ClassBench-ng: Recasting ClassBench After a Decade of Network Evolution

Jiri Matousek[1], Gianni Antichi[2], Adam Lucansky[3]
Jan Korenek[1], Andrew W. Moore[2]

[1]Brno University of Technology [2]University of Cambridge [3]CESNET

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Agenda

Introduction

Analysis of Real Classification Rules

- IP Prefixes
- Ports and Protocol
- OpenFlow

ClassBench-ng

ClassBench-ng Evaluation

Summary

Packet Classification

- matching incoming packets against a set of rules and performing the corresponding action
- the basic operation of each networking device
- examples
    - packet forwarding
    - application of security policies
    - application-specific processing
    - application of quality-of-service guarantees
- packet classification according to IPv4 5-tuple
    - src/dst IPv4 prefix
    - src/dst port
    - protocol

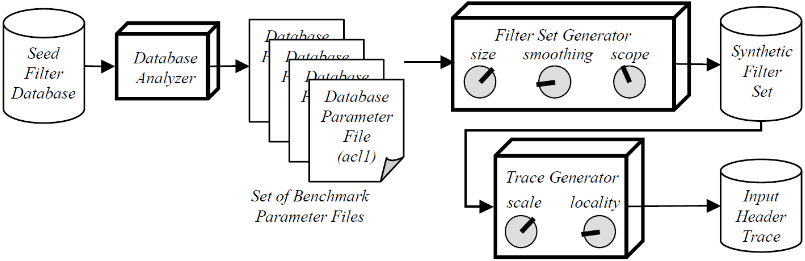
Internet Evolution

- many trends that influence packet classification
  - growing deployment of IPv6 (longer IP prefixes)
  - adoption of SDN with OpenFlow protocol (more header fields)
  - increasing transfer rates (faster classification)
  - increasing number of classification rules (larger data structures)
- Internet evolution stimulates development of new packet classification algorithms
- new algorithms need to be benchmarked

Packet Classification Benchmarking

- lack of publicly available benchmarking data
- benchmarking using synthetically generated rule sets

ClassBench



Taylor, D. E., and Turner, J. S., "ClassBench: A Packet Classification Benchmark,"IEEE/ACM Transactions on Networking, vol. 15, no. 3, pp. 499–511, June 2007

Recasting ClassBench

- today's Internet is no more the one of a decade ago
- questions with respect to ClassBench
  - Are the ideas behind the ClassBench still valid?
  - What are the characteristics of real rule sets with IPv6 prefixes and OpenFlow-specific fields?
  - How to extend the ClassBench with respect to IPv6 and OpenFlow?

Agenda

Introduction

Analysis of Real Classification Rules

- IP Prefixes
- Ports and Protocol
- OpenFlow

ClassBench-ng

ClassBench-ng Evaluation

Summary

Analyzed Real Data Sets

Name	Prefixes or Rules	Source	Date
IPv4 Prefix Sets			
eqix_2015	550 511	http://archive.routeviews.org/	2015-07-02
eqix_2005	164 455		2005-07-02
rrc00_2015	571 351	http://data.ris.ripe.net/	2015-07-02
rrc00_2005	168 525		2005-07-02
IPv6 Prefix Sets			
eqix_2015	23 866	http://archive.routeviews.org/	2015-07-02
eqix_2013	13 444		2013-07-02
eqix_2005	658		2005-07-02
rrc00_2015	24 162	http://data.ris.ripe.net/	2015-07-02
rrc00_2013	14 374		2013-07-02
rrc00_2005	499		2005-07-02
Rule Sets From University Network			
uni_2010	96	university ACL	2010-08-30
uni_2015	122	university ACL	2015-01-14
OpenFlow Rule Sets			
of1	16 889	Open vSwitch in a cloud	2015-05-29
of2	20 250	Open vSwitch in a cloud	2015-05-29
of3	1 757 to 7 456	Open vSwitch in a cloud	2015-06-18 to 2015-07-14

