Efficient Cache Architecture for Table Lookups in an Internet Router

Hayato Yamaki
The University of Electro-Communications
Chofu, 182-8585, Japan

***Abstract*—Table lookup is the most important operation in routers from the aspects of both packet processing throughput and power consumption. To realize the table lookup at high throughput with low energy, Packet Processing Cache (PPC) has been proposed. PPC stores table lookup results into a small SRAM (static random access memory) per flow and reuses the cached results to process subsequent packets of the same flow. Because the SRAM is accessed faster with significant lower energy than TCAM (Ternary Content Addressable Memory), which is conventionally used as a memory for storing the tables in routers, PPC can process packets at higher throughput with lower power consumption when the table lookup results of the packets are in PPC. Although the PPC performance depends on the PPC hit/miss rates, recent PPCs still show high PPC miss rates and cannot achieve sufficient performance. In this paper, efficient cache architecture, constructed of two different techniques, is proposed to improve the PPC miss rate more. The simulation results indicated that the combined approach of them achieved 1.72x larger throughput with 41.4% lower energy consumption in comparison to the conventional PPC architecture.**

Keywords*—Router architecture; table lookup; Packet Processing Cache (PPC); Ternary Content Addressable Memory (TCAM)

I. Introduction

Recent increase in internet communication traffic amount is remarkable owing to the spread of data consuming applications such as cloud services, video streaming, and internet-of-things (IoT) applications. Accordingly, requirements of the throughput in routers become more and more serious. It reaches 400 Gbps or 1 Tbps in recent years. The increase in internet traffic also induces a problem of the power consumption in routers [1], [2]. The reports [3], [4] emphasized that several percentages of the energy generated in the world is consumed by network devices. Thus, routers must consider both the packet processing throughput and energy efficiency. This demand becomes more serious in core routers, which are placed in core networks and handle huge amount of internet traffic.

For routers, table lookup operation is the most slow and power consumed operation in packet processing [5], [6], [7]. When a packet arrives at router, the router needs to lookup tables (e.g., a routing table and ACL (access control list) to obtain the information required to process the packet, such as the output port and filtering decision. These tables are often stored into ternary content addressable memories (TCAMs) to gain high lookup performance, especially in core routers, which are placed at core networks and required to process huge amount of packets. The TCAM can lookup tables at one cycle by comparing all data stored in the TCAM simultaneously. However, owing to this power consuming operation, a TCAM significantly consumes energy in comparison to a same-sized SRAM [8]. According to [9], [10], 40% of the power consumption in routers is due to the TCAMs. TCAM is also insufficient from the viewpoint of the lookup performance. To achieve over 400 Gbps, it is required for routers to process packets every 1.25 nano seconds if the shortest length packets come continuously. However, recent TCAM products show the access latency of approximately 5 nano seconds. Thus, to increase table lookup throughput with reducing the energy, improving the table lookup is needed.

PPC (packet processing cache) is an attractive approach to meet the requirement [11], [12]. PPC can reduce the number of TCAM accesses by storing table lookup results into a small SRAM and reusing the stored results to process subsequent packets. PPC can finish the table lookups of a packet at high speed with low energy consumption when the corresponding TCAM lookup results of the packet are in the cache (i.e., a cache hit). Thus, the PPC performance depends on the number of cache hits/misses, and reducing the number of PPC misses is the most important issue to improve the PPC performance. However, PPC still cannot satisfy this requirement because of the following two problems: the high average PPC miss rate and the low attack tolerance. This study proposes a novel efficient cache architecture, which constructed of Port-aware Cache and Victim IP Cache, for high-throughput and low-power table lookup.

This paper extends the previous work of Yamaki et al [13]. Different from [13], this paper newly adds the more detailed analysis (Fig. 3) and the explanation of the concrete hardware (Fig. 6) in Sec. 4. In addition, to resolve the problem that the performance of Port-aware Cache depends on the network configurations, this study newly proposed Semi-static Port-aware Cache, which decides the best mix of each entry sizes by prior trials at a boot process of a router. Moreover, this study newly investigated the various sizes of Victim IP Cache and evaluated them in Sec. 6. The writing and figures in the manuscript are also improved for easy understanding as a whole. The main contribution of the paper is summarized below.

- This paper indicated that HTTP and DNS packets impacted on the PPC performance significantly from the perspective of the number of packets and flows.
- Port-aware Cache, one of the proposed approach in this paper, can not only prevent increases in PPC misses caused by attacks (8.64% improvement) but also reduce the number of PPC misses caused by HTTP packets (9.02% improvement).

- 64KB Victim IP Cache, the other approach of this paper, can save the 85.5% of all the packets missed in PPC by caching them to a victim cache per destination IP address.
- The simulation results showed that the combination of the two approaches can achieve 1.72x larger throughput with 41.4% lower energy in comparison to the conventional PPC.

The rest of the paper is organized as follows. First, more detailed architecture of PPC and the problems are shown in Section 2. Section 3 introduces the relative works of reducing the cache miss in PPC. In Sections 4 and 5, the two proposed architecture, Port-aware Cache and Victim IP Cache, are proposed, respectively. Section 6 evaluates the proposed architecture, and finally we conclude the work in Section 7.

II. Packet Processing Cache

A. Outline

PPC has been proposed as a supplemental approach of TCAM lookup and can realize high-throughput and low-power table lookup by reducing the number of TCAM accesses. In PPC, a flow is defined based on the five tuples (i.e. source/destination IP addresses, source/destination port numbers, and protocol number) of packets. PPC stores TCAM lookup results per flow into a small SRAM and references the stored results to process subsequent packets of the same flow. Because the packets of the same flow are processed using the same TCAM lookup results, PPC enables to process packets using the SRAM without accessing TCAM when PPC has the TCAM lookup results of the flow.

