Privacy-Preserving Detection of Inter-Domain SDN Rules Overlaps

Privacy-Preserving Detection of Inter-Domain SDN Rules Overlaps

Arnaud Dethise
KAUST
Université catholique de Louvain

Marco Chiesa
Université catholique de Louvain

Marco Canini
KAUST

ABSTRACT

SDN approaches to inter-domain routing promise better traffic engineering, enhanced security, and higher automation. Yet, naïve deployment of SDN on the Internet is dangerous as the control-plane expressiveness of BGP is significantly more limited than the data-plane expressiveness of SDN, which allows fine-grained rules to deflect traffic from BGP's default routes. This mismatch may lead to incorrect forwarding behaviors such as forwarding loops and blackholes, ultimately hindering SDN deployment at the inter-domain level.

In this work, we make a first step towards verifying the correctness of inter-domain forwarding state with a focus on loop freedom while keeping private the SDN rules, as they comprise confidential routing information. To this end, we design a simple yet powerful primitive that allows two networks to verify whether their SDN rules overlap, i.e., the set of packets matched by these rules is non-empty, without leaking any information about the SDN rules. We propose an efficient implementation of this primitive by using recent advancements in Secure Multi-Party Computation and we then leverage it as the main building block for designing a system that detects Internet-wide forwarding loops among any set of SDN-enabled Internet eXchange Points.

CCS CONCEPTS

• **Security and privacy** → **Network security**; • **Networks** → **Routing protocols**;

KEYWORDS

Inter-domain routing, Privacy, Network Verification, SMPC

ACM Reference format:
Arnaud Dethise, Marco Chiesa, and Marco Canini. 2017. Privacy-Preserving Detection of Inter-Domain SDN Rules Overlaps. In *Proceedings of SIGCOMM Posters and Demos '17, Los Angeles, CA, USA, August 22–24, 2017,* 3 pages.
https://doi.org/10.1145/3123878.3131967

1 INTRODUCTION AND MOTIVATION

Adopting SDN in the inter-domain setting has recently been a goal of both academia and industry [3, 6]. These works aim at overcoming the limited routing expressiveness of BGP, the de-facto standard inter-domain routing protocol, which dictates how routing information is exchanged in the Internet. Although BGP prevents forwarding loops (by mandating that route advertisements carry the sequence of traversed networks), it constrains routing to be destination-based. The benefits of SDN over BGP include support for enhanced traffic-engineering, traffic redirection through middleboxes, security, and application-specific peering [6].

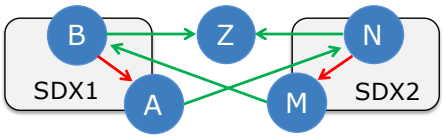

Figure 1: SDN-induced forwarding loop example. Arrows in red denote SDN deflection rules.

SDN deflections and the risk of forwarding loops. To overcome certain limits of BGP, SDN at the inter-domain level enables operators to install match-action forwarding rules that "deflect" traffic according to the operators' intents, e.g., sending HTTP traffic destined to the same destination through two different networks. This operation is dangerous in practice: BGP propagates a single path for each IP prefix whereas SDN may forward traffic destined to that prefix through multiple paths, which could lead to forwarding loops.

Forwarding anomaly example. Birkner et al. [1] investigated the problem of detecting inter-domain forwarding loops caused by deflections installed at different SDN-enabled Internet eXchange Points, also called SDXes. Consider the example in Figure 1. Two networks A and M route their traffic destined to Z through paths $(A\ N\ Z)$ and $(M\ B\ Z)$, respectively. Assume networks B and N decide to install SDN rules to steer their HTTP traffic destined to Z towards A and M, respectively. This operation results in a forwarding loop along $(B\ A\ N\ M)$ as both B and N deflect their traffic through a path where there exist already a deflection for HTTP traffic.

