Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 11 July 2026

Z. Luo, Ed.
H. Yan
CMCC

7 January 2026

Available Session Recovery Protocol
draft-cmcc-asrp-03

Abstract

This document describes an experimental protocol named the Available Session Recovery Protocol (ASRP). The protocol aims to optimize high-availability network cluster architectures, providing a superior cluster high-availability solution for stateful network services such as load balancing and Network Address Translation (NAT). ASRP defines the procedures for session backup and recovery, along with the message formats used in interactions, enabling efficient and streamlined session state management.

Unlike traditional high-availability technologies that back up session states within the cluster, the core innovation of ASRP lies in distributing state information to clients or servers. This approach offers multiple advantages, significantly enhancing the cluster’s elastic scaling capability; supporting rapid recovery from single-point or even multi-point failures; reducing resource redundancy by eliminating centralized backup nodes; and substantially simplifying cluster implementation complexity.

The ASRP protocol provides network clusters with ultimate elastic scalability, facilitating the implementation and deployment of large-scale elastic network clusters.

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 11 July 2026.

Copyright Notice

Copyright (c) 2026 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

Traditional high-availability network clusters based on primary-backup architecture rely on session-state synchronization between the primary and backup nodes. Although functionally complete, such architectures face challenges in the cloud era, including insufficient flexibility for elastic scaling, resource redundancy, and high implementation complexity. To address these issues, the Elastic Stateful Cluster has been proposed.

Elastic Stateful Cluster is a highly available network service cluster composed of multiple coordinated nodes, where the number of nodes can be elastically scaled in or out. It provides stateful network services such as load balancing (SLB) and network address translation (NAT) to external users. To achieve elastic scaling, traditional Elastic Stateful Clusters adopt a fast-path/slow-path design, separating session management from packet forwarding. This enables excellent elastic scalability at the fast-path node layer.

1.1. Traditional Path Elastic Stateful Cluster

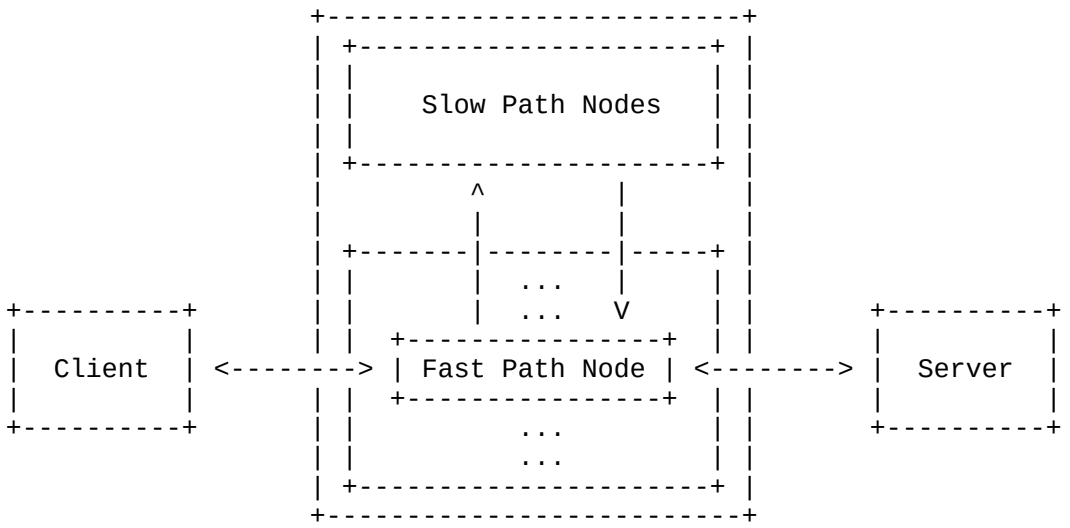

Figure 1: Fast/Slow Path Elastic Stateful Cluster

The slow-path nodes are responsible for session creation and session synchronization, while the fast-path nodes handle high-speed packet forwarding. When a fast-path node fails, external traffic is automatically switched to other healthy nodes, ensuring continuous service availability. The drawback of Elastic Stateful Clusters with this architecture is the limited elastic scalability of the slow-path layer. The slow-path nodes must implement complex session-synchronization mechanisms internally. A typical implementation can be found in the AWS Hyperplane NFV platform.

1.2. ASRP Elastic Stateful Cluster

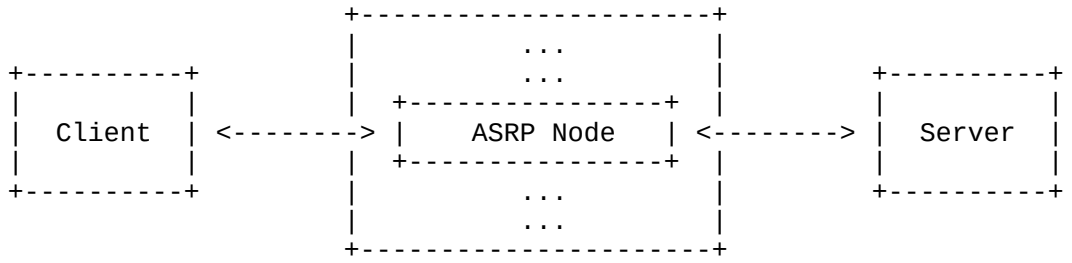

Figure 2: ASRP Elastic Stateful Cluster

The Available Session Recovery Protocol (ASRP) proposes an innovative high-availability solution designed to build a more concise, efficient, and elastic architecture for stateful services. Its core idea is to disruptively distribute session state information to the endpoints of a session (client or server). The backup state is strictly synchronized with the lifecycle of the real session-created as the session is established and cleared upon its termination. By eliminating the need for independent keep-alive and timeout mechanisms, this design ensures the real-time availability of backup information. Based on this mechanism, network nodes within a cluster (e.g., load balancers, NAT devices), when facing failures or during cluster scaling, can quickly rebuild the complete session state directly from the endpoints. This logically achieves "stateless" network nodes (ASRP node).

To achieve this goal, ASRP defines corresponding session backup and recovery mechanisms. The protocol cleverly utilizes the original data packets carrying business traffic (in-band transmission) to convey session state information (for example, embedding protocol messages into the payload of a TCP three-way handshake [RFC9293]). This avoids the overhead of additional control packets for state synchronization in the vast majority of cases. Its implementation shares similarities with TCP Fast Open (TFO, [RFC7413]) and remains completely transparent to upper-layer applications.

In summary, the ASRP protocol, by focusing on endpoint backup and rapid recovery of session state, effectively guarantees session consistency and service continuity for applications on network nodes within a cluster. In an elastic stateful cluster built upon ASRP, network nodes possess the characteristic of being atomic and mutually independent-no communication is required between nodes, and no session synchronization is needed within the cluster. This fundamental design significantly enhances the cluster’s ability to scale elastically, supports rapid recovery from single-point and even multi-point failures, and reduces resource redundancy and system implementation complexity by eliminating centralized backup nodes.

Consequently, the ASRP protocol is particularly suited for network environments requiring frequent elastic scaling, pursuing extremely high resource utilization, and demanding high service stability. It provides a robust solution for the deployment and operation of large-scale, highly elastic network clusters.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Protocol Overview

3.1. Two Operational Modes

For the ASRP protocol to function properly, all network nodes within the cluster must first deploy service programs that support the ASRP protocol. Additionally, the clients or servers responsible for backing up sessions must deploy EBPF modules or kernel modules that support the ASRP protocol. Whether these modules are deployed on the client side or the server side corresponds to the two operational modes of the ASRP protocol, respectively:

3.1.1. Passive(PSV) Mode

In Passive (PSV) mode, the network nodes and the servers are typically located within the same trusted network domain (e.g., inside a data center). The network nodes back up session state information to the servers and recover sessions from the servers when needed. This mode requires both the network nodes and the servers to support the ASRP protocol. Typical application scenarios include traditional load balancers that provide services through a Virtual IP (VIP), or NFV load balancing network elements offering cloud load balancing services.

