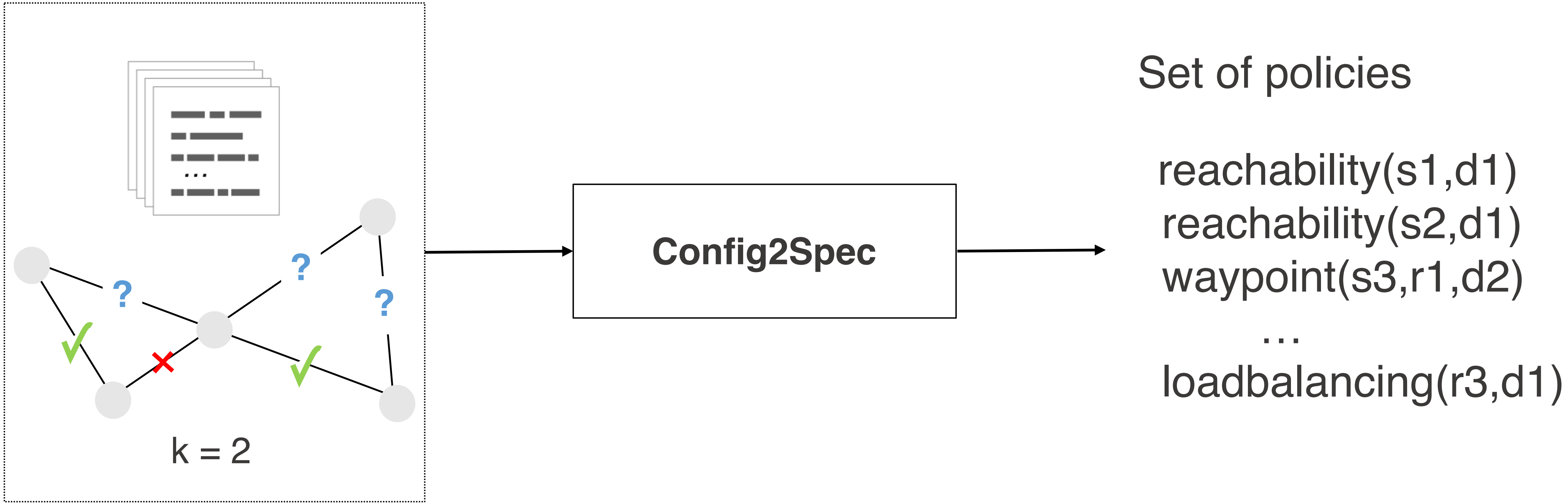
返信先: @HeidyKhlaafさん, @JulianBirchさん

Speaking as a procurement professional, writing specifications is the problem in pretty much anything

Dr Heidy Khlaaf (هايدي خلاف) @HeidyKhlaaf · 2019年12月16日
返信先: @ScornflakeGrrrlさん, @JulianBirchさん
Yup! And trying to make them formal is pretty much a nightmare.

Approach

Automatically mine the network’s full specification from its configuration and the given failure model



Config2Spec

Set of policies

reachability(s1,d1)
reachability(s2,d1)
waypoint(s3,r1,d2)
…
loadbalancing(r3,d1)

How to automatically mine the specification

Combine the strength of data plane analysis and control plane verification

Data plane analysis

all policies for

one concrete env.

