IEICE TRANSACTIONS

on Information and Systems

DOI:10.1587/transinf.2024NTI0001

Publicized:2024/11/21

This article has been accepted and published on J-STAGE in advance of copyediting. Content is final as presented.

A PUBLICATION OF THE INFORMATION AND SYSTEMS SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

INVITED PAPER

Leveraging Heterogeneous Programmable Data Planes for Security and Privacy of Cellular Networks, 5G & Beyond

Toru HASEGAWA[†], ***Fellow***, **Yuki KOIZUMI**[††], **Junji TAKEMASA**[††], **Jun KURIHARA**[†††], ***Members***, **Toshiaki TANAKA**[†††], ***Senior Member***, **Timothy WOOD**[††††], ***and*** **K. K. RAMAKRISHNAN**[†††††], ***Nonmembers***

SUMMARY Cellular networks have become a critical part of our networking infrastructure, enabling ubiquitous communication. However, they are likely to be under threat, and can also be the vehicle through which cellular-connected end-systems can be subject to attacks. This paper introduces our efforts to leverage data plane devices such as programmable network interface cards, switches, and end-hosts to efficiently detect attacks and ensure user privacy at terabit per second speeds. Specifically, our project designs a heterogeneous data plane framework that cohesively combines multiple data plane devices, and designs two security solutions on the framework: security monitoring and privacy protection. This paper briefly introduces the goals and initial results for the two solutions.

key words: *5G, Security, Privacy, Programmable Data Plane*

1. Introduction

As users increasingly use cellular networks for all their communication needs, the traffic carried by cellular networks is growing substantially. This leads to a growing need to provide strong security for mission-critical 5G and beyond cellular networks. Attacks on mobile user devices as well as the cellular infrastructure have already been demonstrated, and the shift to a disaggregated, software-based 5G core may result in an increased attack surface and greater vulnerability. Our goal is to protect the cellular users and the infrastructure by leveraging data plane devices such as programmable network interface cards, switches, and end-hosts to efficiently detect attacks and ensure user privacy at terabit per second speeds.

Existing work on data plane based security monitoring has focused on detecting volumetric attacks such as denial of service through flooding of traffic. While such attacks can be extremely damaging, they are also relatively easy to detect with fingerprinting or sketch-based techniques that can operate at high speeds on programmable switches. In contrast, slow attacks, which attempt to avoid detection, can still be equally or more damaging. They cannot be easily detected, and generally require the memory and computation capacity of end-host security middleboxes to detect or prevent them. Likewise, attackers seeking to violate client privacy by eavesdropping on communication or fingerprinting user traffic patterns cannot be easily prevented, especially at large scale. These threats leave both cellular users and operators vulnerable to attacks. Nonetheless, the flexibility of a software-based 5G core along with the use of new programmable data plane technologies may indeed offer a solution.

We launched a research project to provide strong security monitoring and privacy protection solutions that exploit the high speed of programmable switches, the increased capability and capacity of programmable network cards, and the memory/computational capacity of end-host servers. The project overview is illustrated in Figure 1.

By leveraging the strengths of each of these data plane components, we can develop an efficient and performant 5G security solution. To achieve this goal, we identify four sub-goals. The first sub-goal is to design a heterogeneous data plane framework that cohesively combines multiple data plane devices for network function processing. The second and third sub-goals are set for security and privacy solutions on the heterogeneous data plane. Specifically, the second sub-goal is set to use this heterogeneous data plane framework to develop real-time monitoring of cellular traffic, being situated at the key vantage point of the 5G cellular core. The third sub-goal is set to use the framework to attack prevention and privacy protection. The fourth sub-goal is to integrate all the above results into a holistic system used for evaluating the proposed techniques both in terms of security and performance.

The rest of paper focuses on the initial efforts and results for the first and third sub-goals. Before presenting these efforts and results, we provide some background knowledge of programmable data planes in Section 2. Focusing on the first sub-goal, Section 3 designs a heterogeneous data plane framework for security monitoring (the second sub-goal) to explain how such a framework is used for the monitoring. Focus-

[†]The author is with Faculty of Materials for Energy, Shimane University, Japan.

[††]The authors are with Graduate School of Information Science and Technology, Osaka University, Japan.

[†††]The authors are with Graduate School of Applied Informatics, University of Hyogo, Japan.

[††††]The author is with George Washington University, USA.

[†††††]The author is with University of California at Riverside, USA.

Copyright © 200x The Institute of Electronics, Information and Communication Engineers