Fig. 1 shows the table lookup flow with PPC. Conventionally, PPC entries are constructed of the 13 bytes tags (i.e., five tuples) and 15 bytes data (i.e., TCAM lookup results). The TCAM lookup results include the routing table lookup result of 1 byte, ARP table lookup result of 12 bytes, ACL lookup result of 1 byte, and QoS table lookup result of 1 byte. PPC entries are addressed using the hash value of the five tuples as the index. Typically, a 32KB small SRAM (i.e., approximately 1,024 entries) is used as PPC considering the SRAM latency. It is because there is few latency gap between two memories (i.e., TCAM and PPC), unlike processor caches. For example, L2 cache latency of microprocessors is almost the same of that of TCAM (approximately 5 nano seconds).

B. Problems

The table lookup performance with PPC is mainly determined by a PPC miss rate because PPC accesses are significantly faster with lower energy consumption than TCAM, and they are almost negligible. Thus, achieving low PPC miss rate is the most important issue for PPC. However, PPC has two problems to meet this requirement.

1) Tolerance to Attacks: PPC has little attack tolerance because it may register a large number of attack flows in PPC when one-packet-based attacks, such as port-scan attacks, pass through a router. Consequently, many useful PPC entries are evicted by attack flows, and it causes the significant degradation in the PPC hit rate. The attacks induce two disadvantages to PPC. First, attack flows created by such attacks never hit in PPC because they are mainly constructed of one packet. Second, useful PPC entries are evicted by attack flows.

2) A large number of TCAM accesses: If a PPC miss occurs, a router must access TCAMs several times (four times in the case of Fig. 1) to obtain each table lookup result. Thus, the number of PPC misses significantly impact on the table lookup performance. However, the state-of-the-art PPC still remains the PPC miss rate of 30% [14], [15]. It indicates that 30% of all packets still access to TCAM. Especially considering the power consumption, further improvement in the PPC miss rate is important. The most effective approach to meet this requirement is to increase the PPC entries: However, it is not reasonable from the following two reasons. First, PPC cannot increase the capacity largely due to the access latency, as mentioned before. As the PPC capacity, the size like L1 caches in microprocessors (i.e. 32KB) is acceptable. Second, PPC easily increase the capacity because of the large PPC entry size (28KB per entry).

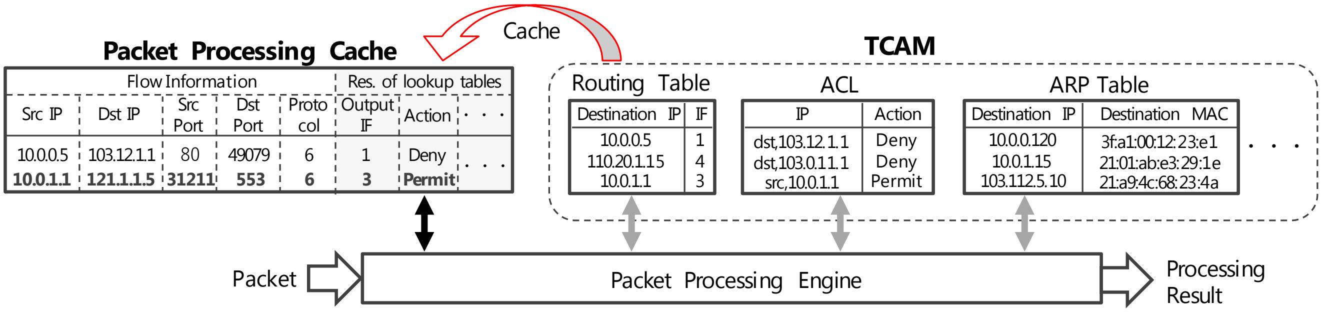
III. Related Studies

In this section, related studies of this work are introduced. Although there are many studies focusing on PPC, the attack tolerance of PPC was not considered in all studies. Thus, this section shows studies of improving the average PPC miss rate.

One approach to improve the PPC miss rate is that reducing the cache tag information and increasing the PPC entries without increasing the capacity. As mentioned in Section 2, the cache tag of PPC is 13-byte flow information, and it is one of the reason that PPC has a few entries. Digest Cache was proposed Chang et al. [16]. This method uses the hash values calculated from the five tuples as cache tags instead of the five tuples. Likewise, Ata et al. proposed the cache which used the three tuples (i.e., source/destination IP addresses and smaller port number) as cache tags instead of the five tuples [17]. These works effectively reduce the TCAM accesses by increasing the number of stored flows. However, the compressing tags cause cache conflicts, and thus, an extra hardware for avoiding the cache conflicts is required.

As other approaches, there are studies of reducing the PPC misses by improving the hash conflicts. The paper [18] emphasized that CRC hash function is not appropriate as the cache indexes of PPC and it caused many cache conflicts. They proposed a novel PPC indexing method, which split the cached area of PPC into two areas and used two different hash functions. They showed two universal hash functions [19] are effective to reduce the cache conflicts. The one problem of this method is that the implementation cost of the universal hash functions is high due to the large input-data size (i.e., 13-byte five tuples).

Improving the cache replacement policy is one of the effective approach to reduce the PPC misses. Kim et al. pointed out that LRU does not fit for PPC because LRU determines the replaced entry based on only the last packet of the flow and cannot consider the flow characteristics [20]. They proposed two types of cache-replacement algorithms which utilize last two packets information of the flow to determine the replaced entry and reduced the PPC misses by several percentages compared with LRU. However, the hardware cost of storing last two-packet information, was not discussed. The increase



Packet Processing Cache

Flow Information					Res. of lookup tables		
Src IP	Dst IP	Src Port	Dst Port	Proto col	Output IF	Action	· · ·
10.0.0.5	103.12.1.1	80	49079	6	1	Deny	· · ·
10.0.1.1	**121.1.1.5**	**31211**	**553**	**6**	**3**	**Permit**	

Routing Table

Destination IP	IF
10.0.0.5	1
110.20.1.15	4
10.0.1.1	3

ACL

IP	Action
dst,103.12.1.1	Deny
dst,103.0.11.1	Deny
src,10.0.1.1	Permit

ARP Table

Destination IP	Destination MAC
10.0.0.120	3f:a1:00:12:23:e1
10.0.1.15	21:01:ab:e3:29:1e
103.112.5.10	21:a9:4c:68:23:4a

Fig. 1. Outline of table lookups with PPC.

in the PPC capacity due to this additional stores becomes a serious problem for the small-sized cache like PPC.