→ prune the large space of polices

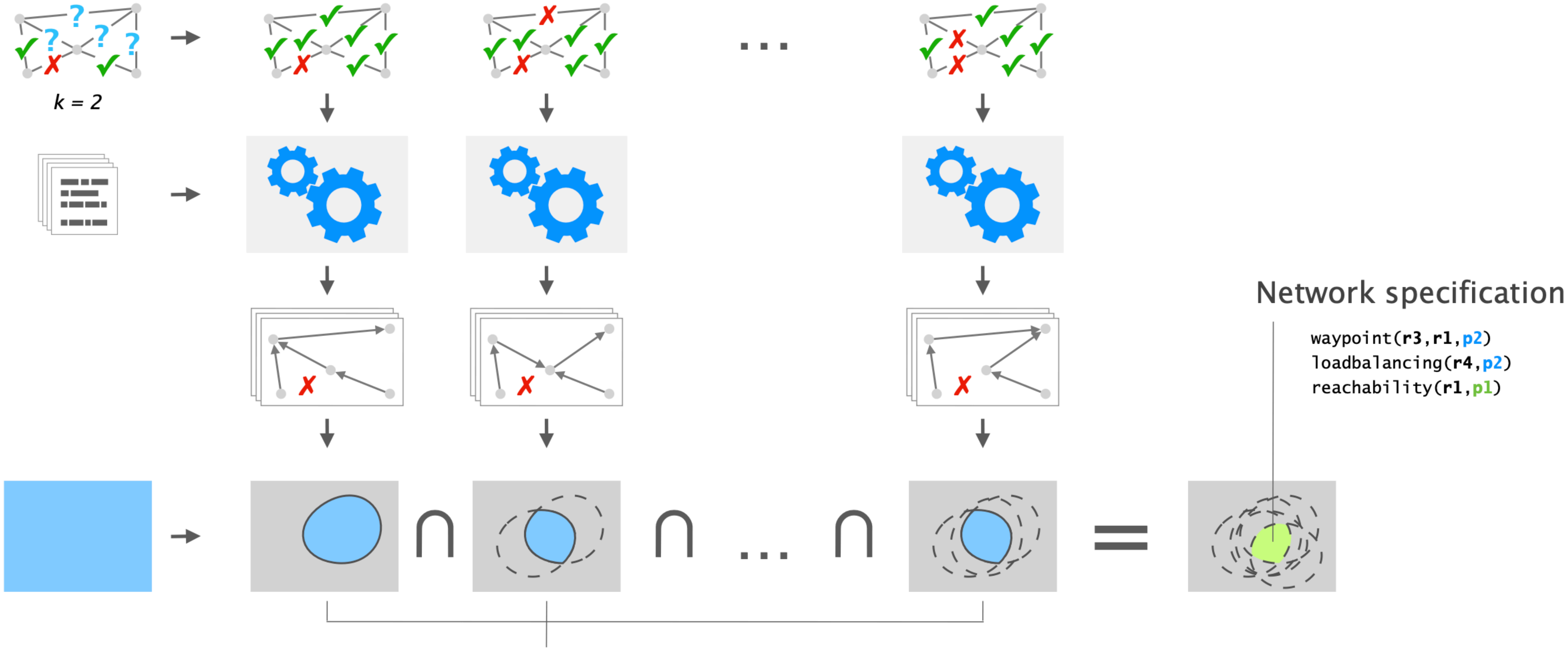
Control plane verification

one policy for

entire failure model

→ validate the remaining polices

Pruning the policies with data plane analysis

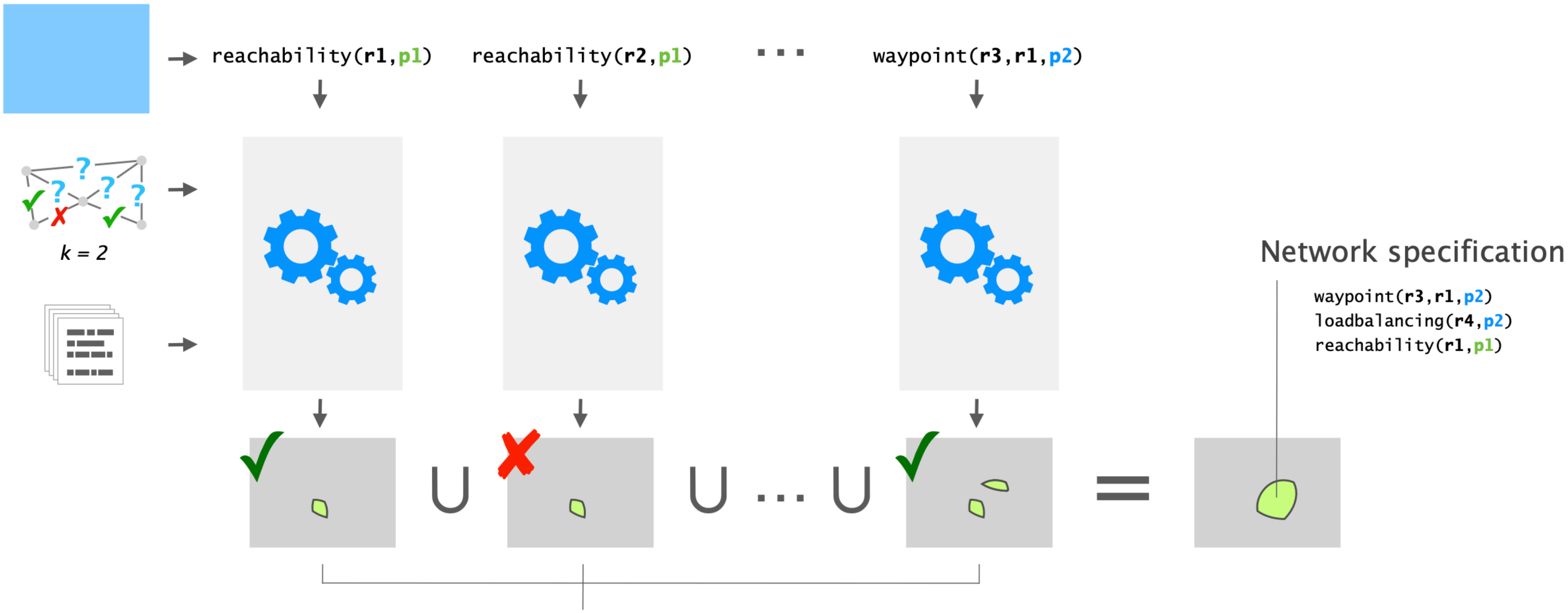


Network specification

```
waypoint(r3,r1,p2)