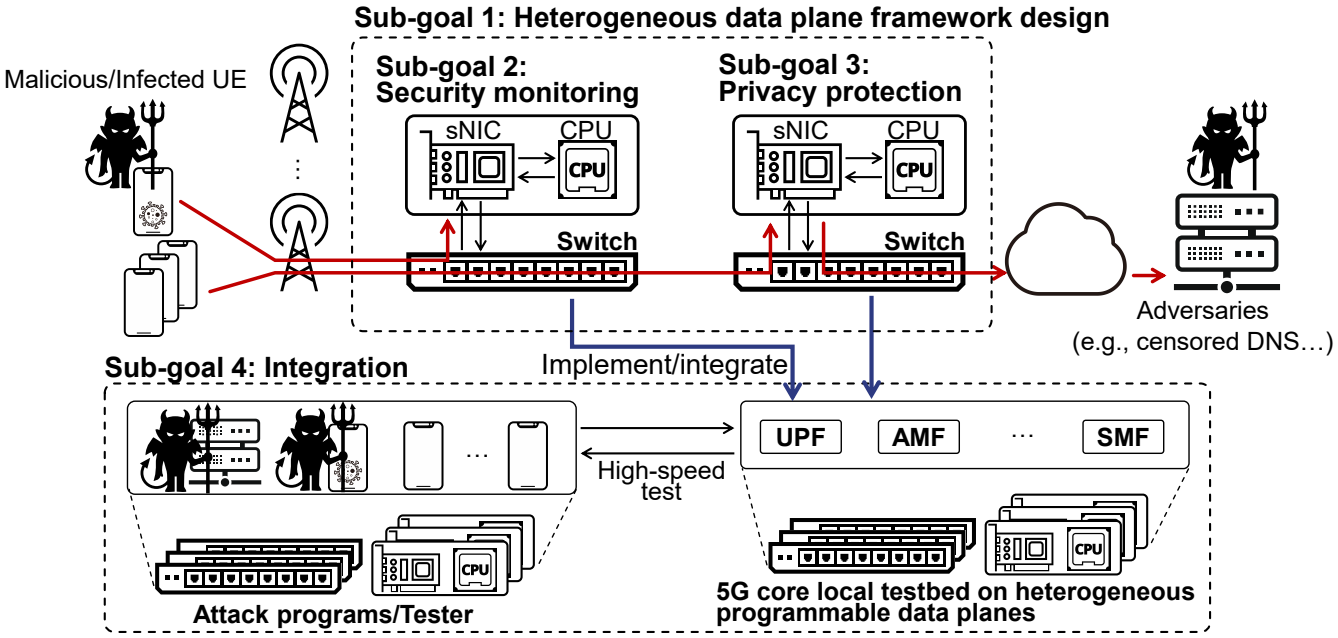

Fig. 1: Project overview: Heterogeneous programmable data planes for security and privacy of cellular networks

ing on the third sub-goal, Section 4 designs protocols for protecting privacy at the network and DNS levels. We review the related works to discuss their differences from our project in terms of security monitoring and privacy protection in Section 5. Finally, we summarize the future direction in Section 6.

2. Programmable Data Plane

Programmable devices, such as P4-capable switches (P4switch), SmartNICs (sNICs), and hosts have varying capabilities. Programmable switches are capable of much higher data plane throughput but programmable sNICs have more computing capability and memory capacity. Programmable switches are increasingly being suggested for monitoring, using telemetry queries at terabit scale [1], [2]. At the same time, sNICs also increasingly support programmability beyond simple protocol offloading. Switches and sNICs are both achieving programmability and with a malleable data path driven by the popularity of the P4 programming language. sNICs enable end-hosts to scale to fairly high-speed, 40–100 Gbps, rates [3], [4], although not quite to the terabit scale. Since sNICs support more operations [5] than switches, it is very beneficial to use them for stateful packet processing, to complement the coarse-grained query processing of programmable switches.

2.1 Programable Switch

A programmable switch, i.e., P4switch, has a pipelined packet processor with Reconfigurable Match Tables (RMT) [6] architecture. In this paper, we assume a Tofino 2 ASIC [7] as a P4switch. It has several pipes, each of which implements an *ingress* and *egress pipeline* functionality. Ingress pipelines modify packet headers and decide output ports, and egress pipelines can additionally modify the headers before sending them out from the switch. The pipeline consists of a *parser*, *deparser*, and a cascade of *pipeline stages* that lies between them. Each stage is equipped with SRAM and TCAM units and ALUs that adopt Very-Large Instruction Word (VLIW) to naturally provide instruction-level parallelism, thereby applying single *match-action* operation on packets.

In exchange for high-speed packet processing, the programmable switch imposes on programmers hardware requirements in terms of memory and computation. Firstly, the size of SRAM and TCAM units available in an entire switch is limited to tens of megabytes and megabytes, respectively. Secondly, the computational capacity of a switch is bounded by its number of pipeline stages, 20 stages in Tofino 2 [7].

2.2 SmartNIC

A sNIC is a catch-all term that encompasses FPGA-based devices [8], [9] (highest performance, but most difficult to (re)program), NICs with specialized compute engines [10], [11] (e.g., Netronome[12] cards that provide a large number of weaker, specialized cores, and allow a mix of P4 and micro-C programs), and NICs with general purpose compute facilities [13], [14] (e.g., Nvidia/Mellanox Bluefield [15] cards that incorporate an ARM processor and allow arbitrary processes to run on the card). Hosts running software middleboxes can also have heterogeneous hardware (GPUs, a mix of low/high power CPUs, etc), and diverse pro-

gramming models (e.g., eBPF functions that interpose on the kernel data plane or DPDK-based applications that bypass the kernel entirely) [16]–[21]. Each of these options exposes different trade-offs between throughput, latency, cost, energy consumption, memory capacity, and ease of programmability.

3. Heterogeneous Programmable Data Plane Framework for Security Monitoring

3.1 Goal and Motivation

As the first step, we address a heterogeneous programmable data plane framework, focusing on security monitoring. This is partly because there are studies monitoring on a framework consisting of either a combination of a P4Switch and a sNIC and that of a sNIC and a host. we seek to build a monitoring framework to detect a wide range of anomalies, caused both by volumetric attacks as well as specific slower targeted attacks that are increasingly successful [22] yet difficult to detect when in the middle of a high traffic volume. We leverage the capabilities of a programmable data plane for securing and managing such high-speed networks.

Security threats in the Internet are growing more and more diverse, and hence utilizing heterogeneous programmable data plane devices is a promising approach to efficiently detecting and mitigating a range of threats. However, a simple combination of such devices may not achieve the expected performance. Although using heterogeneous devices mutually compensate for weak aspects of each other, it incurs additional difficulties in terms of programmability, deployment, and runtime management.

