Semester Thesis:
Securing data-plane driven fast-reroute systems

Stephan Keck

Supervisors:
Thomas Holterbach
Prof. Dr. L. Vanbever

January 28, 2019

Contents

1 Abstract

The emerging field of programmable switches, with its programming language p4 [1], enables researchers to build new data-plane driven systems. Those systems are constrained by line-rate processing and limited memory in the switch. Therefore they have to make fundamental trade-offs. As an example [2], does heavy-hitter detection entirely in the data plane despite those constraints. Blink [3] is another example of a data-plane driven system. It reroutes upon remote failures by observing TCP flows and their retransmissions. As such systems monitor and possibly react to traffic through the switches, there are possibilities for an attacker to circumvent or manipulate those systems.

This thesis shows how a detrimental attack on the current Blink system is feasible even for an attacker that sends a very small proportion of all the packets to a destination prefix. First we provide some background and describe how an attacker would be able to trigger rerouting on a switch running Blink. Then we provide an analytical model for such attacks and analyse the characteristics of TCP flows that get monitored by Blink. After the analysis of the TCP flows, a countermeasure is proposed and analysed. With the countermeasure Blink sustains 80 times stronger attacks than before, while still being able to reroute on the majority of remote failures.

2 Background

2.1 Internet Convergence on Failures

ISP networks are fast to converge on failures in the network, thanks to fast-convergence frameworks [4] that are based on fast failure detection and precomputed next hops. This helps ISPs to recover fast from internal failures. However, remote failures in today's internet are frequent and still slow to converge, taking about 30 seconds on average [5],[6]. SWIFT [6] predict such failures based on the BGP updates, but the first BGP update can take minutes to arrive. The goal of Blink is to reroute in the order of seconds after a remote failure.

2.2 Blink

Blink [3] is a data-plane driven system that reroutes upon remote failures by only observing TCP flows and their retransmissions. It can be run on a single switch which is programmable by P4. Blink leverages the fact that TCP flows exhibit predictable behavior upon disruption. Blink's pipeline has two parts to monitor TCP traffic and retransmission, Flow Selector and Sliding Window. Blink's Flow Selector mechanism randomly selects 64 flows for every prefix that get monitored and observes if they have any retransmissions. Inactive flows get evicted after two seconds and all Flow Selector cells get flushed every 512 seconds. Flows that send a FIN packet also get evicted. If one of the 64 cells of the Flow Selector needs to be filled with a new flow it is filled on a first-in-first-selected basis. Therefore, a flow is more likely to get selected if it sends more packets. The retransmissions of a flow get registered in a Sliding Window of 800 ms. If at least half of the monitored flows retransmit within this Sliding Window, Blink starts to reroute the monitored flows to predefined next-hops and observes if they reach the destination. The Flow Selector timeout of 512 seconds and the Flow Selector inactive flow timeout were chosen to give Blink a high rerouting performance and high efficiency of hardware resources and not to avoid attacks. An overview of those parameters is given in table 1.

Table 1: Parameters of the Blink implementation

Name	Value
Monitored flows	64
Rerouting threshold	32
Flow Selector reset timeout	512 seconds
Flow Selector inactive flow timeout	2 seconds
Sliding Window size	800 ms

2.3 Transmission Control Protocol

The Transmission Control Protocol (TCP) is based on the internet protocol and provides reliable host-to-host connectivity. To send data packets reliably, TCP uses retransmissions when a packet gets dropped. TCP retransmissions are send after the retransmission timeout (RTO) expires. The RTO is given by $RTO = sRTT + 4 * RTTVAR$ with sRTT being a smoothed round-trip time and RTTVAR the round-trip time variation. If a retransmission doesn't succeed, the hosts do a exponential backoff, doubling the RTO. Blink leverages the fact that those retransmissions for a given destination happen very quickly and almost simultaneously after a failure in the network.

3 Possible attacks on Blink

In this chapter we describe how Blink could be attacked with malicious traffic. First, we describe how an attacker could get his flows selected by Blink. Then, we show what attacks can be performed and what their effects are.