IP Prefix Set Representation

- representation using trie (binary prefix tree)
- desired properties of trie description
  - anonymity
  - completeness
  - scalability
- the same trie description as in the original ClassBench
  - prefix length distribution
  - branching probability distributions
  - average skew distribution

$$skew = 1 - \frac{weight(lighter)}{weight(heavier)}$$

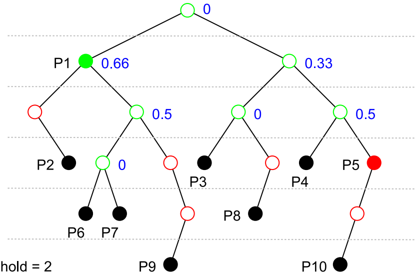
  - prefix nesting threshold

Example of IP Prefix Set Representation

- prefix length distribution
- branching probability distribution
  - probability of 1-child node
  - probability od 2-children node
- average skew distribution
- prefix nesting threshold

Prefixes

P1 = 0*
P2 = 001*
P3 = 100*
P4 = 110*
P5 = 111*
P6 = 0100*
P7 = 0101*
P8 = 1010*
P9 = 01110*
P10 = 11100*

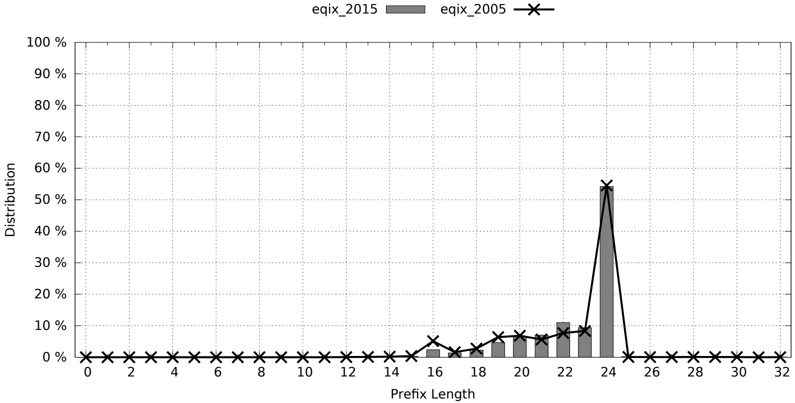


Prefix nesting threshold = 2

lengths	1-child	2-children	skew
0	0	1	0
0.1	0	1	0.5
0	0.25	0.75	0.33
0.4	0.75	0.25	0
0.3	1	0	
0.2			

IPv4 Prefix Sets (2005-2015)

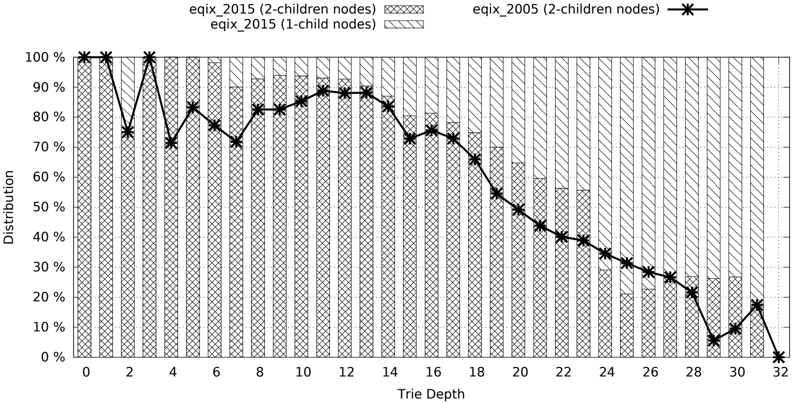
Prefix Length Distribution



- 3-times more prefixes after 10 years of evolution

IPv4 Prefix Sets (2005-2015)

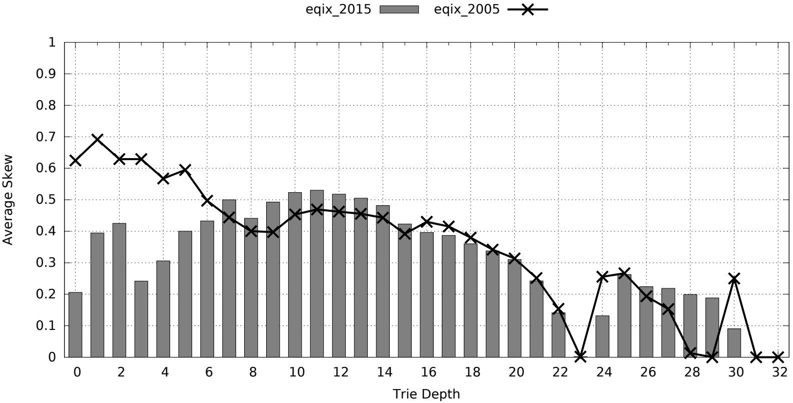
Branching Probability Distributions



- 3-times more prefixes after 10 years of evolution

IPv4 Prefix Sets (2005-2015)