3.1.2. Active(ACT) Mode

In Active (ACT) mode, the network nodes are typically located within the same trusted network domain as the clients (e.g., a corporate intranet). The network nodes back up session state information to the clients and recover sessions from the clients when needed. This mode requires both the network nodes and the clients to support the ASRP protocol. Typical application scenarios include Source Network Address Translation (SNAT) scenarios (such as internal network devices accessing the internet through an SNAT gateway) or NFV SNAT

network elements providing cloud SNAT services.

3.2. Two Routing Behaviors

3.2.1. Symmetric Routing

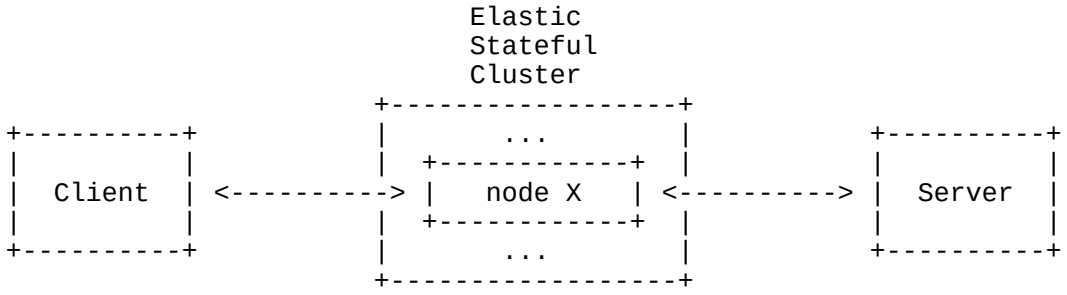

Figure 3: Symmetric Routing

Symmetric routing refers to a path type in which the bidirectional traffic of the same session between a client and a server is always routed to the same node within the cluster. This path consistency is the foundation for the proper functioning of stateful network services (such as NAT [RFC4787] [RFC5508] and firewalls), ensuring that a single node maintains and processes the complete session state information.

Typical examples that demonstrate symmetric routing include,

1. Active-Standby High Availability Architecture

   In this path type, all traffic for a specific session is always handled and maintained by the primary node (e.g., NAT mapping tables, firewall session states). The standby node remains on standby and only takes over when the primary node fails. This architecture inherently ensures symmetric routing of traffic to the primary node.

2. Stateful Load Balancing Cluster with a "Same-Source-Same-Destination" Mechanism

   In this modern extended architecture, network devices (such as OVS or routers) use a "same-source-same-destination" policy to ensure that all packets belonging to the same connection are directed to the same load balancing node, thereby maintaining symmetric routing in a distributed environment.

3.2.2. Asymmetric Routing

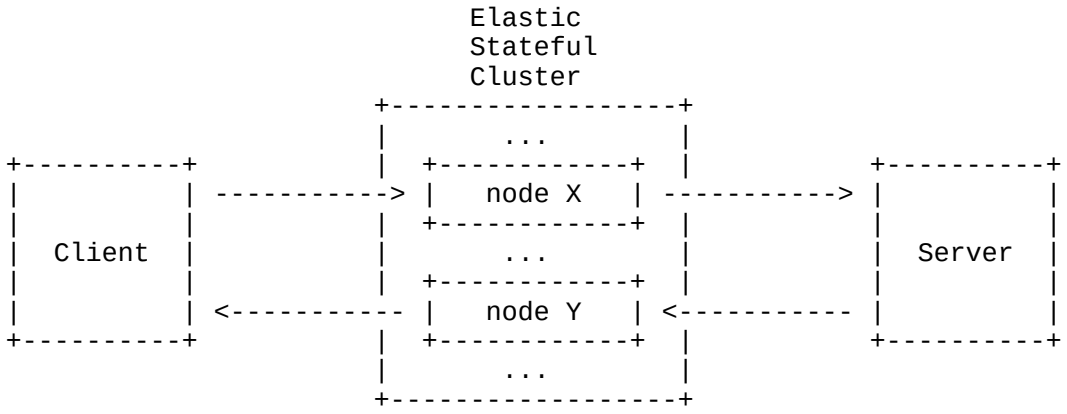

Figure 4: Asymmetric Routing

The bidirectional traffic of the same session may be routed to different nodes within the cluster. Specifically, requests sent from the client to the server may be processed by node X, while responses sent from the server to the client may be processed by node Y. This path inconsistency requires two or more nodes to collaboratively maintain the state information of the same session, posing new challenges to the failure recovery mechanisms of stateful service clusters.

In cloud network environments, asymmetric routing is an extremely common and often default phenomenon. Taking NFV element clusters that require elastic scaling as an example, one of the core goals of their architectural design is to allow nodes to independently and flexibly scale horizontally. In such distributed architectures, without deploying specific traffic steering strategies like "same-source-same-destination," underlying network devices (such as switches or load balancers) typically distribute traffic naturally and evenly across multiple available nodes based on mechanisms like ECMP (Equal-Cost Multi-Path [RFC2991], [RFC2992]), thereby commonly forming asymmetric routing.

3.3. Protocol Message

ASRP achieves distributed backup and on-demand recovery of session state information by exchanging specific protocol messages among clients, servers, and network nodes (such as load balancers and NAT devices). In load balancing scenarios, session states are backed up to individual servers; in Source NAT (SNAT) scenarios, session states are backed up to individual clients.

3.3.1. New Session Message (NS)

Generated by the network node, it is used to send the initial state information of a session to the designated client (in ACT mode) or server (in PSV mode) for backup when a new session is created. The NS message can be inserted into the original service packet for transmission or independently encapsulated and transmitted.

3.3.2. Hello Session Message (HS)

Generated by the client, it is used in ACT mode to declare its support for the ASRP protocol to the network node and to trigger the network node to return an NS message to complete session backup. The message also supports in-band or independent transmission.

3.3.3. Query Session Message (QS)

Generated by the network node, it is used to query the backup party (client or server) for session status when a packet is received from the end where the session is backed up but cannot match a local session. This message is typically transmitted by modifying and echoing the original packet.

3.3.4. Recover Session Message (RS)

Generated as a response to the QS message by the client or server holding the backup, it contains the state information required to recover the session. The network node parses the RS message and reconstructs the local session, thereby achieving failure recovery.

3.3.5. Encap NS/QS/RS Message (ENS/EQS/ERS)

When a packet received by the network node originates from a client or server that does not hold the backup of the session, ASRP messages cannot be transmitted simply by modifying the original packet. Instead, the ASRP message must be encapsulated in a new packet for transmission. In such cases, ASRP defines a format for encapsulating NS, QS, or RS messages within UDP ([RFC0768]) packets for transmission, referred to as ENS, EQS, and ERS messages, respectively.

3.4. Session Creation/Recovery Scenarios

This section elaborates on, through a series of typical scenarios, how the ASRP protocol achieves session backup and recovery via message interaction in the event of network node failures under different operational modes. Each scenario details the involved protocol message flows and the processing steps of each entity.