3.2 Cooperative Monitoring Architecture

We proposed a cooperative monitoring framework that encompasses sNIC, host, and P4Switch, as shown in Fig. 2. Since in-depth analysis cannot be performed for all traffic at terabit speeds, the proposed framework obtains attack indicators at a coarse-grained flow-subset granularity. This filtering allows us to separate potentially malicious flows with only minimal impact on the data path of benign flows. Filtering is implemented using a P4Switch, where the focus is on processing high volumes of traffic. We then steer traffic subsets for finer grained analysis on a sNIC-and-host subsystem, where processor and memory intensive operations are performed on the packet flow.

The proposed framework is based on our initial design [10], which mainly focus only on a sNIC-and-host subsystem. The initial design [10] has a novel in-memory data structure on the sNIC with a malleable flow eviction policy and performs effective aggregation of the state information so that the frequency of updates to the host (using DMA) are reduced significantly.

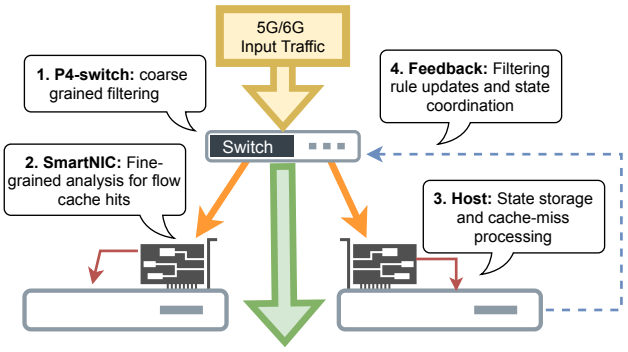

Fig. 2: Architectural overview of cooperating monitoring framework

We extend this system to a cooperative monitoring framework consisting of P4switches, sNICs, and, hosts. To manage the entire system, we design a cooperative control loop (Fig. 2) between the sNIC-host subsystem and the P4Switch to direct an appropriate amount of traffic from the P4Switch to the sNIC-host subsystem. This control loop configures the switch to tune what flow subsets should be forwarded to the sNIC-and-host subsystem. In the sNIC-and-host subsystem, the sNIC acts as an accelerator that enables significantly faster tracking of flow states compared to performing these functions on a host. On the other hand, the host has a much larger memory reservoir, which is needed for some deeper forms of analysis.

The primary challenge in realizing the cooperative monitoring framework is to optimize the deployment of analysis components across the heterogeneous monitoring devices, i.e., P4Switches, sNICs, and hosts, while taking into account different constraints inherent to these heterogeneous devices. Our research explores how the deployment should be adjusted to balance trade-offs in performance, cost, and efficiency. Although simply sending all traffic to hosts for full inspection would provide the highest detection rates, it would incur high costs due to the larger server footprint. In contrast, performing all analysis on the switches may not be able to detect all attacks.

To achieve an ideal deployment of analysis components, we need to primarily consider constraints related to resources and computing capabilities. From a resource perspective, resources on P4switches are fixed in quantity, whereas server resources are likely to be more malleable since they can be multiplexed for a variety of software-based functions. From a computing capability standpoint, P4switches excel at handling high-volume traffic but only support limited types of operations, which are specialized for packet header processing. Conversely, sNICs and servers can handle less traffic but can perform more complex operations than P4switches. Similarly, the sNIC operations are also limited compared to the hosts, as they may lack support for recursive functions and floating-point operations in

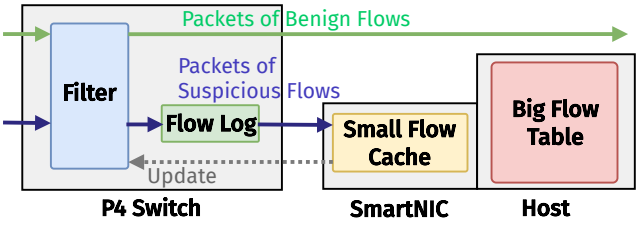

Fig. 3: Prototype implementation of cooperating monitoring

the packet processing pipeline [5].

3.3 Initial Results: Prototype Implementation

We implement a prototype of the cooperative monitoring framework and quantitatively validate its effectiveness of the concept [23]. Figure 3 illustrates the implementation and its packet processing flows. The prototype represents an example solution for the aforementioned challenges in layout of analysis components. Our prototype consists of a filter that preliminarily filters out packets of benign flows at the P4Switch, a small flow cache on sNIC, and a big (whole) flow table on the host.

In our implementation, packets first go through the filter, which detects whether the packets belong to a flow that has been marked as benign. If the packets are marked as benign, then no further analysis is needed and the packet can simply be sent out of the P4switch (green line in Figure 3).

If the packet does not hit in the filter, it is transferred to the host for the further analysis, which detects whether the packet is for a suspicious flow or not (blue line in Figure 3). Instead of immediately sending the packet to the host, its meta-data is stored in a Flow Log stored on the P4Switch. These logs are periodically flushed out to the sNIC/ host in a batch to reduce transferring cost between the P4switch and sNIC. The sNIC maintains a cache of currently active flows, but it does not have sufficient memory to track all flows over a long period of time. When the sNIC uses up its memory space, it evicts some flows to the host, which has significantly more memory available to store flows.

Once the sNIC or host have gathered sufficient information about a flow to identify the flow as benign, then it must update the filter stored on the P4Switch. After the update, the flow information no longer needs to be stored by either the sNIC or host, allowing memory in the sNIC and host to be reclaimed.

