Supplementary Materials for "Flow Table Security in SDN: Adversarial Reconnaissance and Intelligent Attacks"

Mingli Yu, *Student Member, IEEE*, Tian Xie, *Student Member, IEEE*, Ting He, *Senior Member, IEEE*, Patrick McDaniel, *Fellow, IEEE* and Quinn K. Burke, *Student Member, IEEE*

I. Rule Timeouts

One limitation of our previous model of the flow table is that it ignores the possibility that rules may come with idle and/or hard timeouts set by the controller, which can also cause rules to be removed from the flow table in addition to evictions. Nevertheless, most of our results remain applicable in the presence of timeouts as explained below. We will use τ_I to denote the idle timeout and τ_H the hard timeout.

A. Size inference

For RCSE (Algorithm 1) to work properly under timeout, it suffices for the subroutine `forward-backward-probing` to sense hits for the last C probes in the forward pass (when $c \geq C$). As sending the last C probes, waiting for the corresponding rules to be installed, and testing their existence in the flow table takes $T_{2C} + d_I$ time, where T_c denotes the time to send c probes, the timeouts will have no impact if $\min(\tau_I, \tau_H) > T_{2C} + d_I$, which is usually satisfied in practice. Specifically, under a realistic setting (e.g., $\lambda_a = 100$ packets/ms, $C = 1000$, and $d_I = 0.9$ ms), $T_{2C} + d_I$ is only 20.9 milliseconds, but $\min(\tau_I, \tau_H)$ is at least several seconds (e.g., 60 seconds for the default Mininet controller).

B. Policy inference

For RCPD (Algorithm 2) to work correctly under timeout, it suffices for the subroutine `flush-promote-evict-test` to finish before timeout occurs, which is satisfied if $\min(\tau_I, \tau_H) > T_{C+3} + d_I$. As $C \gg 1$, this condition will be satisfied if the above condition for size inference is satisfied.

C. Traffic parameter inference

Our solution for traffic parameter inference (Section IV-A) is based on TTL approximation, where the existing formulas have not considered timeouts. Nevertheless, as FIFO is already approximated by a TTL-based eviction policy with a non-reset timer τ (the characteristic time) [1], the existing TTL approximation formula (8) can be easily extended to accommodate a hard timeout τ_H:

$$h_i^{\text{FIFO}} = \frac{\lambda_i \min(\tau, \tau_H)}{1 + \lambda_i (d_I + \min(\tau, \tau_H))}, \tag{1}$$

as the hard timeout effectively sets another non-reset timer. Similarly, as LRU is already approximated by a TTL-based eviction policy with a reset timer τ [1], the existing formula (10) can be extended to accommodate an idle timeout τ_I:

$$h_i^{\text{LRU}} = \frac{e^{\lambda_i \min(\tau, \tau_I)} - 1}{\lambda_i d_I + e^{\lambda_i \min(\tau, \tau_I)}}, \tag{2}$$

as the idle timeout effectively sets another reset timer. For FIFO with idle timeout, LRU with hard timeout, or either policy with both types of timeouts, the TTL approximation will be a hybrid TTL-based policy with both a non-reset timer and a reset timer, which has not been analyzed before. We leave these cases to future work.

The above analysis implies that our approach of inferring parameters of background traffic in Section IV-A3 remains valid in the presence of hard timeout under FIFO or idle timeout under LRU, as long as the probing rates are designed such that the characteristic time given by (12) or (14) is no larger than the externally-imposed timeout value.

D. DoS attack

Our results in Section IV-B2 remain valid in the presence of hard timeout under FIFO or idle timeout under LRU, as in these cases, the TTL approximation formulas Supp-(1) and Supp-(2) have the same form as before. Specifically, Theorem IV.1 still holds under FIFO with hard timeout τ_H for any meaningful attack target $\bar{h} < \sum_{i=1}^{F} \frac{\lambda_i}{\lambda} \cdot \frac{\lambda_i \tau_H}{1+\lambda_i(d_I+\tau_H)}$ (which is an upper bound on the average hit probability due to the hard timeout), as in this case (17) still holds, and hence (18) and the conclusion therein still hold. Similarly, Theorem IV.1 still holds under LRU with idle timeout τ_I for any $\bar{h} < \sum_{i=1}^{F} \frac{\lambda_i}{\lambda} \cdot \frac{e^{\lambda_i \tau_I} - 1}{\lambda_i d_I + e^{\lambda_i \tau_I}}$ (upper bound on the average hit probability due to the idle timeout). For the same reason, the attack design methods proposed in Section IV-B2 remain valid under any meaningful attack target $\bar{h}$. We leave attack design under other combinations of policy and timeout to future work.