loadbalancing(r4,p2)
reachability(r1,p1)
```

over-approximation of the specification

Check whether a candidate policy belongs to the specification with control plane verification



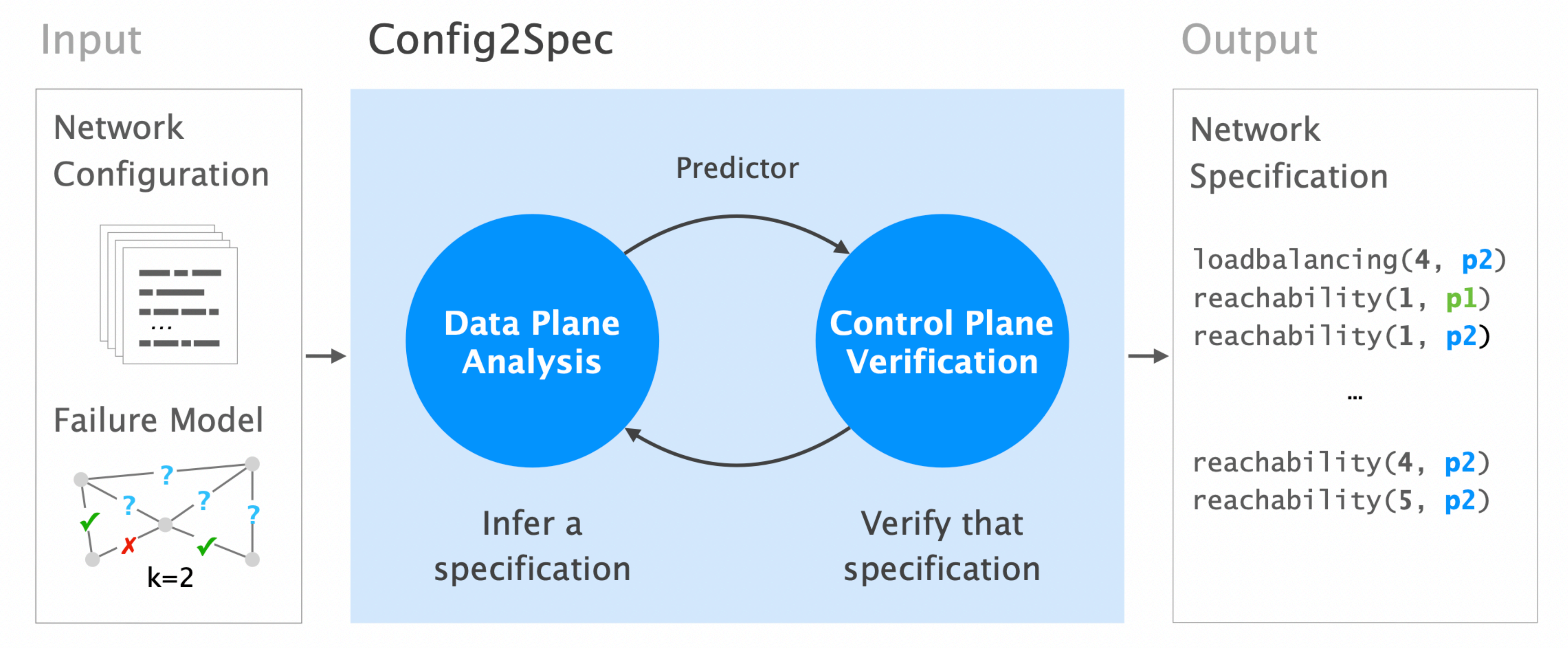
reachability(**r1**,p1) reachability(**r2**,p1) ... waypoint(**r3**,**r1**,p2)