3.4.1. PSV-Scenario-1: Direct Session Creation

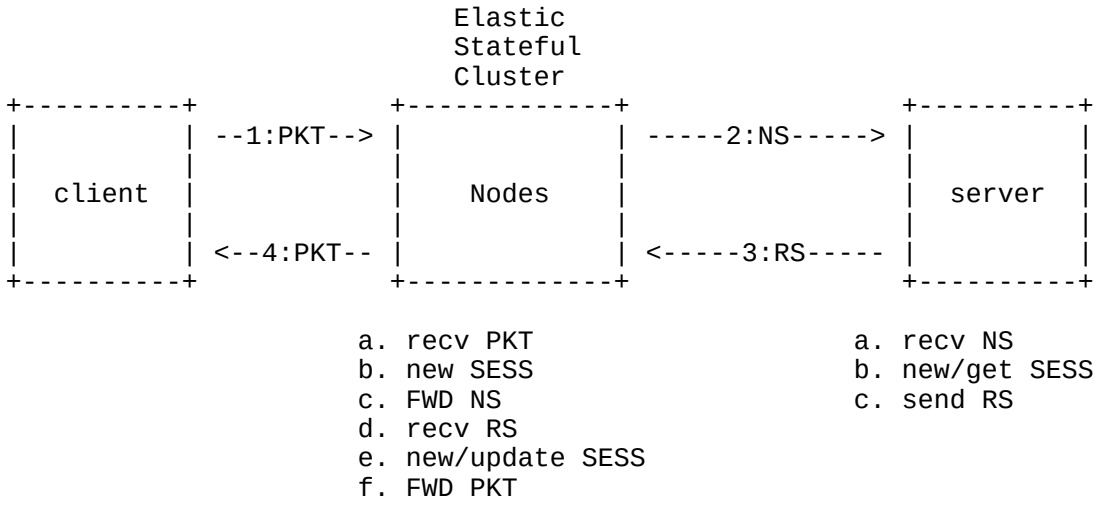

Figure 5: Direct Session Creation in PSV Mdoe

This scenario describes the direct session creation flow in Passive (PSV) mode. The most common example is the TCP SYN packet during connection establishment, which indicates that the client is initiating a new connection. In addition, after receiving a DNS request packet, the network node can also directly enter the new session creation flow. For convenience, SYN packets will be used as typical examples of such packets in the following discussion, and other similar packets will not be elaborated on further.

The processing flow is as follows:

1. Session Creation: When the network node receives a packet from a client (e.g., a TCP SYN) and finds no local session, it directly creates a new session. Subsequently, the network node generates an NS message and forwards it to the selected server.

2. Server Response: After receiving the NS message, the server creates or retrieves the corresponding session state based on its content. When the server sends a response packet, it generates an RS message and sends it back to the network node.

3. Session Synchronization and Packet Forwarding under Asymmetric Routing: Upon receiving the RS message, the network node uses the information within it to complete the establishment or update of the local session, and then forwards the packet according to the session.

Technical Notes:

NS/RS messages can be inserted into original packets (e.g., TCP SYN/ SYN-ACK) or transmitted independently. Similarly, other messages in subsequent scenarios can also be inserted into original packets or transmitted independently.

3.4.2. PSV-Scenario-2: Session Recovery for Server

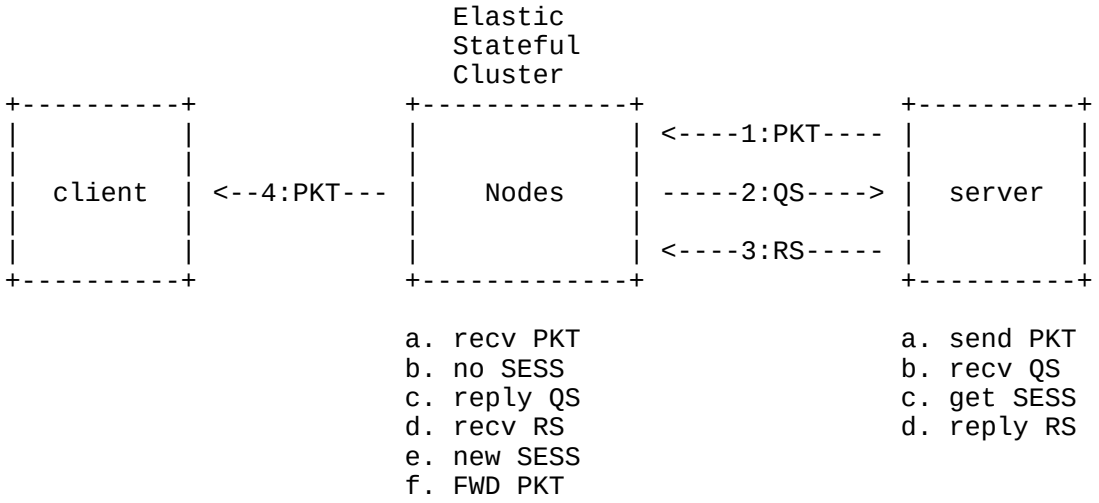

Figure 6: Session Recovery for Server in PSV Mode

This scenario describes the server packet-triggered session recovery flow in Passive (PSV) mode.

The processing flow is as follows:

1. Session Query: After receiving a packet from a server, the network node searches its local session table. If no corresponding session is found, the network node generates a QS message and sends it back to the server.

2. Server-Assisted Reply: Upon receiving the QS message, the server looks up the corresponding locally stored backup session state information based on the message content, generates an RS message, and sends it back to the network node.

3. Session Recovery: After receiving the RS message, the network node creates a new local session and then forwards the packet according to the session.

3.4.3. PSV-Scenario-3: Session Creation/Recovery

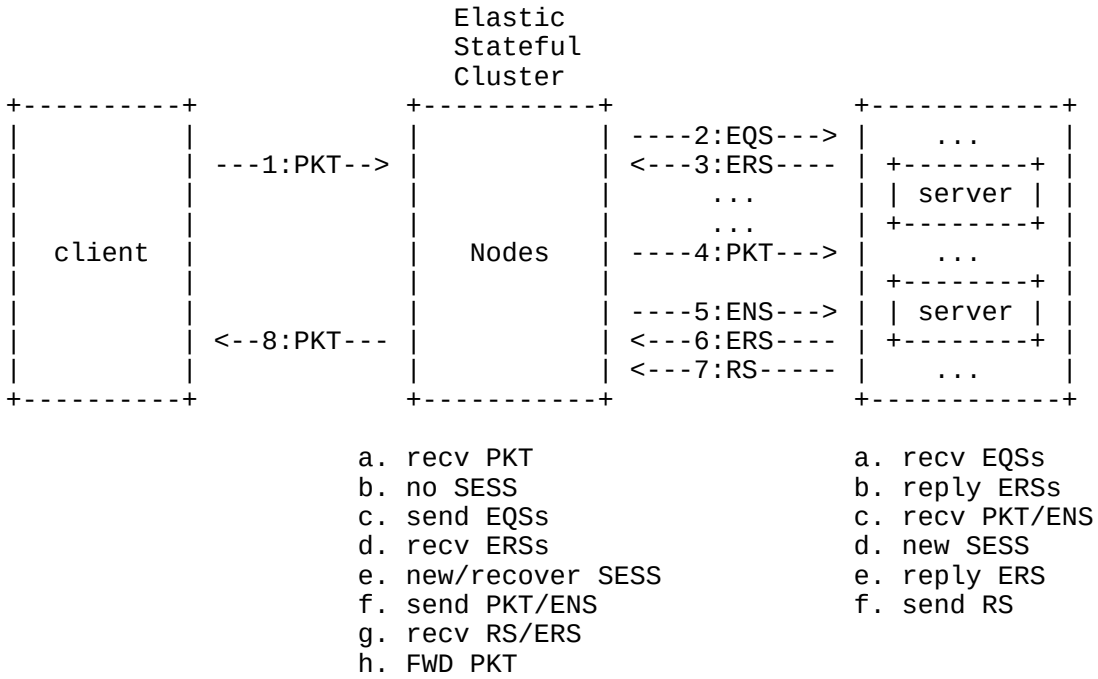

Figure 7: Session Creation/Recovery for Client in PSV Mode

This scenario describes a situation in Passive (PSV) mode where a network node receives a client packet that neither triggers the creation of a new session (e.g., a non-TCP SYN packet) nor matches any existing session entry. The network node needs to determine whether the packet belongs to an existing session to decide how to process it. In this case, the network node must use the ASRP protocol to query the session state and proceed based on the query result. ASRP relies on the cluster employing a deterministic server selection algorithm (such as a consistent hashing algorithm) to locate the target server to be queried. The Deterministic Bucket Mapping Consistent Hashing (DBMCH) algorithm is an enhanced algorithm that ensures the network node can still query the correct target server even when the number of servers changes. Its principles are detailed in Appendix A.

