Internet Area Working Group K. Xu
Internet-Draft Tsinghua University & Zhongguancun Laboratory
Intended status: Informational X. Feng
Expires: 1 January 2026 Y. Yang
Tsinghua University
Q. Li
Tsinghua University & Zhongguancun Laboratory
30 June 2025

Enhancing ICMP Error Message Authentication Using Challenge-Confirm Mechanism

draft-xu-intarea-challenge-icmpv4-01

Abstract

The Internet Control Message Protocol (ICMP) plays a crucial role in network diagnostics and error reporting. However, it is a challenge to verify the legitimacy of a received ICMP error message, particularly when the ICMP error message is embedded with stateless protocol data. As a result, adversaries can forge ICMP error messages, leading to potential exploitation and off-path attacks.

This document explores solutions to this problem by first presenting a straightforward but flawed stateful challenge-response mechanism. It explains how this "strawman" approach, while preventing simple spoofing, introduces a severe vulnerability to state-exhaustion Denial-of-Service (DoS) attacks.

Building on this analysis, the document then proposes a robust, stateless challenge-response mechanism inspired by TCP SYN-Cookies. This final proposal eliminates the need to store per-challenge state by computationally generating challenges. It limits state management to minimal flags on existing sockets or a bounded probabilistic data structure. This approach effectively authenticates ICMP error messages while inherently resisting both off-path spoofing and state-exhaustion DoS attacks, thus significantly improving the robustness of ICMP. Additionally, it discusses security and deployment considerations to ensure its practical implementation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

This Internet-Draft will expire on 1 January 2026.

Copyright Notice

Table of Contents

1. Introduction

The Internet Control Message Protocol (ICMP) [RFC792] is an integral part of network operations, providing essential feedback on transmission issues such as unreachable destinations or packet fragmentation requirements. However, the legitimate verification of ICMP error messages is inherently vulnerable by design. To enable senders to correlate error reports with the original packets for effective network diagnostics, ICMP error messages, as specified in [RFC792], MUST include the header information and a portion of the payload of the original message that triggered the error. When the original message originates from stateless protocols like UDP or ICMP, the embedded original message header lacks contextual information (e.g., sequence numbers, acknowledgement numbers, and ports in stateful protocols like TCP). This makes it difficult for the receiver to effectively verify the legitimacy of the error messages. Consequently, attackers can forge ICMP error messages embedded with stateless protocol payloads to bypass the legitimate verification of the receiver, tricking the receiver into erroneously accepting and responding to the message, which can lead to malicious activities.

For example, off-path attackers can forge ICMP "Fragmentation Needed" messages, embedding stateless protocols like UDP or ICMP Echo Reply, degrading throughput and harming latency-sensitive applications. This can also induce TCP segment fragmentation [NDSS2022MTU] and enabling IP ID-based TCP session hijacking [CCS2020IPID]. Moreover, forged ICMP Redirect messages embedded with stateless protocol data can be used to trick victims into modifying their routing, facilitating off-path traffic interception, modification, and credential theft [USENIXSECURITY2023ICMP], [SP2023MITM].

This document explores how to securely authenticate these ICMP error messages. It first examines an intuitive challenge-confirm solution but demonstrates its fatal flaw: vulnerability to Denial-of-Service (DoS) attacks. It then presents a refined, stateless mechanism that solves the original problem without introducing new vulnerabilities.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. TCP terminology should be interpreted as described in [RFC9293].

3. Problem Statement

Current ICMP specifications have limitations that enable off-path attackers to forge ICMP error messages embedded with stateless protocol data, compromising network security and reliability. The key issues are as follows:

3.1. Source-Based Blocking Ineffectiveness

Certain ICMP error messages, like the "Fragmentation Needed" messages, can originate from any intermediate router along the packet's path. Given this wide range of possible sources, legitimate messages can come from numerous locations. As a result, source-based blocking is rendered ineffective because it becomes difficult to distinguish between legitimate and forged messages based on the source address alone.

3.2. Authentication Evasion based on Embedded Packet State

Although ICMP requires including as much of the original (offending) packet as possible in error messages without exceeding the appropriate MTU limits, the stateful or stateless nature of the embedded packet protocol directly impacts the authentication difficulty and security strength of ICMP error messages. According to ICMP specifications [RFC792], [RFC1122], error messages should include at least the first 28 octets of the original packet to aid in identifying the affected process and verifying legitimacy.

