

Internet Area Working Group
Internet-Draft
Intended status: Informational
Expires: 1 January 2026

K. Xu
Tsinghua University & Zhongguancun Laboratory
X. Feng
Tsinghua University
A. Wang
Southeast University
30 June 2025

Enhancing ICMPv6 Error Message Authentication Using Challenge-Confirm
Mechanism

draft-xu-intarea-challenge-icmpv6-01

Abstract

The Internet Control Message Protocol for IPv6 (ICMPv6) is essential for network diagnostics but is vulnerable to off-path spoofing attacks, especially when error messages relate to stateless transport protocols like UDP. An attacker can forge these messages to degrade performance or enable Man-in-the-Middle 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 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 ICMPv6 error messages while inherently resisting both off-path spoofing and state-exhaustion DoS attacks, thus improving the robustness of ICMPv6.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 1 January 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Problem Statement	4
3.1. Source-Based Blocking Ineffectiveness	4
3.2. Authentication Evasion based on Embedded Packet State . .	4
3.2.1. Stateful Embedded Packets (e.g., TCP)	4
3.2.2. Stateless Embedded Packets (e.g., UDP, ICMPv6) . . .	5
4. A Strawman Proposal: Stateful Challenge-Confirm	5
4.1. The Flaw in the Strawman: Denial-of-Service Vulnerability	6
5. The Proposed Solution: A Stateless Challenge-Confirm Mechanism	7
5.1. Core Principle: Eliminating State with Cryptographic Computation	7
5.2. Challenge-Confirm Mechanism	7
5.3. Protocol-Specific State Management	9
5.4. Challenge-Confirm Option	9
6. Security Considerations	13
7. IANA Considerations	13
8. References	14
8.1. Normative References	14
8.2. Informative References	15
Acknowledgments	15
Authors' Addresses	15

1. Introduction

The Internet Control Message Protocol for IPv6 (ICMPv6) serves as the cornerstone of operational signaling in IPv6 networks. It performs critical functions such as Path MTU Discovery [RFC8201], Neighbor Discovery [RFC4861], and reporting errors encountered during packet processing [RFC4443]. However, the legitimate verification of ICMPv6 error messages is inherently vulnerable by design. To enable senders to correlate error reports with the original packets for effective network diagnostics, ICMPv6 error messages, as specified in [RFC4443], 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 ICMPv6, 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 ICMPv6 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 ICMPv6 "Packet Too Big" messages, embedding stateless protocols like UDP or ICMP Echo Reply, to lower hosts' Path MTU to the IPv6 minimum of 1280 bytes [RFC8200], disrupting network throughput and latency-sensitive applications like video conferencing. This manipulation also simplifies off-path TCP hijacking [Feng2021]. Additionally, attackers can exploit forged ICMPv6 Redirect messages to tamper with a victim's gateway, enabling Man-in-the-Middle (MitM) attacks. Even with WPA/WPA2/WPA3 security, attackers can impersonate legitimate APs, bypass encryption, and hijack traffic [Feng2023]. These diverse attack vectors starkly underscore the critical and urgent necessity for robust authentication mechanisms in ICMPv6 for error message processing.

This document explores how to securely authenticate these ICMPv6 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 ICMPv6 specifications have inherent limitations that allow off-path attackers to forge ICMPv6 error messages, undermining network security and reliability. The primary issues are:

3.1. Source-Based Blocking Ineffectiveness

Certain ICMPv6 error messages, such as Packet Too Big messages, can originate from any intermediate router along the packet's path. This ubiquity makes source-based blocking ineffective, as legitimate messages can come from multiple sources.

3.2. Authentication Evasion based on Embedded Packet State

Although [RFC4443] stipulates that "Every ICMPv6 error message (type < 128) MUST include as much of the IPv6 offending (invoking) packet (the packet that caused the error) as possible without making the error message packet exceed the minimum IPv6 MTU", the inherent characteristics of the embedded packet protocol directly influence the difficulty of authenticating ICMPv6 error messages and their overall security strength.

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

When attackers embed stateful protocol packets, such as TCP segments, in forged ICMPv6 error messages, receivers can theoretically utilize the inherent state information of the TCP protocol for a certain degree of verification. The TCP protocol establishes and maintains state between communicating parties through sequence numbers, acknowledgment numbers, and ports. These connection-based TCP state information are difficult for attackers to accurately guess. Receivers can attempt to verify whether these connection-specific secret information in the embedded TCP header matches their maintained TCP connection state, thereby judging the authenticity of the ICMPv6 error message [RFC5927].