Yamaki et al. also proposed methods of reducing the PPC misses [12]. They focused on one-packet flows created applications such as DNS (domain name system), DoS (denial of service) attack, and port scan attacks because these flows never hit in PPC and proposed methods of denying packets from these flows. The simulation results showed that DNS-Aware Cache, one of the proposed methods, can reduce the number of PPC misses by 6%.

Although these approaches are effective to reduce the number of PPC misses, PPC still shows high PPC miss rate, as mentioned in the previous section. In addition, these approaches are not effective to prevent the attack influence.

IV. Port-aware Cache

This study first proposes Port-aware Cache to improve the PPC miss rate and reduces a negative impact of attacks. Port-aware Cache stores flows per application group by assigning different cache areas to each application group. As a result, an increase in PPC misses caused by attacks can be avoided by isolating the impact of each application group. Furthermore, it also improves the average PPC miss rate by assigning suitable number of entries in each application group.

A. Motivation

To identify the construct of a flow (e.g., the number of packets composing a flow), the application of the flow is one of the important information. We explain it referring DNS (domain name system), HTTP (hypertext transfer protocol), and several types of attacks for examples. In general, a DNS flow consists of one packet because of the simple request-reply communication. In this case, DNS communication creates two one-packet flows (the request flow and the reply flow). On the other hand, an HTTP flow consists of a large number of packets because HTTP protocol requires 3-way handshake at first and sends internet contents subsequently. Like this, it is expected that one-packet-based attacks, such as vulnerability-based attacks and port-scan attacks, create a significant large number of flows with a small number of packets. Basically, the application is identified from the port numbers of the packet.

Fig. 2 shows the top 5 ports of sending packets in a network and the amount of packets and flows. In this measurement and following analyses, An in-house traffic-analysis program and core-network traffic in Japan, called WIDE traffic, are used. Details of them are described in Sec. 6.

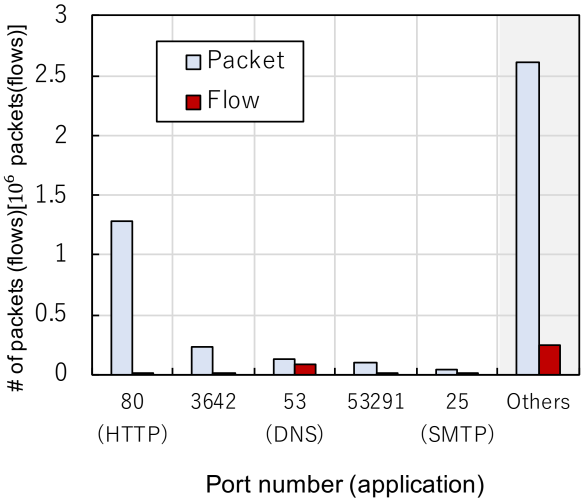


Fig. 2. Top 5 ports of sending packets in a network and the amount of packets and flows.

As shown in Fig. 2, although the HTTP packet amount is dominant in the network (30%), the HTTP flow amount is not a large portion in the network (4.8%). On the other hand, although the DNS flow amount is dominant in the network (25%), the DNS packet amount is not a large portion in the network (2.8%). This is caused by the difference in the application protocol, as mentioned above. This result also mean that HTTP and DNS especially impact on the PPC miss rate because of a large amount of packets or flows. More specifically, PPC miss rate greatly depends on the number of PPC misses caused by HTTP packets, and DNS flows affect on PPC entries due to the considerable insertion.

Attack flows also have individual characteristics. As mentioned in Sec. 2.2, some attack flows are constructed of a few packets and sent to internet drastically in a short period. Consequently, useful PPC entries are evicted and polluted. Fig. 3 and 4 shows the impact of attacks on the PPC miss counts. These figures show the breakdown of PPC hits and PPC misses from the perspective of applications with two sizes

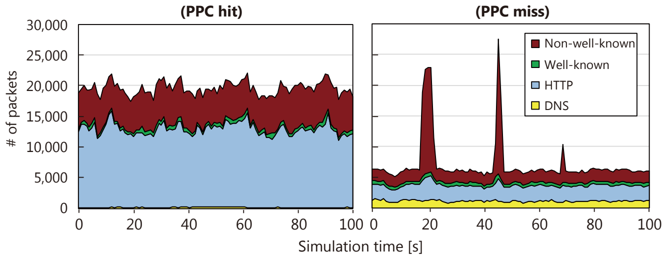


Fig. 3. The breakdown of PPC hits (the left figure) and PPC misses (the right figure) from the perspective of applications with 1K PPC entries.

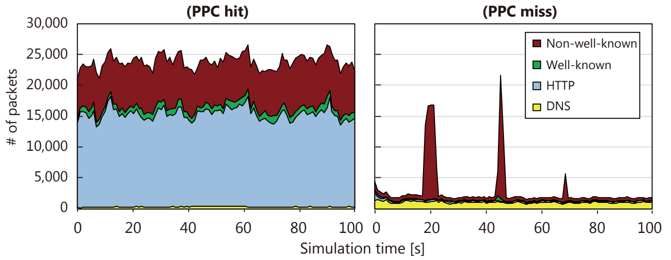


Fig. 4. The breakdown of PPC hits (the left figure) and PPC misses (the right figure) from the perspective of applications with 1M PPC entries.

of PPC. The non-well-known in the graph is defined as the packets whose port numbers are not well-known ports. In this measurement, three attacks were observed at 19s, 43s, and 68s. These figures indicate that a drastic increase in PPC misses is caused by attack flows. In addition, attack flows indirectly cause an increase in PPC misses caused by HTTP and other-application packets when the PPC size is small. It is because attack flows evict a large number of useful flows of HTTP and other applications when attacks occur. Note that PPC misses caused by DNS packets are not affected by attacks and the number of PPC entries because DNS packets hardly hit in PPC. On the other hand, most packets of HTTP and other applications have potential to hit by preparing a large number of entries.

