HYPERSCAN UPDATES

WWW.HYPERSCAN.IO

xiang.w.wang@intel.com

Agenda

- Hyperscan Ecosystem Overview
- Hyperscan Roadmap
- Hyperscan Performance
- nDPI Case Study
- New open source DPI on FD.io: Universal DPI

Hyperscan:

An industry fastest Regular Expression, Literal Matching Algorithm on Intel platform

Applications

Integrated Open Source Solutions and User Cases (40+ Customers, 37 Open Source Projects)

- IDS/IPS: SNORT, SURICATA
- SD-WAN/DPI: ntop
- Web Application Firewall: modsecurity Open Source Web Application Firewall, NGINX, NAXSI

Language Bindings

- golang, Java, python
- Spam Filtering System: RSPAMD spam filtering system
- Github
- Clickhouse Database

Operating Systems

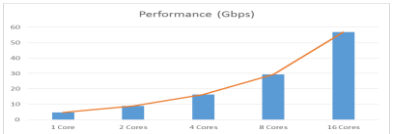
- FreeBSD, fedora, ubuntu, debian, gentoo linux, Windows, OS X

Intel Architectures

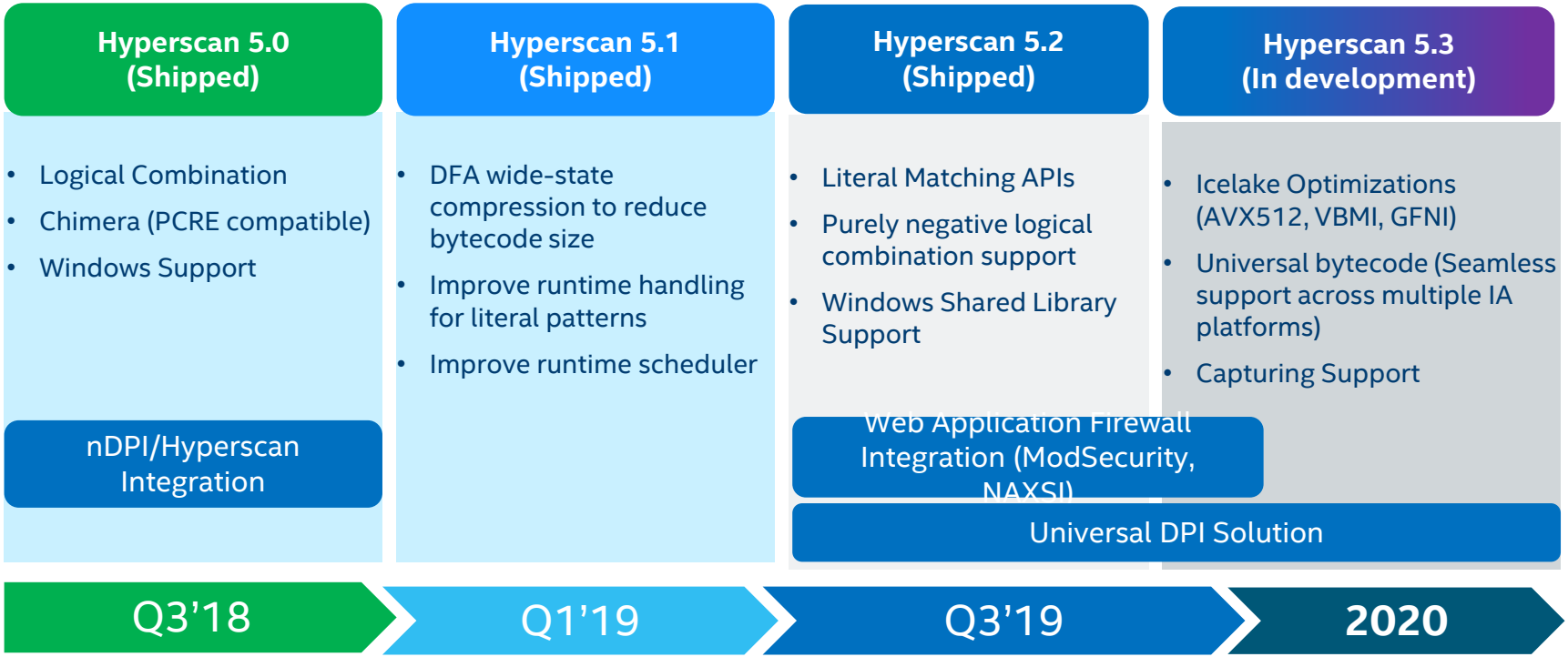
Seamless Support from Atom to Xeon processor

