IPSECME Working Group
Internet-Draft
Intended status: Standards Track
Expires: 23 April 2026

L. Xia
W. Jiang
Huawei Technologies

20 October 2025

Stateless Encryption Scheme of Enhanced Encapsulating Security Payload (EESP)
draft-xia-ipsecme-eesp-stateless-encryption-02

Abstract

This draft first introduces several use cases for stateless encryption, analyzes and compares some existing stateless encryption schemes in the industry, and then attempts to propose a general and flexible stateless encryption scheme based on the summarized requirements.

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 23 April 2026.

Table of Contents

1. Introduction

Recently, with the emergence of more new scenarios such as high-performance cloud services, AI large model computing, and 5G mobile backhaul networks, higher requirements have been put forward for the hardware friendliness, performance, and flexibility of the IPsec ESP protocol. A new protocol design, EESP [I-D.ietf-ipsecme-eesp] [I-D.ietf-ipsecme-eesp-ikev2], is being discussed and formulated. EESP focuses on solving issues such as introducing more fine-grained sub-child-SAs, adapting the ESP header and trailer format, and allowing parts of the transport layer header to be unencrypted, and implementing flexible expansion of EESP new features through options.

In addition to the issues listed above that are being addressed, stateless encryption is also a very important point. Its basic idea is to dynamically calculate data keys based on a small number of master keys (for AES-GCM, the encryption key and authentication key are combined), which helps optimize hardware resource limitations, performance optimization, and key negotiation complexity in large-scale IPsec session scenarios. This draft first introduces several use cases for stateless encryption, analyzes and compares some existing stateless encryption schemes in the industry, and then attempts to propose a general and flexible stateless encryption scheme based on the summarized requirements.

2. Use Cases

2.1. General Computing of Cloud Service

Public cloud services provide IPsec VPN access for massive users, and the servers in their infrastructure need to support massive IPsec session access. If hardware supports IPsec, the hardware should support session-based encryption and decryption, and the data keys of different sessions are isolated. The server needs to maintain the security connection context between the server and a large number of clients, and the hardware with limited memory cannot store the huge context. Note that the client and server do not belong to the same trusted domain in this case.

The stateless encryption scheme in the [PSP] solution proposed by Google is used to address the above hardware memory overhead problem. Its main principle is to derive a data key based on the master key on the server side, and the client side obtains the data key through an out-of-band method. It has:

* Pros: Save half of total session contexts. Furthermore, since the master key is owned by server and not shared, key leakage affects only one server;

* Cons: When a large number of new sessions are created, the data key negotiation is along the out of band slow path in real time, the first packet transmit will be delayed, and which results in performance degrade.

2.2. Cluster Communication in HPC Network

As shown in the below figure, encrypted communication is required between different instances of large-scale HPC jobs, the security session number is at the scale of O(M * N * N). So, an efficient security context management mechanism is required to solve the problem of large-scale security sessions. Note that all communication instances of a HPC job belong to the same trusted domain.