Average Skew Distribution

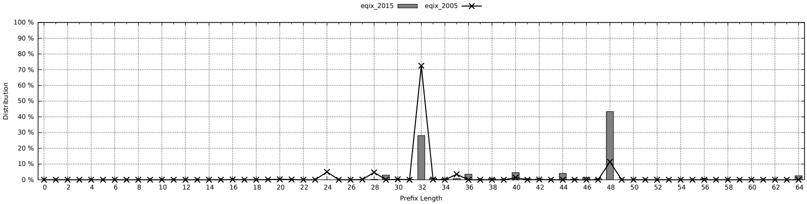


- 3-times more prefixes after 10 years of evolution

IPv6 Prefix Sets (2005-2015)

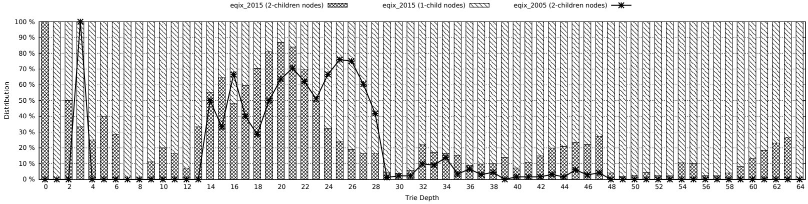
- 36-times more prefixes after 10 years of evolution
- the most common prefix length shifted from 32 (RIRs/ISPs) to 48 (end users/organization)

Prefix Length Distribution

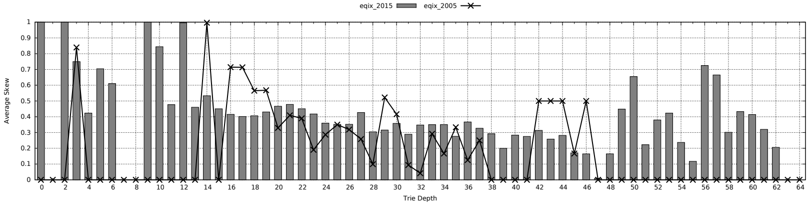


IPv6 Prefix Sets (2005-2015)

Branching Probability Distributions



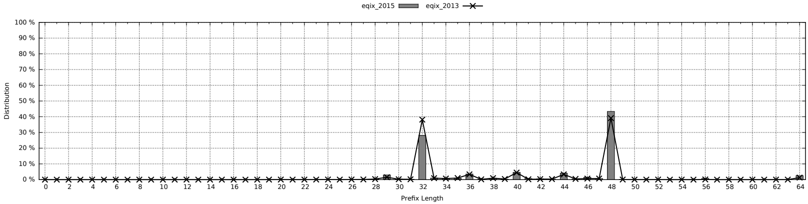
Average Skew Distribution



IPv6 Prefix Sets (2013-2015)

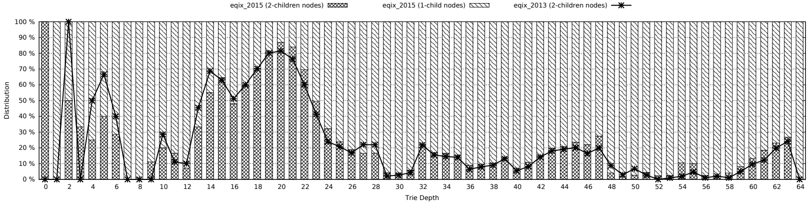
- 2-times more prefixes after 2 years of evolution
- only minor changes in prefix length distribution

Prefix Length Distribution

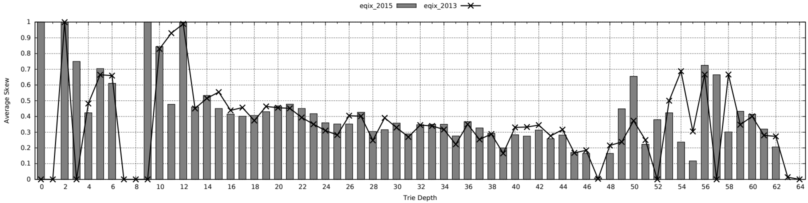


IPv6 Prefix Sets (2013-2015)