- intel inside ATOM ... intel inside CORE i5 ... intel inside XEON

Linear Core Scalability, Intel Optimized



Hyperscan Roadmap

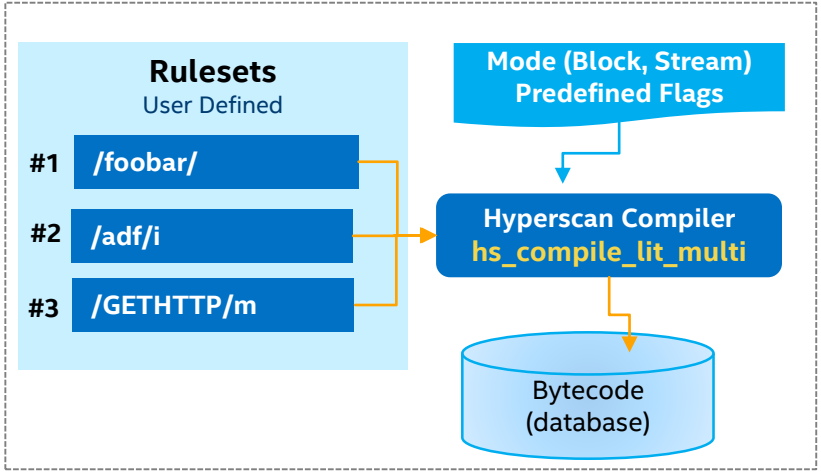


* Hyperscan is supported on all Intel Platform from Atom, Core to Xeon Scalable Processor Family

Release 5.2 (Q3'19)

Literal Matching APIs

- A new set of APIs **hs_compile_lit()** and **hs_compile_lit_multi()** for literal matching.
- No need to convert patterns into HEX representation to avoid regex syntaxes.
- Simplified and faster compile and runtime process.

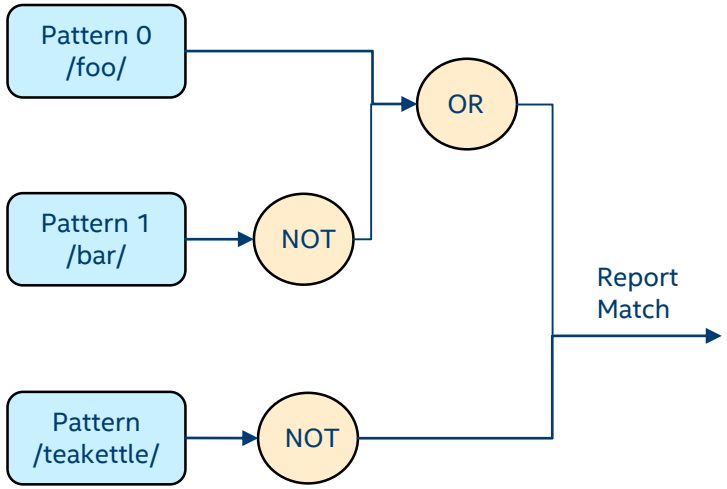


Release 5.2 (Q3'19)

Logical combination

- Report matches only when find defined logical combination (unordered AND, NOT and OR) of patterns
- Add support for purely negative patterns, e.g.

(pattern0 OR (NOT pattern1)) AND (NOT pattern2)