We preliminarily evaluate the effectiveness of our design through simulation. We implement the simulator that consists of the switch and host components according to Figure 3 and inject PCAP traces provided by MAWI project [24] to the simulator. We inject PCAP traces provided by MAWI project [24] to the simulator. We merge 750 traces taken from different days to amplify the traffic volume, resulting in a total packet rate of about 100 Mpps. We require the host to analyze the first 7 packets of flows belonging to common protocols like HTTP(s)/SSH, and the first 3 packets of flows on other protocols. The trace included a very large number of port scanning slow attack traffic (about 87% of TCP flows have only one packet, potentially indicating a port scan attempt). The details are explained by our paper [23].

4. Privacy Protection Framework

4.1 Goal and Motivation

Softwarization of 5G networks increases the risk that compromised nodes perform privacy attacks against flows encrypted using Transport Layer Security (TLS). In the 5G environment, user privacy may be leaked through communications in both the control and data planes. We address privacy attack prevention framework for mobile data services in 5G and beyond environments. A typical scenario would be to allow mobile cloud operators to protect their users from privacy attacks like fingerprinting and censorship attacks. The framework we develop will be integrated with User Plane Function (UPF nodes) to protect the 5G Core NFs from both security and privacy attacks.

We design two prevention methods against privacy attacks at the network and DNS levels, which reveal clients' IP addresses and URLs which they access. A primary challenge in these privacy protection methods is to achieve both high-speed and sophisticated packet handling simultaneously.

The objective of the privacy protection framework is to design a *terabit-scale* privacy protecting *middlebox* implemented by leveraging multiple devices in the heterogeneous data plane. Precisely, the framework provides *anonymity*. Our approach is beneficial because the transparent middlebox approach is general, and the service is applicable to various functions/servers running in the 5G environment without any change of 5G infrastructure, especially because Web and DNS communications, which contain rich private information, are key components of the 5G environment.

At present, we have designed and implemented the lightweight anonymity protocol [25] and the DNS anonymization method [26], respectively, for the objectives. The rest of section describes the details of the the lightweight anonymity protocol design.

4.2 Lightweight Anonymity Protocol

As the first step towards privacy at the network level, we address a lightweight anonymity protocol which enables it for providers to provide privacy service. An important feature it that the only headers including

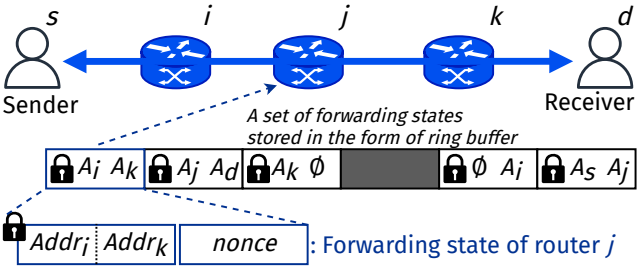

Fig. 4: The assumed system model and header format

source and destination IP addresses are encrypted (obfuscated). We design the lightweight anonymity protocol suitable for P4switches and implement the protocol on a P4switch. This implementation provides network layer anonymization at terabit per second speeds. The P4 code is publicly available at our Github repository†.

(1) System Model

Unlike onion routing like Tor, where the relay node encrypts a whole packet including both the header and the payload, the lightweight anonymity protocol [27] only encrypts a part of the header to achieve the good trade-off point between high throughput and anonymity.

Figure 4 shows the network model and the header format of the lightweight anonymity protocol. The anonymizing router (e.g., router j) establishes a route before forwarding a data packet, encrypts the forwarding state that contains IP addresses of the previous and the next anonymizing routers (e.g., routers i and k), with its own secret key and embeds the forwarding state into the packet header. When the anonymizing router forwards a data packet, it (e.g., router j) extracts its own forwarding state (e.g., state A_i, A_k for router j) from the packet header, and decrypts it to determine the forwarding destination. An attacker who eavesdrops on a packet cannot decrypt the forwarding state that is created by any router other than the anonymizing router that he controls, resulting in a user's anonymity.

(2) Design

We design an anonymizing router under the computational constraint of the P4switch. Related to the computational constraint, the P4switch employs a pipeline architecture for packet processing ASIC, and a packet can only use 24 pipeline stages when it passes through a pipeline once. Although the ASIC can perform complicated computations by making the packet passes through the pipeline multiple (e.g., N) times, the packet must re-enter the pipeline via the switch port, which resulting in throughput degradation to $1/N$ times. For example, given 3.2 Tbps P4Switch, we can allow only two pipeline passes to achieve more than terabit/s (e.g., 1.6 Tbps) throughput.

†https://github.com/Hasegawa-Laboratory/phi4p4

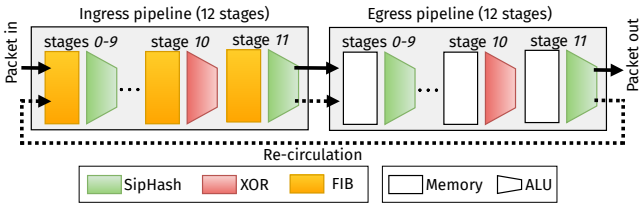

Fig. 5: Pipeline layout for Tofino ASIC of P4Switch

We carefully choose the cryptosystem and header structure to minimize pipeline stage consumption, as specifically described below:.

Cryptosystem: We choose the one-time pad cypher for encrypting and decrypting forwarding states in packet headers because its encryption and decryption are identical and repeat the same computation, which allows encryption and decryption to share the same pipeline stages. The one-time pad cipher encrypts a message M, as below: First, it computes a pad P, which is a random number generated by the pseudo-random function (PRF) with a nonce N and a key key, as $P = PRF(key, N)$. Second, it obtains the encrypted message E by performing XOR of M and P as $E = M \oplus P$. In the same way, the decrypted message is obtained by performing XOR of P and E. As a PRF, we choose SipHash, a keyed cryptographic hash function, similar to an existing study [28].