3.1 Getting selected by Blink

The Blink paper states that Blink might be vulnerable to an attacker sending fake retransmissions. For an attacker who sends similar traffic to the legitimate traffic, it is stated, that he would need 50% of the traffic to be able to trigger rerouting. It is expected that this is hard for prefixes with high traffic volume, which are monitored by Blink. Furthermore an attack with the majority of the traffic would be noticeable and could be mitigated at runtime. However, it is also mentioned that an attacker who knows the Blink mechanism could make an attack with much fewer resources, because Blink can't differentiate attacker from legitimate traffic, nor was the Flows Selector reset timeout designed to prevent attacks. This thesis will focus on attacks that try to leverage the way Blink monitors traffic with limited resources.

A Blink savvy attacker leverages the fact that legitimate flows get evicted often. Every time a legitimate flow gets evicted, he has a chance to get one of his flows monitored. He sends his flows at a constant rate, with an inter packet delay of less then two seconds, to not get evicted until the Flow Selector's 512 second reset timeout expires.

3.2 Possible attacks

We found three possible attacks on the Blink system. The possibilities depend on how many of the attacker flows are selected by the Flow Selector.

- Trigger rerouting:
  If an attacker manages to get at least 32 malicious flows monitored, he can trigger the fast rerouting by sending retransmissions on all of his flows at roughly the same time.

- Deteriorating Blink's performance:
  An attacker can deteriorate Blink's rerouting mechanism by not sending retransmissions even though his packets didn't arrive at the destination. If he controls more then half of the flows, Blink would not be able to reroute on a remote failure.

- Fake forwarding loops:
  In a third possibility, the attacker could always send the same packet over and over with a very low inter packet delay in order to simulate a forwarding loop. When Blink fast reroutes a forwarding loop would be detected and the next-hop would be avoided, making Blink less efficient

or even unable to reroute. This attack would be easier to run because only one flow needs to be selected in order to cancel a next-hop.

The first attack could be harmful to the entire network, since an attacker could cause unknown behavior in the system. Therefore, this thesis focuses on this attack. We show how likely such an attack can be and also introduce a countermeasure. In the latter possibilities, the worst case is that Blink is unable to reroute even though it uses hardware resources. These false negatives don't harm the network and rerouting can still be done based on BGP.

4 Attack Scenario

In the last chapter, we described how an attacker can get his flows monitored by Blink and what the attackers possibilities are based on the number of his flows that got selected. The attacker needs his flows to be at least in half of the cells of the Flow Selector in order to trigger the rerouting mechanism of Blink. In this chapter, we provide an analytical model that shows how many of his flows an attacker is expected to get selected by Blink and subsequently how likely the attacker is to trigger rerouting.

4.1 Analytical Model

The time budget (t_B) for an attacker to get his flows monitored by Blink is given by the time until Blink resets all entries in the Flow Selector after 512 seconds. Because an attacker could send TCP retransmissions at the same time as sending his TCP messages or he could try multiple times and always send retransmissions after some smaller time than t_B, it is assumed that the attacker has the whole time until a timeout occurs as his time budget t_B. Many variables have to be assumed in order to make calculations for the expected number of malicious flows that are in the Flow Selector after any given time up to t_B.

In a first step only the overall packet rates of legitimate and attackers traffic p_l and p_m are taken into consideration, because it provides the probabilities to get chosen by the Flow Selector. This means that both, the number of actual flows and their corresponding packet rates, get simplified into this variable. The selection is done following a first-in first-selected basis, the probability that a malicious flow is selected is $q_m = p_m/(p_m + p_l)$. Similarly the probability for a legitimate flow is $q_l = p_l/(p_m + p_l)$. In order to maintain a legitimate flow in a cell, a new legitimate flow has to replace it after every eviction. As explained in section 4.3 an attacker would never stop his flows in order to maintain the selected ones in the Flow Selector. A legitimate flow gets evicted after a certain time, this is called the eviction time t_E. This time depends on the activity and length of the legitimate flow.