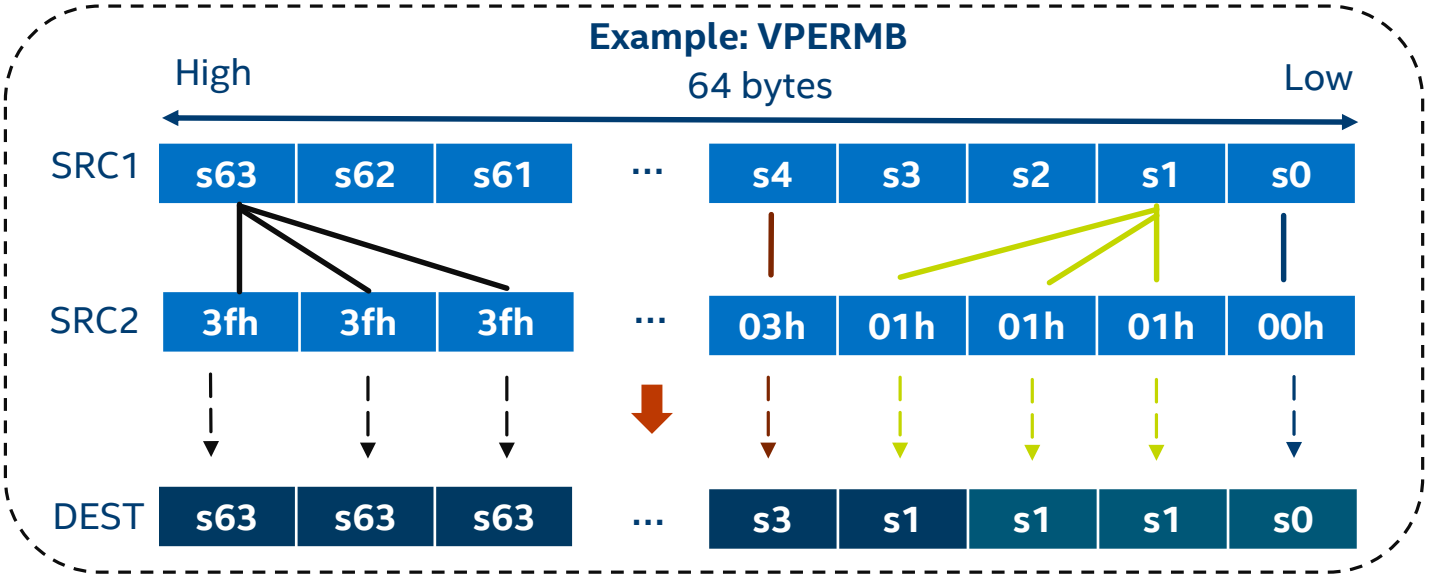
Windows

- Add support for DLL shared library build for Window 32/64-Bit OS

Release 5.3 (In Development)

Icelake optimizations

- AVX512 VBMI utilization for literal matching, acceleration models, lookaround, etc.



Release 5.3 (In Development)

Capturing Support

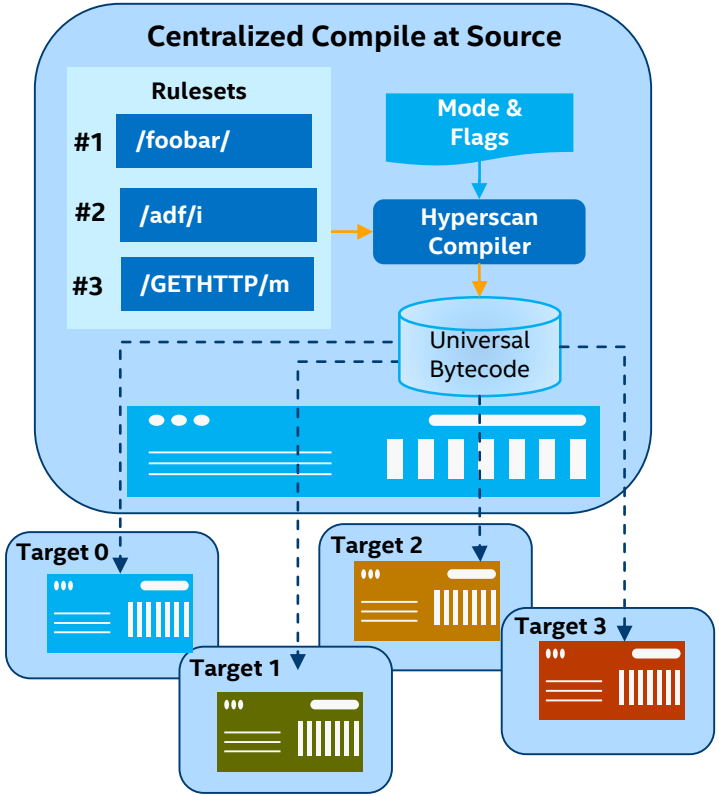
- Hyperscan ignores capturing syntax and only reports matching offsets for the whole pattern
- Add capturing syntax support widely used in WAF, IDS/IPS, etc

e.g. foobar(badge)brush, offsets of “badge” will be reported together with overall pattern matching offsets

Release 5.3 (In Development)