Header Structure of Forwarding States: We design a header structure that stores forwarding states so that it satisfies the requirements for the anonymity and minimizes pipeline resource consumption. We construct a ring buffer that stores forwarding states in the packet header, as shown Figure 4. The anonymizing router inserts and eliminates forwarding states by rotating them to obtain its own forwarding state. Such the rotating operation is implementable by using the parser and deparser unit of the P4Switch, which exists outside the pipeline, thereby consuming no pipeline resources.

(3) Prototype Implementation

We implement a prototype router with P4 language and Tofino's library. The resulting pipeline layout is illustrated in Figure 5.

We experimentally measure the packet forwarding rate of the prototype router. To realize terabit/s-scale measurement, we construct the testbed consisting of one host machine and one P4Switch, as shown in Figure 6. The host machine generates test traffic at 200 Gbps, the P4Switch amplifies it to 1.2 Tbps by using the packet replicator of the P4Switch. Then, the P4Switch performs anonymity protocol processing at this 1.2 Tbps traffic. This means that we can measure up to 1.2 Tbps throughput. As a measurement result, we obtain 1.16 Tbps throughput. This result clarifies the feasibility of Tbps-class anonymity proto-

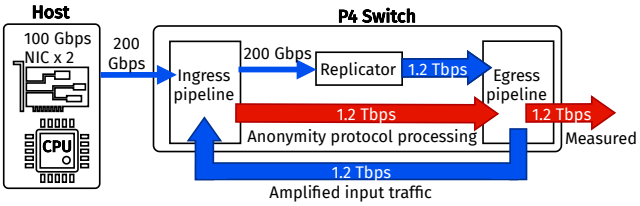

Fig. 6: Testbed for throughput measurement

col processing, which is further explained in details by our paper [25].

4.3 Annonymized DNS Protocol

Recall that most of user activities on the Internet begin from querying resolvers of Domain Name System (DNS) for domain names to communicate with hosts over the network. This implies that the *user privacy in DNS queries* would be a final piece remained to preserve the user privacy of network activities. From this observation, our project focused on the *DNS anonymization* to preserve the user privacy over the network in addition to the aforementioned network-level anonymity protocol.

Here we briefly describe our motivation and problem formulation. First recall that since the traditional DNS lacks fundamental security and privacy features in its design, several security and privacy enhancements of DNS had been actively investigated. Specifically, in the context of user privacy in DNS queries, several relay-based anonymization schemes have been recently introduced, e.g., Oblivious DNS [29], Oblivious DNS over HTTPS (ODoH) [30], [31], Anonymized DNSCrypt[32]. These schemes decouple the users' origin (IP addresses) and content of queries by simply encrypting queries in an end-to-end manner between users and resolvers, and employing dedicated intermediate relays between users and resolvers. However, there exists one significant drawback in such novel DNS anonymization schemes using relays: The lack of *collusion resistance*, i.e., the privacy could be completely corrupted when the relay colludes with the target resolver. Considering the current deployment of DNS, users do not have various choices of full-service resolvers enabling DNS message encryption and usually use ones operated by large and limited entities [33], e.g., Google, Cloudflare, etc. Much like encryption-enabled resolvers, relays would be operated by such limited big players as mentioned in [34, Section 7.1]. Indeed, as far as we know, just a few big CDN providers provide such relays in Apple iCloud Private Relay, a deployment of ODoH. Hence it may increase concerns about collusion and surveillance.

(1) Architectural Concept

From the above observation, we aim to solve issues related to collusion in existing anonymization schemes and to present a practical solution that ensures user anonymity. To this end, we have introduced a new architectural concept, termed μ*ODNS* (*Mutualized Oblivious DNS*) [26] for achieving user anonymity in DNS queries. μODNS has been designed to preserve the user's anonymity in the context of DNS even when facing untrusted resolvers and potentially colluding network nodes. Additionally, μODNS aims to minimize the performance degradation resulting from privacy enhancements. Architecturally, μODNS can be seen as an extension of existing relay-based schemes, enabling multiple relays and route randomization. In this sense, it adopts an approach akin to Tor [35], but with a highly specialized and streamlined focus on DNS.

(2) Protocol Design

As motivated to widely deploy our technology on the Internet, we have also presented not only the concept of μODNS but also the comprehensive exposition of the design and implementation of μODNS as *Proof of Concept* (PoC). The PoC protocol of μODNS was given by extending existing DNS anonymization schemes, specifically ADNSCrypt [32] and ODoH [30], [31], [34], to effectively realize the concept of μODNS while maintaining compatibility with the original schemes. We have also conducted a performance evaluation of μODNS using these PoC implementation deployed on the Internet. The results of this evaluation have demonstrated that μODNS can minimize the performance degradation caused by its privacy enhancements, achieving performance levels comparable to the underlying original protocol.

(3) Current Status

Currently the implementation of μODNS is publicly available on the Internet as open-source software (OSS) and public services[†]. Since there still remains several tasks to deploy our novel concept to the Internet as a *feasible* service attaining *light-speed* latency, the μODNS OSS project repositories are being actively maintained and extended to enable several new features considering the performance and security enhancements for deployment and operation.

5. Related Work

This section summarizes existing studies that leverage programmable data planes to build security monitoring and privacy protection solutions.