B. Architecture of Port-aware Cache

From the discussion in Section 4.1, this study proposes Port-aware Cache In Port-aware Cache, PPC area is divided into 3 ranges: the DNS, HTTP, and other-application ranges. From the results of Fig. 3 and 4, the well-known ports and non-well-known ports are not distinguished because the flows of the well-known ports hardly impact on the PPC hits/misses.

The whole architecture of Port-aware Cache is depicted in Fig. 5. For the access to each application range, the address of PPC is recalculated from the CRC hash value of the five tuples and the smaller port number. This calculation is processed in the Modifier module. Details of the Modifier module is shown in Fig. 6. Offset 1, offset 2, and offset 3 depicted in Fig. 6 show the number of entries in DNS, HTTP, and other-application ranges, respectively. The modifier module adequately selects the address based on the smaller port number.

There are two advantages of Port-aware Cache. First, it is expected that Port-aware Cache reduces the number of PPC misses by assigning appropriate number of PPC entries to each application range. As mentioned in Section 4.1, DNS packets not only rarely hit in PPC but also disturb PPC entries. From this reason, assigning a few PPC entries to the DNS range is better. Unlike DNS, HTTP packets have many opportunities to hit in PPC. Consequently, assigning a large portion of PPC entries to the HTTP range is better. Second, Port-aware Cache can suppress the negative impact of attacks to the PPC miss rate by isolating each application range. As shown in Fig. 3 and 4, attack packets not only cause PPC misses of their own but also evicts useful flows, such as HTTP flows, and impede PPC hits. This problem is resolved in Port-aware Cache by isolating flows to each application range.

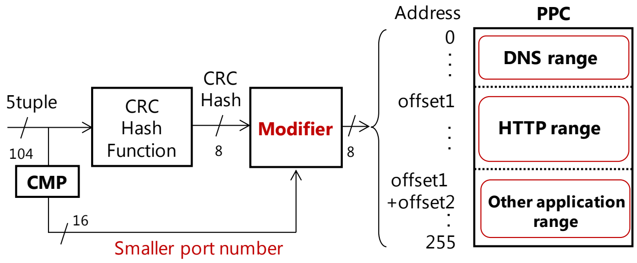


Fig. 5. Architecture of Port-aware Cache.

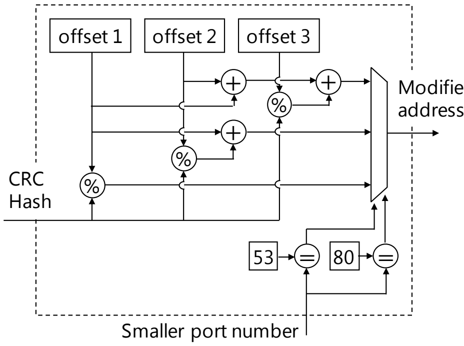


Fig. 6. Details of Modifier.

C. Semi-static Port-aware Cache

In this section, Semi-static Port-aware Cache, which is an improvement in Port-aware Cache, is introduced. In Port-aware Cache, the entry sizes of each application range (i.e., the offsets 1, 2, and 3 in Fig. 6) are an important factor to decide the cache performance. In addition, it is considered that the best mix of the entry sizes of each application area varies depending on networks. From these reasons, Semi-static Port-aware Cache decides the best mix of the entry sizes from prior trial at the boot process of a router.

Fig. 7 shows the block diagram of Semi-static Port-aware Cache. Semi-static Port-aware Cache explores the best mix by trying various configurations of Port-aware Cache using a packet log of 1 minute captured at a boot process of a router. However, trying all the possible configurations is not realistic due to the explosive combinations. Thus, Semi-static Port-aware Cache uses heuristic approach to explore the best

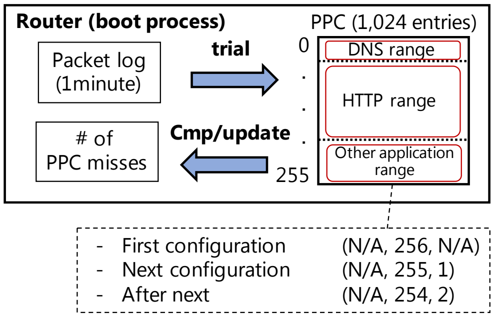


Fig. 7. Block diagram of Semi-static Port-aware Cache. The configuration (A, B, C) means the assigned addresses in the DNS range, the HTTP range, and the other-application range, respectively.

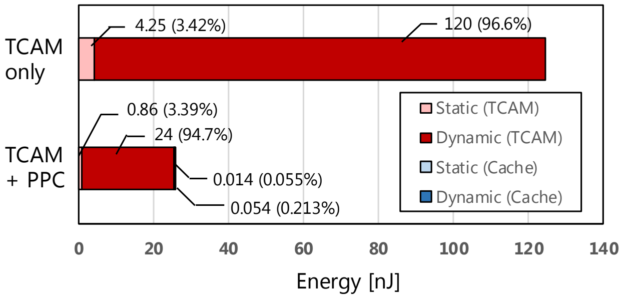


Fig. 8. Breakdown of energy consumption of the table lookups per packet.

mix of the entry sizes. The basic idea is to assign PPC entries to the HTTP range as many as possible.

More specifically, as the beginning of the trial, all the PPC entries are assigned to the HTTP range, as shown in the first configuration depicted in Fig. 7. The next process is as follows. (1) The number of PPC misses in the case of PPC with the first configuration is measured using the packet log, and the result is stored to a register. (2) The PPC configuration is updated by subtracting one to the HTTP range and adding one to the other-application range. (3) The number of PPC misses is newly measured using the same packet log, and the result is compared to the registered result. (4) If the new result is smaller than the registered one, the new result is stored to the register, and the process is returned to (2). (4)' On the other hand, if the registered result is smaller, the process is returned to (2), and the DNS range is added by one instead of adding one to the other-application range hereafter. (5) If the registered result is smaller again, the current configuration is decided as the best mix. By applying Semi-static Port-aware Cache, the maximum number of trials to decide the best mix of the entry sizes becomes the maximum number of indexes (i.e., 256 times in the case of 4-way PPC with 1,024 entries).

