Link-flooding defense entirely on the data plane

Jiarong Xing, Wenqing Wu, Ang Chen

Rice University

The recent proposed link-flooding attacks (LFAs) [2] bring a great threat to our networks. The LFA attackers identify network critical links by using tools like traceroute, and then flood them to cut off the network connectivity by creating lots of low-rate flows with decoy servers around the targeted area. Defending against LFAs is difficult because the attacker usually uses low-rate legitimate flows and only sends "wanted" traffic (e.g. one HTTP GET request per second) to make it *indistinguishable* from legitimate user flows. Besides, the attacker could select multiple targeted links to launch *rolling attacks*. Thus, even if the attack on one critical link is detected, she can change her attacking gesture easily.

Over the years, people never stopped proposing new defenses for LFAs including filtering-based solutions that identify and block malicious flows, and rerouting-based solutions that absorb malicious traffic by routing around the congestion. However, existing defenses are *reactive* and *inefficient* of which the filtering and rerouting could take one minute to tens of minutes after being attacked, at which time damage has been caused. Worse still, their inefficiency makes them lose effectiveness for rolling attacks because the attacker can change attacking gestures faster than the response of defenses.

Our defense: DPDef. In this paper, we propose DPDef, a distributed proactive defense entirely on data planes that can mitigate LFAs at the beginning of the attacks at RTT timescales. In DPDef, we use the recently emerged programmable switches which provide programmability of data planes with high-level languages like P4 [1], allowing people to customize packet headers and new protocols. Besides, they can perform per-flow monitoring and per-packet inspection over header fields at linespeed (Tbps), which enables a more accurate and efficient LFA detection. DPDef maintains per-flow state on data plane to identify suspicious flows based on attack-specific features in the attack launching phase (before the attack takes effect), like persistent low-rate flows to a destination prefix. When the LFA takes effect, DPDef will reroute malicious traffic to alternative paths computed on the fly to mitigate the congestion of the targeted link. At the same time, it also creates an "illusion of success" to the attacker to prevent following attacks by inflicting packet drops from malicious flows and disabling traceroute temporarily.

Malicious flow detection. DPDef marks suspicious flows in the attack launching phase based on some attack-specific features. For example, naïve attackers might use non-conforming TCP traffic to create a large volume of traffic with a small number of connections, which can be detected by inspected their responses to TCP congestion control signals like packet drops. Smart attackers could use lots of low-rate protocol confirming flows to make themselves indistinguishable, which can be detected by looking for persistent low-rate flows to a certain destination prefix.

Routing around congestion. The intuitive way to mitigate LFAs after marking suspicious flows is to block all of them. However, the false positive of the marking process will cause collateral damage to legitimate flows. To avoid this problem, DPDef uses a conservative defense that reroutes suspicious traffic to alternative paths. DPDef disseminates *probes* (packets with special headers) that carry path utilization metrics on each programmable switch to find the least-congested alternative path on the fly. To minimize the influence on legitimate flows, we only reroute suspicious traffic to alternative paths.

Following attacks prevention. If an attacker realizes that her traffic is rerouted, she might change her attack gesture to flood other links. To prevent the following attacks, we create an "illusion of success" to the attacker by inflicting packet drops from malicious flows randomly. Besides, in the rerouting phase, DPDef will disable traceroute temporarily to prevent the attacker from obtaining the updated network topology.

Ongoing work. We have a proof of concept prototype running on a customized version of `ns3` with P4 `bmv2` model support. We use a network topology that has two critical links and the attacker performs rolling attacks against these links whenever she detects a routing change. The results show that DPDef can reroute malicious traffic at RTT timescales. And its efficiency makes it robust to rolling attacks. DPDef is an ongoing project, we are working on several open questions. For example, we are trying to find more attack-specific features that can help us detect suspicious flows more accurately. Besides, we need to distinguish real LFAs and normal temporary link congestions. Finally, the programmable data planes only have a small memory, so we need to design memory-efficient data structures for the LFA defense.

References

[1] P. Bosshart, D. Daly, et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95, 2014.

[2] M. S. Kang, S. B. Lee, and V. D. Gligor. The crossfire attack. In *2013 IEEE symposium on security and privacy*, pages 127–141. IEEE, 2013.