3.2.1. Stateful Embedded Packets (e.g., TCP)

When off-path attackers embed stateful protocol packets, such as TCP segments, in forged ICMP error messages, the receiver has some means of verification. The TCP protocol uses sequence numbers, acknowledgment numbers, and ports to establish and maintain a connection state between communicating parties. This connection-specific information is difficult for off-path attackers to accurately guess. Receivers can check if the connection-related details in the embedded TCP header match the TCP connection state they maintain. For example, they can verify if the sequence number in the TCP segment embedded in the ICMP error message aligns with the expected sequence number for an ongoing TCP connection. This way, they can make an informed judgment about the authenticity of the ICMP error message.

3.2.2. Stateless Embedded Packets (e.g., UDP, ICMP)

In contrast to stateful TCP, when attackers embed stateless protocol packets, such as UDP or ICMP messages, in forged ICMP error messages, receivers lose the ability to perform effective state verification. UDP and ICMP protocols are inherently designed as stateless protocols, where the source does not maintain any session state information. The UDP or ICMP messages embedded in ICMP error messages contain almost no state information that can be used for context verification. In addition to performing some basic protocol format checks, receivers have virtually no way to determine the authenticity of ICMP error messages based on the embedded stateless packet header. This lack of state verification greatly reduces the authentication strength of ICMP error messages, making it easier for attackers to implement authentication evasion and use forged error messages for malicious attacks.

4. A Strawman Proposal: Stateful Challenge-confirm

A logical way to verify that an ICMP error originates from an on-path entity is to issue a challenge and await a correct confirm. This proves that the entity that sent the error is also on the path of subsequent traffic for that flow.

Let’s consider a simple, stateful challenge-confirm mechanism.

The operational flow would be as follows:

1. Receive Error: Host A receives an ICMP error message (e.g., Fragmentation Needed) that claims to be from an on-path router R, regarding a UDP flow to Host B.

2. Generate and Store Challenge: Host A does not trust the message. It generates a large, unpredictable random number (a nonce). It then stores this nonce in a local cache, associating it with the flow’s identifiers (e.g., the 4-tuple) and a timeout.

3. Issue Challenge: Host A sends the next UDP packet for retransmission for that flow to Host B. This packet includes the nonce in the IP Option field.

4. Receive Confirmation: If router R is legitimately on the path and the error condition persists, it will drop the challenge packet again and generate a new ICMP error message. This new message will contain the header of the challenge packet, including the IP Option with the nonce.

5. Validate: Host A receives the new error message, extracts the nonce, and looks it up in its cache. If the nonce is found and matches the one stored for that flow, the error is deemed authentic. Host A can now safely process the error (e.g., update its PMTU for that flow).

4.1. The Flaw in the Strawman: Denial-of-Service Vulnerability

The stateful approach, while functionally correct, introduces a critical security vulnerability: state-exhaustion Denial-of-Service (DoS) attacks.

An attacker can exploit the behavior described in Step 2. The attacker can send a high volume of forged ICMP error messages to Host A, each for a different (and possibly non-existent) flow. For each of these forged messages, Host A is forced to perform the following actions:

* Generate a cryptographically secure random number.

* Allocate memory for a cache entry to store the nonce, flow identifiers, and a timer.

* Manage the timer for this entry.

By sending thousands of such messages per second, the attacker can force the victim host to exhaust its memory or CPU resources dedicated to managing the challenge cache. This is a classic state-exhaustion DoS attack, analogous to a TCP SYN flood. A solution that opens up such a significant DoS vector is not suitable for deployment on the public Internet.

One might consider rate-limiting the processing of incoming ICMP error messages as a potential mitigation. However, this is insufficient. The fundamental problem lies in the host’s inability to distinguish legitimate ICMP errors from forged ones before expending processing resources. As a result, an attacker can easily saturate the rate limit with fake messages, effectively preventing the host from receiving and responding to genuine network errors. This turns the rate limit mechanism into an amplifier of denial, suppressing critical feedback from the network while consuming system resources. Therefore, any viable defense must allow for early authentication of ICMP messages before the host allocates significant per-message state.