Verifying inter-domain forwarding properties: a trade-off between accuracy and privacy. Existing verification techniques (e.g. HSA [8]) enable operators to accurately detect forwarding anomalies such as forwarding loops or blackholes. However, these approaches were designed for the intra-domain setting and usually operate on a snapshot of the network state. In the inter-domain setting, one can hardly obtain such a snapshot: SDN deflection rules will likely be treated as business information that operators are unwilling to share [1, 3]. In this regard, Birkner et al. [1] strike a balance between the privacy of the forwarding state and the accuracy of the detected forwarding cycles. Namely, the match part of the

SIGCOMM Posters and Demos '17, August 22–24, 2017, Los Angeles, CA, USA

ACM ISBN 978-1-4503-5057-0/17/08.
https://doi.org/10.1145/3123878.3131967

SDN rules is kept private but false positives are possible, i.e., reporting forwarding loops in loop-free routing states.

A privacy-preserving primitive to the rescue! We advocate that operators should be able to verify the inter-domain forwarding state without sacrificing the accuracy of the verification nor their privacy, thus removing an obstacle to inter-domain deployment of SDN. By focusing on verification rather than configuration, unlike previous works using SMPC (e.g. [2, 7]), our assumptions about the actors become more practical. We take a first step and introduce a privacy-preserving primitive, called Distinct-Match, that allows any two networks, each one holding a set of SDN rules, to verify whether the match space of the two sets of rules overlap. We show that Distinct-Match can implemented with Secure Multi-Party Computation (SMPC), a tool that is renowned to be heavy in computation and communication, while still giving practical times for inter-domain configuration. Our implementation of Distinct-Match can be evaluated in the same order of magnitude as the network delay between the two SMPC parties (e.g., 400ms with 100ms delay), or at most an order higher for a large number of rules (e.g., 2,538ms with 100ms delay and 5,000 rules), while keeping the SDN rules private (§2). We finally show how this primitive can be leveraged to accurately detect forwarding loops among SDXes without leaking private information (§3).

2 THE DISTINCT-MATCH PRIMITIVE

We model an SDN rule as a match part, i.e., a sequence of n bits with value 0, 1, or 'x' (i.e., the wildcard bit), and an action part, e.g., forward a packet through a specific interface. Two SDN rules are *distinct*, i.e., do not overlap, if there exists at least one index $i = 1, \ldots, n$ such that the i'th bits of the two rules are not equal and both different from 'x'. As an example, (1x1) and (110) are distinct while (1x1) and (111) are not.

Distinct match evaluation in SMPC. We leverage SMPC to verify whether two SDN rules, each owned by a distinct party (network), are distinct without leaking any private information. Each party will exchange an XOR-encrypted version of its inputs and they will jointly evaluate the function. We model this as a boolean function with the ABY [4] framework, which provides two ways to evaluate it with different constraints and efficiency trade-offs: Yao's garbled circuits [9] and the GMW protocol [5] (see table 1).

2.1 Circuit design

The first stage of the circuit receives as inputs two SDN rules. An SDN rule r_i comprises a pattern p_i and a mask m_i. The j'th bit of p_i is equal to the j'th bit of r_i unless it is a wildcard, in which case, the written bit can be anything. The j'th bit of m_i is 1 if the j'th bit of r_i is different from 'x', 0 otherwise. To verify if two rules are distinct, we first compute bitwise $(p_1 \oplus p_2) \wedge (m_1 \wedge m_2)$ then verify if at least one bit is 1. For each pair of rules, this returns one bit of value 1 if the rules are distinct, 0 otherwise. This circuit can also be extended to return the next-hop identifier in case of overlap.

Overlap among many rules. In general, one network, called *client*, may want to verify whether installing an SDN rule would overlap with any of the SDN rules already installed in the forwarding plane by a different network, called *server*. We reinforce the privacy guarantees by hiding the rule-per-rule results, either through shuffling the outputs or reducing them to a single bit (as in the results below).