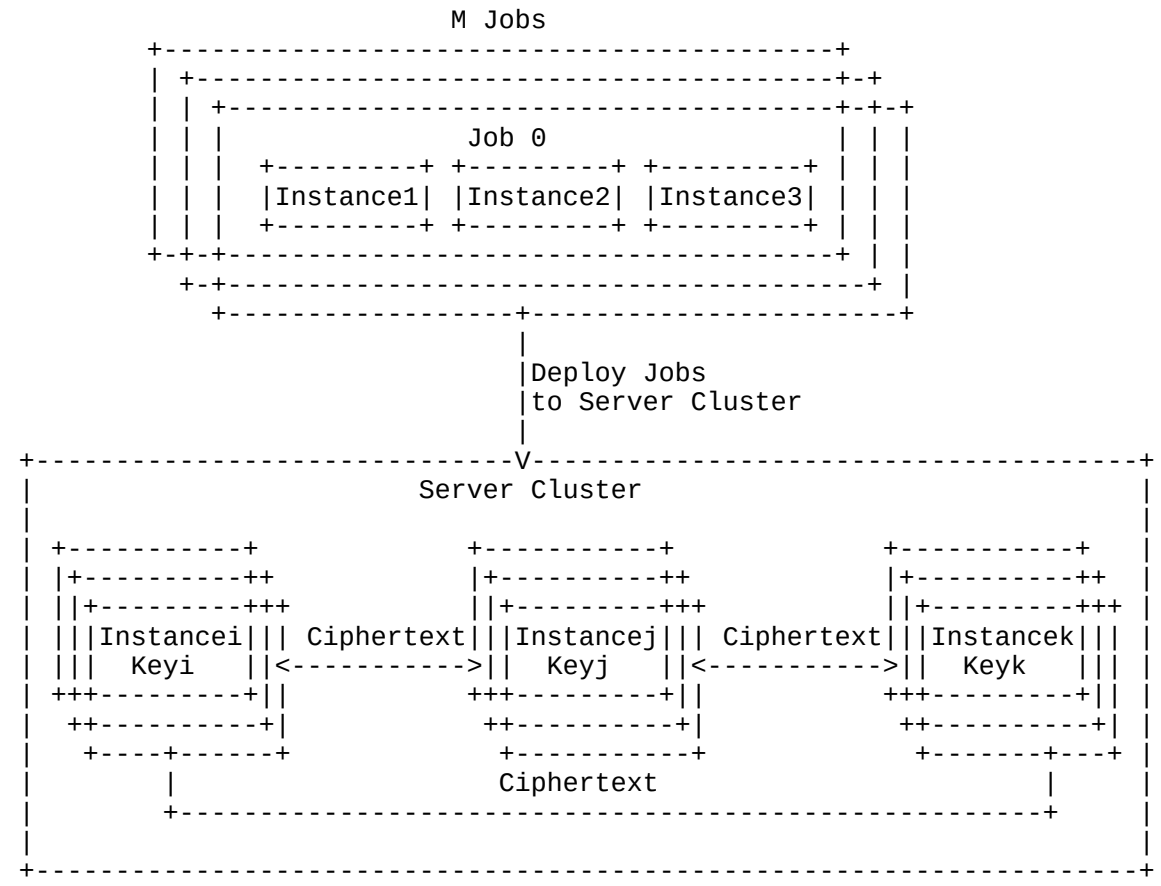

Figure 1: Encrypted Communication for Large Scale HPC Networks

The stateless encryption scheme defined by [UEC_TSS] can be used to solve the above problem. The main principle is that all communication instances of a HPC job belong to the same trust domain and share the same master key for both receiving and sending directions. It has:

* Pros:

  - Better than Google PSP, it saves all security session contexts;

  - The communication parties do not need to store data keys, and the increase of the number of instances and connections of the HPC job does not affect the number of security contexts;

- Without out of band slow path data key negotiation, the first packet delay is small;

- Data keys can be updated through the TSC.epoch.

* Cons:

- Master key leakage affects the entire trusted domain;

- The context content can be generated based on the SSI / Source IP / Destination IP field. Although the context content is flexible, the calculation overhead increases.

2.3. NIC/DPU Pool for General Computing

To cope with large-scale traffic access (e.g., computing server access to storage networks) and efficiently utilize network card resources, NIC resource pooling is an effective solution. For north-south traffic from client access to servers, the NIC resource pool must be transparent to the application, allowing a client to access resources behind any NIC in the pool. When using encrypted connections, all NICs must share the same key for a client’s access. At this point, the NICs in a resource pool belong to the same trust domain, so stateless encryption sharing the master key is applicable. This saves data key synchronization between NICs and reduces the storage of security sessions and data keys on them in scenarios with a large number of secure client connections. The client obtains the data key for this encrypted connection through an out-of-band method, which can be derived from the master key and context. Encrypted connections and contexts can be isolated based on flows or VM instances. As shown in the figure below:

