Challenges in Realizing Privacy-aware Cloud-based DDoS Mitigation Mechanism

Su-Chin Lin*, Wei-Ning Chen* and Hsu-Chun Hsiao*†
*Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
†Research Center for IT Innovation, Academia Sinica, Taiwan

I. Introduction

In recent years, cloud-based Distributed denial of service (DDoS) mitigation has been widely deployed. Unfortunately, existing cloud-based schemes [4] severely degrade user privacy, breaking the end-to-end security guarantee [6]. An untrusted or compromised cloud could expose users' sensitive data as demonstrated by the Cloudbleed vulnerability [2] against Cloudflare, one of the biggest DDoS defense providers.

Our previous work, DAMUP [9], proposed a lightweight architecture based on the observation that the victim server can often better differentiate benign and malicious flows. When under DDoS attack, DAMUP utilizes a cryptographic token established during the initial connection, to filter the attack traffic. The client embeds a valid token to mark its traffic benign, while the adversary does not have the token; that is, every connection is first redirected through the proxy, and those without valid tokens will be blocked.

Illustrated in Figure 1, the DAMUP proxy is deployed in clouds, since it can usually resist higher intensity attacks. When the server is under attack, the proxy will only forward traffic with legal tokens, which are embedded in TLS handshake. Thereby, the DAMUP solution ensures content privacy since the proxy does not have to know the server's TLS private key. The gateway is the edge router of an organization. To reduce the damage when the server's public address is exposed [12], [3], the proxy constructs a secure tunnel between the proxy and the gateway. The gateway will only forward packets that come from the secure tunnel to the server.

Currently, our DAMUP implementation is open-sourced on GitHub [1], including all configurations and a complete step-by-step tutorial. However, there still remain several implementation challenges that need to be tackled. In the poster, we will focus on the implementation details and discuss promising directions to address these challenges. The challenges include privacy concerns, scalability issues, and compatibility with TLS 1.3.

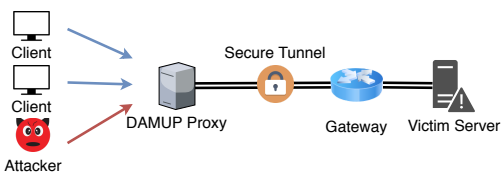

Fig. 1: DAMUP system architecture

II. Implementation

A. Implementation and Challenges

We embeds the token in the Server Name Indication (SNI) [5] field in the TLS handshake. The token consists of a Client ID and a HMAC-SHA256 signature. The proxy validates the user by verifying the signature. It will proxy the traffic to the server if the verification succeed. Regarding the implementation, we modified the open-source `sniproxy` library [10] to support the desired features. The `sniproxy` should be built on the cloud services, which have considerable amounts of bandwidth such that it can handle the tokens.

The secure tunnel ensures that all traffic to the server will be via the proxy. We leverage a open-source IPsec library strongSwan as the realization. The implementation is based on IKEv2 [8], which provides confidentiality and data origin authentication. Thus it's resistant to IP spoofing.

Nevertheless, there are some implementation challenges regarding the scalability and privacy. First, to prevent token abused by the adversary, the proxy should monitor the traffic for each token. However, this is not scalable since the proxy has to keep token states. The design of a stateless token is preferred. Second, the current implementation of token may lead to another privacy concern. An ISP or a malicious middlebox may take advantage of the cryptographic token to track the user. Third, with respect to compatibility, there are drafts in IETF [7] which considers encrypting the SNI field, albeit in TLS 1.3, the SNI is still left in plain text [11]. Hence, the token may have to be embedded in other TLS extension field.

B. Possible Solutions

The stateless and anti-tracking token is possible. We can shorten the lifetime of the token, such that the token is more difficult to be tracked. Furthermore, the traffic limit can be also included in the token. Nonetheless, the overhead of deriving a new token, as well as the signature verification, should be taken into consideration.

For the encryption of SNI, although we can always embed the token in other TLS extension fields, the configuration cost of the user is substantial. It is tricky because the user might require more modification to embed the token. We have to strike a balance between implementation and the user's cost.

REFERENCES

[1] https://github.com/bookgin/damup, accessed: 2018-07-05.

[2] "Cloudbleed bug: Everything you need to know," https://www.cnet.com/how-to/cloudbleed-bug-everything-you-need-to-know, accessed: 2018-07-05.

[3] "Hatcloud - bypass cloudflare with ruby," https://github.com/HatBashBR/HatCloud, accessed: 2018-07-05.

[4] "Step 1: How does cloudflare work? – cloudflare support," https://support.cloudflare.com/hc/en-us/articles/205177068-Step-1-How-does-Cloudflare-work-, accessed: 2018-07-05.

[5] S. Blake-Wilson, M. Nystrom, D. Hopwood, J. Mikkelsen, and T. Wright, "Transport layer security (tls) extensions," Internet Requests for Comments, RFC Editor, RFC 3546, June 2003.

[6] Z. Durumeric, Z. Ma, D. Springall, R. Barnes, N. Sullivan, E. Bursztein, M. Bailey, J. A. Halderman, and V. Paxson, "The security impact of https interception," in *NDSS*, 2017.

[7] C. Huitema and E. Rescorla, "Issues and requirements for sni encryption in tls," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-tls-sni-encryption-03, May 2018, http://www.ietf.org/internet-drafts/draft-ietf-tls-sni-encryption-03.txt. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-tls-sni-encryption-03.txt

[8] C. Kaufman, "Internet key exchange (ikev2) protocol," Internet Requests for Comments, RFC Editor, RFC 4306, December 2005, http://www.rfc-editor.org/rfc/rfc4306.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4306.txt

[9] S.-C. Lin, P.-W. Huang, H.-Y. Wang, and H.-C. Hsiao, "Damup: Practical and privacy-aware cloud-based ddos mitigation."

[10] D. Lundquisti, "sniproxy," "https://github.com/dlundquist/sniproxy".

[11] E. Rescorla, "The transport layer security (tls) protocol version 1.3," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-tls-tls13-28, March 2018, http://www.ietf.org/internet-drafts/draft-ietf-tls-tls13-28.txt. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-tls-tls13-28.txt

[12] T. Vissers, T. Van Goethem, W. Joosen, and N. Nikiforakis, "Maneuvering around clouds: Bypassing cloud-based security providers," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1530–1541.