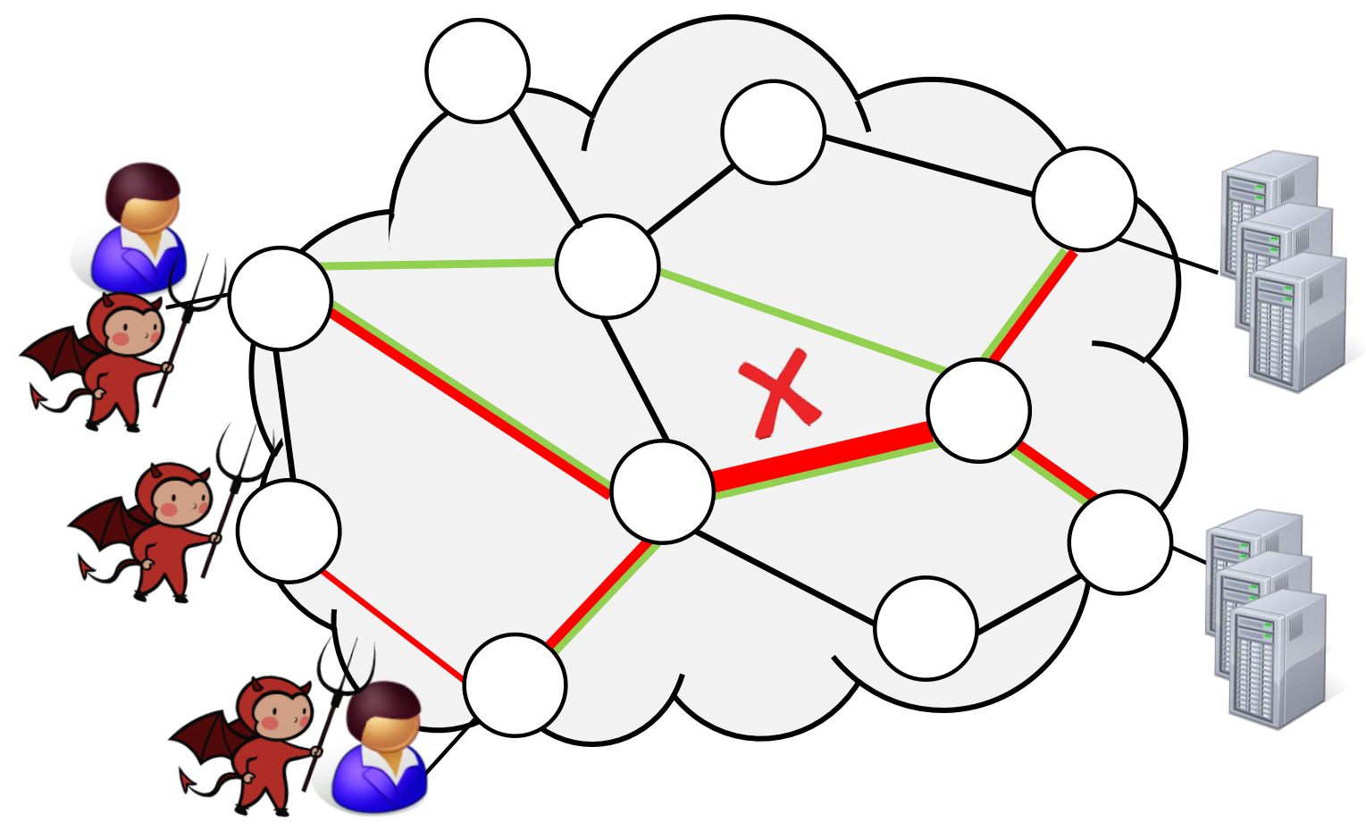
Data Plane Link-Flooding Defense against Rolling Attacks

Jiarong Xing, *Wenqing Wu and Ang Chen*
Rice University
Email: jxing@rice.edu

1. Problem

- ***Link-flooding attacks (LFAs):*** Attacking the victim by flooding critical links without triggering control plane actions.
- Step 1: Obtain network topology via ***traceroutes.***
- Step 2: Pinpoint ***critical links.***
- Step 3: Flood by sending ***normal low-rate*** flows.
- Step 4: Keep ***changing attack parameters*** *dynamically*.