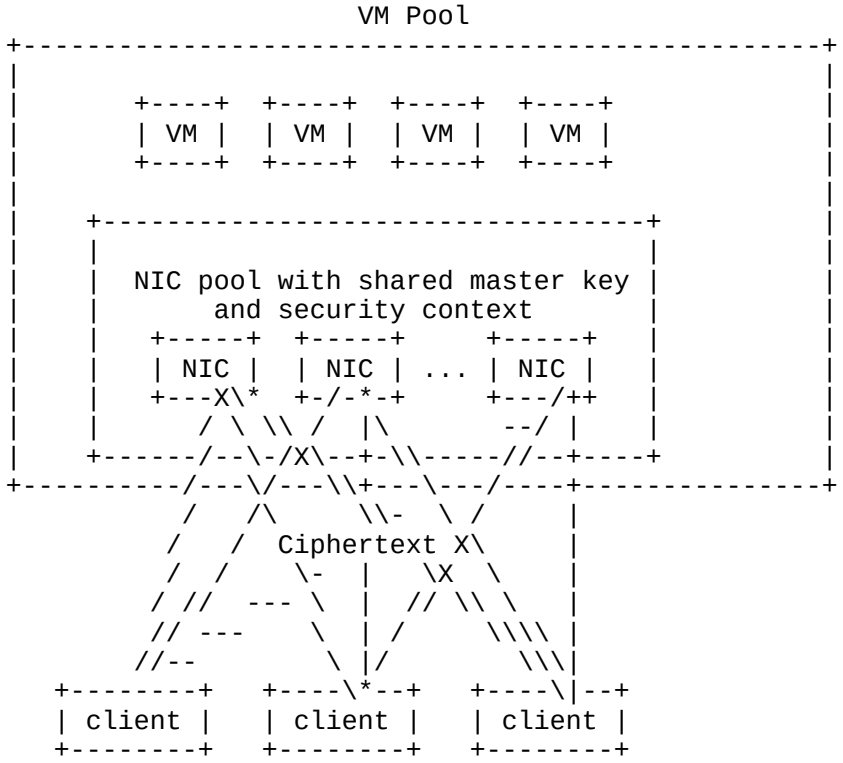

Figure 2: Encrypted Communication for NIC Pool

Similarly, the NIC resource pool can also be used for east-west traffic access between VMs. In this case, all NICs are in the same security domain and can share a master key, and different data keys can be dynamically generated based on different encryption connection contexts.

2.4. AI Computing

As shown in the figure below, in a AI computing network, a computing task is collaboratively executed by a group of CPUs & XPUs located in the same trust domain or across trust domains (in the case of cross-trust domains, they are interconnected as proxies through DPU). For CPUs & XPUs within the same trust domain, stateless encryption sharing the same master key can eliminate the complexity and latency of key negotiation between chips. For interconnection across trust domains, the DPU needs to perform encryption connection proxy functions between two trust domains (local trusted domain and global trusted domain). At this time, the DPU simultaneously possesses the master keys of the two trust domains, calculates the data key for

intra-domain communication in each domain based on its context, and then uses the calculated two data keys to complete the secure connection proxy across trust domains.