Branching Probability Distributions



Average Skew Distribution



Ports Representation

- 5 port classes are distinguished within analysis
  - **WC** – wildcard
  - **HI** – user port range (1024 : 65535)
  - **LO** – well-known system port range (0 : 1023)
  - **AR** – arbitrary range
  - **EM** – exact match

Ports and Protocol

Transport Layer Protocol

- increasing number of rules specifying UDP protocol
- increasing number of rules with wildcarded protocol

Data Set	**Protocol Specification** **wildcard**	**TCP**	**UDP**
uni_2010	26.04 %	71.88 %	2.08 %
uni_2015	38.52 %	54.92 %	6.56 %

Source and Destination TCP/UDP Port

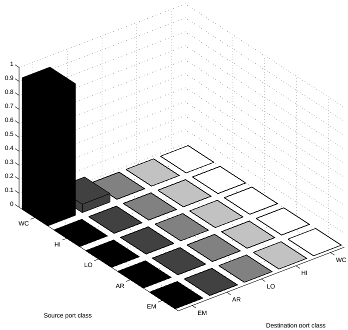
- increasing number of rules with AR or WC destination port specification

Port	**WC**	**HI**	**LO**	**AR**	**EM**
			uni_2010		
Source	100.00	0.00	0.00	0.00	0.00
Destination	26.04	0.00	0.00	5.21	68.75
			uni_2015		
Source	100.00	0.00	0.00	0.00	0.00
Destination	38.52	0.00	0.00	8.20	53.28

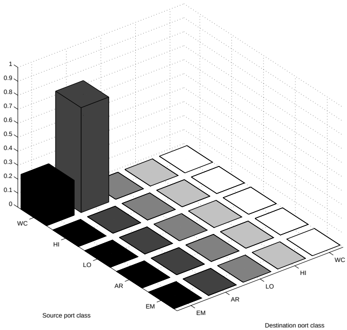
Source-Destination Port Pair Class

- port pair class (PPC) helps to understand interdependencies between source and destination port classes
- analysis of PPC for TCP and UDP protocols in uni_2015

TCP



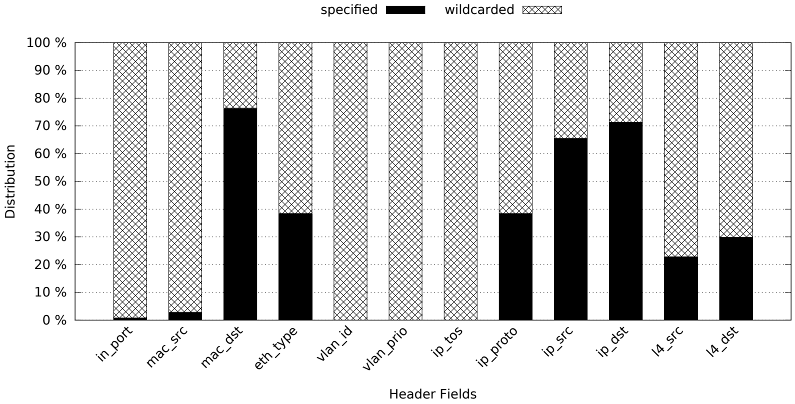
UDP



OpenFlow 1.0 Rules

- OpenFlow 1.0 extends the standard 5-tuple with 7 header fields
  - ingress port
  - src/destinaiton MAC address
  - EtherType
  - VLAN ID
  - VLAN priority
  - DSCP (former IP ToS)