5. The Proposed Solution: A Stateless Challenge-Confirm Mechanism

To solve the DoS vulnerability, we must remove the requirement to store per-challenge state. The solution is inspired by TCP SYN-Cookies [RFC4987], where state is not stored but is instead encoded cryptographically and later re-computed for validation.

5.1. Core Principle: Eliminating State with Cryptographic Computation

Instead of generating and storing a random nonce, the host computes a deterministic nonce on demand. This nonce is a cryptographic hash of information that defines the flow, combined with a secret key known only to the host.

Challenge Nonce = F(secret_key, src_IP, dest_IP, [other_flow_info])

* secret_key: A high-entropy secret value held by the host's operating system. This key MUST be rotated periodically (e.g., every few minutes) to limit the impact of any potential key compromise and to mitigate replay attacks.

* F: A keyed-hash function, such as HMAC-SHA256, truncated to the size of the nonce field.

With this approach, a nonce can be generated when needed (for an outgoing challenge) and verified later (on a returning confirmation) by simply re-computing it. There is no need to store it in a cache.

5.2. Challenge-Confirm Mechanism

The refined, stateless process is as follows:

* Receive and Validate Error: Host A receives an ICMP error message. It first validates the embedded header's 4-tuple against its list of active sockets/connections. If no matching socket exists, the message is silently discarded. No state is created.

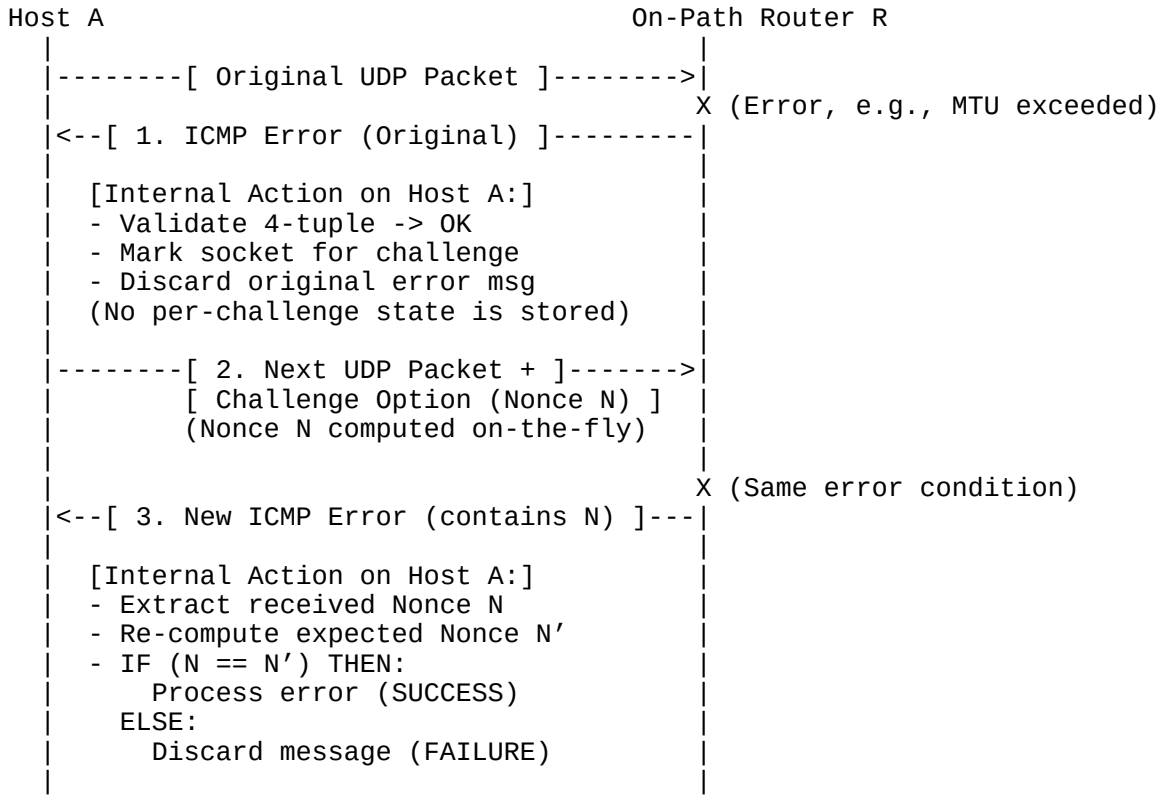
* Mark Flow for Challenge: If a matching socket is found, Host A does not create new state. Instead, it sets a simple flag on the existing socket control block, marking it as "requires challenge". The initial ICMP error is then discarded.