The processing flow is as follows:

1. Query Local Session: The network node receives an original packet from the client and searches its local session table. If no local session is found, it calculates the corresponding server for the packet using a consistent hashing algorithm or the DBMCH algorithm.

2. Query Backup Session: The network node generates EQS messages and sends them sequentially to each candidate server (typically one), querying the candidate servers for the backup session. The server returns the query result via an ERS message.

3. Process Query Results: If a session is found, the local session is recovered using the ERS message, and the original packet is forwarded. If none of the servers has a corresponding session, the process proceeds to the new session creation flow. An ENS message is sent to the server selected by the algorithm.

4. Server Creates New Session: Upon receiving the ENS message, the server creates a locally associated session and replies with an ERS message carrying no service data as an acknowledgment. In cases of asymmetric routing, the first service packet sent from the server to the client will simultaneously carry an RS message, allowing nodes along the path to recover the session.

5. Session Synchronization and Packet Forwarding under Asymmetric Routing: After receiving the RS message, the network node first recovers the local session based on the message and then forwards the packet according to the session.

Technical Notes:

Reason for using ENS/EQS/ERS: In this scenario, the IP addresses used for communication between the network node and the server may be completely different from those in the original packet. Therefore, NS/QS/RS messages need to be encapsulated within UDP packets (as ENS/EQS/ERS) to ensure routing reachability.

3.4.4. ACT-Scenario-1: Session Creation/Recovery

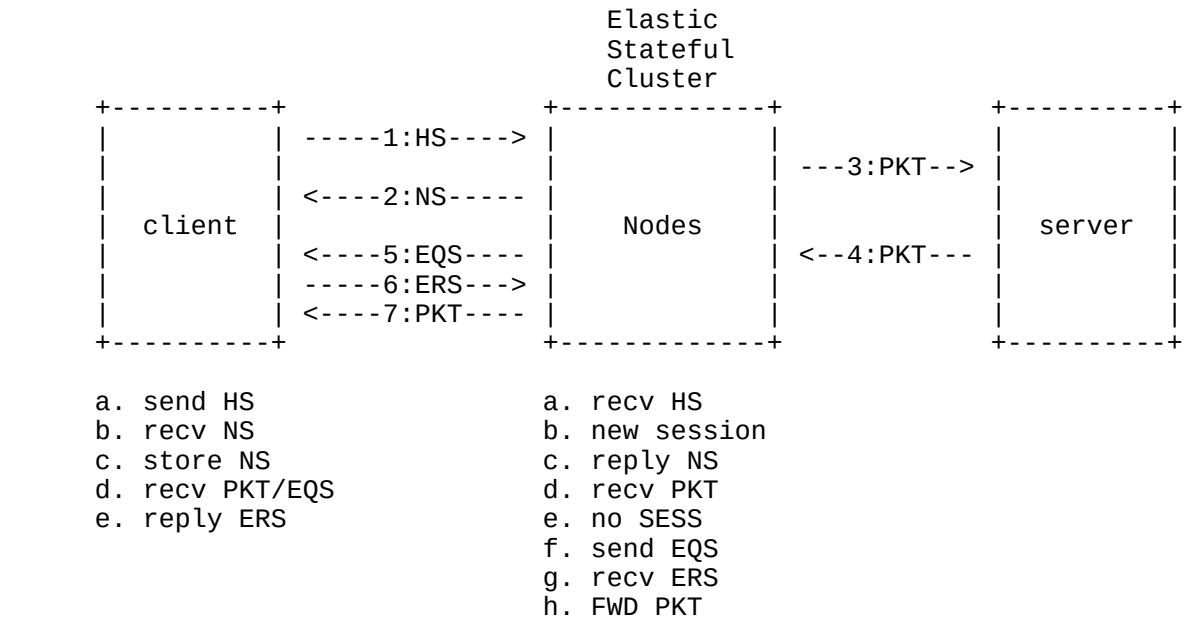

Figure 8: Session Creation/Recovery in ACT Mode

This scenario describes the session establishment process in Active (ACT) mode and the session recovery flow triggered by server-side packets.

The processing flow is as follows:

1. Session Creation: When a client initiates a new session, it first sends a packet containing an HS message to the network node, announcing its support for ASRP. Upon receiving the HS message, the network node creates a new session following the normal procedure, generates an NS message to return to the client, and forwards the original packet according to the session.

2. Processing Server Response Packets: If a matching session is found, the packet is forwarded based on that session. If no matching session is found, the network node locates the corresponding client through a mapping relationship and sends an EQS message to it.

3. Client-Assisted Recovery: After receiving the EQS message, the client replies with an ERS message to the network node.

4. Session Recovery: Upon receiving the ERS message, the network node restores the session state locally and then forwards the packet according to the session.

Technical Notes:

Fixed Mapping Mechanism: During the server packet-triggered recovery phase, the network node must be able to deterministically locate the client that backs up the session. ASRP recommends using a static, configurable mapping policy as the foundation for achieving efficient and reliable recovery. If such a mapping cannot be established, it is not advisable to use ASRP in this scenario. For SNAT services, ports can typically be used to map clients, with different clients assigned configurable, distinct port ranges. When a server packet arrives at the network node, the network node can locate the client based on the destination port.

3.4.5. ACT-Scenario-2: Session Recovery for Client

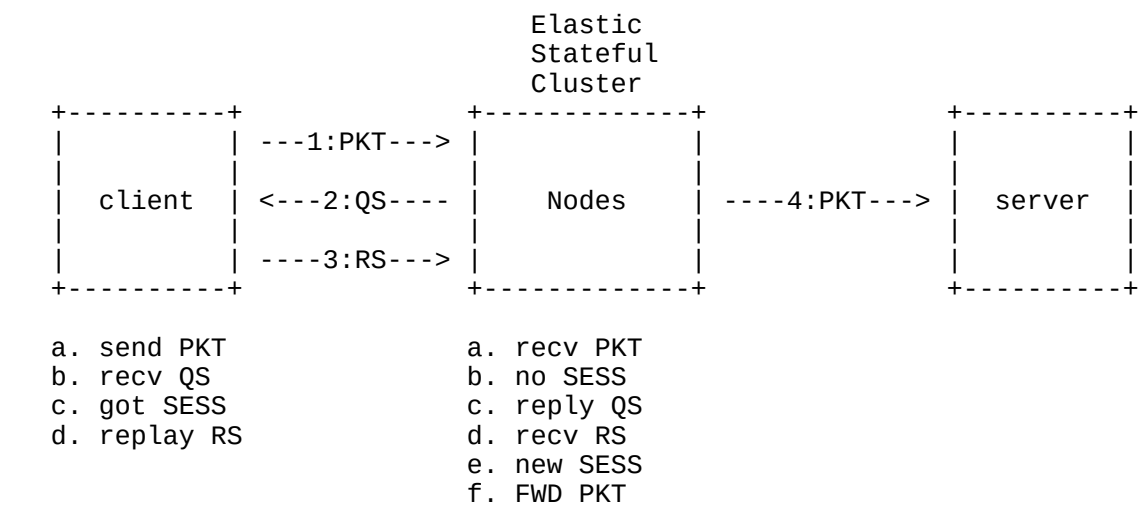

Figure 9: Session Recovery for Client in ACT Mode

This scenario describes the client packet-triggered session recovery flow in Active (ACT) mode.