With this assumptions, the probability P for an attacker to get a flow in one specific cell during the time budget is:

$$P(1) = (1 - q_l)^{t_B/t_E}$$

Because each cell has its own associated flows, their allocation can be seen as independent trials (Bernoulli trial). The probability of getting n of the 64 flows affected is calculated by

$$P(n) = nCk * q_l^k * (1 - q_l)^{(n-k)}$$

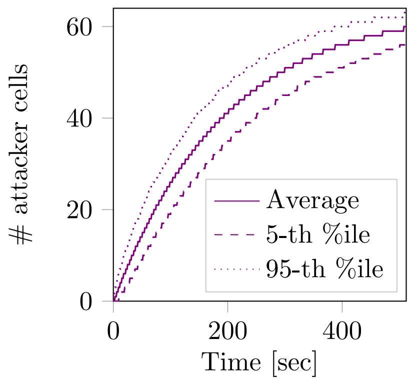
with n being the number of cells the attacker controls, nCk being the binomial coefficient and $k = 64$ the number of cells. This leads to a binomial distribution. Table 2 gives an overview of the parameters.

The purple lines in figure 1a show the number of cells an attacker is expected to control at any time up to t_B. The upper line shows the 95-th percentile and the lower line the 5-th percentile.

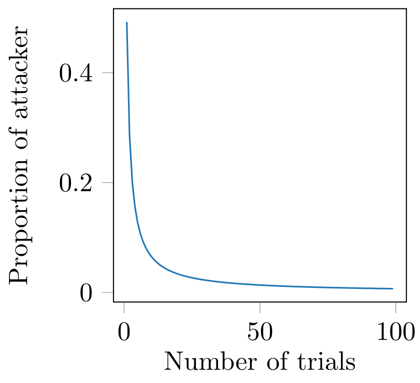
The calculations show that multiple variables have a big influence on a possible attack. One obvious factor is the proportion of malicious traffic. Another relevant factor is the number of chances that the attacker gets to be selected by Blink per cell of the Flow Selector. This is the number of trials $n_t = t_B/t_E$. Doubling the proportion of malicious flows has approximately the same influence as doubling the time budget or halving the average eviction time.

Table 2: Parameters of the analytical model

Name	Abbreviation	Value
Time budget (Flow Selector reset timeout)	t_B	512 seconds
Eviction time	t_E	-
Number of trials	n_t	$n_t = t_B/t_E$
Monitored flows	k	64
Attacker flows in Flow Selector	n	-
Proportion of legitimate traffic	q_l	-
Proportion of malicious traffic	q_m	-



(a) Calculation of an attack with 2% of all traffic and eviction time of 4 seconds



(b) Proportion of attacker traffic needed to have a 50% chance of success

Figure 1: Results from the analytical model

4.1.1 Probability of an successful attack

With the analytical model one can calculate how big the percentage of attacker traffic of all traffic needs to be in order to make a successful attack. The calculation in figure 1b were done for a range of number of trials, n_t. The x-axis shows the number of trials and the y-axis the proportion of all packets the attacker needs to send in order to make a successful attack within the given number of trials. The attacker strength for a 50% chance of attack was calculated. As the number of trials (n_t) rises, the variance of the number of malicious flows in the cells also increases. This means that, for a big n_t, an attacker has better chances to get lucky, i.e. to succeed in an attack even though he has a smaller proportion of the traffic than the calculation suggests.

An attacker with 10% of the flows can be successful after only $6 * n_t$ and an attacker with 1% of the flows after $65 * n_t$. With the time budget given by the Blink implementation with $t_B = 512$ the legitimate flows have to stay active for minutes in order to secure the system from a small attacker. Now that we have seen that the unknown time that the flows spend being monitored by Blink plays a significant role, we will find out how long it takes for legitimate flows to get evicted.

4.2 Eviction Time

As we saw in the previous subsection, the eviction time is an important parameter in determining how strong an attacker has to be to make a successful attack. Fortunately, the Blink paper provides logs of the Flow Selector for the anonymised internet traffic traces used in the papers evaluation.