2.2 Evaluation

We evaluated our primitive with respect to different delays between the parties and different numbers of server's rules. Table 1 shows the average execution time over 50 executions of the Distinct-Match operation where each rule is 13 bytes long so as to encode source and destination IPv4 addresses, transport ports, and protocol. The client and server run on the same machine with 32 logical cores at 2.60 GHz with 128 GB of RAM. `tc-netem` is used to simulate the communication delay on the local interface. We distinguish between *setup* time, which is independent on the actual inputs and can be pre-computed, and *online* time, which instead depends on the actual inputs. The baseline is the round-trip time between networks plus 1ms of local, non-private computation.

		Number of rules				Baseline
Delay		1	50	500	5000	5000
1 ms	setup	2.86	8.66	38.6	281	-
	online	4.77	8.46	23.7	33.2	3
10 ms	setup	20.8	41.5	89	334	-
	online	40.8	59.3	89.8	264	21
100 ms	setup	201	401	655	2252	-
	online	401	599	839	2528	201

Table 1: Average computation time (in ms) for the comparisons. Blue (Green) cells use Yao (GMW).

The main performance bottleneck is the communication delay, due to the multiple rounds of messages that need to be exchanged. Yao's circuit provides faster computation for smaller problem size, while GMW becomes more efficient when comparing more than 2,000 rules as it scales logarithmically in the input size. We observed that Yao scales linearly in the size of the rules while GMW scales better: for 5,000 rules, each using 37 bytes to include IPv6 addresses, the computation time is 15% higher.

3 USE CASE: CYCLE DETECTION

To showcase the benefits of Distinct-Match, we devise a mechanism to detect forwarding cycles among SDXes. Compared to [1], our approach admits fewer false positives and provides a stronger privacy. We outsource the SMPC computation to the SDXes, which allows us to make extensive use of pre-computation, and can achieve even lower runtimes by moving the most costly parts of the circuit outside of the SMPC. This leads to a maximum speedup of a factor of ∼5 compared to Table 1 for 5,000 rules, with the SDXes learning whether some bits in the rules match or not but not learning the content of these rules, which is secretly shared by the members. Whenever an SDX member needs to install a new SDN deflection rule r, it verifies whether there exist networks downstream that deflect packets matching r, and will iteratively follow deflections towards the next hop. If a chain leads back to the initial SDX member, a forwarding loop is found. We built a proof-of-concept of the system and leave its complete evaluation as part of future work.

Acknowledgements

This research is (in part) supported by European Union's Horizon 2020 research and innovation programme under the ENDEAVOUR project (grant agreement 644960).

REFERENCES

[1] Rüdiger Birkner, Arpit Gupta, Nick Feamster, and Laurent Vanbever. 2017. SDX-Based Flexibility or Internet Correctness?: Pick Two!. In *SOSR*.
[2] Marco Chiesa, Daniel Demmler, Marco Canini, Michael Schapira, and Thomas Schneider. 2016. Towards Securing Internet eXchange Points Against Curious onlooKers. In *Applied Networking Research Workshop*.
[3] Marco Chiesa, Christoph Dietzel, Gianni Antichi, Marc Bruyere, Ignacio Castro, Mitch Gusat, Thomas King, Andrew W. Moore, Thanh Dang Nguyen, Philippe Owezarski, Steve Uhlig, and Marco Canini. 2016. Inter-Domain Networking Innovation on Steroids: Empowering IXPs withSDN Capabilities. *IEEE Communications Magazine* 54, 10 (Oct 2016), 102–108.
[4] Daniel Demmler, Thomas Schneider, and Michael Zohner. 2015. ABY – A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In *NDSS*.
[5] Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to play any mental game. In *STOC*.
[6] Arpit Gupta and et al. 2014. SDX: A Software Defined Internet eXchange. In *SIGCOMM*.
[7] Debayan Gupta, Aaron Segal, Aurojit Panda, Gil Segev, Michael Schapira, Joan Feigenbaum, Jenifer Rexford, and Scott Shenker. 2012. A New Approach to Interdomain Routing Based on Secure Multi-party Computation. In *HotNets*.
[8] Peyman Kazemian, George Varghese, and Nick McKeown. 2012. Header Space Analysis: Static Checking for Networks.. In *NSDI*.
[9] Andrew C Yao. 1982. Protocols for Secure Computations. In *FOCS*.