The processing flow is as follows:

1. Session Query: When the network node receives a packet from a client and finds no local session, and if the packet does not contain an HS message, it generates a QS message and sends it back to the client (i.e., the originator of the packet).

2. Client-Assisted Recovery: Upon receiving the QS message, the client queries its locally stored corresponding session backup, then generates an RS message and sends it back to the network node.

3. Session Recovery: After receiving the RS message, the network node restores the session locally and then forwards the packet according to the session.

4. Protocol Details

4.1. Message Formats

The ASRP protocol defines protocol signature, multiple message types, and their associated packet formats. All messages are encoded using a TLV (Type-Length-Value) structure. All numeric fields use network byte order (big-endian).

All ASRP messages consist of the following five parts:

1. Type: 1 byte, used to uniquely identify different ASRP messages.

2. Flags: 1 byte, indicating message attributes. Two flag bits are currently defined:

   ASRP_F_ACT (0x1): If set, indicates the message belongs to ACT mode; otherwise, it belongs to PSV mode.

   ASRP_F_MSG (0x2): If set, indicates this message is transmitted independently; otherwise, it indicates the message is embedded within the original service packet for transmission.

3. Length: 2 bytes, representing the total length of the entire ASRP message.

4. Session-Tuple: Carries the address and port information of the session. Its length depends on the address type ([RFC0791] [RFC8200]):

   IPv4: Contains source IPv4 address, destination IPv4 address, source port, and destination port, totaling 12 bytes.

   IPv6: Contains source IPv6 address, destination IPv6 address, source port, and destination port, totaling 36 bytes.

5. Session-Data: A variable-length field that carries the private state information of the network node. Its specific content is implementation-defined.

4.1.1. ASRP Signature

ASRP messages are placed before the packet data. In actual transmission, not all packets carry ASRP messages, so a ASRP Signature or a new TCP option needs to be added to the packet to quickly determine whether it contains an ASRP message.

For the TCP protocol, a new TCP option can be defined in the TCP header to indicate that an ASRP message is located at the beginning of the TCP data. The newly added TCP option type is TCPOPT_ASRP(60), with a length of 2. Its format is as follows:

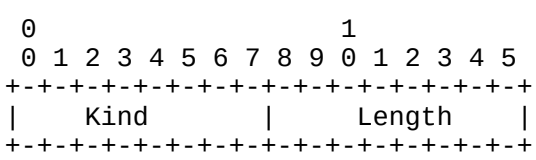

Figure 10: ASRP TCP Option

For other transport-layer protocols, similar to the Proxy Protocol, a 12-byte ASRP Signature is used at the beginning of the data: 0x0D 0x0A 0x0D 0x0A 0x00 0x0D 0x0A 0x41 0x53 0x52 0x50 0x0A

This Signature contains a CRLF pattern, a null byte, and the specific ASCII sequence ASRP. The probability of this sequence occurring in normal data streams is less than 2^{-96}, making it easy to debug and identify: during packet capture analysis, the clear ASRP identifier is visible.

After an ASRP message is inserted into a packet, ASRP Signature must be applied to the packet. This is the default operation and will not be reiterated in subsequent discussions.

4.1.2. NS Message Format

The NS message is used by the network node to back up session state information to the client or server. There are two types of NS messages, corresponding to IPv4 and IPv6 respectively.

Type Assignments:

ASRP_NS4 = 1 (IPv4)

ASRP_NS6 = 2 (IPv6)

NS (IPv4) Message Format:

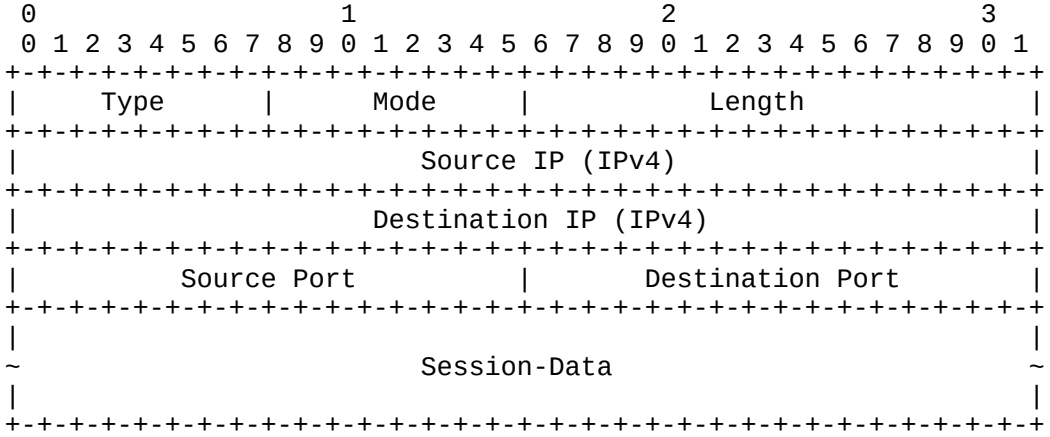

Figure 11: ASRP NS(IPv4) Message Format

NS (IPv6) Message Format:

The structure of IPv6 NS messages is the same as IPv4, with the main difference being the IP address length in the Session-Tuple field. IPv6 uses 128-bit addresses, so both Source IP and Destination IP occupy 16 octets each, expanding the entire Session-Tuple field to 36 octets.

The NS message is inserted before the original packet data. The packet carrying the NS message is referred to as an NS packet, and its format is as follows:

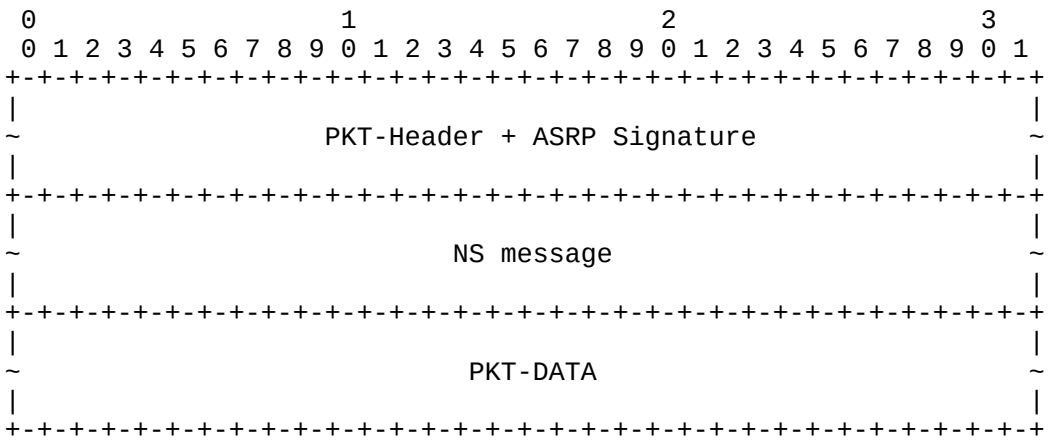

Figure 12: ASRP NS Packet Format

In PSV mode, the NS message is inserted into the original packet, and the source and destination addresses remain unchanged. The NS packet is then forwarded to the server.

In ACT mode, the NS message is inserted into a newly allocated packet. The source and destination addresses and ports of the original packet are copied and swapped, and the NS packet is returned to the client.

It is important to note that the NS message internally contains only one Session-Tuple. However, a session on a network node requires two Session-Tuples, representing the connections between the network node and the client, and between the network node and the server, respectively. To improve transmission efficiency, ASRP extracts the other Session-Tuple directly from the packet header of the NS message packet. Similarly, QS and RS messages also require extracting the other Session-Tuple from the message packet header. This will not be elaborated on further in subsequent sections.

4.1.3. HS Message Format

The HS message is sent by the client during the initial connection establishment phase to declare its support for the ASRP protocol capability to the network node.