With logs like in table 3, the top 20 prefixes with the most throughput were

Table 3: Log from the Flow Selector of inserted and evicted flows

Time	Action	Src. IP	Dst. IP	Src. port	Dst. port
0.00	Add_flow	231.99.49.4	206.16.36.172	14647	80
0.00	Add_flow	153.107.141.214	50.21.229.59	33487	9482
0.00	Add_flow	101.126.224.8	183.101.92.58	9050	32881
0.00	Add_flow	201.37.180.68	141.57.205.187	80	30243
0.01	Remove_flow	201.37.180.68	141.57.205.187	80	30243

chosen for each trace and the eviction times calculated for each prefix. The results are 229 eviction time distributions, the median of the average eviction time is 10 seconds. Figure 2 shows the distribution of the eviction times of all evaluated prefixes. Firstly, when looking at the 20-th percentile line in dashed blue, one can see that some prefixes have a significant percentage of flows that were shorter than two seconds in the Flow Selector and that most prefixes have more than 20% of the flows within a few seconds. 4.8% of the prefixes have the majority of their flows getting evicted after less than 0.8 seconds. For those prefixes it might be hard to defend them from attacks. Most of the prefixes have many evictions just after two seconds and a fast decline in evictions thereafter.

In figures 3a, 3b and 3c three examples of eviction time distributions are shown. Figure 3a has many flows that get evicted after less than three second. With an average of 3.79 it is a rather bad case for Blink. Figure 3b shows an average case although also many flows end after two to three seconds there are many flows that stay active longer, thus helping to prevent attacks. Figure 3c shows one of the be better cases for Blink.

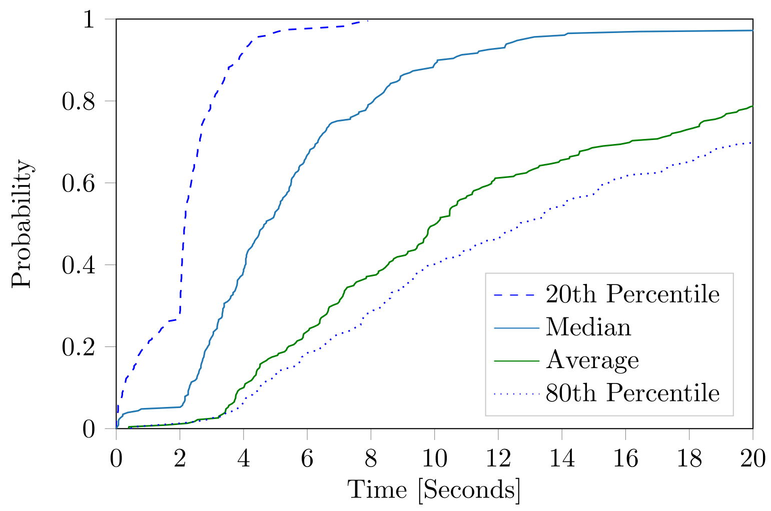


Figure 2: Cumulative distribution function of the eviction times for the prefixes with the most throughput

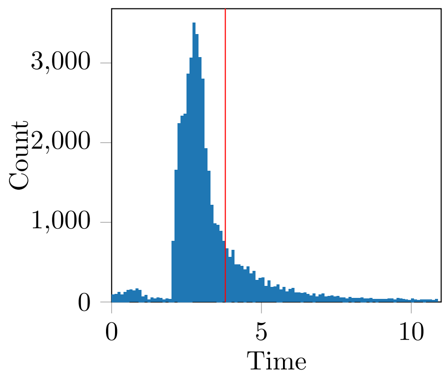
With those insights on simulations on mininet are necessary to show that attacks work as calculated in section 4.1.

4.3 Simulations on mininet

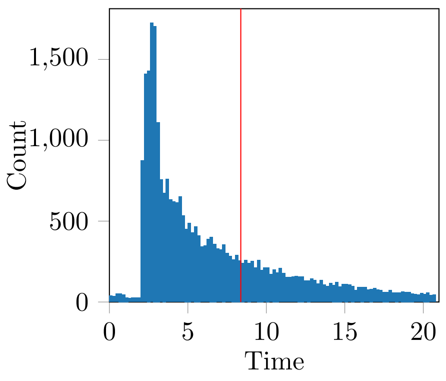
In a first step a simple topology was used. 2'000 legitimate flow and 105 malicious flows ($q_m \approx 0.05$) are used. This should show, that the attacker could be successful using 5% of the flows. For the three distributions shown in figure 3, multiple simulations were done to check the analytical model.