3.2.2. Stateless Embedded Packets (e.g., UDP, ICMPv6)

In contrast to stateful TCP, when attackers embed stateless protocol packets, such as UDP or ICMPv6 messages, in forged ICMPv6 error messages, receivers lose the ability to perform effective state verification. UDP and ICMPv6 protocols are inherently designed as stateless protocols, where the source does not maintain any session state information. The UDP or ICMPv6 messages embedded in ICMPv6 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 ICMPv6 error messages based on the embedded stateless packet header. This lack of state verification greatly reduces the authentication strength of ICMPv6 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 ICMPv6 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 ICMPv6 error message (e.g., Packet Too Big) 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 that flow to Host B. This packet includes the nonce in a new IPv6 Destination Option.
4. **Receive Confirmation:** If router R is legitimately on the path and the error condition persists, it will drop the challenge packet and generate a new ICMPv6 error message. This new message will contain the header of the challenge packet, including the IPv6 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 ICMPv6 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 = $F(\text{secret_key}, \text{src_IP}, \text{dest_IP}, [\text{other_flow_info}])$

- * `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 ICMPv6 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 ICMPv6 error is then discarded.
- * **Issue Computed Challenge**: The next time the application sends a 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 IPv6 Destination Option, and the packet is sent.

- * **Receive and Verify Confirmation:** If a legitimate on-path node returns a new ICMPv6 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.

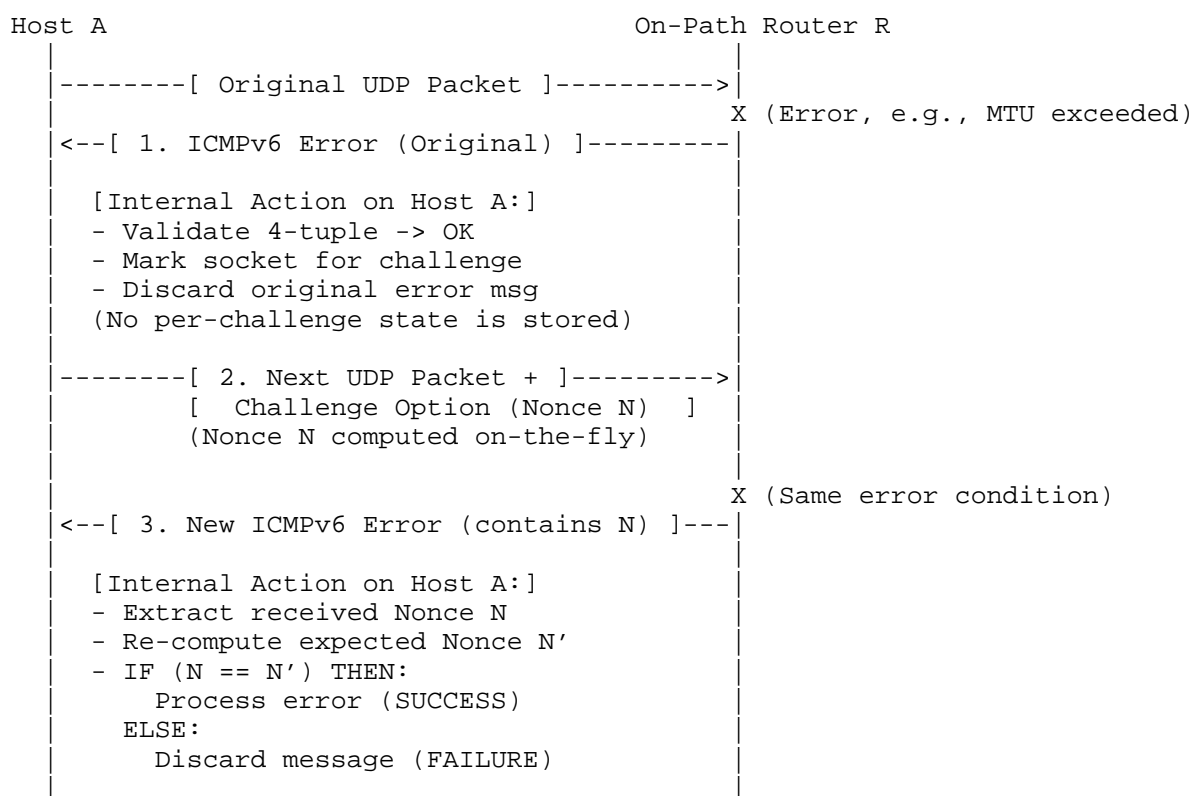


Figure 1: Challenge-Confirm Mechanism

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 ICMPv6 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 Sketch or Bloom Filter should be used. On Error Reception: The host hashes a flow identifier (e.g., source IP, destination IP, ICMPv6 Identifier) and increments the corresponding counter(s) in the sketch. On Packet Transmission: When sending a new ICMPv6 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 ICMPv6 error message containing a stateless protocol payload includes the following option (as shown in Figure 2) in the IPv6 header. Similarly, the ICMPv6 error message triggered in response to this challenge packet should also include the same option in the header of the embedded IPv6 challenge packet (as shown in Figure 3).