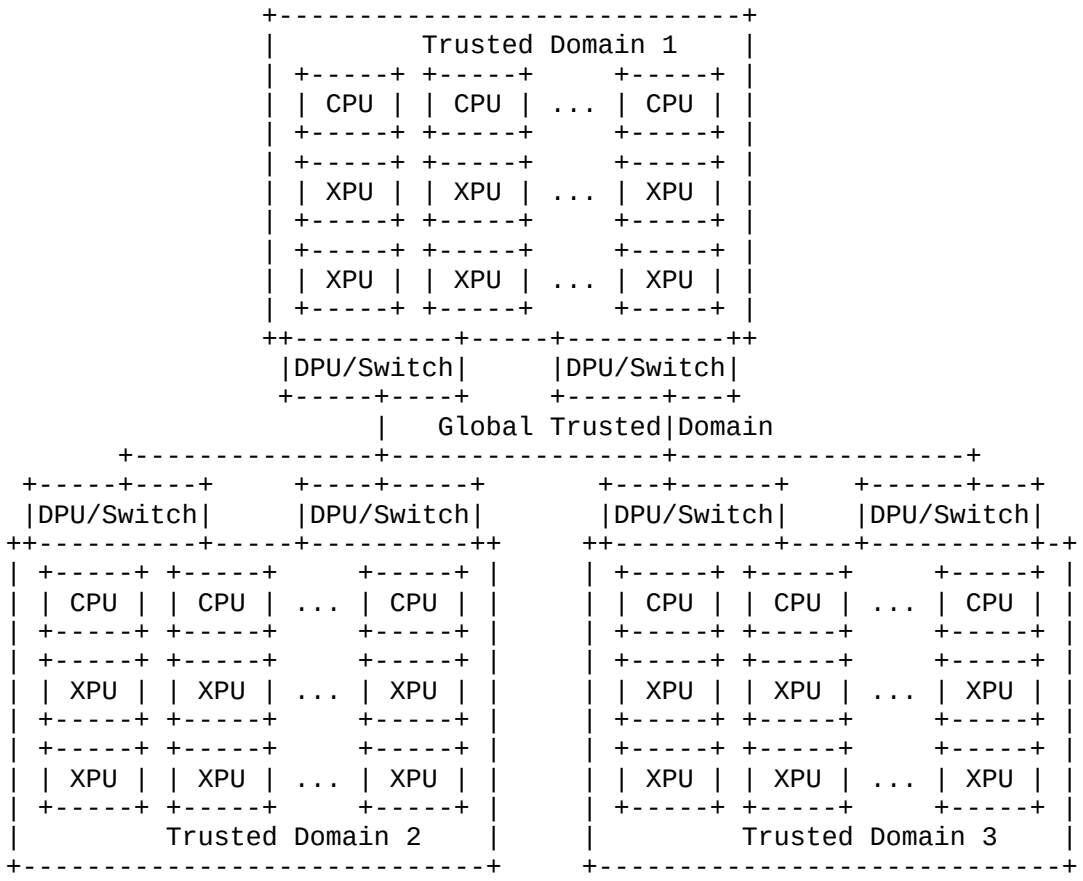

Figure 3: Encrypted Communication for AI Computing Network

3. Requirement Summary

Based on the above use cases, the requirements for a general and flexible stateless encryption scheme are as follows:

* Support entities within a trust group to share the same master key;

* Master key supports multi-level combination design. In a trust group, the master key is composed of multiple root keys of different types and levels, such as trust region root key, user

group root key, task group root key, etc. This enhances the overall security of the master key and supports fine-grained encryption traffic isolation (e.g., all entities in a trust region, entities of the same user group in a trust region, entities of the same task group in a trust region, etc.);

* Different types of root keys have different security levels and lifecycles, and corresponding key rotation mechanisms need to be defined. The master key update will trigger the data key update;

* The key rotation of each type of root key should support multiple key rotations, such as pre_key, current_key, and next_key, to support rapid rotation while ensuring that real-time encryption and decryption are not affected;

* The key derivation of the data key is based on the master key, context, and KDF. KDF must support packet-by-packet data key calculation in most cases (except when the data key is cached in memory), which requires extremely high performance and must support cryptographically secure, hardware-concurrent high-performance algorithms;

* To support real-time derivation of the Data Key, context information and IV information need to be carried with the message. To support different scenarios and different granularities of data key calculation and encryption traffic isolation (based on stream, based on source IP, based on source ID, etc.), multiple combinations of context and IV need to be supported, and different combination algorithms need to be distinguished through specific fields in the message;

* Context information enables dynamic updates of the data key, such as carrying an epoch in the context. When the epoch changes, the data key is also refreshed accordingly;

* It is necessary to support encryption proxy capabilities across trust regions. At the edge nodes across trust regions (such as DPU, Switch, etc.), support for master keys and stateless encryption of two trust groups (one is in local trust region and the other is in global trust region) is required, and proxy conversion of message encryption and decryption between the two trust groups must be completed.

4. EESP Stateless Encryption Scheme

Stateless Encryption is designed for large-scale general-purpose computing, AI computing, and pooled networks. It addresses the challenges of storing and managing security contexts by using computation to replace storage (key derivation) and flexible encryption and decryption, thereby enabling secure communication between nodes within and across domains. Therefore, to ensure that the endpoint can perform correct encryption and decryption without the need to store and manage security contexts, the stateless encryption extension must include the necessary fields required for calculating data key and performing the follow up encryption and decryption:

* Key Derivation Fields: Used to calculate the data key for data packets;

* Initial Vector Fields: Since AES-GCM is the primary data encryption algorithm, per-packet initialization vector (IV) should never be repeated for the same encryption key. A single duplicate IV can undermine the encryption of the entire stream;

* Confidentiality and integrity protection range Fields: Provide flexibility in the range of message confidentiality and integrity protection.

4.1. Master Key Management

Each trust group shares a master key. The master key supports being composed of multiple root keys, including: the trust zone root key, the user group root key, and the task group root key. This mechanism enhances the overall security of the master key and supports fine-grained encryption traffic isolation. The multiple root keys that make up the group key are securely distributed by different controllers (infrastructure providers, user group administrators, task group administrators) through different controllers/KMS. An example of the data structure definition for the root key is as follows:

```
RootKeyStruct ::= SEQUENCE {
    root_key_id    OCTET STRING,
    root_keys_index    SEQUENCE (SIZE(3)) OF INTEGER
        root_keys_value      SEQUENCE (SIZE(3)) OF OCTET STRING
}
```

Based on the trust region, use group, and task group under the trust group, the corresponding root_key_id can be found respectively. Then, within the structure corresponding to this ID, the combination

of the root_keys_index and root_key_value arrays forms three sets of root_key information (pre_key, current_key, and next_key) used for key rotation. This three-key rotation ensures the timely update of the root key (when the root key is rotated, it is replaced with the latest current_key) and guarantees that real-time encryption and decryption are not affected. The specific method for key rotation is as follows: a new next_key is generated, the original next_key is replaced with the new current_key, and the original current_key is replaced with the new pre_key.