The analytical model for this comparison was adjusted because of the limited number of flows. The number of flows that are expected to be in the Flow Selector after any time step got subtracted from the original 105 attacker flows to calculate the new proportion of legitimate flows q_l for the next time step. It is not taken into account that there are hash collisions, even though the attacker is expected to match only 51.75 cells with his 105 flows. Also, the calculation neglects the distribution of the eviction times and takes the average eviction times for the formula for $P(n)$ from 4.1.

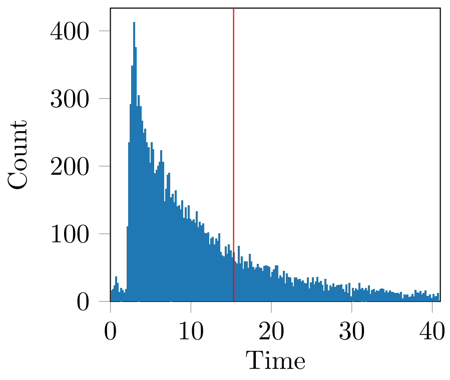
As we can see, the analytical model matches the simulations on mininet well. Therefore, the attacks we described in section are a realistic threat to Blink.



(a) Average of 3.79 seconds

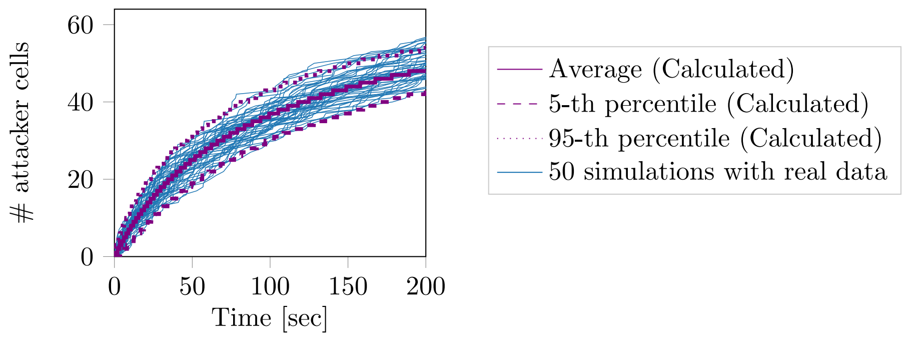


(b) Average of 8.37 seconds

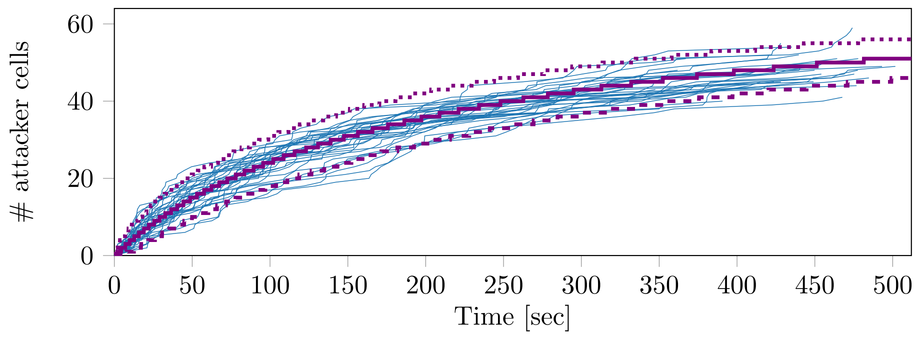


(c) Average of 15.31 seconds

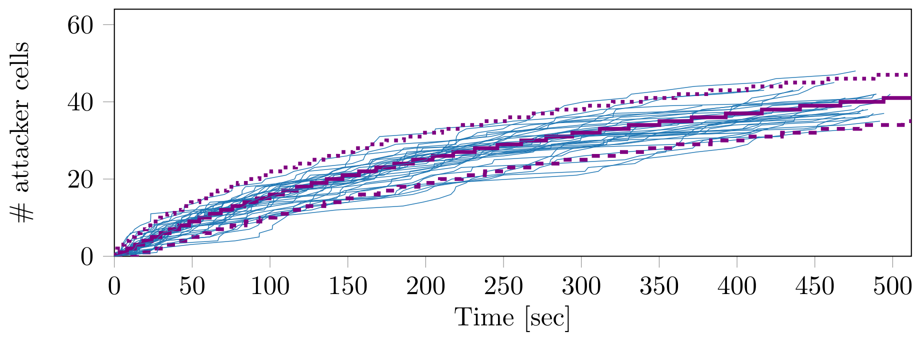
Figure 3: Eviction time distributions for three different prefixes with the average in red