The references refer to the sections and formulas in the main body by default. References to the sections and formulas in supplementary materials are prefixed by Supp.

TABLE I
EXPERIMENT RESULTS OF SIZE INFERENCE WITH TIMEOUTS

metric	FIFO aware	FIFO agnostic	LRU aware	LRU agnostic
error mean (%)	411.86	0.1	0.43	0.1
error std (%)	312.71	0.05	0.77	0.05
#probes mean	4920	$1.114e4$	$2.410e5$	$1.102e4$
#probes std	28.71	2.458	2082	1.756

TABLE II
EXPERIMENT RESULTS OF POLICY INFERENCE WITH TIMEOUTS

metric	FIFO	LRU
error probability	0.0987	0
#probes ('mean (std)')	2004 (488)	5030 (0)

E. Experiments

We validate the impact of timeouts by repeating the experiments in Section VI under the timeouts of $\tau_I = 60$s and $\tau_H = 600$s. Although the timeouts have different default values for different controllers, 60s of idle timeout is the default choice by Mininet [2] (no hard timeout), and we add a hard timeout to test the effect of both timeouts. However, the default policy in Open vSwitch is neither LRU nor FIFO in the presence of both timeouts. Therefore, we modify its code to implement the desired replacement policies, available at [3]. We repeat the experiments in Section VI, as shown in Tables Supp-I–II and Fig. Supp-1–3 (ignoring the idle timeout under FIFO and the hard timeout under LRU in computing the TTL approximation). The results are very similar to those in Section VI, implying that our solutions remain applicable under realistic timeout values.

II. POSSIBLE DEFENSES

In this section, we briefly discuss possible strategies to defend against the identified attacks.

A. Attack Prevention

As the proposed attacks are based on the two primitives in Section II-B that have been validated on current SDN implementations, one strategy to defend against the identified attacks is to modify the implementation to invalidate at least one of the primitives.

One strategy is to make all the hits and misses indistinguishable in terms of response time. However, such a strategy comes at a high performance cost, as it will slow down packets that result in hits (which are normally the majority of cases) to the level of packets that result in misses. In [4], an alternative strategy was proposed, where after a flow has not been active (i.e., generating packets) for a period of time T_{th}, the switch will delay new packets of this flow within a small window W to mimic the response times under table misses, even if these packets result in hits. From the attacker's perspective, adopting this strategy on top of an existing replacement policy is equivalent to adding an "idle timeout" of T_{th} and a "rule installation delay" of $d_I = W$ after the first miss following a timeout, as packets arriving during this delay will be detected as misses. As explained in Section Supp-I-A, this defense has no impact on our size/policy inference algorithms as long as $T_{th} > T_{2C} + d_I$. Furthermore, it will not affect our methods for traffic parameter inference or DoS attack under LRU as long as T_{th} is no smaller than the characteristic time.

To test the efficacy of this defense strategy, we repeat the experiments of RCSE and RCPD in Section VI while implementing the strategy in [4] with $W = 100$ ms (as recommended by [4]) and various values of T_{th}. The results, shown in Fig. Supp-4–5, demonstrate a tradeoff between defense efficacy and communication performance, measured by the percentage of legitimate packets resulting in hits that are delayed by the defense mechanism. To notably increase the error of size/policy inference, T_{th} has to as small as a few milliseconds, causing a significant portion of legitimate packets to be delayed. Fundamentally, timing attacks (including the proposed attacks) exploit the performance difference between packets handled within the data plane and those involving the control plane, and thus cannot be completely eliminated as long as such difference exists. Techniques designed to destroy the statistics used in a specific attack (e.g., [4]) may not be effective for other attacks. It remains open whether there exists a universal defense with tolerable performance penalty against all possible timing attacks in SDN.

B. Other Defenses

When DoS attacks are launched, the most effective defense is to quickly detect the attacks (e.g., [5]) and block the attack traffic. In addition, it is also desirable to have graceful performance degradation when the detection is not successful. Motivated by Observation 2 in Section V-C, our recent results [6] suggested that different policies degrade differently under a given attack strategy, wherein an adaptive policy selection scheme was proposed to select the most resilient policy for the perceived type of attack.

REFERENCES

[1] M. Dehghan, B. Jiang, A. Dabirmoghaddam, and D. Towsley, "On the analysis of caches with pending interest tables," in *ICN*, September 2015.

