COIN H. Singh
Internet-Draft MNK Labs and Consulting
Intended status: Informational M-J. Montpetit
Expires: August 22, 2021 Concordia Univeristy
February 18, 2021

Requirements for P4 Program Splitting for Heterogeneous Network Nodes
draft-hsingh-coinrg-reqs-p4comp-03

Abstract

For distributed computing, the P4 research community has published a paper to show how to split a P4 program into sub-programs which run on heterogeneous network nodes in a network. Examples of nodes are a network switch, a smartNIC, or a host machine. The paper has developed artifacts to split program based on latency, data rate, cost, etc. However, the paper does not mention any requirements. To provide guidance, this document covers requirements for splitting P4 programs for heterogeneous network nodes.

Table of Contents

1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Introduction

The research paper [FLY] covers splitting a P4 program into sub-programs to run the sub-programs on heterogeneous network nodes. There are certain issues to discuss first because some P4 code cannot be split to run elsewhere. There are other issues as well. For brevity, this document uses the terms smartNIC and NIC interchangeably.

In a data center, host machines are connected to a switch. In an Enterprise network, P4 data plane replicates ARP [RFC0826] and IPv6 ND [RFC4861] messages for layer-2 address resolution. If a program split moves ARP and IPv6 code to smartNIC, the hosts should also move to smartNIC. If hosts do not move, the switch resolves layer-2 destinations and messages the NIC with ARP or IPv6 ND table update. But the switch is forwarding traffic at 12 Tbps and for any layer-2 lookup, the switch has to message the NIC which slows down switch forwarding. If hosts also move with ARP and IPv6 ND to the NIC, there are still issues. A NIC with two 100G ports will not be able to support all 25G hosts on a switch with 32 ports. So multiple NICs are used. If a switch is used in bridged mode, there is a single

link-local domain for ARP and IPv6 ND. If the switch is used as a layer-3 switch, then one interface with layer-3 addresses can operate the switch. With multiple NICs, each NIC has its own link-local domain and if configured, a layer-3 interface. So hosts on one NIC go through an additional router to communicate with hosts on another NIC. On the switch, running in bridged mode, the router is not needed.

In a public cloud, Azure resolves layer-2 destination with a central controller and thus the switch does not use any data plane broadcast or IPv6 ND multicast addresses. However, this network faces the same issue mentioned above when multiple NICs are used. Google resolves layer-2 via a proprietary Neighbor Discover protocol [GOOG]. How does Flightplan [FLY] deal with three such disparate networks?

Regarding BGP, if a CLOS network runs BGP, BGP operates between LEAF and SPINE switches. If BGP data plane table splits to a smartNIC, you have to assign an IP address for BGP peer on host CPU. Now the host CPU runs BGP control plane and NIC stores BGP data plane tables. But both Azure and AWS (Amazon Web Services) do not run any SDN or BGP control plane on host because such network activity steals key cycles from host CPU. There is another major problem. Hosts routinely move in the data center to load balance. With a host move, the BGP peer may move to a totally different subnet and break the BGP network.

The punt or divert path of a data plane processes ARP, IPv6 ND, and any routing control messages. Production quality switches (or routers) also run a punt rate-limiter in the data plane so that the switch/router CPU is not inundated. In a heterogeneous network, it is not just how close one punts packets to CPU, but also what else moves with punt path? Certainly the data plane punt rate-limiter also moves.

3. Terminology and Abbreviations

CPU - Central Processing Unit.

DPDK - Data Plane Development Kit from Intel.

ClOS - leaf and spine switched network redundant topology.

FPGA - Field Programmable Gate Array.

NIC - Network Interface Card.

npu - Network Processing Unit.

smartNIC - a NIC with processor/FPGA.

TCAM - Ternary Content-Addressable Memory.

VPP - Vector Packet Processing from Cisco.

4. Requirements

The requirements are:

1. If the heterogeneous network includes a switch, the ARP and IPv6 ND data plane P4 code should not be split to run outside the switch.