(a) Average eviction time of 3.79 seconds



(b) Average eviction time of 8.37 seconds



(c) Average eviction time of 15.31 seconds

Figure 4: Number of Flow Selector cells occupied by an attacker flow for the three example prefixes

5 Countermeasures

Designing a countermeasure for the discussed attack on Blink requires considering multiple trade-offs. When trying to improve security, two main objectives have to be considered. Blink is efficient on hardware resources and fast reroutes on over 80% of the complete failures in the synthetic traces evaluated in the Blink paper. A countermeasure should try to keep efficiency and rerouting performance high, while mitigating attacks.

5.1 Resetting Flow Selector cells

As we saw in the previous chapter, the attacker has a relatively easy way to trigger rerouting in a Blink enabled system, because he has much time to insert his flows. Therefore, the countermeasure reduces this time significantly. If all the cells are reset at once, Blink is not able to reroute for some time. In order to maintain a high availability of monitored flows, the Flow Selector in the countermeasure resets the cells consecutively. A period can be chosen and the Flow Selector cycles through all the cells, resetting one cell in every period. To make sure that a cell is only reset once within its time slot, a new 1-bit variable is included.

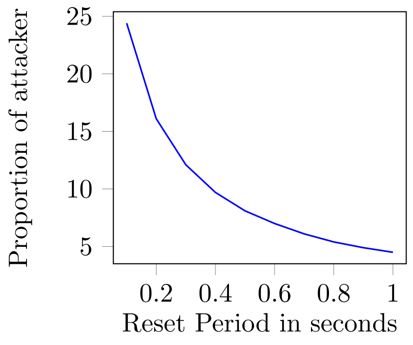
A mild approach to mitigate attacks would be to reset one cell of the Flow Selector every second and therefore set the overall reset time to 64 seconds. A more aggressive approach would be to reset eight cells every second and therefore reset one cell every 125ms.

5.2 Evaluation

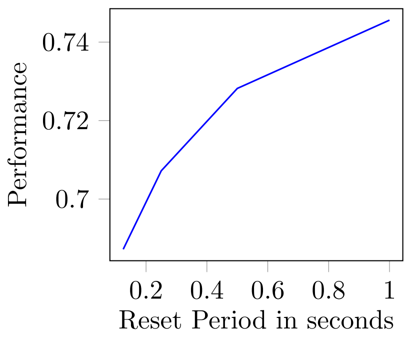
For the analysis of the countermeasure a worst case scenario was considered. This is that every flow gets evicted after the timeout for inactive flows, which is only two seconds. It can be seen in figure 2, that, for many prefixes, most flows are evicted after two to four seconds. In the scenario with an eviction time of two seconds the Blink system without any countermeasure is unsafe from attackers who have just 0.27% of the total traffic.

5.2.1 Mitigation of malicious attacks

Calculations with this eviction time show for the mild approach of resetting one cell every second, the attacker has to have a strength of 4.5% and with the aggressive strategy even 21.5% of all traffic. This means that an attacker has to make about 80 times more traffic in order to trigger Blink's rerouting mechanism. The proportion of traffic an attacker needs to trigger rerouting against different reset periods, with an eviction time of two seconds, can be seen in figure 5a. Figure 6 shows how many cells an attacker is expected to gain (black) compared to a simulation in mininet (blue). The attacker has in this case 10% of all the traffic and the countermeasure uses a cell reset time of 125ms.



(a) Attacker strength needed in % to have a 50% chance of success against the countermeasure with eviction time of two seconds



(b) Rerouting performance in % for different reset periods

Figure 5: Evaluation of the countermeasure

During the 25 minutes of simulation, the attacker was maximally able to get 25 flows selected by Blink's Flow Selector and therefore couldn't trigger rerouting.