Network specification

waypoint(**r3**,**r1**,p2)
loadbalancing(**r4**,p2)
reachability(**r1**,p1)

under-approximation of the specification

Automatically mine the spec. using the two approaches



```
loadbalancing(4, p2)
reachability(1, p1)
reachability(1, p2)

...

reachability(4, p2)
reachability(5, p2)
```

Papers in this presentation

Mining network specifications in the configuration

・Config2Spec

Incremental Network Configuration Verification [10]

・RealConfig

Proactive verification of DNS configurations [11]

・GROOT

Background and Motivation

Network configuration changes are frequent and often small

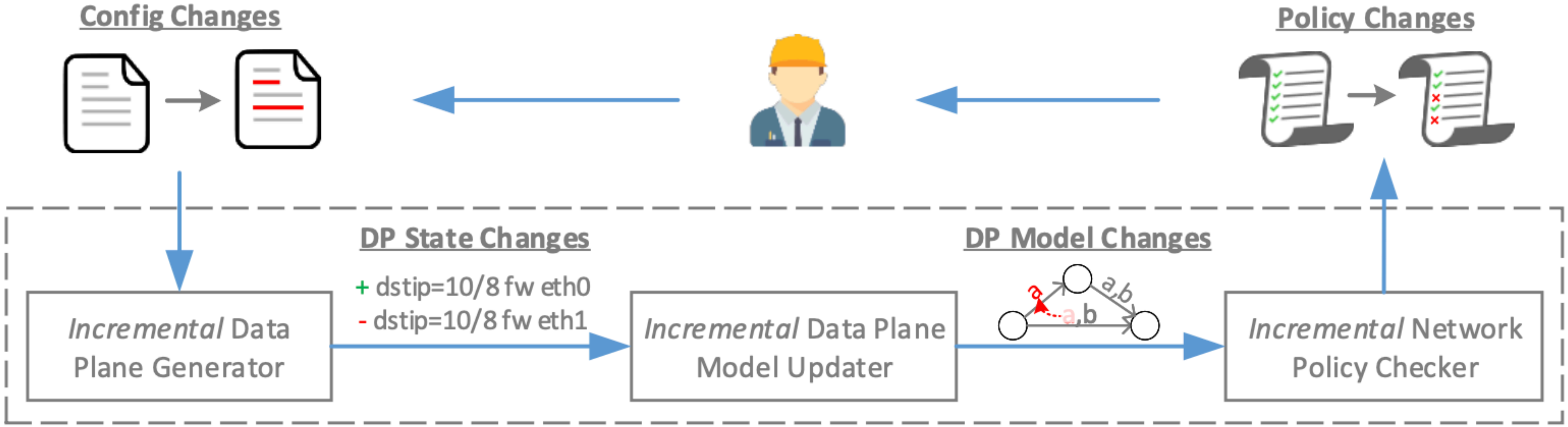
Current verification tools are **not optimized** for configuration changes

	General-purpose tools	Domain-specific algorithms
Analyze	Minesweeper [6]	ARC [13]
Simulate	Batfish(original) [5] Plankton [8]	Batfish(current) [5]