Universal Bytecode

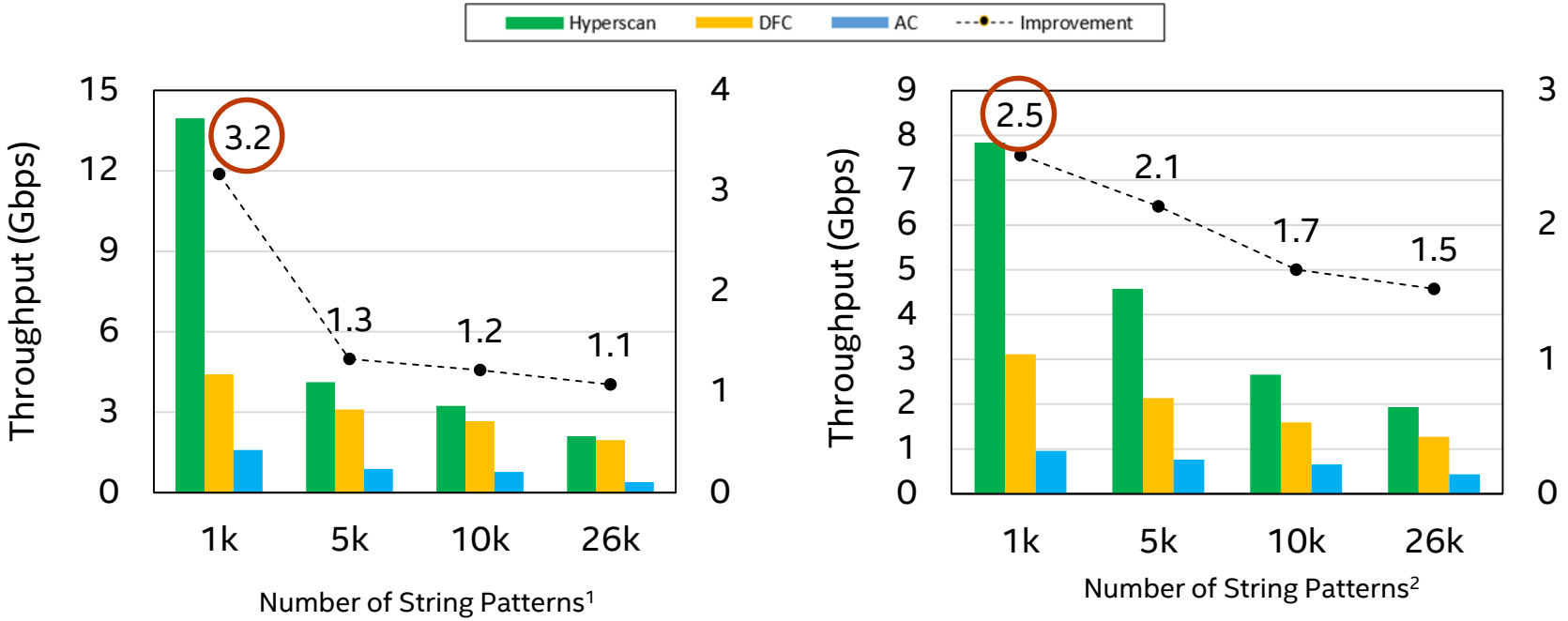
- Centralized compile of a universal bytecode at source machine
- Distribute this universal bytecode to multiple target machines with different Intel architectures
- Each target machine automatically pick up bytecode section best for its architecture during scan