- ***Features of link-flooding attacks:***
- Cost-sensitive
- Data plane attacks
- Indistinguishability.
- Rolling attacks.

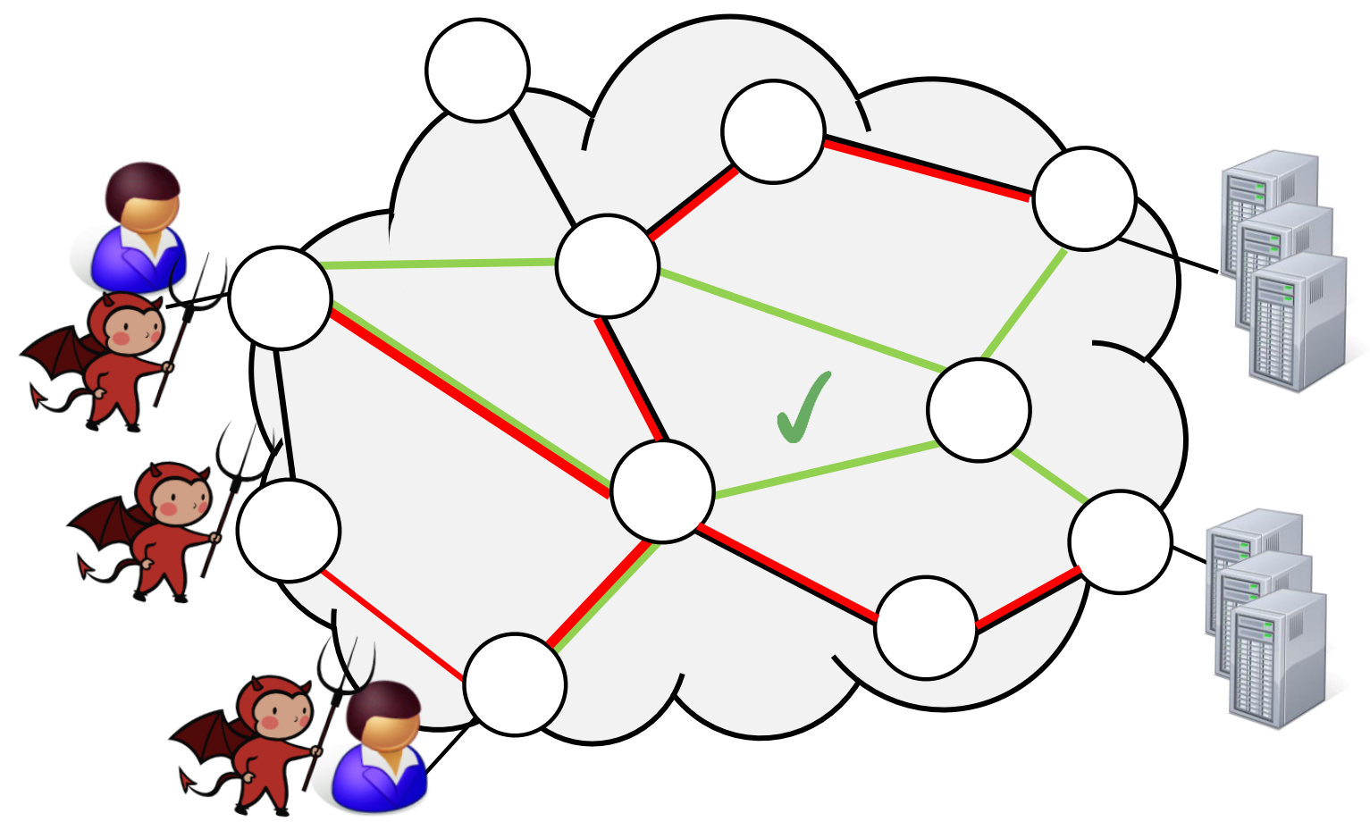


LFAs flood critical links

2. State of the art

- Existing work mitigates LFAs **on the control plane**.

- **Filtering** malicious traffic based on certain features.
- **Rerouting** traffic from the congested link to others.



Mitigating LFAs via rerouting

- Existing LFA defenses have ***inherent limitations:***
- ***Hard to filter*** indistinguishable low-rate malicious flows.
- Rerouting is ***slow***, so cannot handle rolling attacks.
- Data planes attacks might ***not even trigger*** control plane actions because they are low-rate.