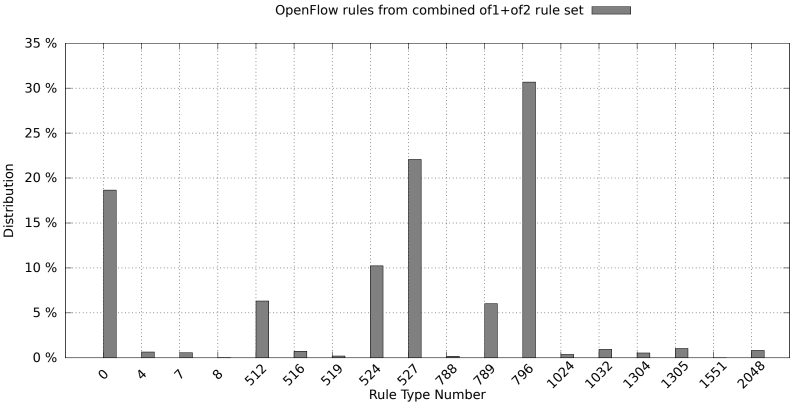
OpenFlow Header Fields Values



Rule Set	in_port	mac_src	mac_dst	eth_type	ip_proto	ip_src	ip_dst	l4_src	l4_dst
of1	123 (0.866)	27 (0.032)	593 (0.047)	1 (<0.001)	3 (0.003)	478 (0.046)	109 (0.009)	4 (0.029)	48 (0.022)
of2	140 (0.864)	19 (0.081)	791 (0.050)	1 (<0.001)	3 (0.001)	390 (0.028)	97 (0.007)	4 (<0.001)	8227 (0.927)
of1+of2	182 (0.599)	45 (0.042)	1176 (0.041)	1 (<0.001)	3 (<0.001)	498 (0.020)	119 (0.004)	6 (0.001)	8237 (0.742)

OpenFlow Rule Types

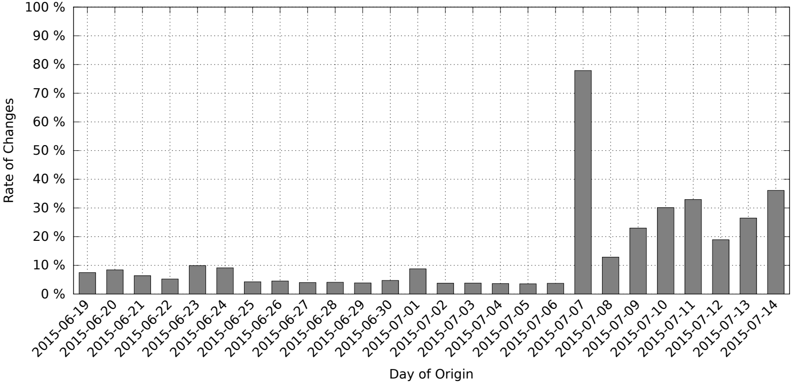
- OpenFlow rule type describes which header fields are wildcarded/specified in rules of this type
- rule type can be represented as 12-bit binary number
    - theoretically 4096 different rule types
    - practically only 18 utilized rule types