Evaluation of Hyperscan

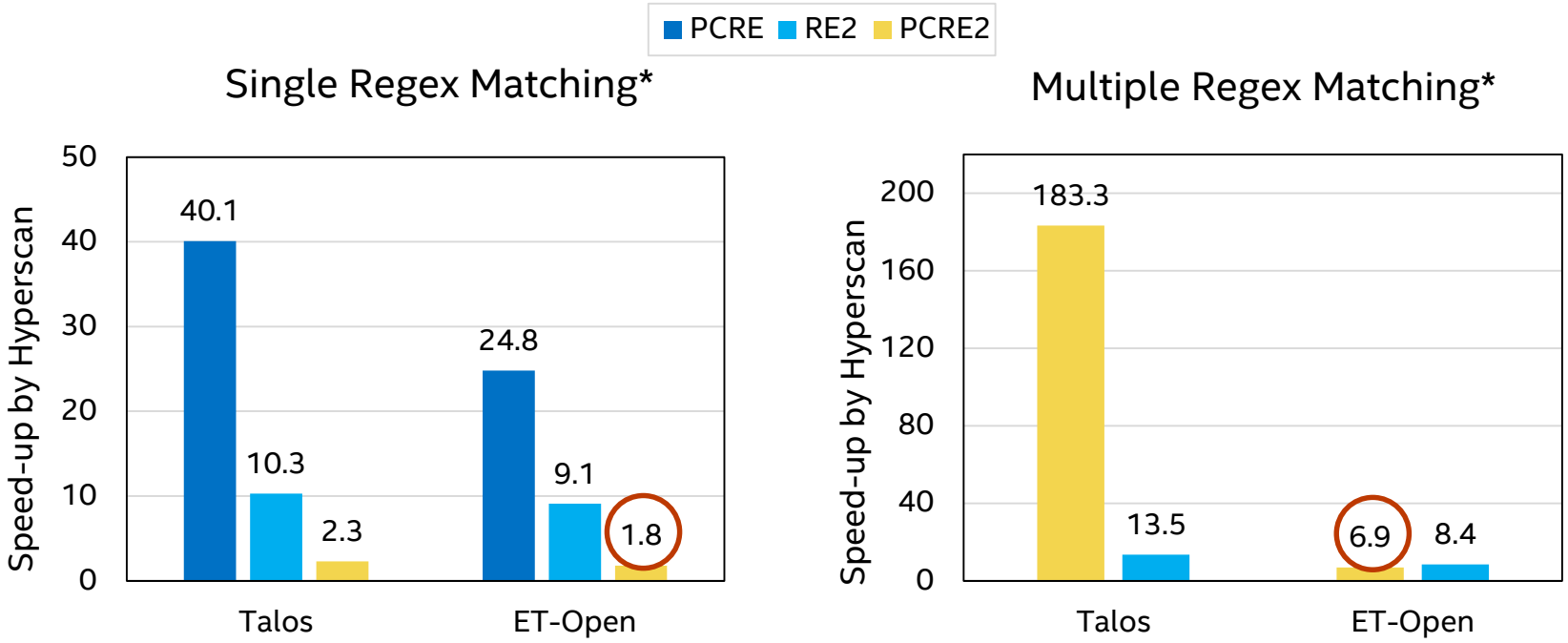
- Performance of literal matching and regex matching vs. state-of-the-art solutions
- Experiment setup:
  - Machine: Intel Xeon Platinum 8180 CPU @ 2.50GHz (48 GB of RAM)
    - ❖ Runs with a single core
    - ❖ GCC 5.4
  - Ruleset: Snort Talos (May 2015), Snort ET-Open 2.9.0, Suricata rulesets 4.0.4
  - Workload: random traffic, real-world web traffic

Multi-literal Matching Performance with Snort ET-Open



[1] Random workload.
[2] Real web traffic trace.

Regex Matching Performance



* Test with Snort Talos (1,300 regexes) and ET-Open (2,800 regexes) rulesets under real Web traffic trace.

nDPI

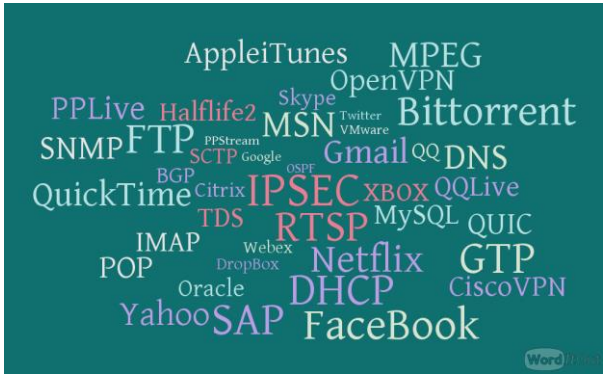
Open source DPI solution under GNU LGPL license (https://github.com/ntop/nDPI)

Support more than 240 protocols/applications

- Messaging (Facebook, Whatsapp)
- Multimedia (YouTube, iTunes)
- Conferencing (Webex, CitrixOnLine)
- Streaming (iQiyi, Netflix)
- Business (VNC, Citrix)