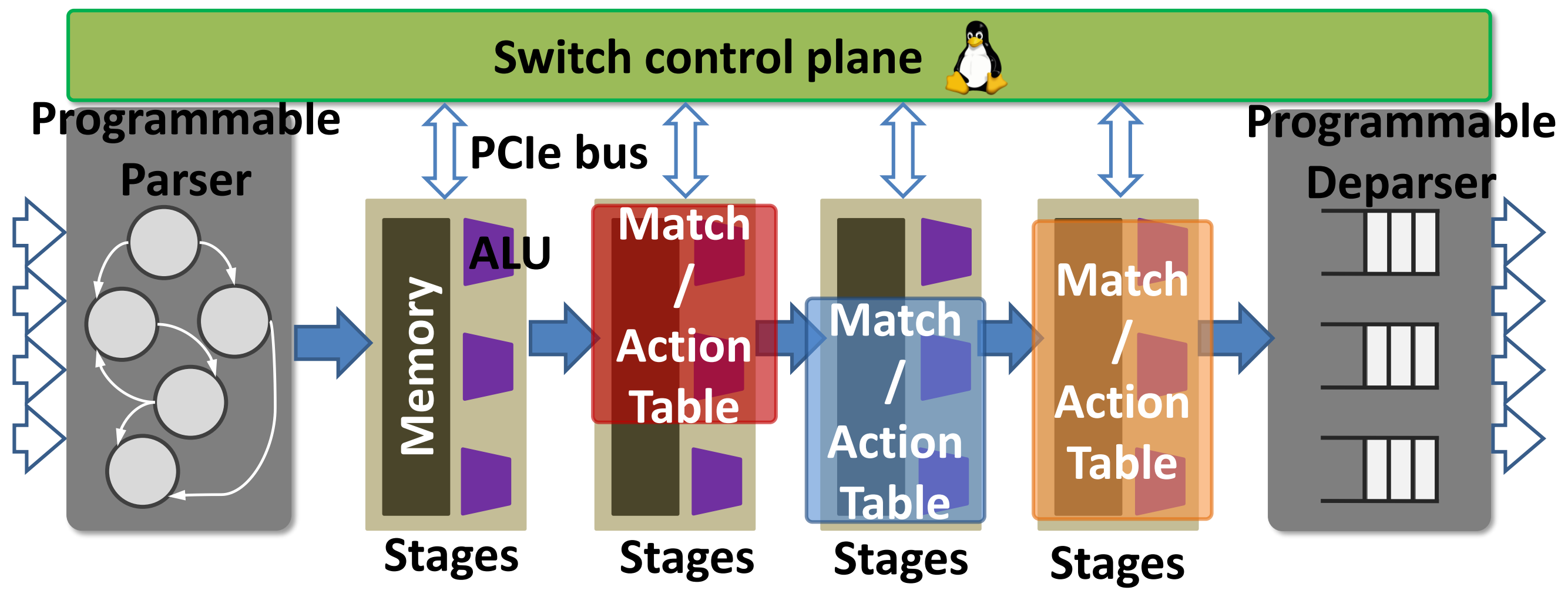
3. Key insight

- ***Data plane defenses against data plane LFAs.***

- ***What are benefits of data plane defenses?***
- ***High visibility.*** Fine-grained detections.
- ***High agility.*** Fast responses to rolling attacks.

- New opportunity:

New defenses on programmable data planes.
- User-defined protocols
- Customized packet processing
- Sophisticated hardware state
- Programmable with high-level languages, e.g. P4.