V. Victim IP Cache

Victim IP Cache is also proposed in this study to further improve the TCAM access rate. It is placed between PPC and TCAMs and accessed if PPC misses occur. When a packet misses in PPC, the packet accesses Victim IP Cache before accessing TCAM.

A. Motivations

As discussed in Section 2, a router with PPC still requires to access TCAMs because of a large PPC miss rate. Fig. 8 depicts the comparison of the energy consumption of the table lookup with TCAM only approach and that with PPC-based approach. We also showed the breakdown of the energy consumption in Fig. 8. Details of the method for calculating the energy consumption of the table lookup is explained in Section 6. The graph indicates that PPC significantly improves the energy consumption of the table lookup by reducing the number of TCAM accesses. However, the TCAM still consumes a large portion of the energy consumption of the table lookup, while the energy consumed by the SRAM is almost negligible. For this, further improvement in the number of TCAM access is required.

To realize this, this study focused on IP Cache, which is another study of caching packets. Unlike PPC, IP Cache stores only routing table lookup (and ARP table lookup) results and reuses them by using destination IP addresses of packets as the lookup keys. Past studies show that IP Cache can achieve the PPC hit rate of more than 90% [6]. In this paper, IP Cache and PPC are combined to reduce the whole number of TCAM accesses more.

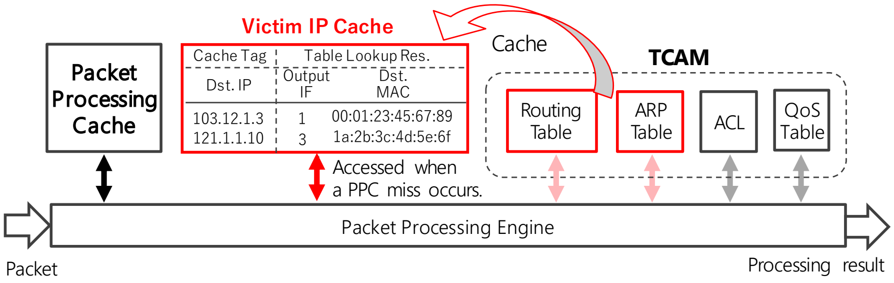
B. Architecture of Victim IP Cache

Victim IP Cache supports PPC by storing the some table lookup results of PPC miss flows. Different from PPC, which uses five tuples as cache tags, Victim IP Cache caches only the results of the routing table and ARP table, and thus, it uses only the destination IP address as cache tags. When a packet misses in PPC, the packet accesses Victim IP Cache. At this time, if a cache hit occurs in Victim IP Cache, the TCAM accesses for searching the routing table and ARP table are omitted. As explained in the previous section, because IP Cache has a possibility to obtain higher cache hit rate than PPC owing to the high temporal locality, Victim IP Cache may save a large number of packets which miss in PPC.

Fig. 9 shows the outline of the table lookups with Victim IP Cache. Victim IP Cache entries are constructed of 4-byte destination IP address as cache tags and 1-byte output port information and 6-byte destination MAC address as cache data, namely, 11 bytes per entry. This low entry size makes the number of Victim IP Cache entry size larger than PPC, and thus, Victim IP Cache has a possibility to obtain higher cache hit rate. Different from a flow-based victim cache, the entries of Victim IP Cache are not shifted to PPC when cache hits occur in Victim IP Cache.

VI. Evaluations

This section provides the evaluation of the proposed cache architecture using an in-house PPC simulator and packet traces captured in real networks. We first show the evaluation of Port-aware Cache from the perspectives of the PPC miss reduction and the attack tolerance. Next, Victim IP Cache is evaluated by comparing to a typical victim cache. Finally, the combination of Port-aware Cache and Victim IP Cache is evaluated from the perspectives of the throughput and energy consumption.



Cache Tag	Table Lookup Res.	
Dst. IP	Output IF	Dst. MAC
103.12.1.3	1	00:01:23:45:67:89
121.1.1.10	3	1a:2b:3c:4d:5e:6f

Fig. 9. Outline of table lookups with Victim IP Cache.

TABLE I. Simulator parameters.

	Parameter	Value
	Clock frequency	2 GHz
Cache	Number of entries	1,024
	Number of ways	4
	Replacement policy	LRU
	Access latency	0.5 ns
TCAM	Access latency	5 ns

TABLE II. Details of network traces.

Trace Name	Packets/sec.
IPLS [22]	99,264
UFL [22]	51,319
MRA [22]	41,372
FRG [22]	32,722
CNIC [22]	31,023
WIDE [23]	24,657
PSC [22]	22,807
APN [22]	20,793
TXG [22]	11,610
COS [22]	7,972
BUF [22]	7,827

TABLE III. Simulation result of Semi-static Port-aware Cache.

		UFL	WIDE	APN	TXG
(Conv. PPC)	PPC miss rate [%]	26.6	25.5	39.2	16.9
(Static)	PPC miss rate [%]	28.8	23.2	45.0	30.9
	DNS range [entries]	12	12	12	12
	HTTP range [entries]	704	704	704	704
	Other-app. range [entries]	308	308	308	308
(Semi-static)	PPC miss rate [%]	26.4	23.2	38.2	16.7
	DNS range [entries]	4	12	44	16
	HTTP range [entries]	380	704	36	88
	Other-app. range [entries]	640	308	944	920

A. Experimental Setup

For the simulation, an in-house PPC simulator, which was written in C++, and 11 types of packet traces were used. Details of the parameters set in the simulator and packet traces are summarized in Tables I and II. We configured the associativity of PPC to 4-way set associative and the total number of PPC entries to 1,024 entries, which were the typical configuration of PPC. The SRAM latency was set to 0.5 nano seconds from the estimation of CACTI 6.5 [21], which was a major tool for simulating the memory. The packet traces used in this simulation were captured in various universities or laboratories networks (obtained from RIPE Network Coordination Centre [22]) and a core network in Japan (obtained from WIDE MAWI WorkingGroup [23]).