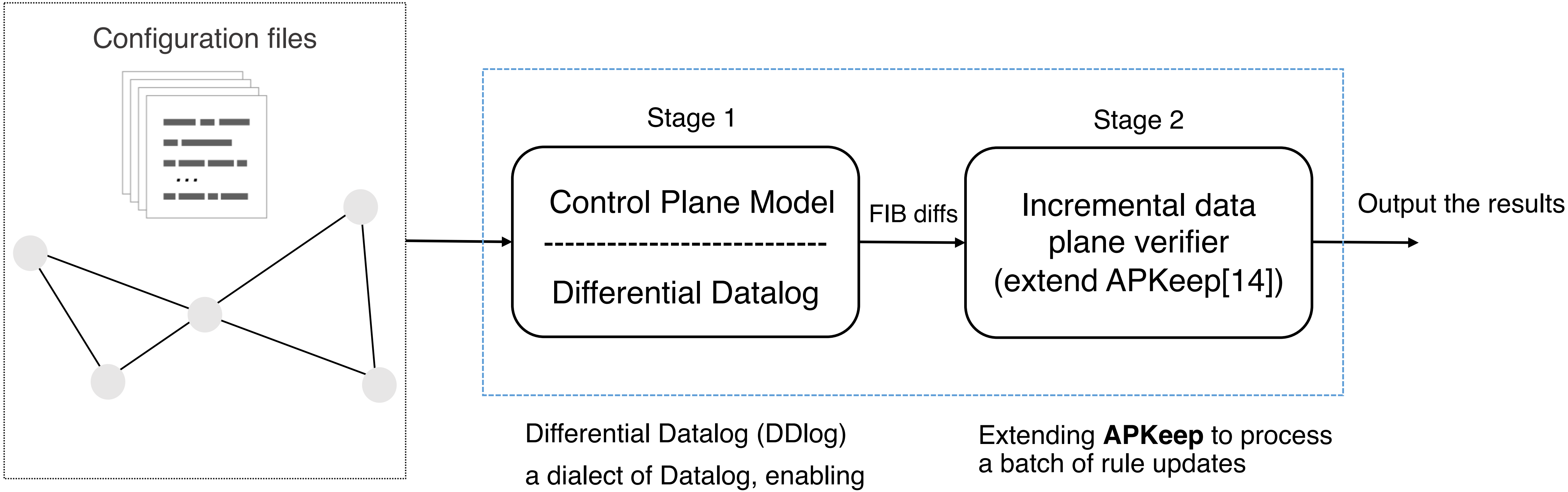
Approach

Design a **incremental** network verification for configuration changes

- combine **incremental** DP verification with **incremental** DP generation


Config Changes
Policy Changes
DP State Changes
+ dstip=10/8 fw eth0
- dstip=10/8 fw eth1
DP Model Changes
a
a,b
a,b
Incremental Data Plane Generator
Incremental Data Plane Model Updater
Incremental Network Policy Checker

Internal of the verification flow



Configuration files

Stage 1

Control Plane Model

Differential Datalog

FIB diffs

Stage 2

Incremental data plane verifier (extend APKeep[14])

Output the results

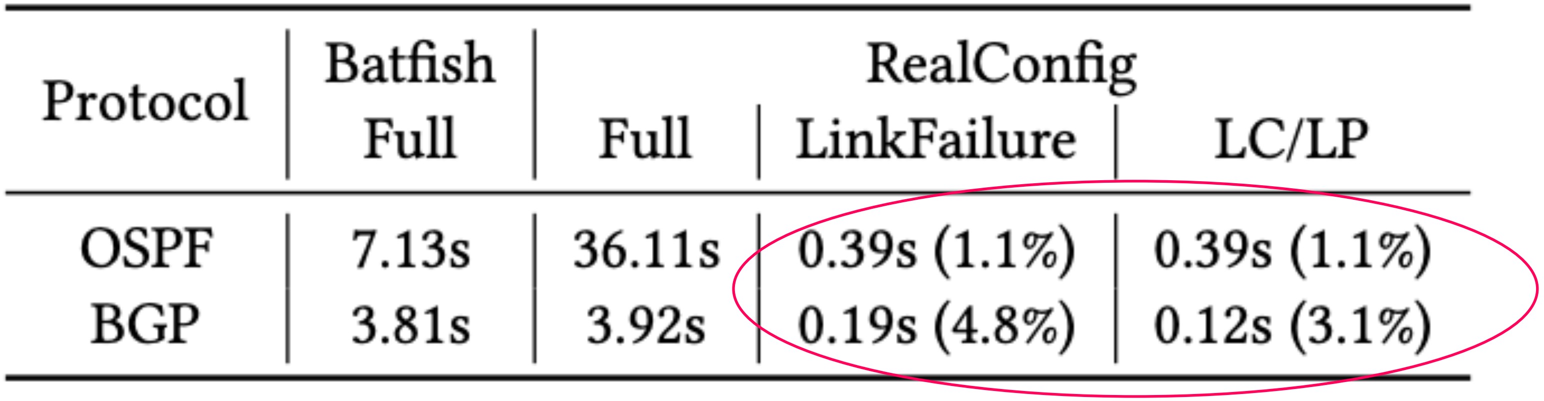
Differential Datalog (DDlog)

a dialect of Datalog, enabling automatic incremental computation

Extending **APKeep** to process a batch of rule updates

Preliminary Results

Average data plane generation time for the fat tree network



Protocol	Batfish	RealConfig		
	Full	Full	LinkFailure	LC/LP
OSPF	7.13s	36.11s	0.39s (1.1%)	0.39s (1.1%)
BGP	3.81s	3.92s	0.19s (4.8%)	0.12s (3.1%)