Type Assignment:

ASRP_HS = 3

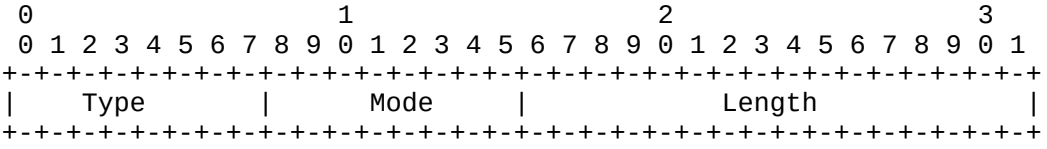

Figure 13: ASRP HS Message Format

The HS message is inserted at the beginning of the original packet data. The packet carrying the HS message is referred to as an HS packet, and its format is similar to that of the NS packet, with the only difference being the type of message it carries.

4.1.4. QS Message Format

The QS message is used by the network node to query session information.

In PSV mode, if the network node receives a packet from the server and cannot find a matching session, it uses the QS message to query the server for the session.

In ACT mode, if the network node receives a packet from the client, and the packet does not contain an HS message while no matching session is found, it uses the QS message to query the client for the session.

Type Assignment:

ASRP_QS = 4

The QS message format is identical to HS message, differing only in the type identifier.

The QS message is inserted before the original packet data, and the source and destination addresses and ports are swapped (in order to return the QS message to the client or server that backs up the session). The packet carrying the QS message is referred to as a QS packet, and its format is similar to that of the NS packet, with the only difference being the type of message it carries.

4.1.5. RS Message Format

The RS message is used to recover the network node session. There are three types of RS messages, corresponding respectively to IPv4 sessions, IPv6 sessions, and null sessions.

Type Assignments:

ASRP_RS4 = 5 (IPv4)

ASRP_RS6 = 6 (IPv6)

ASRP_RSN = 7 (null)

When the RS message type is ASRP_RS4 or ASRP_RS6, its message format is identical to the NS message of the corresponding IP version, differing only in the type identifier.

When the RS message type is ASRP_RSN, the message length is 4 bytes, indicating that the message does not contain information for session recovery (i.e., a null session). Its message format is similar to that of the HS message.

The RS message is the response to the QS message. The packet carrying the QS message is referred to as a QS packet, and its format is similar to that of the NS/HS packet, with the only difference being the type of message it carries.

4.1.6. ENS/EQS/ERS Message Format

When the packet received by the network node originates from a client or server that does not back up the session, the original destination address of the original packet may differ entirely from that of the ASRP message packet. Therefore, encapsulation is required before transmission. ASRP employs a UDP encapsulation [RFC3828] format to encapsulate NS/QS/RS message packets. The encapsulated NS/QS/RS messages are referred to as ENS/EQS/ERS messages.

The format of the ENS/EQS/ERS message packet is as follows:

IP + UDP + NS/QS/RS Packet

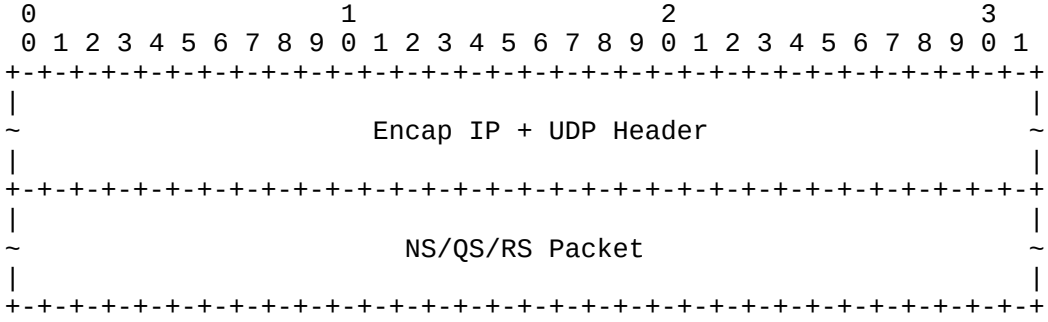

Figure 14: ASRP ENS/EQS/ERS Packet Format

The source and destination addresses in the Encap IP header ensure the normal delivery of messages between the network node and the client or server. In the Encap UDP header, the source port is adjustable, while the destination port defaults to 55555 (configurable). The receiving end uses the destination port to distinguish between ENS, EQS, and ERS packets.

4.2. Message Processing

4.2.1. ASRP Signature Detection

For the TCP protocol, the TCP option TCPOPT_ASRP(60) is checked. If this option is present, it indicates that the beginning of the data contains an ASRP message.

For other protocols, the Signature at the beginning of the data is matched first. A successful match indicates that an ASRP message follows the Signature.

In subsequent discussions regarding the processing of ASRP messages in packets, it is assumed that ASRP Signature has already been handled and will not be reiterated.

4.2.2. NS Message Processing

The NS message is generated by the network node when creating a new session and is used to back up the session to the client or the server.

In PSV mode (Figure 5), after the network node enters the direct session creation flow, it generates an NS message and sends it to the server.

In ACT mode (Figure 8), after the network node creates a session, it generates an NS message and sends it to the client.

The NS message can be inserted into a packet for transmission or transmitted independently. Upon receiving the NS message, the client or server stores the backup session state information locally and associates it with the local session.

Avoiding IP Fragmentation:

If the original packet exceeds the MTU after inserting the NS message, to avoid IP fragmentation, the NS message should be sent separately and prioritized, with the Flags field set to ASRP_F_MSG, before transmitting the original packet.

NS Message Loss Handling:

In PSV mode, the NS message is typically inserted into the first packet (e.g., TCP SYN). If the first packet is lost, retransmitting the first packet will also generate an NS message.

In ACT mode, if the NS message is lost, the network node regenerates the NS message through subsequent HS messages.

4.2.3. HS Message Processing

HS messages are generated only in ACT mode (Figure 8). When a client initiates a new session, it generates an HS message, inserts it into a packet, and sends it to the network node. After sending the HS message, the client waits for an NS message. If the NS message is not received, the client continues to insert the HS message into subsequent packets sent at certain intervals (recommended to be millisecond-level) to remind the network node to return the NS message.

When the network node receives an HS message, if no matching local session is found, it creates a session, generates an NS message, and sends it to the client. If the network node subsequently receives HS messages and matches them to a local session, it also immediately sends an NS message to the client.

Avoiding IP Fragmentation:

If the original packet exceeds the MTU after inserting the HS message, to avoid IP fragmentation, the HS message should be sent separately and prioritized, with the Flags field set to ASRP_F_MSG, before transmitting the original packet.

HS Message Loss Handling:

If the NS message is not received, the client continues to insert the HS message into subsequent packets sent at certain intervals (recommended to be millisecond-level).

4.2.4. QS Message Processing

QS messages are generated by the network node and are used to query sessions.

In PSV mode (Figure 6), when the network node receives a packet from a server and cannot find a matching session, it returns a QS message to the server to query the session.

In ACT mode (Figure 9), when the network node receives a packet from a client, cannot find a matching session, and the packet does not contain an HS message, it returns a QS message to the client to query the session.

The QS message is attempted to be inserted into the original packet, and then the source and destination addresses and ports of the original packet are swapped. This returns the QS message together with the original packet to the client or server, with the advantage that the original packet does not need to be discarded or cached.

Avoiding IP Fragmentation:

If the original packet exceeds the MTU after inserting the QS or the corresponding RS message, to avoid IP fragmentation, the QS message should be sent separately with the Flags set to ASRP_F_MSG, and the original packet can be discarded or cached.

QS Message Loss Handling:

Subsequent packets will continue to generate QS messages, continuing the attempt to recover the session.

4.2.5. RS Message Processing

RS messages are generated by the client or server and processed by the network node to recover sessions.