The PPC simulator can simulate the table lookup operations in a router with PPC. First, the flow information (i.e., the five tuples) of a packet is extracted from a trace file in accordance with the timestamp of the packet. After reading a packet, the packet is sent to PPC and judged whether a PPC hit or miss. If the packet hits in PPC, the simulator finishes processing of the packet. On the other hand, if the packet misses in PPC, the packet is sent to a TCAM module. After passing the TCAM access latency, the processing of the packet is finished, and a new PPC entry is added. The simulator measures the PPC miss rate every seconds.

B. Evaluation of Port-aware Cache

1) Usefulness of Semi-static Port-aware Cache: First, Semi-static Port-aware Cache was evaluated. Table III shows the PPC miss rates of Semi-static Port-aware Cache measured in four networks. Note that this paper showed the results of the four networks because of the same trend. For comparison, this paper also showed the PPC miss rates of conventional PPC and naive Port-aware Cache (referred to as Static). In naive Port-aware Cache, the entry sizes of the DNS range, HTTP range, and other-application range were set to 12, 704, and 308 entries, respectively, which are the best mix derived from the WIDE trace.

Table III indicates that Semi-static Port-aware Cache can improve the PPC miss rates of all networks compared to conventional PPC, while naive Port-aware Cache improves the PPC miss rate of only the WIDE trace. It means that the best mix of the entry sizes varies depending on the networks and that Semi-static Port-aware Cache fits this demand.

2) Attack Tolerance: The attack tolerance of Port-aware Cache was evaluated. To evaluate this, the WIDE trace was used because eight attacks were observed in this trace at 19 second, 46 second, 69 second, 286 second, 465 second, 723 second, 737 second, and 783 second. Fig. 10 shows the PPC miss rates of Port-aware Cache and conventional PPC. When compared to conventional PPC, Port-aware Cache can reduce the number of PPC misses by 8.64% on average against attacks.

Moreover, we analyzed the breakdown of PPC misses in the WIDE trace from 0 second to 100 seconds and showed it in Fig. 11. In comparison to Fig. 3, Port-aware Cache can not

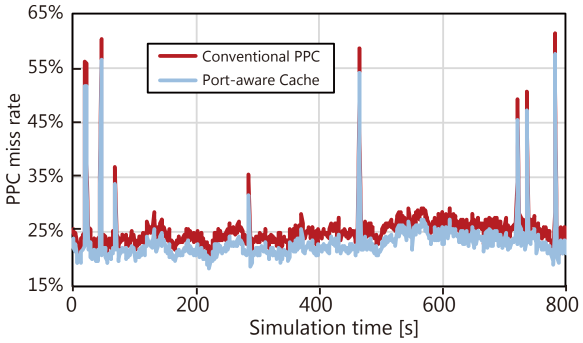


Fig. 10. PPC miss rates of Port-aware Cache and conventional PPC in WIDE trace.

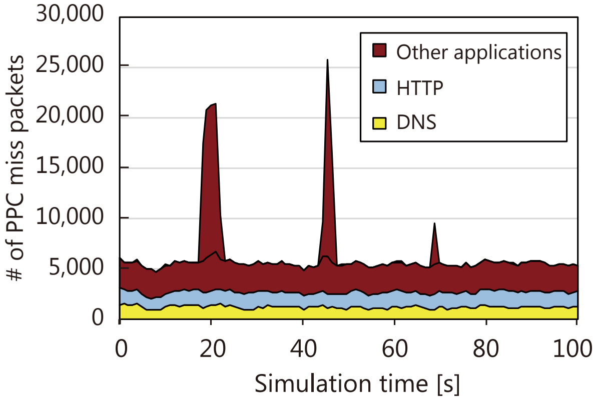


Fig. 11. The breakdown of PPC misses in the case of Port-aware Cache.

only prevent increases in PPC misses caused by attacks but also reduce the number of PPC misses caused by HTTP packets. However, PPC misses caused by other-application packets are still remained largely, and thus, reducing them is an important issue for further improvement.

C. Evaluation of Victim IP Cache

1) Improvement in avg. PPC miss rate: To reveal the usefulness of Victim IP Cache, we also implemented a typical flow-based victim cache and compared the PPC miss rates. The typical flow-based victim cache stores packets missed in PPC per flow. The PPC miss rates are summarized in Table IV. The number of victim cache entries was varied from x1 (compared to PPC, i.e., 32KB) to x8 (i.e., 256KB). As shown in the table, the number of PPC misses can be reduced significantly by Victim IP Cache. For example, 64KB Victim IP Cache improved the PPC miss rate by 78.7% compared to conventional PPC, while the 64KB typical flow-based victim cache improved them by 32.9%. The results showed that Victim IP Cache is more effective than a typical flow-based victim cache. As a future work, there is a room for further improvement in the PPC miss rate by combining the typical flow-based victim cache and Victim IP Cache.

2) Throughput and Energy: As evaluated in the previous section, Victim IP Cache can significantly improve the cache miss rate. However, different from PPC, the cache miss rate of Victim IP Cache does not directly represent the throughput and energy of the table lookup because packets hit in Victim IP Cache must access TCAMs to search the ACL and QoS table. This subsection introduces throughput and energy models of the table lookup with PPC and Victim IP Cache and estimated them based on the calculation. The throughput and energy models are already considered in [14], and this study extends them to evaluate Victim IP Cache.

First, the throughput model was extended. The table lookup throughput obtained by PPC and Victim IP Cache, represented as T, is calculated as (1).

$$T = \frac{l}{t_{avg.}} = \min\left(\frac{l}{t_{ppc}}, \frac{l}{t_{vic}m_{ppc}}, \frac{l}{t_{tcam}(2m_{diff}/n + m_{vic})}\right) \tag{1}$$

Here, t_{ppc}, t_{vic}, t_{tcam}, and $t_{avg.}$ represent the PPC, Victim IP, TCAM, and average lookup latency, respectively, and m_{ppc} and m_{vic} represent the cache miss rates of PPC and Victim IP Cache, respectively. Moreover, m_{diff} represents the gap of the cache miss rates between m_{ppc} and m_{vic}. The variables n and l in (1) represent the number of tables in a router and the packet length, respectively. In this study, we supposed four tables and 64 bytes as n and l, respectively. Equation (1) means that the table lookup throughput is restricted by the minimum throughput among PPC, Victim IP Cache, and TCAM. In comparison to conventional PPC, Victim IP Cache can achieve higher throughput by increasing the achievable throughput of TCAM.

