Empowering Enterprises for Industry 4.0 Transformation

Verifiable Networks

Nate Foster
Cornell University

2020
Should this packet be here?

Properties
- Connectivity
- Loop-freedom
- Waypointing
- Resilience
- Isolation
Internet Protocol

DARPA INTERNET PROGRAM

PROTOCOL SPECIFICATION

September 1981

preparing for

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia  22209

by

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California  90291

September 1981

Internet Protocol

TABLE OF CONTENTS

- PREFACE
- HEADERS
- PARSE
- **TABLE OF CONTENTS**

```c
/* -*- P4_16 -*- */
#include <core.p4>
#include <v1model.p4>

const bit<16> TYPE_IPV4 = 0x800;

/**************************** HEADERS ***********************************/

typedef bit<9> egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;

header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16> etherType;
}

header ipv4_t {
    bit<4> version;
    bit<4> ihl;
    bit<8> diffserv;
    bit<16> totalLen;
    bit<8> identification;
    bit<3> flags;
    bit<13> fragOffSet;
    bit<8> ttl;
    bit<16> protocol;
    ip4Addr_t headerChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}

struct metadata {
    /* empty */
}

struct headers {
    ethernet_t   ethernet;
    ipv4_t       ipv4;
}

/**************************** PARSE ***********************************/

parser MyParser(packet_in packet,
    out headers hdr,
    inout metadata meta,
    inout standard_metadata_t standard_metadata) {
    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition accept;
    }
}
```
Verifiable Networks

Static Approaches
- Use programmability to get precise model
- Check network properties via static analysis

Dynamic Approaches
- Use programmability to instrument network
- Check observations against formal specification
Static Verification
Static Verification

**Approach**
- Start with a program
- Extract mathematical model
- Check properties of interest
Example: ACL

Internet

Server

10.0.1.1

10.0.1.1 deny

ACL
This is not a toy example!

Network path not found?

How to use an army of tests to understand and diagnose your network!
Demo
1. Start with P4 Program
2. Annotate with assertions
3. Translate to imperative code
4. Apply standard optimizations
5. Generate first-order formula
6. Send to theorem prover
    - **Passed**
    - **Counterexample**

Input packet:
- [Parser] start
- [Parser] _parse_ethernet
- [Packet] ethernet.dst          000000000000
- [Packet] ethernet.srvcc 00000000
- [Packet] ethernet.ether_type = 0xf7ff
- [Assert] (not (= ipv4.valid 1w0))
Evaluation: switch.p4

Statistics
- 5,599 LoC
- 58 parser states
- 120 match–action tables

Control-plane interface
- 758 LoC
- ~2 days’ programmer effort
- Default actions (31)
- Fabric wellformedness (14)
- Table actions (66)
- Guarded reads (10)
- Action data (14)

Found 10 bugs
- Parser bugs (2)
- Action flaws (4)
- Infeasible control-plane (3)
- Invalid table read (1)
Dynamic Verification
Limits of Static Verification

High-Level Specification
Verifier
Mathematical Model

SDN Controller

Operational
Logical

BGP
Traffic Engineering
Load Balancing
Overlay Virtualization

...
Run-Time Verification

**Approach**
- Start with a specification
- Compile to local predicates
- Wrap components with a reference monitor that checks for violations
Verifiable Closed-Loop Control

SDN Controller

BGP
Traffic Engineering
Load Balancing
Overlay Virtualization

High-Level Specification

Verifier

Fine-Grained Network Telemetry
Research Opportunities

• **Efficiency:** can we make collection and analysis of fine-grained telemetry scalable and efficient?

• **Expressiveness:** can we use run-time techniques to verify rich properties (e.g., QoS, causality, etc.)?

• **Trust:** can we trust the network components used to generate and analyze telemetry data?
Thank You!

Collaborators

• Calin Cascaval (Barefoot)
• Bill Hallahan (Yale)
• JK Lee (Barefoot)
• Jed Liu (Barefoot)
• Nick McKeown (Stanford)
• Guru Parulkar (Stanford / ONF)
• Sundararajan Renganathan (Stanford)
• Cole Schlesinger (Amazon)
• Milad Sharif (Barefoot)
• Robert Soulé (Lugano)
• Bruce Spang (Stanford)
• Han Wang (Barefoot)

http://www.cs.cornell.edu/~jnfoster