* Issue Computed Challenge: The next time the application sends the retransmitted packet on this marked socket, the networking stack intercepts it. It computes the challenge nonce using the secret key and the packet's flow information. This nonce is placed in a Challenge-Confirm IP Destination Option, and the packet is sent.

* Receive and Verify Confirmation: If a legitimate on-path node returns a new ICMP error, it will contain the challenge packet. Host A receives this new error, extracts the embedded nonce, and recomputes the expected nonce using the same secret key and flow information.

* Process or Discard: If the received nonce matches the re-computed one, the error is authentic, and Host A can act on it. If it does not match, the message is a forgery or is stale, and it is discarded.

This flow achieves the same anti-spoofing goal as the strawman but without creating state for unverified messages, thus defeating the DoS attack. Figure 1 illustrates the complete interaction, including both the legitimate process and how an off-path attacker's attempts are thwarted.



5.3. Protocol-Specific State Management

The mechanism for "marking a flow" in Step 2 is lightweight and transport-specific.

UDP: Upon receiving a validatable ICMP error, the host sets a flag on the corresponding UDP socket’s control block.

TCP: While TCP has its own protections, this mechanism can supplement it. A flag can be set on the TCB.

ICMP: For connectionless protocols like ICMP Echo, which lack a socket state, a probabilistic, fixed-size data structure like a Count-Min Sketch or Bloom Filter should be used. On Error Reception: The host hashes a flow identifier (e.g., source IP, destination IP, ICMP Identifier) and increments the corresponding counter(s) in the sketch. On Packet Transmission: When sending a new ICMP packet, the host queries the sketch. If the query indicates this flow has likely received a recent error, it attaches the computed challenge. This probabilistic approach ensures that state remains bounded, preventing DoS attacks against ICMP-based applications.

5.4. Challenge-Confirm Option

To support the Challenge-Confirm mechanism, this document defines a new Challenge-Confirm Option. The challenge packet for a received ICMP error message containing a stateless protocol payload includes the following option (as shown in Figure 3) in the IP header. Similarly, the ICMP error message triggered in response to this challenge packet should also include the same option in the header of the embedded IP challenge packet (as shown in Figure 4).