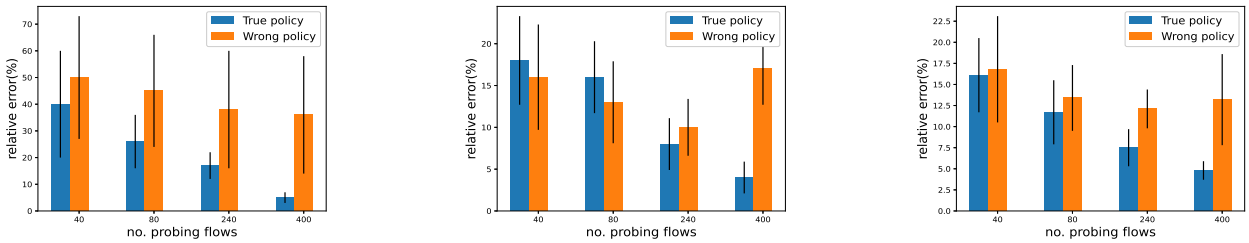
[2] "Openflow tutorial," https://homepages.dcc.ufmg.br/~mmvieira/cc/OpenFlow Tutorial - OpenFlow Wiki.htm.

[3] T. Xie, N. Nambiar, and T. He, "Configurable rule replacement policies in SDN: Implementation in Open vSwitch," https://github.com/SophieCXT/SDN-Implementation-in-Open-vSwitch, 2021.

[4] H. Cui, G. O. Karame, F. Klaedtke, and R. Bifulco, "On the fingerprinting of software-defined networks," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2160–2173, 2016.

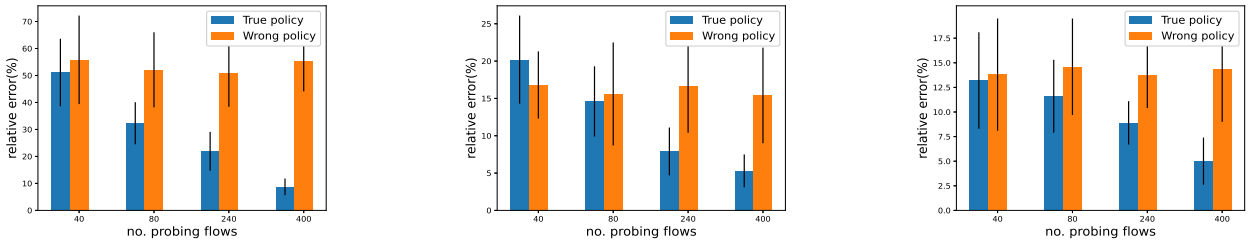
[5] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "SPHINX: detecting security attacks in Software-Defined Networks," in *NDSS*, 2015.

[6] T. Xie, T. He, P. McDaniel, and N. Nambiar, "Attack resilience of cache replacement policies," in *IEEE INFOCOM*, May 2021.



(a) estimate λ (b) estimate F (c) estimate α

Fig. 1. Joint parameter inference under FIFO in Mininet with timeouts



(a) estimate λ (b) estimate F (c) estimate α

Fig. 2. Joint parameter inference under LRU in Mininet with timeouts

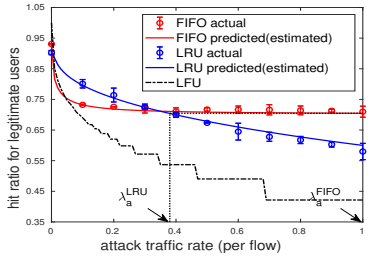
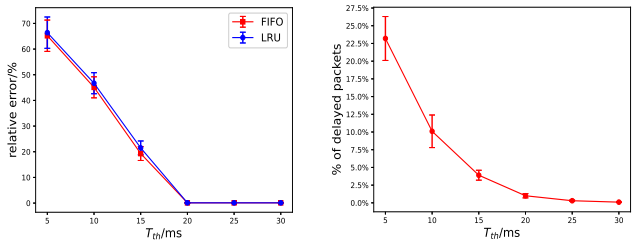
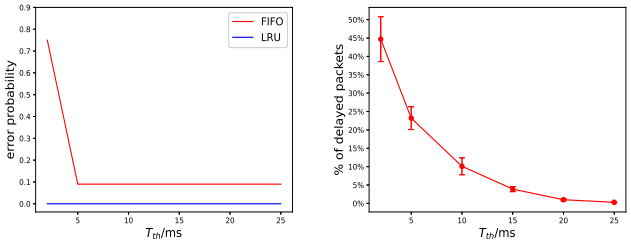


Fig. 3. DoS attack: predicted and actual hit ratios for background traffic in Mininet with timeouts



(a) relative error (b) delayed legitimate packets

Fig. 4. Defense: policy-agnostic size inference under varying T_{th} (n=10, $\lambda_a/\lambda = 50$)



(a) error probability (b) delayed legitimate packets

Fig. 5. Defense: size-aware policy inference under varying T_{th} ($N = 10$, $\lambda_a/\lambda = 500$)