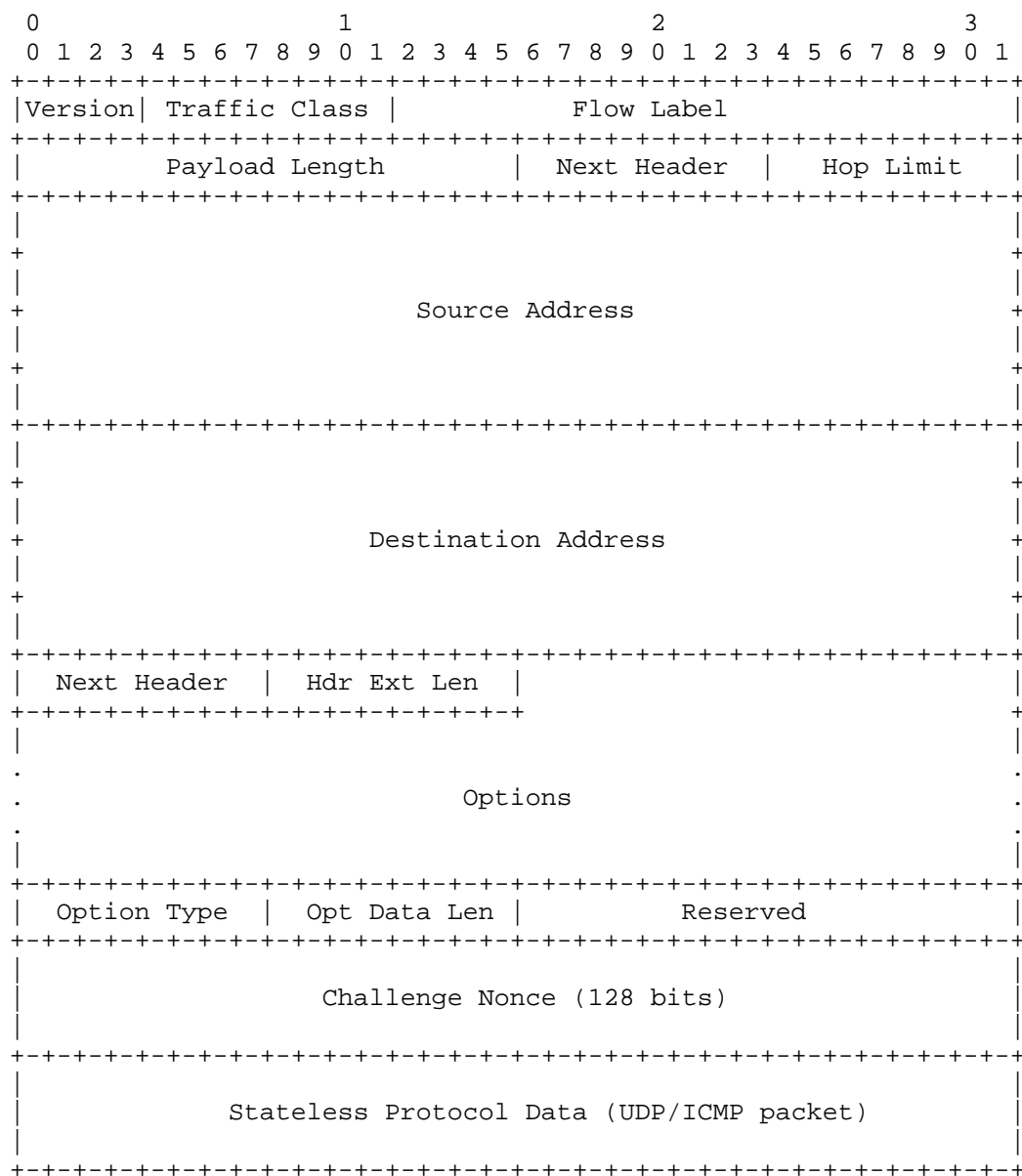


Figure 2: The IPv6 Challenge Packet with Challenge-Confirm Option

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 computed.



Figure 4: New ICMPv6 Error Responding to the Challenge Packet

6. Security Considerations

The proposed enhancements aim to bolster ICMPv6 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 ICMPv6 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 Destination Option as per [RFC8200]. 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 "Destination Options and Hop-by-Hop Options" registry [RFC2780].

This draft requests the following IPv6 Option Type assignments from the Destination Options and Hop-by-Hop Options sub-registry of Internet Protocol Version 6 (IPv6) Parameters (<https://www.iana.org/assignments/ipv6-parameters/>).

Hex Value	Binary Value	Description	Reference
	act chg rest		
TBD	00 0 -		This draft

Table 1

8. References

8.1. Normative References

- [Feng2021] Feng, X., Li, Q., Sun, K., Fu, C., and K. Xu, "Off-path TCP hijacking attacks via the side channel of downgraded IPID", IEEE/ACM transactions on networking , 2021.
- [Feng2023] 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.
- [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>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/rfc/rfc4443>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/rfc/rfc4861>>.

- [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>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/rfc/rfc8200>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/rfc/rfc8201>>.
- [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>>.

8.2. Informative References

- [RFC5927] Gont, F., "ICMP Attacks against TCP", RFC 5927, DOI 10.17487/RFC5927, July 2010, <<https://www.rfc-editor.org/rfc/rfc5927>>.

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

Ao Wang
Southeast University
Nanjing
China
Email: wangao@seu.edu.cn