The architecture of a programmable switch

4. Our approach

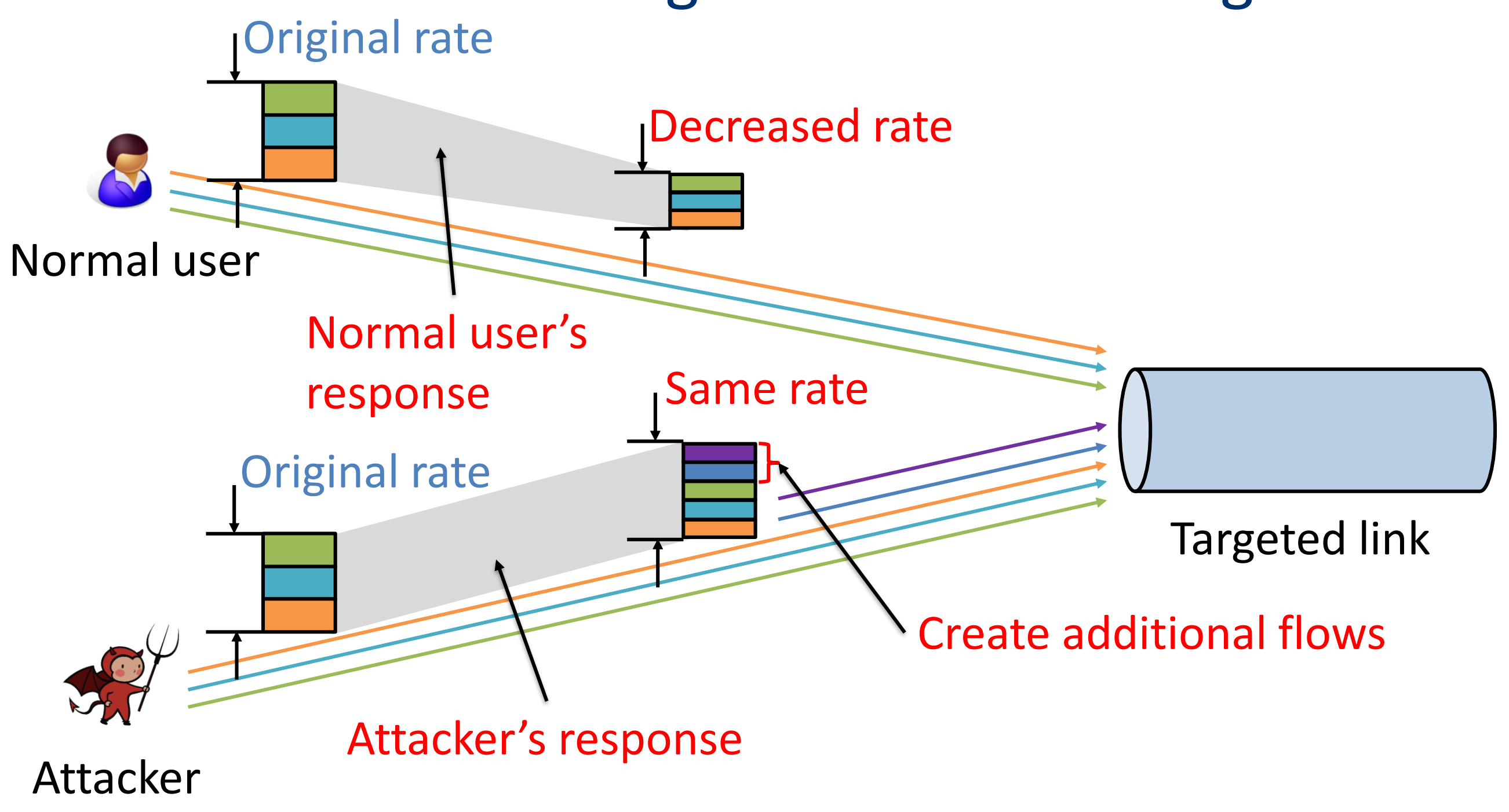
Two-stage data plane link-flooding defense

- ***In-place classification:*** Identify malicious IPs by observing ***sending rate change patterns*** when the link is congested.
- "In-place" ***minimizes disturbance*** to normal traffic.

- ***Fast rerouting:*** Reroute traffic to other links to alleviate congestion of the target link on the data plane.
- Calculate alternative paths ***on the data plane*** makes the rerouting ***fast***.