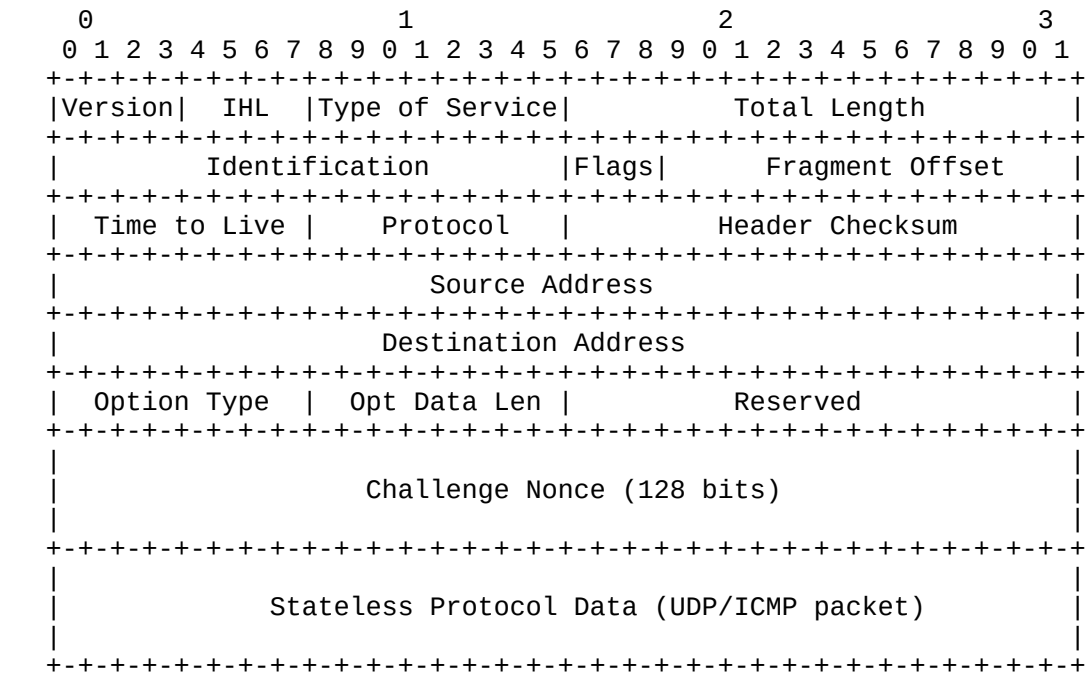


Figure 3: The Challenge Packet Header with Random Number in IP Options

The fields in Challenge-Confirm Option are defined as follows:

* *Option Type*: 8-bit identifier for the challenge-confirm option. The final value requires IANA assignment.

* *Opt Data Len*: 8-bit unsigned integer specifying the length of the option data field in bytes.

* *Reserved*: 16-bit field reserved for future use. MUST be set to zero on transmission and ignored on reception.

* *Challenge Nonce*: 128-bit random number generated according to [RFC4086] requirements.

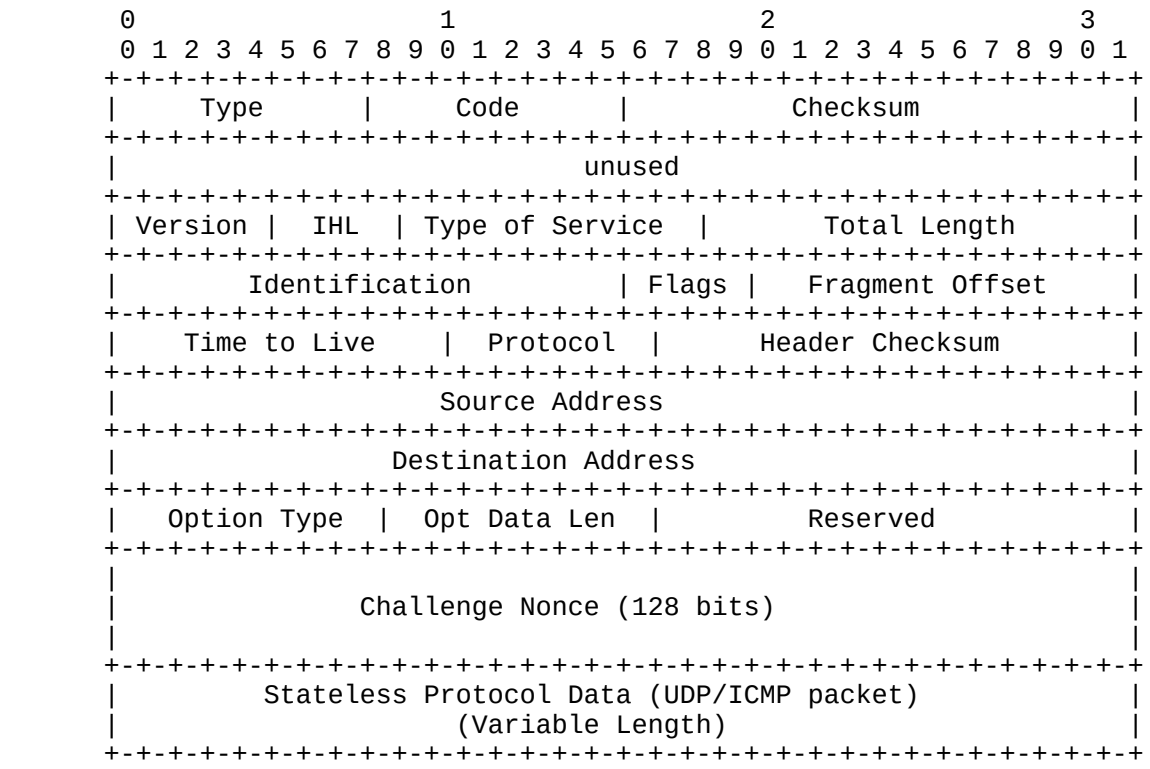


Figure 4: New ICMP Error Responding to the Challenge Packet

6. Security Considerations

The proposed enhancements aim to bolster ICMP security by addressing specific vulnerabilities related to message authentication. Key security aspects include:

* *Authentication Strength*: The security of the authentication depends on the unguessability of the computed nonce, which is guaranteed by the use of a strong keyed-hash function and a secret key with sufficient entropy [RFC4086].