Recently, there has been substantial research on the benefits of emerging programmable data plane technologies [1], [2], [36] to provide strong and high-speed security monitoring. However, there has been relatively little work on effectively combining different data plane components. Programmable switches have been used

[†]See https://junkurihara.github.io/dns for the detailed information of the OSS and public service.

for telemetry queries at terabit scale [1], [2]. They have been shown to identify attack indicators and can trigger alerts quite well for heavy hitters and other volumetric attacks. But the P4switch's hardware constraints pose a major challenge non-volumetric, stateful detection, where protocol-level inspection is required. Detecting a covert timing channel, for example, requires a time-series analysis, whereas detecting stateful attacks, such as SSH brute-forcing and port scanning attacks, requires a sophisticated classification task and a large memory space to maintain the status of a large number of flows. Since it is difficult to perform such tasks and maintain the status of a large number of flows in switches, several existing approaches, for example NetWarden [37] and FlowLens [38], exploit the control plane of switches for performing complex tasks, such as ML-based classification.

There has been diverse contributions that aim to provide privacy protection by designing anonymity protocols. Traditionally, sender anonymity provided by onion routing, such as *Tor* has been a popular way to prevent attackers from identifying the source or destination of a packet. However, current approaches are not suitable to the 5G environment because it is not trivial to place multiple anonymity routers at different operators' networks. At least three Tor routers are required to reduce the probability that adversaries compromise them to impersonate the sender. Thus we adopt *relationship anonymity*, provided by network level onion routing like PHI [27] and achieve it by leveraging *IP address obfuscation*. Here, a node between trusted and untrusted networks obfuscates (encrypts) IP addresses by leveraging the large IPv6 address spaces [39]. There has been also some studies that address anonymized DNS protocols, which are summarized in Section 4.3.

6. Future Direction and Conclusion

This paper introduces our efforts to design a heterogeneous programmable data plane framework and to apply the framework to security monitoring and privacy prevention. The preliminary implementation of the protypes clarify the possibility that such a framework is useful to them at terabit second speeds. [10], [23], [25], [26], [40]

The future directions for the four sub-goals are summarized below: First, the heterogeneous program data plane framework is strengthened for privacy protection as well as security monitoring. A key extension is implementation of cryptosystems on sNICs as well as P4switches [40]. Besides, we seek an optimal computation allocation method to a P4Switch, sNICs and a host, whereas the cooperative security monitoring framework heuristically allocates computations on the above devices. Second, the cooperative security monitoring framework is applied to cellular networks like 5G networks to verify its resilience against attacks in such networks. Third, the lightweight anonymity protocol is extended so that payloads as well as headers are protected against strong adversaries. In addition, μODNS is improved for it to be actually deployed in the Internet. Resilience against DoS attacks is such an improvement. Finally, we integrate the above solutions to a UPF node by leveraging heterogeneous programmable data planes.

Acknowledgments

This work was supported in part by NSF CNS-2210379, CNS-2210380, and NICT Contract No. 22401.

References

[1] A. Gupta, R. Harrison, M. Canini, N. Feamster, J. Rexford, and W. Willinger, "Sonata: Query-driven streaming network telemetry," Proceedings of Anual Conference of the ACM Special Interest Group on Data Communication, SIGCOMM, pp.357–371, Aug. 2018.

[2] X. Chen, S. Landau-Feibish, M. Braverman, and J. Rexford, "BeauCoup: Answering many network traffic queries, one memory update at a time," Proceedings of Anual Conference of the ACM Special Interest Group on Data Communication, SIGCOMM, pp.226–239, Aug. 2020.

[3] J. Sonchack, A.J. Aviv, E. Keller, and J.M. Smith, "Turboflow: information rich flow record generation on commodity switches," Proceedings of European Conference on Computer Systems, EuroSys, pp.1–16, April 2018.

[4] Z. Zhao, H. Sadok, N. Atre, J.C. Hoe, V. Sekar, and J. Sherry, "Achieving 100Gbps intrusion prevention on a single server," Proceedings of USENIX Symposium on Operating Systems Design and Implementation, OSDI, pp.1083–1100, Nov. 2020.

[5] Netronome, "The joy of Micro-C." https://cdn.open-nfp.org/media/documents/the-joy-of-micro-c_fcjSfra.pdf, 2014.

[6] P. Bosshart, G. Gibb, H.S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN," ACM SIGCOMM Computer Communication Review, vol.43, no.4, pp.99–110, 2013.

[7] "Intel® Tofino™ 2." https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/tofino-2-series.html.

[8] J.W. Lockwood, N. McKeown, G. Watson, G. Gibb, P. Hartke, J. Naous, R. Raghuraman, and J. Luo, "NetFPGA–an open platform for gigabit-rate network switching and routing," Proceedings of IEEE International Conference on Microelectronic Systems Education, MSE, pp.160–161, June 2007.

[9] X. Zhang, X. Shao, G. Provelengios, N.K. Dumpala, L. Gao, and R. Tessier, "CoNFV: A heterogeneous platform for scalable network function virtualization," ACM Transactions on Reconfigurable Technology and Systems, vol.14, no.1, pp.1:1–1:29, Aug. 2020.

[10] S. Panda, Y. Feng, G. Kulkarni, Sameer, K.K. Ramakrishnan, N. Duffield, and L. Bhuyan, "SmartWatch: Accurate traffic analysis and flow-state tracking for intrusion prevention using SmartNICs," Proceedings of ACM International Conference on emerging Networking EXperiments and Technologies, CoNEXT, Dec. 2021.