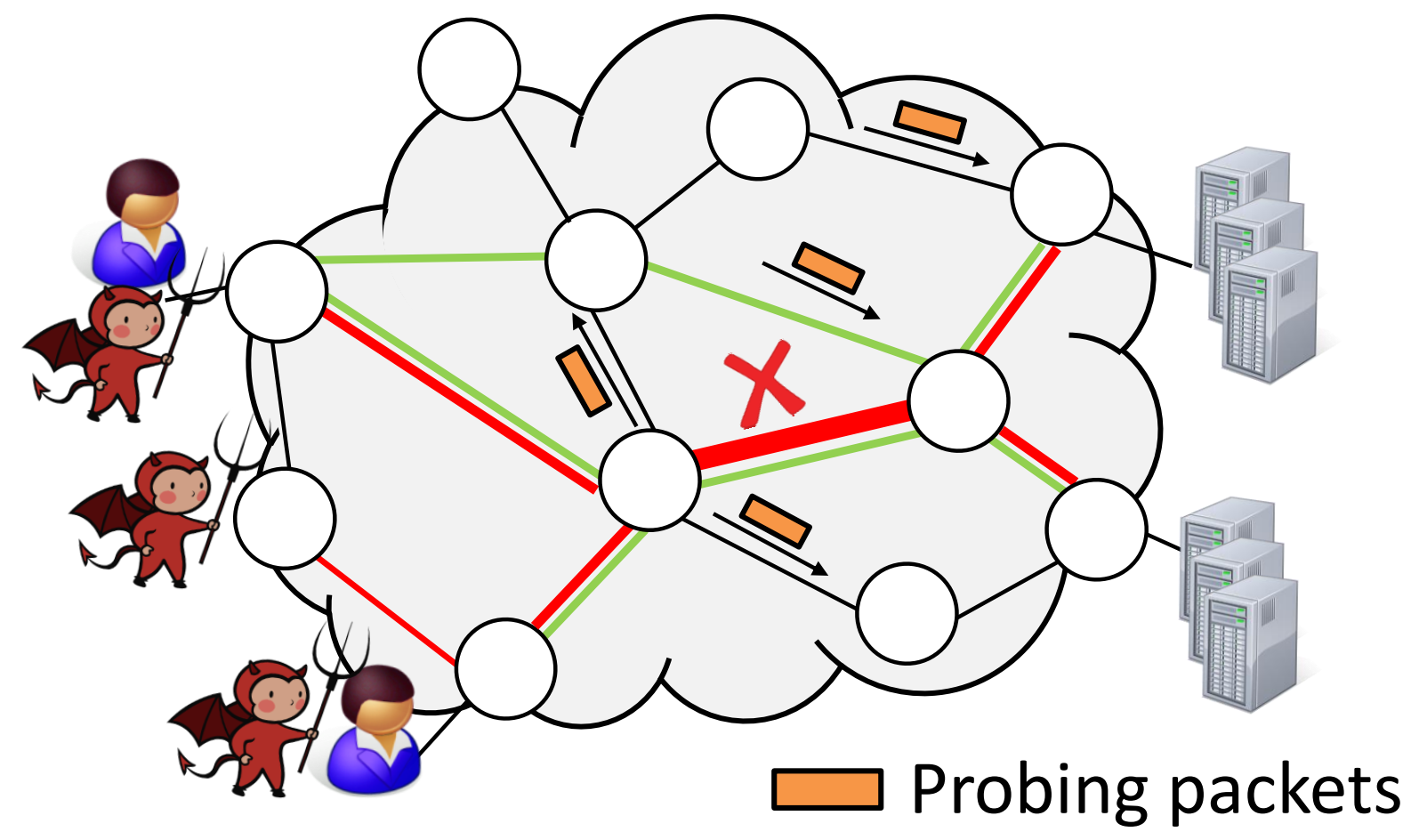
5. In-place classification

- ***Different sending rate change patterns:***
- ***Normal IP:*** The sending rate will decrease.
- ***Malicious IP:*** The sending rate will not change.



Different response to TCP congestion control signals

6. Fast rerouting



Discovering alternative paths via probing packets

- ***How to calculate alternative paths on the data plane?***
- Each switch broadcasts ***probing packets*** that contain link utilization to other switches (Hsu-Contra-NSDI'20).

- ***Would rerouting to longer paths hurt performance?***
- Preserve performance by only rerouting suspicious traffic.

- ***How to prevent the attacker from changing her parameters?***
- Disable traceroute while rerouting.
- Create an illusion of success by randomly dropping packets from suspicious flows.

This work is partially supported by the National Science Foundation through the grant CNS-1801884.