Flexible to add new protocol dissector

Hyperscan Accelerated literal matching



nDPI Database

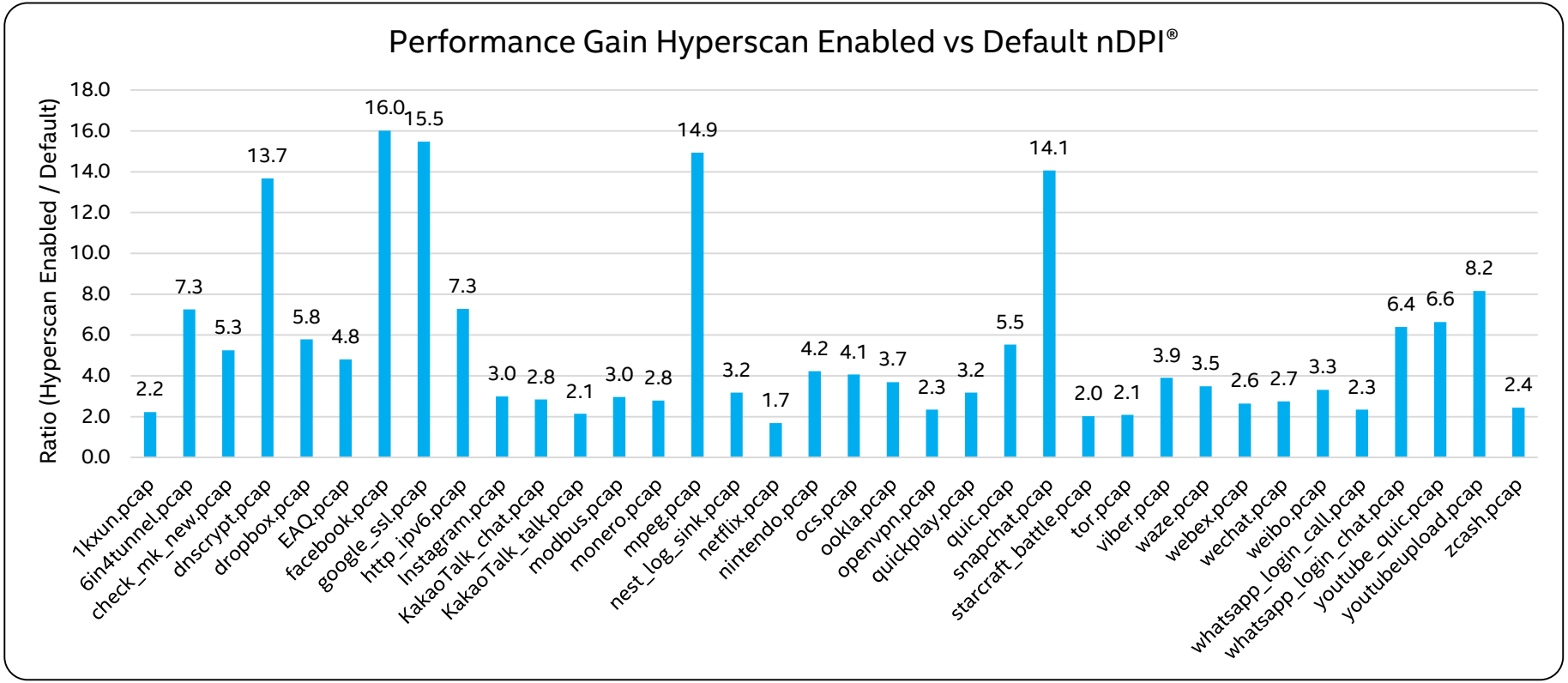
- Maintain a protocol database that mostly contains HTTP URLs
- Multiple literal matching (Hyperscan or Aho Corasisk) to detect URLs in packets
- Example: Wechat rules

```
{ ".wechat.com", NULL, "\\.wechat\\.com",    "WeChat",   NDPI_PROTOCOL_WECHAT,
NDPI_PROTOCOL_CATEGORY_CHAT, NDPI_PROTOCOL_FUN },
{ ".wechat.org", NULL, "\\.wechat\\.org",  "WeChat", NDPI_PROTOCOL_WECHAT,
NDPI_PROTOCOL_CATEGORY_CHAT, NDPI_PROTOCOL_FUN },
{ ".wechatapp.com", NULL, "\\.wechatapp",  "WeChat",  NDPI_PROTOCOL_WECHAT,
NDPI_PROTOCOL_CATEGORY_CHAT, NDPI_PROTOCOL_FUN },
{ ".we.chat", NULL, "\\.we\\.chat", "WeChat",  NDPI_PROTOCOL_WECHAT,
NDPI_PROTOCOL_CATEGORY_CHAT, NDPI_PROTOCOL_FUN },
```

nDPI Performance Evaluation

- nDPI® uses an application called ndpireader to give an application level protocol analysis on each pcap.
- Throughput in measured in Kpps(packets/sec).
  - The nDPI® throughput, Average Memory and Setup Time and noted for each run.
- nDPI® reads a total of 76 pcaps acquired from top sites. In this report we are only showing performance of top 36 pcaps.