5.2.2 Influence on Performance

When cells get reset consecutively and quickly it is clear that less overall cells are available to register retransmissions. The evaluation of Blink was used to find out how big the drop in performance would be, if an aggressive approach is chosen. Unfortunately the traces created for the performance measurements simulate the complete failure already after ten seconds. Because of this no analysis of a slow reset mechanism was possible.

Blink's original mechanism rerouted 83.9% of the time, when a complete failure was simulated. The most aggressive countermeasure, with a reset period of 125 ms, that was evaluated rerouted about 69% of the time. This is a loss of nearly 15% of all cases in comparison with

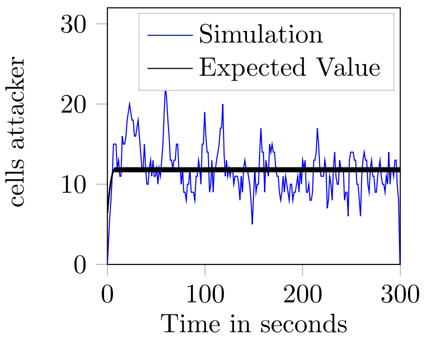


Figure 6: Simulation of the attack on a prefix with eviction times of 2 seconds and an attacker with 10% of the traffic against a reset period of 125 ms

the results from the Blink paper. The mild option reroutes about 73% of the time, however it has to be considered that the impact might not be fully shown, due to the short simulation. The results can be seen in figure 5b.

This loss in performance could be revised by lowering the reset threshold or by increasing the number of monitored flows.

5.2.3 Influence on Efficiency

Blink only uses 6162 bits of memory for each prefix monitored. One goal of the countermeasure is to keep the Blink system efficient. The one bit of additional data that is occupied by the countermeasure is negligible. The same counts for the number of write and read accesses that have to be made in order to maintain the reset state of all the cells.

6 Further Possibilities

There are many other possibilities for countermeasures that could be added for Blink, which are listed and explained below.

- Retransmission distribution:
  Use the controller to look if the given signal is an actual failure. The controller could rely on the RTT distribution of the traffic, which gives an idea of how the bursts of retransmissions should look like over time.

- Bidirectional flows:
  Monitoring bidirectional flows could make an attack harder because he has to establish connections instead of sending any fake traffic. Similarly monitoring the throughput could make an attack harder, since the attacker has to send much traffic instead of just many packets.

- Throughput monitoring:
  Monitoring not only the packets but also their payload would increase the bandwidth needed for an attacker.

- Make the time budget adaptive:
  Making the time budget adaptive would enable Blink to set the rolling time budget of the Flow Selector according to the current characteristics of the traffic, which result in a specific eviction time distribution. However, an attacker knowing the algorithm of Blink could force Blink to use a time budget by sending many long flows, as he would anyways do in the attack described in 4.1.

7 Conclusion

We saw that the most detrimental attack on Blink is to trigger rerouting on a network switch. It was shown that the attack on the current Blink system was feasible for an attacker that sends less than 1% of all the flows to the destination prefix and in a worst case scenario even with 0.27%. After the analysis of the characteristics of the TCP flows that get monitored, a countermeasure was proposed and analysed. An attacker with up ton 21.5% of the traffic can be prevented from triggering rerouting, with a trade-off on Blink's rerouting performance and without influencing Blink's hardware efficiency. This means that an attacker has to have 80 times more traffic to make a successful attack.

References

[1] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95, 2014.

[2] Vibhaalakshmi Sivaraman, Srinivas Narayana, Ori Rottenstreich, S Muthukrishnan, and Jennifer Rexford. Heavy-hitter detection entirely in the data plane. In *Proceedings of the Symposium on SDN Research*, pages 164–176. ACM, 2017.

[3] Blink: Fast connectivity recovery entirely in the data plane. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, Boston, MA, 2019. USENIX Association.

[4] Mike Shand and Stewart Bryant. Ip fast reroute framework. Technical report, 2010.

[5] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed internet routing convergence. *IEEE/ACM transactions on networking*, 9(3):293–306, 2001.

[6] Thomas Holterbach, Stefano Vissicchio, Alberto Dainotti, and Laurent Vanbever. Swift: Predictive fast reroute. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 460–473. ACM, 2017.