OpenFlow Rule Set Dynamics

- dynamics of OpenFlow rule set expressed with the help of symmetric difference

$$A \Delta B = (A \setminus B) \cup (B \setminus A)$$



Agenda

Introduction

Analysis of Real Classification Rules

- IP Prefixes
- Ports and Protocol
- OpenFlow

ClassBench-ng

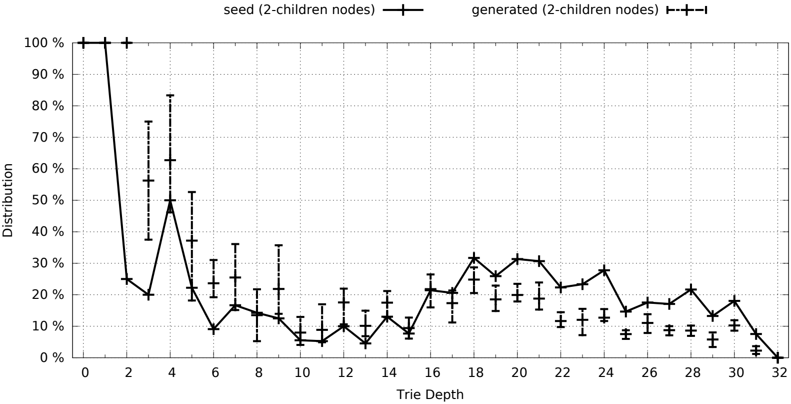
ClassBench-ng Evaluation

Summary

ClassBench Generation Accuracy

- comparison of 10 runs against original values

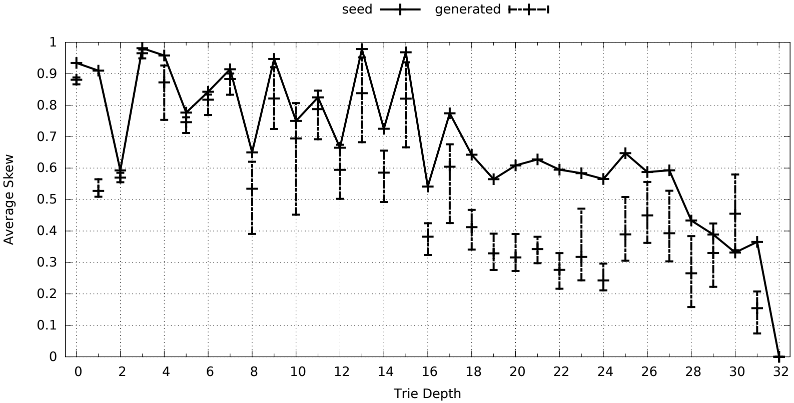
Branching Probability Distribution



ClassBench Generation Accuracy

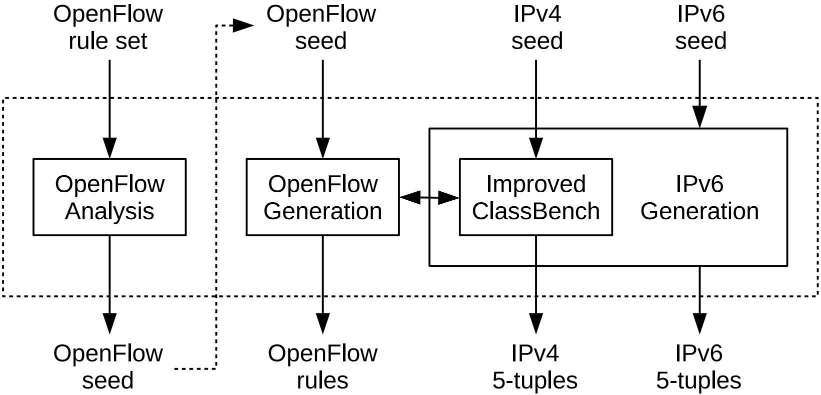
- comparison of 10 runs against original values

Average Skew Distribution



ClassBench-ng

- built upon the original ClassBench
- improves IPv4 prefixes generation accuracy
- supports IPv6 prefixes generation and OpenFlow



Improved ClassBench

- IPv4 prefixes generation is improved using trie pruning algorithm
  - starts from 100-times bigger src/dst prefix sets
  - removes individual prefixes to adjust prefix set parameters to given values
- three steps of trie pruning algorithm
  1. branching probability adjustment (↓)
  2. skew distribution adjustment (↑)
  3. prefixes length distribution adjustment (↓)
- first two steps try to remove as less prefixes as possible
- each step aims to not alter the already ajusted characteristics

OpenFlow Analysis

- generates OpenFlow seed from OpenFlow rule set (in `ovs-ofctl` format)
- 3 parts of OpenFlow seed
  - rule type distribution
  - 5-tuple seed
  - OpenFlow-specific seed
- 4 types of representation within OpenFlow-specific seed
  - values (`in_port`, `eth_type`)
  - parts (`mac_src`, `mac_dst`)
  - size (`vlan_id`)
  - null (`vlan_prio`, `ip_tos`)

OpenFlow Generation

- consists of 3 steps
  1. uses Improved ClassBench to generate given number of IPv4 5-tuples
  2. removes IPv4 5-tuple fields that are not part of the given OpenFlow rule type
  3. adds OpenFlow-specific header fields that are part of the given OpenFlow rule type
- does not allow to generate inconsistent rules (e.g., rule specifying VLAN ID and EtherType `0x0800`)

Agenda

Introduction

Analysis of Real Classification Rules

- IP Prefixes
- Ports and Protocol
- OpenFlow

ClassBench-ng

ClassBench-ng Evaluation

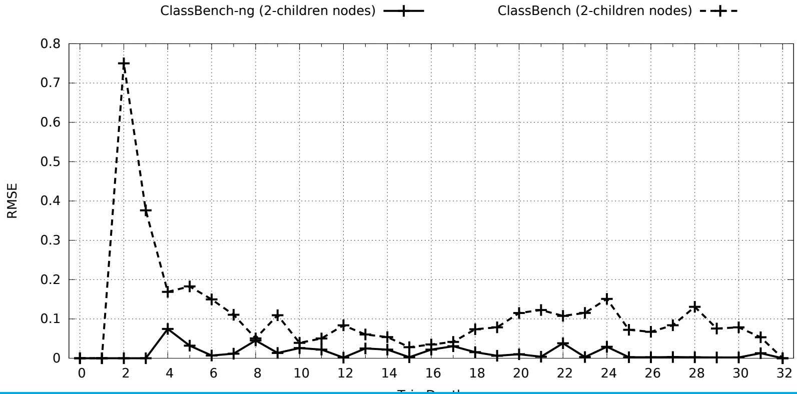
Summary

Improved ClassBench Evaluation

- comparison of IPv4 prefixes generation accuracy of ClassBench and ClassBench-ng using RMSE