When a client or server receives a QS message, it searches locally for the backed-up session state information, then generates an RS message and returns it to the network node. If no corresponding session state information is found, it returns an RS message with an empty session. The method to convert a QS message packet into an RS message is to replace the QS message in the packet with an RS message and swap the source and destination addresses and ports of the packet.

After receiving a non-empty RS message, the network node uses the session information in the RS message to recover the session locally.

Avoiding IP Fragmentation:

IP fragmentation for the RS message packet should be avoided when generating the QS message.

RS Message Loss Handling:

Subsequent packets will continue to generate QS messages, continuing the attempt to recover the session.

4.2.6. ENS/EQS/ERS Message Processing

ENS/EQS messages are generated by the network node to create/query sessions, while ERS messages are generated by the client or server to recover sessions.

In PSV mode (Figure 7), when the network node receives a client packet and cannot match a session or directly create a new session, it uses EQS/ERS/ENS messages to communicate with the server, with functions similar to QS/RS/NS messages.

In ACT mode (Figure 8), when the network node receives a server packet and cannot match a session, it uses EQS/ERS messages to communicate with the client, with functions similar to QS/RS messages.

Avoiding IP Fragmentation:

If the original packet exceeds the MTU after inserting the ENS message, to avoid IP fragmentation, the ENS message should be sent separately and prioritized, followed by forwarding the original packet.

If the original packet exceeds the MTU after inserting EQS/ERS messages, to avoid IP fragmentation, the EQS/ERS messages should be sent separately with the Flags set to ASRP_F_MSG. The original packet needs to be cached and forwarded according to the session after the session is created/recovered.

ENS/EQS/ERS Message Loss Handling:

ENS message loss: The network node continuously sends ENS messages until an ERS response is received.

EQS/ERS message loss: Query timeout, delete related data and packets.

5. Security Considerations

5.1. General Defenses Against Message Forgery Attacks

The security design of the ASRP protocol is based on its typical deployment model.

Deployment Boundaries and Access Control: ASRP recommends deploying network nodes and the clients or servers that back up sessions within the same trusted internal network domain. Under this model, all ASRP protocol packets communicate within the internal address space. By implementing appropriate network segmentation (e.g., using firewall

policies or security groups) and rigorously checking the source addresses of packets, forged ASRP packets originating from untrusted external networks can be effectively prevented from reaching the target nodes.

Session Legitimacy Validation: When processing any ASRP packet that may establish a new session (e.g., packets carrying NS or RS messages), network nodes must perform basic validation according to the specific policies of the upper-layer application or service. For example, in a load balancing scenario, a node should verify whether the session points to a known and healthy server; in a NAT scenario, it should validate whether the address translation complies with predefined rules. This prevents the establishment of illegal sessions at the application level.

Internal Threat Assessment: Even if an attacker is located within the trusted network and can forge ASRP packets, the scope of impact is inherently limited. The attacker can only forge sessions where they themselves are the endpoint (e.g., impersonating a client to request recovery of a non-existent connection). Such forged sessions are indistinguishable in form from those established through normal access, do not directly endanger the security of other users or nodes, and cannot elevate the attacker’s privileges or grant access to unauthorized resources.

Trade-offs Regarding Enhanced Authentication: Theoretically, stronger authentication mechanisms could be introduced for ASRP, such as having the network node generate a random number (Cookie) during session backup and include it in the transmission, then requiring the other party to echo this Cookie during the recovery phase. However, considering that ASRP primarily operates within trusted domains and faces a limited external attack surface, introducing such mechanisms would increase protocol complexity and processing overhead. Therefore, this protocol does not recommend mandating such inter-node Cookie validation in its base design.

5.2. Mitigation Against EQS/ERS Flood Attacks

In the PSV mode’s scenario 3 (Figure 7) and the ACT mode’s scenario 1 (Figure 8), a network node may send a large number of EQS query messages to multiple backup parties upon session loss. This behavior, if maliciously exploited or resulting from a fault, could lead to flood attacks [RFC4987].

To mitigate such risks, implementers should consider the following protective measures:

Message Aggregation: When a network node needs to query multiple sessions from the same backup party (e.g., a specific server) within a short timeframe, the implementation SHOULD support aggregating multiple QS messages into a single EQS packet for transmission. This can significantly reduce the number of control packets, lowering network and processing overhead.

Rate Limiting and Traffic Shaping: Each network node MUST implement monitoring and limiting of the rate at which EQS packets are sent. A reasonable threshold (e.g., the maximum number of EQS packets allowed per second) should be established. When the rate exceeds this threshold, the node should adopt a packet discarding policy, such as:

Discarding newly arrived service packets that would trigger queries, or, discarding lower-priority pending EQS query requests.

This measure aims to prevent EQS packet floods caused by node failures, configuration errors, or malicious traffic, thereby protecting the backup parties (clients or servers) from resource exhaustion attacks. The parameters for rate limiting should be configurable to adapt to deployment environments of different scales.

State Monitoring and Alerting: Implementations are advised to log the frequency and patterns of EQS/ERS messages. Abnormally high query rates should trigger logging and system alerts, enabling administrators to promptly detect potential network issues or security attacks.

6. IANA Considerations

This document defines the Application-layer Session Recovery Protocol (ASRP), an application-layer protocol whose message types and internal identifiers are scoped to its own specification. Therefore, no registration in the "Assigned Internet Protocol Numbers" registry is required.

However, to support in-band signaling over TCP and encapsulated control messaging over UDP, the following IANA allocations are requested.

6.1. TCP Option

The ASRP protocol uses a TCP option to signal that the beginning of the TCP payload contains an ASRP message (e.g., HS, NS, or RS). This option is defined as follows:

Option Name: TCPOPT_ASRP

Option Kind: To be assigned by IANA

Length: 2 bytes

The format and semantics of this option are specified in Section ASRP Signature.

IANA is requested to assign a value from the "TCP Option Kind Numbers" registry (within the "TCP Parameters" registry) for TCPOPT_ASRP and to reference this document.

Note: Early implementations used Kind 60 for experimentation. However, this document requests a permanent, unassigned value suitable for standards-track use.

6.2. UDP Port

Encapsulated ASRP control messages (ENS, EQS, ERS) are transmitted over UDP between network nodes and servers or clients. To support interoperable testing and early deployments, this document requests a standardized UDP port assignment.

Service Name: asrp-encap

Transport Protocol: udp

Port Number: To be assigned by IANA (from the User Ports range, 1024-49151)

Description: UDP port for receiving encapsulated ASRP protocol messages (ENS/EQS/ERS).

For experimental implementations and interoperability testing prior to IANA assignment, UDP port 55555 MAY be used as a temporary default. This port falls within the dynamic/private port range (49152-65535) reserved for local or temporary use and documentation examples [RFC6335].

IANA is requested to assign a permanent port number in the "User Ports" range (1024-49151) for the "asrp-encap" service in the "Service Name and Transport Protocol Port Number Registry", with a reference to this document.

7. References

7.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

[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/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[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/info/rfc8200>.

7.2. Informative References

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <https://www.rfc-editor.org/info/rfc2991>.

[RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path Algorithm", RFC 2992, DOI 10.17487/RFC2992, November 2000, <https://www.rfc-editor.org/info/rfc2992>.

[RFC3828] Larzon, L., Degermark, M., Pink, S., Jonsson, L., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <https://www.rfc-editor.org/info/rfc3828>.

[RFC4787] Audet, F., Ed. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, DOI 10.17487/RFC4787, January 2007, <https://www.rfc-editor.org/info/rfc4787>.

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

[RFC5508] Srisuresh, P., Ford, B., Sivakumar, S., and S. Guha, "NAT Behavioral Requirements for ICMP", BCP 148, RFC 5508, DOI 10.17487/RFC5508, April 2009, <https://www.rfc-editor.org/info/rfc5508>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/info/rfc7413>.