20 to 92 times speed-up

Papers in this presentation

Mining network specifications in the configuration

- Config2Spec

Incremental Network Configuration Verification [10]

- RealConfig

Proactive verification of DNS configurations [11]

- GROOT

Current DNS management

Black-box testing

- Live testing from multiple vantage points

Limitations

✕ Incomplete

✕ Not exhaustive

DNS config review

- check the configurations manually

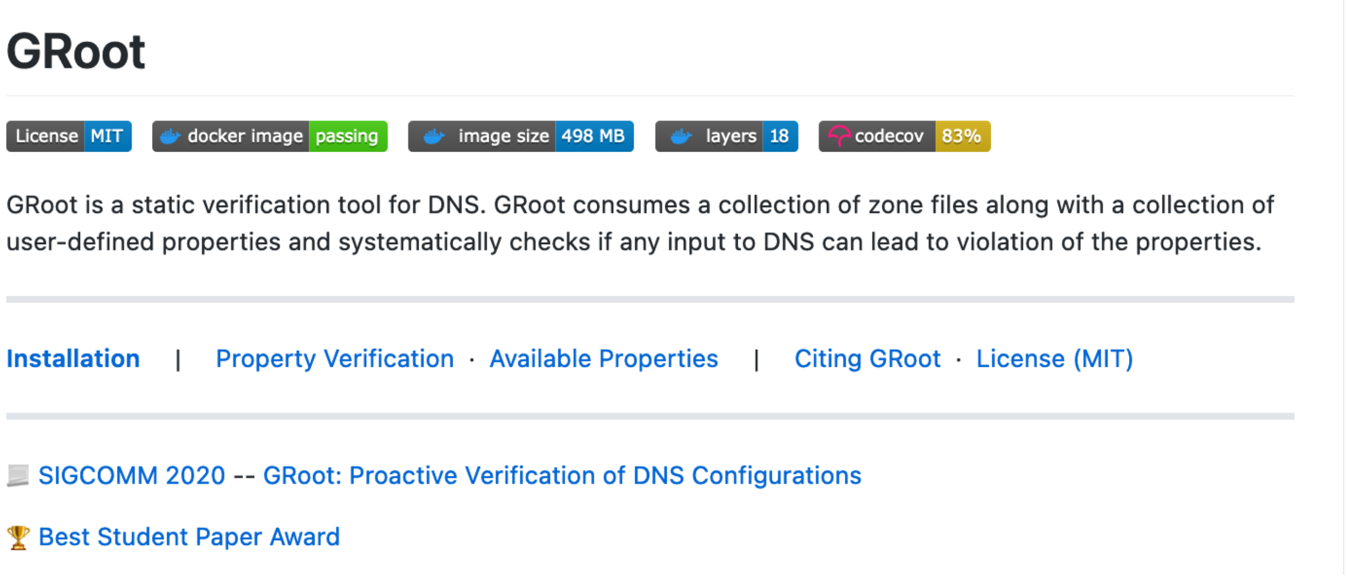
Limitations

✕ Not automated

✕ Error prone

Approach

- Design a first proactive verification tool for DNS configs
- First formal model of DNS resolution (RFC 6672) for automating zone file review



GRoot

GRoot is a static verification tool for DNS. GRoot consumes a collection of zone files along with a collection of user-defined properties and systematically checks if any input to DNS can lead to violation of the properties.