[11] Z. Ni, G. Liu, D. Afanasev, T. Wood, and J. Hwang, "Ad-

vancing network function virtualization platforms with programmable NICs," Proceedings of IEEE International Symposium on Local and Metropolitan Area Networks, LANMAN, pp.1–6, July 2019.
[12] Netronome, "Netronome NFP-6000 Datasheet." https://www.netronome.com/media/documents/PB_NFP-6000-7-20.pdf, 2020.
[13] M. Liu, S. Peter, A. Krishnamurthy, and P.M. Phothilimthana, "E3: Energy-efficient microservices on SmartNIC-accelerated servers," Proceedings of USENIX Annual Technical Conference, ATC, pp.363–378, 2019.
[14] M. Liu, T. Cui, H. Schuh, A. Krishnamurthy, S. Peter, and K. Gupta, "Offloading distributed applications onto smartNICs using iPipe," Proceedings of Anual Conference of the ACM Special Interest Group on Data Communication, SIGCOMM, pp.318–333, Aug. 2019.
[15] NVIDIA, "NVIDIA Bluefield-3 DPU." https://www.nvidia.com/en-us/networking/products/data-processing-unit/, 2021.
[16] J. Hwang, K. Ramakrishnan, and T. Wood, "NetVM: High performance and flexible networking using virtualization on commodity platforms," Proceedings of USENIX Symposium on Networked System Design and Implementation, NSDI, April 2014.
[17] L. Rizzo, "netmap: A novel framework for fast packet I/O," Proceedings of USENIX Annual Technical Conference, ATC, pp.101–112, 2012.
[18] S. Palkar, C. Lan, S. Han, K. Jang, A. Panda, S. Ratnasamy, L. Rizzo, and S. Shenker, "E2: A framework for NFV applications," Proceedings of ACM Symposium on Operating Systems Principles, SOSP, pp.121–136, 2015.
[19] T. Zhang, "Design of NFV platforms: A survey," arXiv, vol.2002.11059 [cs], Feb. 2020.
[20] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici, "ClickOS and the art of network function virtualization," Proceedings of USENIX Symposium on Networked Systems Design and Implementation, NSDI, pp.459–473, April 2014.
[21] S. Miano, F. Risso, M.V. Bernal, M. Bertrone, and Y. Lu, "A framework for ebpf-based network functions in an era of microservices," IEEE Transactions on Network and Service Management, vol.18, no.1, pp.133–151, March 2021.
[22] "Imperva research labs reveals abnormal increase in DDoS attack length, despite popularity of short term attacks." https://www.globenewswire.com/news-release/2020/06/23/2052054/0/en/Imperva-Research-Labs-Reveals-Abnormal-Increase-in-DDoS-Attack-Length-Despite-Popularity-of-Short-Term-Attacks.html, June 2020.
[23] C. Wei, S. Tu, T. Hasegawa, Y. Koizumi, K.K. Ramakrishnan, J. Takemasa, and T. Wood, "A fast monitor for slow network attacks," Proceedings of IEEE Cloud Summit 2024 (poster), pp.1–4, June 2024.
[24] The MAWI (Measurement and Analysis on the WIDE Internet) Working Group, "Mawi working group traffic archive." http://mawi.nezu.wide.ad.jp/mawi/.
[25] Y. Yohinaka, J. Takemasa, Y. Koizumi, and T. Hasegawa, "Feasibility of network-layer anonymity protocols at terabit speeds using a programmable switch," Proceedings of IEEE International Conference on Network Softwarization(NetSoft), pp.1–5, June 2022.
[26] J. Kurihara, T. Tanaka, and T. Kubo, "μODNS: A distributed approach to DNS anonymization with collusion resistance," Computer Networks, vol.237, p.110078, Dec. 2023.
[27] C. Chen and A. Perig, "PHI: Path-hidden lightweight anonymity protocol at network layer," Proceedings of Privacy Enhancing Technologies Symposium, PETS, pp.100–117, Oct. 2017.
[28] S. Yoo and X. Chen, "Secure keyed hashing on programmable switches," Proceedings of ACM SIGCOMM Workshop on Secure Programmable network INfrastructure, pp.16–22, 2021.
[29] P. Schmitt, A. Edmundson, A. Mankin, and N. Feamster, "Oblivious DNS: Practical privacy for DNS queries," Proceedings of Privacy Enhancing Technologies Symposium, PETS, pp.17–19, 2019.
[30] S. Singanamalla, S. Chunhapanya, J. Hoyland, M. Vavruša, T. Verma, P. Wu, M. Fayed, K. Heimerl, N. Sullivan, and C. Wood, "Oblivious DNS over HTTPS (ODoH): A practical privacy enhancement to DNS," Proceedings on Privacy Enhancing Technologies, vol.4, pp.575–592, 2021.
[31] E. Kinnear, P. McManus, T. Pauly, T. Verma, and C.A. Wood, "Oblivious DNS over HTTPS." RFC9230, June 2022.
[32] DNSCrypt Project, "Anonymized DNS." https://github.com/DNSCrypt/dnscrypt-proxy/wiki/Anonymized-DNS, March 2021. Commit ID: 0b21176.
[33] C. Deccio and J. Davis, "DNS privacy in practice and preparation," Proceedings of ACM CoNEXT, Orlando, FL, USA, pp.138–143, 2019.
[34] S. Singanamalla, S. Chunhapanya, M. Vavruša, T. Verma, P. Wu, M. Fayed, K. Heimerl, N. Sullivan, and C.A. Wood, "Oblivious DNS over HTTPS (ODoH): A practical privacy enhancement to DNS," arXiv, vol.2011.10121 [cs], Nov. 2020.
[35] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second generation onion router," Proceedings of USENIX Security Symposium, Aug. 2004.
[36] D. Kim, Z. Liu, Y. Zhu, C. Kim, J. Lee, V. Sekar, and S. Seshan, "TEA: Enabling state-intensive network functions on programmable switches," Proceedings of Anual Conference of the ACM Special Interest Group on Data Communication, SIGCOMM, pp.90–106, Aug. 2020.
[37] J. Xing, Q. Kang, and A. Chen, "NetWarden: Mitigating network covert channels while preserving performance," Proceedings of USENIX Security Symposium, Aug. 2020.
[38] D. Barradas, N. Santos, L. Rodrigues, S. Signorello, F.M.V. Ramos, and A. Madeira, "FlowLens: Enabling efficient flow classification for ML-based network security applications," Proceedings of Network and Distributed System Security Symposium, NDSS, Feb. 2021.
[39] L. Wang, H. Kim, P. Mittal, and J. Rexford, "Programmable in-network obfuscation of DNS traffic," Proceedings of NDSS DNS Privacy Workshop, 2021.
[40] Y. Yohinaka, J. Takemasa, Y. Koizumi, and T. Hasegawa, "On implementing chacha on a programmable switch," Proceedings of 5th P4 Workshop in Europe 2022, pp.1–3, Dec. 2022.