nDPI Performance Ratio – Hyperscan vs Aho Corasick

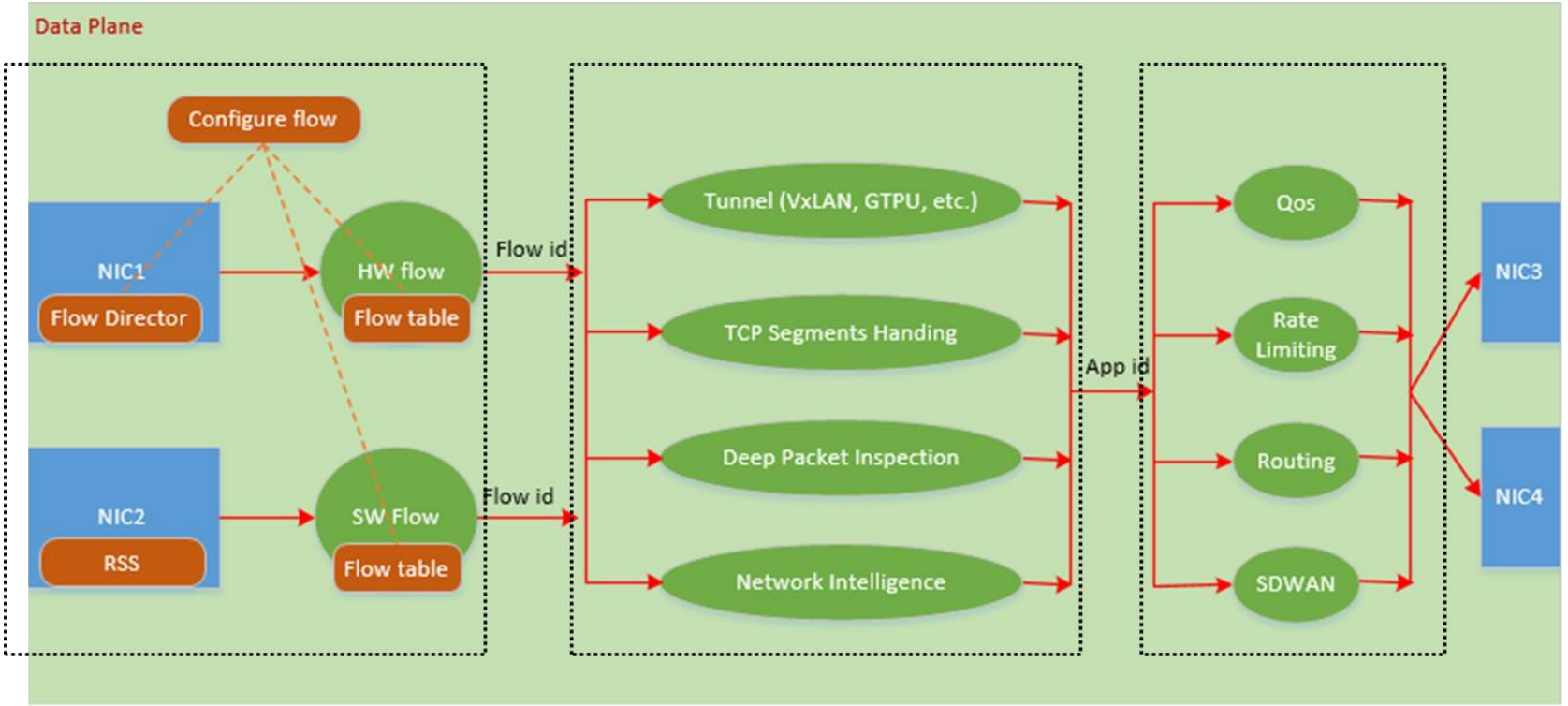


Pcap Based/ 1C1T/ nDPI reader, Intel* Xeon* Gold 6152 @ 2.10GHz

UDPI: Universal Deep Packet Inspection

- A new project under FD.io: https://wiki.fd.io/view/UDPI
- Flow Classification & Expiration
  - HW flow offloading leveraging rte_flow on DPDK
  - SW flow classification
- Application Detection
  - Leverage Hyperscan Stream Mode
  - Reassembly TCP segments on the fly
- Application-based Actions
  - HQos, Rate Limiting, Policy Routing, SDWAN, etc.
- Support Hundreds of Protocols & Applications
  - TLS/HTTPS, HTTP, DNS, QUIC, etc.