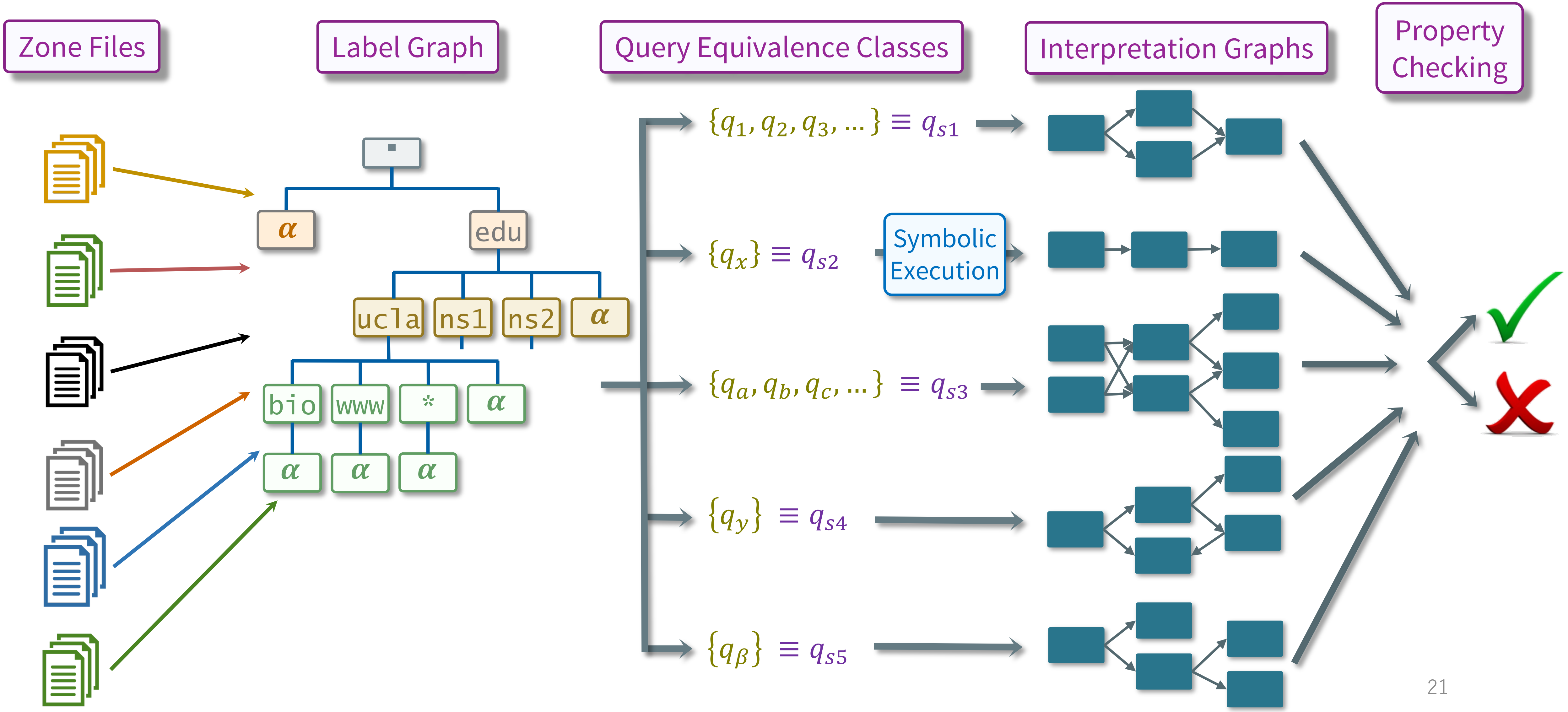
Installation | Property Verification · Available Properties | Citing GRoot · License (MIT)

📃 SIGCOMM 2020 -- GRoot: Proactive Verification of DNS Configurations

🏆 Best Student Paper Award

Written in C++ !! 😭 (歓喜の涙)

Verification flow for the configuration



Experiment using campus network

Zone files of campus.edu. and 895 subdomains

- 120k resource records
- 4259 CNAME records
- 63 wildcard records

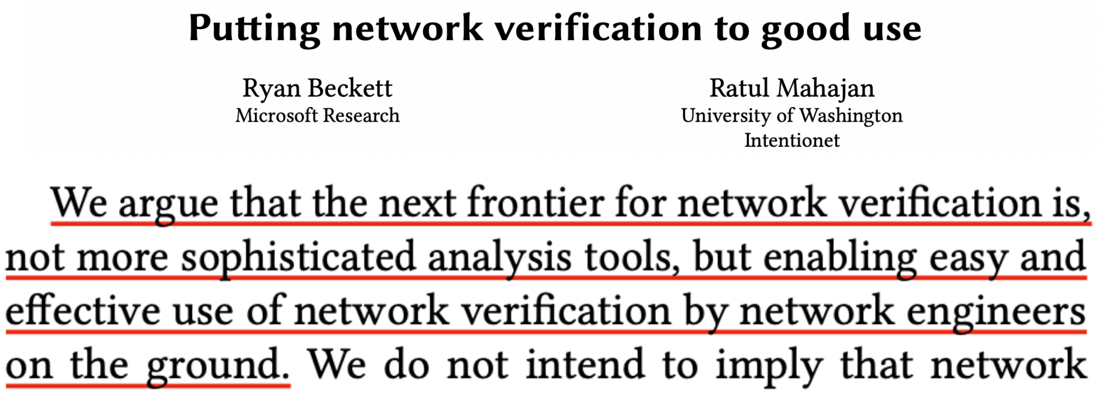
Property	Number of bugs
Delegation Consistency	**49**
No lame delegation	**9**
No rewrite loops	**2**
No missing glue records	**1**
No rewrite blackholing	**48**
No zero TTL records	**0**