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

Appendix A. Deterministic Bucket Mapping Consistent Hashing

A.1. Principles of the DBMCH algorithm

In passive (PSV) mode, the core reason why the ASRP protocol can recover sessions is that sessions can be stably mapped to a backend server. When needed, the ASRP protocol is used to query the backend server for the session related to the packet.

When the backend servers remain unchanged, consistent hashing algorithms can well meet the above core requirement. However, when backend servers change (such as an increase or decrease in number), consistent hashing algorithms can only guarantee that most mappings remain unchanged, while a small portion of mappings change. This causes related sessions to be unrecoverable. To solve this problem, and considering the characteristics of the ASRP protocol, the Deterministic Bucket Mapping Consistent Hashing (DBMCH) algorithm is proposed. The principles of the DBMCH algorithm are as follows:

1. Virtual Buckets: Set a fixed number N of virtual buckets. The number N (such as 65536) should be much larger than the maximum expected number of servers to ensure balance. Once set, the value of N does not change.

2. Fixed Mapping of Sessions to Buckets: Use a stable hash function to deterministically map a session to a virtual bucket in the range 0:N-1, ensuring that all packets of the same session always hit the same bucket.

3. Server List within Buckets: Each virtual bucket maintains an ordered list of servers. The first server in this list is called the bucket’s preferred server, specifically responsible for handling new connections.

4. Three Principles for Adding and Removing Backend Servers: The difference in the lengths of server lists across virtual buckets should not exceed 1; minimize the total length of all server lists; and adjust the load balance of new sessions among nodes by selecting preferred servers.

5. Safe Removal of Non-Preferred Servers from Virtual Buckets: Once all old connections mapped to "non-preferred servers" are closed, these servers can be safely removed from the corresponding virtual bucket’s list. Ideally, after a period, most virtual buckets will retain only the preferred server, causing the average list length to approach 1, which reduces query costs.

A.2. ASRP and DBMCH Work Together

1. Hash-Based Location: Hash the five-tuple of the arriving packet to determine its corresponding virtual bucket.

2. Obtain List: Retrieve the server list for that virtual bucket.

3. Query mechanism: Using ASRP’s query mechanism (EQS/ERS), sequentially query the backend servers in the server list within the virtual bucket to retrieve the backup session and restore the local session.

A.3. Estimation of the Length of Server List

The core conclusion is: The length of the server list within a virtual bucket grows slowly and has a definite upper bound. Its growth is primarily related to two factors: (1) the number of scaling operations, and (2) the ratio of the number of newly added servers to the current scale during each scaling operation.

Quantitative analysis of the following two typical scaling strategies clearly demonstrates this characteristic:

1. Incremental Scaling Strategy

   Assume an initial server count of K, and each scaling operation adds K new servers (i.e., the scale doubles each time). After 8 consecutive scaling operations, the total number of servers reaches 9K. During this process, the mathematical expectation (which can approximate the average number) of servers in any

virtual bucket is roughly 1 + 1/2 + 1/3 + ... + 1/8 approximately equal to 2.7. Therefore, the maximum number of servers per bucket is 3, and the minimum is 2.

Scale Example: This means that scaling from a small cluster of 4 servers to 36 servers, or from 32 servers to 288 servers, the maximum number of servers per bucket remains merely 3.

2. Aggressive Scaling Strategy

Assume an initial server count of K, and each scaling operation adds 8K new servers (i.e., significant expansion). After 4 scaling operations, the total number of servers reaches 33K. The mathematical expectation of servers per bucket is roughly 1 + 8/9 + 8/17 + 8/25 + 8/33 approximately equal to 2.92. Similarly, the maximum number of servers per bucket is 3, and the minimum is 2.

Scale Example: This means scaling from 4 servers to 132, or from 32 servers to 1056, the maximum number of servers per bucket still stabilizes at 3.

Implications and Guidance

The above analysis demonstrates that the DBMCH algorithm possesses excellent scalability. Even when the cluster scale grows by tens or even hundreds of times, the number of server probes required to recover any connection remains extremely limited (typically no more than 3). This theoretically ensures the efficiency of the ASRP session recovery mechanism, making it suitable for large-scale, elastically scaling cloud-native environments.

In practical operations, it is recommended to adopt relatively centralized scaling batches (such as the aggressive strategy mentioned above) to better control the average probing cost.

A.4. DBMCH Example

Assume the number of virtual buckets N = 12; initial servers are a, b, and c, then server d is added, and finally server a is removed.

A.4.1. Initial Servers

Initial servers: a, b, c

Bucket primary:	a	a	a	a	b	b	b	b	c	c	c	c
Virtual bucket:	a	a	a	a	b	b	b	b	c	c	c	c

A.4.2. Add Server

Add server: d

```
Bucket primary: a   a   a   d   b   b   b   d   c   c   c   d
Virtual bucket: a   a   a   ad  b   b   b   bd  c   c   c   cd
```

A.4.3. Remove Server

Remove server: a

```
Bucket primary: d   c   b   d   b   b   b   d   c   c   c   d
Virtual bucket: d   c   b   d   b   b   b   bd  c   c   c   cd
```

A.4.4. Safely Removing Non-primary Servers

Since new sessions always select the primary server, after some time, if all sessions mapped to b and c in buckets 7 and 11 have terminated, then b and c can be safely removed from those buckets. After removal, the number of servers in every bucket becomes 1:

```
Bucket primary: d   c   b   d   b   b   b   d   c   c   c   d
Virtual bucket: d   c   b   d   b   b   b   d   c   c   c   d
```

A.5. Advantages and Disadvantages

A.5.1. Advantages

1. Complete Session Stability

   During dynamic scaling (expansion or contraction) or node failures, the algorithm guarantees that the mapping of all existing connections remains absolutely unchanged. This provides a solid foundation for ASRP-based session recovery and is a key factor in achieving high availability.

2. Excellent Load Balancing for New Connections

   After each scaling operation, the algorithm reallocates virtual buckets so that each server serves as the preferred server for an equal number of buckets. This ensures that newly established connections are evenly distributed across all available servers, effectively preventing load imbalance.

3. Enhanced Robustness through Synergy with ASRP

The algorithm maintains a server list for each virtual bucket, providing a well-defined target set for ASRP's session probing mechanism (EQS/ERS). Working in synergy, they enable precise location and recovery of any connection after a node failure, achieving connection recoverability at the cluster level.

A.5.2. Limitations and Considerations

1. Probe Overhead and Server Count per Bucket

   For connectionless protocols (e.g., UDP) or TCP connections that need recovery, ASRP must sequentially probe the server list within a bucket via EQS messages. In the worst-case scenario of frequent scaling operations and continuous cluster growth, the number of servers in a bucket may gradually accumulate, leading to decreased probing efficiency.

2. Operational Recommendations

   To control the average length of bucket lists, it is recommended to adopt a batch-operation strategy when planning scaling events: "reduce the frequency of operations, but adjust more servers each time."

3. Optimization Directions

Implement dedicated optimizations for long-lived connections to further reduce the number of servers in virtual buckets.

In summary, the DBMCH algorithm provides an innovative solution for seamless scaling and high availability of stateful services. Its value is particularly pronounced when combined with distributed session state backup (ASRP). By understanding its characteristics, designers and operators can maximize its benefits and control its potential overhead through reasonable cluster sizing planning and operational strategies.

Appendix B. Acknowledgments

The authors would like to thank all individuals who have provided valuable feedback and contributions during the development of this document.

Authors' Addresses

Zhaoyu Luo (editor)
CMCC
No. 58 Kunlunshan Road

Suzhou
215000
China
Email: luozhaoyu@cmss.chinamobile.com

Haishuang Yan
CMCC
No. 58 Kunlunshan Road
Suzhou
215000
China
Email: yanhaishuang@cmss.chinamobile.com