Next, we extended the energy model of the table lookup in a router. The energy consumed by the table lookup with PPC and Victim IP Cache, represented as E, is calculated as (2).

$$\begin{aligned} E =& D_{ppc} + D_{vic}m_{ppc} + D_{tcam}(2m_{diff} + nm_{vic}) \\ &+ (S_{ppc} + S_{vic} + S_{tcam})t_{avg.} \end{aligned} \tag{2}$$

Here, D_{ppc}, D_{vic}, and D_{tcam} represent the dynamic energy of PPC, Victim IP Cache, and TCAM per access, while S_{ppc}, S_{vic}, and S_{tcam} represent the summation of the static power of each memories. Equation (2) means that introducing Victim IP Cache increases the static power and dynamic energy of Victim IP Cache although it can reduce the dynamic energy of TCAM.

Based on (1) and (2), the throughput and energy consumption of the table lookup with PPC and Victim IP Cache were estimated. The latency and energy consumption of each memory were estimated using CACTI. The calculated values of the throughput and energy consumption are summarized in Tables V and VI. As shown in Table V, Victim IP Cache realizes 1.65x higher throughput in comparison to conventional PPC. However, it also shows that the TCAM access still restrict the table lookup throughput. Furthermore, as shown in Table VI, the total energy consumption of the table lookup can be reduced by 39.1% using Victim IP Cache in comparison to the conventional PPC while typical flow-based victim cache reduce it by 32.6%. Thus, introducing the victim caches is effective to reduce the energy consumption of the table lookup although it additionally consumes the energy of the victim caches.

TABLE IV. AVERAGE PPC MISS RATES OF VICTIM IP CACHE IN VARIOUS NETWORKS.

	IPLS	**UFL**	**MRA**	**FRG**	**CNIC**	**WIDE**	**PSC**	**APN**	**TXG**	**COS**	**BUF**	**Mean**
Conventional PPC	26.5%	26.2%	39.3%	5.14%	8.21%	23.4%	11.3%	36.4%	15.3%	11.6%	20.7%	17.3%
w/ typical victim cache x1	22.7%	17.7%	32.7%	3.60%	5.22%	19.5%	8.59%	30.3%	11.0%	9.17%	19.7%	13.4%
x2	19.4%	13.8%	24.3%	3.35%	4.43%	16.6%	7.71%	25.9%	9.54%	8.32%	19.4%	11.6%
x4	14.6%	9.71%	17.0%	3.14%	3.48%	14.3%	6.87%	22.3%	7.78%	7.34%	19.2%	9.58%
x8	9.78%	6.30%	11.6%	2.94%	2.66%	12.7%	5.76%	19.4%	5.95%	6.60%	19.0%	7.73%
w/ Victim IP Cache x1	15.9%	17.6%	28.6%	2.71%	4.23%	11.2%	3.11%	23.2%	2.94%	1.16%	0.42%	5.37%
x2	10.9%	11.7%	23.1%	2.24%	2.44%	8.86%	1.75%	17.4%	1.49%	0.88%	0.29%	3.68%
x4	5.69%	6.07%	14.4%	1.89%	1.09%	7.34%	1.04%	13.4%	0.76%	0.79%	0.27%	2.42%
x8	2.79%	2.71%	6.66%	1.65%	0.51%	6.41%	0.78%	10.6%	0.53%	0.77%	0.27%	1.65%

TABLE V. ACHIEVABLE THROUGHPUT OF PPC, VICTIM CACHES, AND TCAM.

	Conventional	**w/ typical victim**	**w/ Victim IP Cache**
PPC	1,024 Gbps	1,024 Gbps	1,024 Gbps
Victim Cache	N/A	2,964 Gbps	2,964 Gbps
TCAM	**593 Gbps**	**886 Gbps**	**978 Gbps**

TABLE VI. ENERGY CONSUMPTION AND THE BREAKDOWN.

	Conventional	**w/ typical victim**	**w/ Victim IP**
TCAM (dynamic)	20.7 nJ (96.3%)	13.9 nJ (95.7%)	12.6 nJ (95.7%)
(static)	0.73 nJ (3.41%)	0.49 nJ (3.39%)	0.45 nJ (3.39%)
Caches (dynamic)	0.03 nJ (0.16%)	0.06 nJ (0.39%)	0.06 nJ (0.43%)
(static)	0.02 nJ (0.07%)	0.08 nJ (0.55%)	0.07 nJ (0.55%)
Total	**21.5 nJ**	**14.5 nJ**	**13.1 nJ**

D. Combined Approach

Finally, the combined approach of Port-aware Cache and Victim IP Cache was evaluated. Table VII shows the estimation of the average PPC miss rate, throughput, and energy consumption. The combined approach of Port-aware Cache and Victim IP Cache can achieve 1.72x higher throughput of the table lookups with 41.4% smaller energy per packet. We showed the combined approach reaches 1Tbps with remarkable small energy consumption compared to conventional PPC.

VII. CONCLUSION

PPC has been proposed to realize high-throughput and low-energy table lookup in routers. PPC shows significant impact on the table lookup throughput and the energy consumption; however, there is a room for improvement. To further improve PPC performance, this paper proposed two novel cache architecture, called Port-aware Cache and Victim IP Cache.

Port-aware Cache divides the cache space into three areas: DNS, HTTP, and other-application areas. it can isolate the influence of flows, such as attack flows, and improve the PPC miss rate by assigning appropriate number of PPC entries to each area. The simulation result indicated that Semi-static Port-aware Cache can decide the best mix of each entry sizes by prior trials at a boot process of a router and improve the PPC miss rate by 9.02% and the number of PPC misses in attacks by 8.64%. For further improvement, it is required to reduce the number of PPC misses caused by applications without HTTP and DNS.