Property	Number of warnings
No rewrite to outside domain	**378**
No resolution at an external NS	**324**
Number of rewrites ≤ 2	**24**

Discussion

- Many studies have improved the scalability of the verification with the specification (like Plankton [8])

- Recent studies have tackled the problems to make the verification tools more user friendly (like Config2Spec [7])



Putting network verification to good use

Ryan Beckett
Microsoft Research

Ratul Mahajan
University of Washington
Intentionet

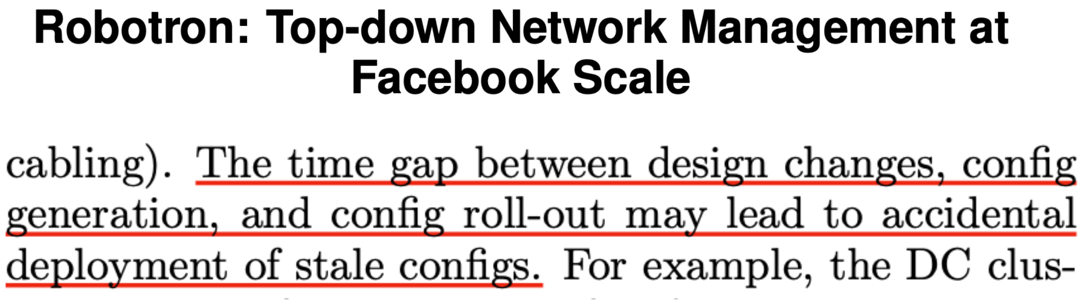
We argue that the next frontier for network verification is, not more sophisticated analysis tools, but enabling easy and effective use of network verification by network engineers on the ground. We do not intend to imply that network

Putting network verification tool good to use (HotNets’19) [12]

Discussion

- Will the verification tool of the DNS configuration follow the same research transition as the network verification tools ?

- There are many open problems in the real operation env.



Robotron: Top-down Network Management at Facebook Scale

cabling). The time gap between design changes, config generation, and config roll-out may lead to accidental deployment of stale configs. For example, the DC clus-

Robotron: Top-down Network Management at Facebook Scale (SIGCOMM’16) [14]

My future work

- Think a rough direction of my research (too early … ?)
- Skip through the related papers
- Implementing some mockups or anything related to the research to get the image at program level

Reference

[1] Detecting BGP Configuration Faults with Static Analysis
Nick Feamster and Hari Balakrishna (NSDI'05)

[2] FIREMAN: A Toolkit for FIREwall Modeling and Analysis
Lihua Yuan et al., (S&P'06)

[3] Debugging the Data Plane with Anteater
Haohui Mai et al., (Sigcomm'2011)

[4] Veriflow : Verifying Network-Wide Invariants in Real Time
Ahmed Khurshid et al.,(NSDI'13)

Reference

[5] A General Approach to Network Configuration Analysis
Ari Fogel et al., (NSDI'15)

[6] A General Approach to Network Configuration Verification
Ryan Beckett et al., (Sigcomm'2017)

[7] Config2Spec: Mining Network Specifications from Network
Configurations Rüdiger Birkner et al.,(NSDI'20)

[8] Plankton: Scalable network configuration verification
through model checking, Santhosh Prabhu et al., (NSDI'20)

Reference

[9] Finding Network Misconfigurations by Automatic Template Inference, Siva Kesava Reddy Kakarla et al., (NSDI’20)

[10] Incremental Network Configuration Verification
Peng Zhang et al., (HotNets’20)

[11] GRoot: Proactive Verification of DNS Configurations
Siva Kesava Reddy Kakarla et al.,(SIGCOM’20)

[12] Putting network verification tool good to use
Ryan Beckett et al., (HotNets’19)

Reference

[13] Fast Control Plane Analysis using an Abstract Representation
Aaron Gember-Jacobson et al., (SIGCOM’16)

[14] Robotron: Top-down Network Management at Facebook Scale
Yu-Wei Eric Sung et al., (SIGCOMM’16)