2. Likewise ARP or IPv6 ND Proxy data plane code should not be split to run outside the switch.

3. BGP table should not be split and move outside the switch. Distributed BGP is a research topic.

4. A switch likely includes TCAM and thus the P4 program may use P4 ternary table match kind. If such a table is moved to another node due to program split, the node the code moves to is important. A FPGA (field-programmable gate array) does not use TCAM and a host machine may not either. The FPGA and host use hash-based table lookup. Depending on the table key size, an appropriate hash is required. Either the splitting tool prompts the user for what hash to use or deduces what hash - user input is desirable. For example, for a 6-tuple IPv4 key, a 128 bit key is used and for the same 6-tuple, the IPv6 key uses 320 bits. Appropriate hashes are required for such keys.

5. Splitting algorithms should not develop High Availability. Network deployments already use dual switches, or CLOS topology for redundancy. BFD [RFC5880] is recommended for use with liveliness detection.

6. Any automated tool that splits a P4 program to run on heterogeneous nodes, should provide a manual override. For example, a P4 program is compiled for a switching asic. The compiler raises an error saying code fits in N+2 pipeline stages but the asic has only N stages. In this case, an automated tool will just split the program. However, a manual override allows the programmer to tweak the code manually to fit. With manual tweaking I have been able to fit code in N-1 stages after getting an initial error from compiler for code using N+2 stages. Manual override could kick in if the number of stages used is (N + 16% x N).

7. The splitting tool should define clearly what is the punt path for P4 code running on a host. The reason is because the host CPU is the data plane, so where is the punted packet to CPU sent? For DPDK, I expect Linux user space to receive punted packets. For VPP, it supports a punt node.

5. Changes to P4 Compiler to Block Split

Using P4 Annotations to pass information to p4c (P4 compiler) backend [P4C] to not split certain code is not desirable. This document proposes to change p4c. A new table implementation property called atomic is added to p4c. If this atomic table implementation property is configured for a table in the P4 program, then the table and its actions and any table invocation code block are not split.

6. Discussion

The two largest public cloud operators are Amazon AWS and Microsoft Azure [NIC]. Both operators run Software Defined Networking (SDN) in the smartNIC. The reason is running SDN stack in software on the host requires additional CPU cycles. Burning CPUs for SDN services takes away from the processing power available to customer VMs, and increases the overall cost of providing cloud services. Azure uses a FPGA on smartNIC and programs the FPGA in Verilog, not P4. Amazon uses multi-core npu (Graviton uses 64 cores) on smartNIC and does not program Graviton in P4. Both these operators do not use host CPU or network switch for SDN operations. In future, even if both operators program smartNIC in P4, the operators do not have heterogeneous nodes running SDN. Likewise, in future, the switch runs a new SDN feature, e.g. switch caching of popular lookup, then there are heterogeneous nodes to apply Flightplan to.

7. Security Considerations

Use IPSec [RFC4301] to secure any control plane communications.

8. IANA Considerations

None.

9. Acknowledgements

Thanks (in alphabetical order by first name) to Nik Sultana for reviewing this document.

10. References

10.1. Normative References

[RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <https://www.rfc-editor.org/info/rfc826>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

10.2. Informative References

[FLY] Sultana, N., Sonchack, J., Giesen, H., Pedisich, I., Han, Z., Shyamkumar, N., Burad, S., DeHon, A., and B. T. Loo, "Flightplan: Dataplane Disaggregation and Placement for P4 Programs", November 2020, <https://flightplan.cis.upenn.edu/flightplan.pdf>.

[GOOG] Singh, A., "Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google Datacenter Network", September 2016, <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/7a2ef8424cdc3be32a4cb96bf3e3483eaf0b8949.pdf>.

[NIC] Firestone, D., "Azure Accelerated Networking: SmartNICs in the Public Cloud", April 2018, <https://www.microsoft.com/en-us/research/uploads/prod/2018/03/Azure_SmartNIC_NSDI_2018.pdf>.

[P4C]   Community, P., "P4_16 Reference Compiler - Github", May 2018, <https://github.com/p4lang/p4c>.

Authors' Addresses

Hemant Singh
MNK Labs and Consulting
7 Caldwell Drive
Westford, MA 01886
USA

Email: hemant@mnkcg.com
URI: https://mnkcg.com/

Marie-Jose Montpetit
Concordia Univeristy
1455 Boulevard de Maisonneuve O
Montreal, Quebec 01886
Canada

Email: marie@mjmontpetit.com