Victim IP Cache supports PPC by caching the routing table and ARP table lookup results of packets which miss in PPC. Because IP Cache has a possibility to achieve higher cache hit rate than PPC, it may save a large number of PPC miss packets. The simulation results showed that 64KB Victim IP Cache can further improve the cache miss rate by 85.5%. As a result, the energy consumption of the table lookup can be reduced by 39.1% compared to conventional PPC.

Finally, the combined approach of Port-aware Cache and Victim IP Cache was considered. The simulation results also showed that both the highest throughput and lowest energy consumption can be achieved by the combined approach. It showed 1.72x higher table-lookup throughput with 41.4% smaller energy per packet in comparison to conventional PPC.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number JP18K18022.

REFERENCES

[1] The Ministry: Tabulation and Estimation of Internet Traffic in Japan, 2016, http://www.soumu.go.jp/main_content/000462459.pdf

[2] METI: Green IT Initiative in Japan, http://www.meti.go.jp/english/policy /GreenITInitiativeInJapan.pdf.

[3] Fan, J., Hu, C., He. K., Jiang, J., and Liuy, B.: Reducing power of traffic manager in routers via dynamic on/off-chip scheduling, 2012 Proc. IEEE INFOCOM, pp.1925-1933, Orlando, FL (2012).

[4] Zheng, X. and Wang, X.: Comparative study of power consumption of a NetFPGA-based forwarding node in publish-subscribe Internet routing, Computer Communications, vol. 44, pp.36-43 (2014).

[5] Gamage, S. and Pasqual, A.: High performance parallel packet classification architecture with PoPular Rule Caching, 18th IEEE International Conference on Networks (ICON), pp.52-57, Singapore (2012).

[6] Talbot, B., Sherwood, T., and Lin, B.: IP caching for terabit speed routers, IEEE Global Telecommunications Conference (GLOBECOM'99), vol.2, pp.1565-1569, Brazil (1999).

[7] Guinde, N. B., Rojas-Cessa, R., and Ziavras, S. G.: Packet classification using rule caching, 2013 Fourth International Conference on Information, Intelligence, Systems and Applications (IISA 2013), pp1-6, Piraeus (2013).

[8] Agrawal, B. and Sherwood, T.: Ternary CAM Power and Delay Model: Extensions and Uses, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.16, no.5, pp.554-564 (2008).

[9] Nawa, M. et al.: Energy-efficient high-speed search engine using a multi-dimensional TCAM architecture with parallel pipelined subdivided structure, 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), pp.309-314, Las Vegas, NV (2016).

[10] Hewlett-Packard Development Company.: Energy Efficient Networking - Business white paper, http://h17007.www1.hp.com/docs/mark/4AA3-3866ENW.pdf.

[11] Okuno, M. and Nishi, H.: Network-Processor Acceleration-Architecture Using Header-Learning Cache and Cache-Miss Handler, The 8th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2004), pp.108-113 (2004).

TABLE VII. SUMMARY OF PPC MISS RATE, THROUGHPUT, AND ENERGY CONSUMPTION.

	Conventional	**w/ Port-aware Cache**	**w/ Victim IP Cache**	**w/ combination**
Miss rate (PPC) [%]	17.3%	16.4%	17.3%	16.4%
(victim) [%]	N/A	N/A	3.68%	3.65%
Throughput [Gbps]	592 Gbps	624 Gbps	978 Gbps	1,021 Gbps
Energy [nJ]	21.5 nJ	20.4 nJ	13.1 nJ	12.6 nJ

[12] Yamaki, H. and Nishi, H.: An Improved Cache Mechanism for a Cache-based Network Processor, In Proceedings of the International Conference on Paralle and Distributed Processing Techniques and Applications (PDPTA'12), pp.1-7, Las Vegas, NV (2012).

[13] Yamaki, H.: Efficient Cache Architecture for Packet Processing in Internet Routers, In Proceedings of the 2020 Future of Information and Communication Conference (FICC 2020), Vol.1, pp.338-352, CA, USA (2020).

[14] Yamaki, H. and Nishi, H.: Line Replacement Algorithm for L1-scale Packet Processing Cache, In Adjunct Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing Networking and Services (MOBIQUITOUS 2016), pp.12-17, Hiroshima, Japan (2016).

[15] Yamaki, H., Nishi, H., Miwa, S., and Honda, H.: Data Prediction for Response Flows in Packet Processing Cache, 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC), pp.1-6, San Francisco, CA (2018).

[16] Chang, F., Feng, W. C., and Li, K.: Efficient Packet Classification with Digest Caches, In Proceedings of Third Workshop Network Processors and Applications (NP-3), 2005.

[17] Ata, S., Murata, M., and Miyahara, H.: Efficient cache structures of IP routers to provide policy-based services, IEEE International Conference on Communications (ICC 2001), vol.5, pp. 1561-1565, Helsinki (2001).

[18] Liao, G., Yu, H., and Bhuyan, L.: A new IP lookup cache for high performance IP routers, In Proceedings of the 47th Design Automation Conference (DAC), pp.338-343, Anaheim, CA (2010).

[19] Carter, J. L., and Wegman, M. N.: Universal classes of hash functions (Extended Abstract), In Proceedings of the ninth annual ACM symposium on Theory of computing (STOC '77), pp.106-112, New York, NY (1977).

[20] Kim C., Caesar M., Gerbear A., and Rexford J.: Revisiting Route Caching: The World Should Be Flat, In Proceedings of the 10th International Conf. on Passive and Active Network Measurement (PAM '09), pp.3-12 (2009).

[21] Muralimanohar, N. et al.: ptimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0, In Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 40), pp.3-14, Chicago, USA (2007).

[22] RIPE Network Coordination Centre: Réseaux IP Européens Network Coordination Centre RIPE NCC, http://www.ripe.net/.

[23] WIDE MAWI WorkingGroup: MAWI Working Group Traffic Archive, http://mawi.wide.ad.jp/mawi/.