* *Denial of Service (DoS) Resistance*: This is the principal security advantage over stateful designs. The mechanism is resilient to state-exhaustion attacks because: 1. It creates no state for ICMP errors that do not correspond to an existing, active transport-layer socket. 2. For valid flows, the state added is minimal (a flag) or probabilistically bounded (a sketch), preventing uncontrolled resource consumption.

* *Replay Attack Mitigation*: The periodic rotation of the secret_key provides the primary defense against replay attacks. A captured nonce-confirmation pair will become invalid after the key is changed. The rotation interval presents a trade-off between security and the maximum legitimate round-trip time for a challenge-confirm exchange.

* *Backward Compatibility*: The mechanism is fully backward-compatible. Hosts not implementing this specification will ignore the Option as per [RFC792]. Intermediate routers are unaffected. Only end hosts wishing to enhance their security need to implement the changes.

7. IANA Considerations

The Challenge-Confirm Option Type should be assigned in IANA's "IPv4 Option Type field" registry [RFC2780].

This draft requests the following IP Option Type assignments from the IP Option Numbers registry in the Internet Protocol (IP) Parameters registry group (https://www.iana.org/assignments/ip-parameters).

Copy	Class	Number	Value	Name	Reference
TBD	TBD	TBD	TBD	TBD	This draft

Table 1

8. Normative References

[CCS2020IPID] Feng, X., Fu, C., Li, Q., Sun, K., and K. Xu, "Off-path TCP exploits of the mixed IPID assignment", ACM Conference on Computer and Communications Security (CCS) , 2020.

[NDSS2022MTU] Feng, X., Li, Q., Sun, K., Xu, K., Liu, B., Zheng, X., Yang, Q., Duan, H., and Z. Qian, "PMTUD is not Panacea: Revisiting IP Fragmentation Attacks against TCP", Network and Distributed System Security Symposium (NDSS) , 2022.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/rfc/rfc1122>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC2780] Bradner, S. and V. Paxson, "IANA Allocation Guidelines For Values In the Internet Protocol and Related Headers", BCP 37, RFC 2780, DOI 10.17487/RFC2780, March 2000, <https://www.rfc-editor.org/rfc/rfc2780>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/rfc/rfc4086>.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <https://www.rfc-editor.org/rfc/rfc4987>.

[RFC792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/rfc/rfc792>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/rfc/rfc9293>.

[SP2023MITM] Feng, X., Li, Q., Sun, K., Yang, Y., and K. Xu, "Man-in-the-middle attacks without rogue AP: When WPAs meet ICMP redirects", IEEE Symposium on Security and Privacy (SP) , 2023.

[USENIXSECURITY2023ICMP] Feng, X., Li, Q., Sun, K., Qian, Z., Fu, C., Zhao, G., Kuang, X., and K. Xu, "Off-Path Network Traffic Manipulation via Revitalized ICMP Redirect Attacks", USENIX Security Symposium (Security) , 2023.

Acknowledgments

The authors would like to thank the IETF community, particularly members of the INT-AREA working groups, for their valuable feedback and insights during the development of this proposal.

Authors' Addresses

Ke Xu
Tsinghua University & Zhongguancun Laboratory
Beijing
China
Email: xuke@tsinghua.edu.cn

Xuewei Feng
Tsinghua University
Beijing
China
Email: fengxw06@126.com

Yuxiang Yang
Tsinghua University
Beijing
China
Email: yangyx22@mails.tsinghua.edu.cn

Qi Li
Tsinghua University & Zhongguancun Laboratory
Beijing
China
Email: qli01@tsinghua.edu.cn