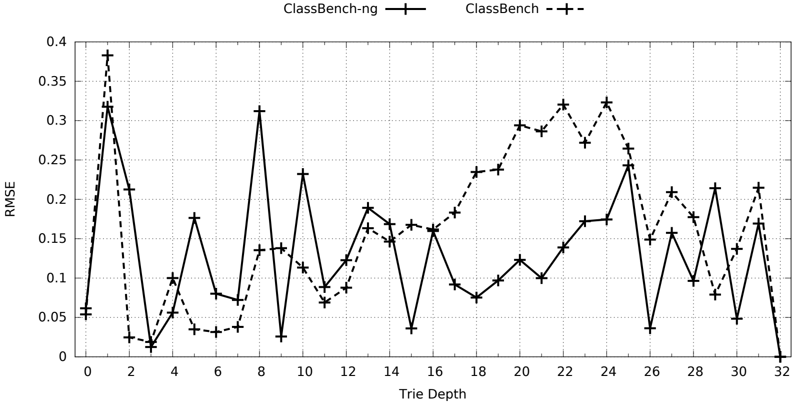
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\bar{y} - y_i)^2}$$

Branching Probability Distribution



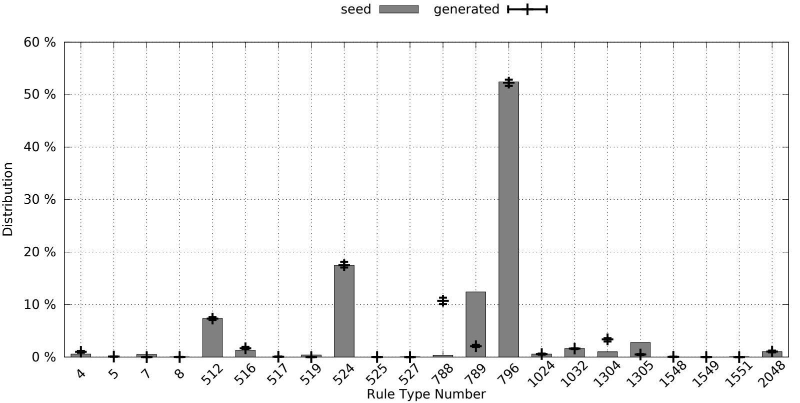
Improved ClassBench Evaluation

Skew Distribution



- comparison of 10 runs against original values

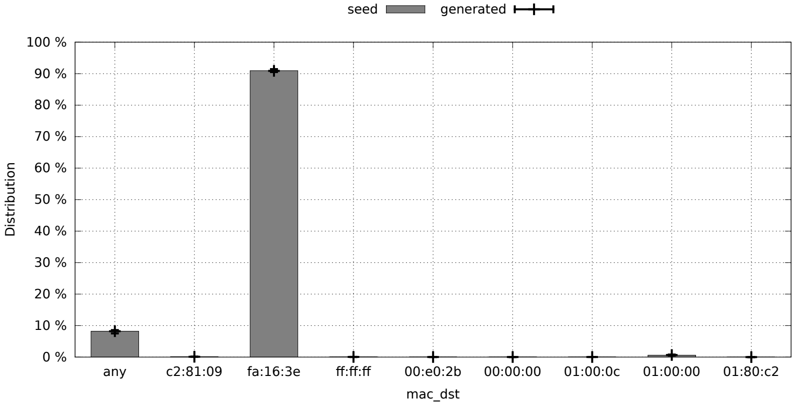
OpenFlow Rule Types



OpenFlow Generation Evaluation

- comparison of 10 runs against original values

Destination MAC address (vendor part)



Agenda

Introduction

Analysis of Real Classification Rules

- IP Prefixes
- Ports and Protocol
- OpenFlow

ClassBench-ng

ClassBench-ng Evaluation

Summary

Summary

- detailed analysis of real classification rule sets
  - IPv4/IPv6 prefixes from core routers
  - ACL rules from university network
  - OpenFlow rules from datacenter
- ClassBench-ng tool that is able to
  - accurately generate IPv4/IPv6 5-tuples
  - analyze real OpenFlow rule sets
  - accurately generate OpenFlow rules
- ClassBench-ng is planned to be released in January 2017

Thank you for your attention