Toru Hasegawa Toru Hasegawa received the B.E., the M.E. and Dr. Informatics degrees in information engineering from Kyoto University, Japan, in 1982, 1984 and 2000, respectively. After receiving the master degree, he worked as a research engineer at KDDI R&D labs.(former KDD R&D labs.) for 29 years, and then worked as a professor at

Graduate school of Information and Science, Osaka University for 11 years. He is currently a specially-appointed professor at Shimane University. His current interests are future Internet, Information Centric Networking, mobile computing and so on. He has published over 100 papers in peer-reviewed journals and international conference proceedings including MobiCom, ICNP, IEEE/ACM Transactions on Networking, Computer Networks. He has served on the program or organization committees of several networking conferences such as ICNP, P2P, ICN, CloudNet, ICC, Globecom etc, and as TPC co-chair of Testcom/Fates 2008, ICNP 2010, P2P 2011 and Global Internet Symposium 2014. He received the Meritorious Award on Radio of ARIB in2003, the best tutorial paper award in 2014 from IEICE and the best paper award in 2015 from IEICE. He is a fellow of IPSJ and IEICE.

Yuki Koizumi Yuki Koizumi is an associate professor at the Graduate School of Information Science and Technology, Osaka University, Japan. He received his master's degree and his doctoral degree in information science from Osaka University in 2006 and 2009, respectively. His research interests include the internet architecture, privacy, security, in-network computing, and programmable data plane.

Junji Takemasa Junji Takemasa is an assistant professor of Graduate school of Information and Science, Osaka University. He received the master's and Ph.D. degrees in information science from Osaka University, Japan, in 2016 and 2019, respectively. After receiving the Ph.D. degree, he worked as a research engineer at KDDI Research, Inc. for one and half years and moved to Osaka University. His research interests include programmable network, future internet and anonymous communication. He is a member of IEICE, IPSJ and IEEE.

Jun Kurihara Jun Kurihara received degrees of the B.E., M.E. and Ph.D. in Engineering from Tokyo Institute of Technology in 2004, 2006 and 2012, respectively. From 2006 to 2017, he was with KDDI Corp. and KDDI R&D Labs., Inc. as a strategic planner and a research engineer. He is currently an associate professor at Graduate School of Information Science, University of Hyogo and also a principal researcher at Zettant Inc. He was a visiting researcher at Palo Alto Research Center (PARC), CA, USA from 2013 to 2014, and Carnegie Mellon University, PA, USA in 2022. His research interests include coding theory, networking architecture and privacy in networking. He received the Best Paper Award from IEICE in 2014.

Toshiaki Tanaka Toshiaki Tanaka received the B.E. and M.E. degrees in Communication Engineering from Osaka University in 1984 and 1986 respectively, and the Ph.D. degree from Kyushu University in 2007. He had been with KDDI Corp. from 1986 to 2021, and was the Vice President of KDDI Research Inc. and the Fellow of KDDI Corp. since 2017. He is currently a professor at Graduate School of Information Science, University of Hyogo. His research interests include 5G security, network security, data trust and privacy. He received the MEXT Commendation for Science and Technology in 2014, the IEICE Achievement Award in 2015, and the TTC Information and Communication Technology Award in 2019.

Timothy Wood Timothy Wood is a Professor in the Department of Computer Science at George Washington University. Before joining GW, he received a doctoral degree in computer science from the University of Massachusetts Amherst in 2011 and a bachelor's degree in electrical and computer engineering from Rutgers University in 2005. His research studies how new virtualization technologies can provide application agnostic tools that improve performance, efficiency, and reliability in cloud computing data centers and software-based networks. His PhD thesis received the UMass CS Outstanding Dissertation Award and he has won four best paper awards, a Google Faculty Research Award, and an NSF Career award.

K. K. Ramakrishnan Dr. K. K. Ramakrishnan is a Distinguished Professor of Computer Science and Engineering at the University of California, Riverside. He joined AT&T Bell Labs in 1994 and was with AT&T Labs-Research from its inception in 1996, until 2013, as a Distinguished Member of Technical Staff. Before 1994, he was a Technical Director and Consulting Engineer in Networking at Digital Equipment Corporation. Between 2000 and 2002, he was at TeraOptic Networks, Inc., as Founder and Vice President.

Dr. Ramakrishnan is an ACM Fellow, an IEEE Fellow, and an AT&T Fellow, recognized for his fundamental contributions to communication networks, including his work on congestion control, traffic management, and VPN services. His work on the "DECbit" congestion avoidance protocol received the ACM Sigcomm Test of Time Paper Award in 2006, and he received the AT&T Technology Medal in 2012 for his work on Mobile Video Delivery. He recently received the 2024 ACM SIGCOMM Award recognizing his lifetime contribution to the field of communication networks. He has published over 300 papers and has 186 patents issued in his name. K. K. received his MTech from the Indian Institute of Science (1978), MS (1981), and Ph.D. (1983) in Computer Science from the University of Maryland, College Park, USA.