UDPI Architecture



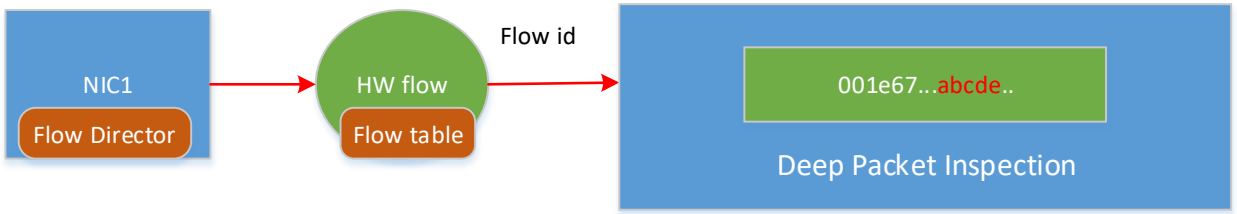
1. Flow Classification

2. Application Detection

3. Application-based Action

Block Mode and TCP Segments Reassembly

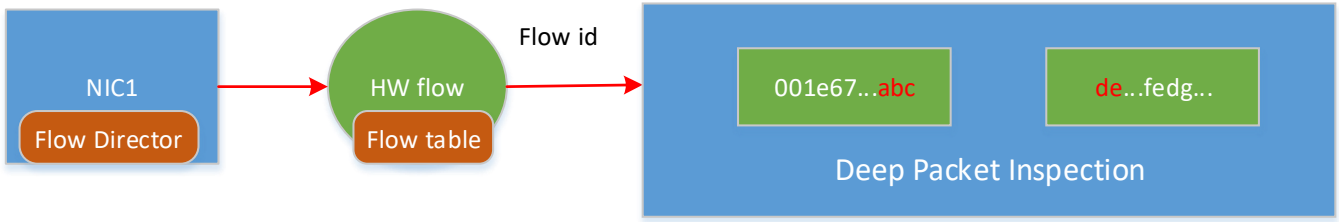
- Block mode can scan rules only in a complete payload.
- If defining a rule "abcde", then for Block Mode, "abcde“ should be in a complete PDU payload.



- Requirements for TCP Segments process:
  - Reassembly TCP segments first to a complete PDU payload.
  - Scan PDU payload through Block mode.
  - Fragment TCP segments again.
- This degrades the performance.
- Most DPI open source projects leveraging Hyperscan performs in this way.

Stream Mode and TCP Segments Reassembly

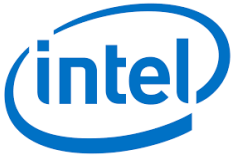
- Stream mode can scan rules straddling into different TCP segments.
- If defining a rule "abcde", then for Stream Mode, then "abc" can be reside in packet 1, and "de" can be in packet 2.



- Requirements for TCP Segments process:
  - Reassemble TCP segments reassembly on the fly.
  - Can handle out-of-order tcp segments.
  - Can handle overlapping segments.
- This helps to improve the performance.
- UDPI project is implemented in this way.

UDPI Founders and Committers

13 organizations joined and **20 initial committers**.



Committers:

- Xiang Wang, @ Intel,
- Yang Hong, @ Intel,
- Harry Chang, @ Intel,
- Jian Gu, @ ZTE,
- Jianghua Shan, @ China Telecom,
- Yang Zhang, @ China Telecom,
- Xingfu Li, @ HuachenTel,
- Shuai Wu, @ Inspur,
- Yuying Xia, @ Yxlink,
- Chenggang Fan, @ Sunyainfo,
- Feng Gao, @ Tencent,
- Zhong Liu, @ China Unicom,
- Yong Zhao, @ Huawei,
- Haiquan Chen, @ QingCloud,
- Jim Thompson, @ Netgate,
- Pengjie Li, @ Alibaba,
- Zhao Zhang, @ Alibaba,
- Zhangpeng Xie, @ Alibaba,
- Drenfong Wang, @ Intel,
- Hongjun Ni, @ Intel,

Conclusion

- Solid and mature with better performance than state-of-the-art solutions
- Still a WIP in many senses, especially for Icelake optimizations
- Used in large number of commercial deployments and delivers significant performance boost for DPI applications
- UDPI provides a complete open source DPI solution and call for collaboration and contributions