4.2. Data key Derivation at Both Ends of the Communication

When secure communication is required within a trust group, the source point performs the following processing:

* data key derivation:
  - Obtain the master key: Based on the trust group information, combine the relevant root keys (e.g., through XOR calculation) to derive it;
  - Calculate the context information: Based on the source point IP/ID, or connection ID, etc., along with Epoch, the context is calculated using a specific algorithm. Using the source point IP/ID to calculate the context ensures that different secure sessions at the destination point have different data keys, thereby preventing the compromise of encryption security that could occur if different sessions had the same data key and the IV was also the same;
  - Execute KDF to derive the data key: use the aforementioned master key and context as inputs to the KDF;
* IV Calculation: Based on the source point IP/ID or connection ID, along with Epoch, random numbers, and counters, the IV is computed using a specific algorithm;
* Determine the scope of confidentiality and integrity protection: COffset and IOffset respectively;
* Encrypt the message using the data key and IV, and construct the security header: The security header field contains all the information mentioned above. The example diagram is as follows:

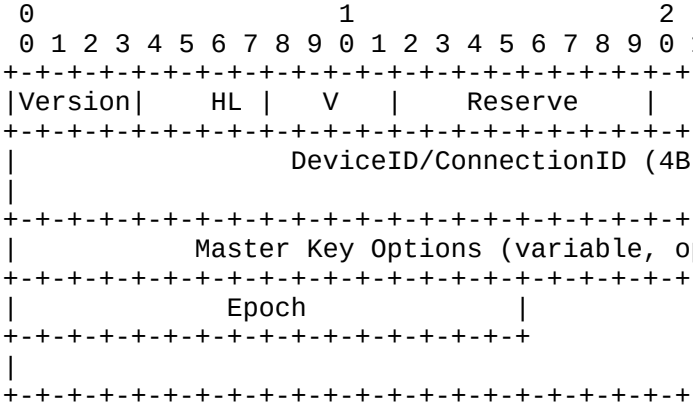

Figure 4: Example of the Security Header Format for Stateless Encryption

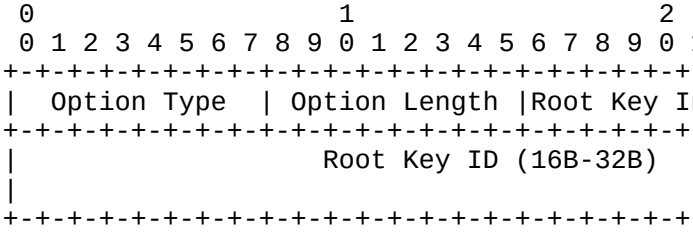

Figure 5: Example of the Master Key Option of Security Header Format for Stateless Encryption

Correspondingly, the destination node is processed as follows:

* Read the security header: Obtain all parameters required for key derivation;

* Data key derivation：

  - Obtain the master key: Based on the master key option in the security header, combine the relevant root keys (e.g., through XOR calculation) to obtain it;

  - Calculate the context information: Based on the source point IP/ID or connection ID in the security header, along with Epoch, compute the context using a specific algorithm;

  - Execute KDF to derive the data key: use the aforementioned master key and context as inputs to the KDF;

* IV Calculation: Based on the source point IP/ID in the security header, or connection ID, etc., along with Epoch, random numbers, and counters, the IV is calculated according to a specific algorithm;

* Determine the scope of confidentiality and integrity protection: COffset and IOffset respectively;

* Decrypt the message using the data key and IV.

5. Security Considerations

* A highly secure control plane is required to ensure that the master keys managed by users/systems are not leaked or lost;

* The control channel establishment phase requires two-way authentication and authorization to ensure the integrity and confidentiality of the master key during the master key distribution phase. At the same time, it ensures that the group master key is only distributed to the corresponding group members;

* The endpoint requires secure storage of the master key and data key locally;

* The key derivation process must ensure that the data keys calculated by cryptographic engines on different entities are unique. This means that the input for key derivation must include a unique ID to prevent two cryptographic engines from using the same data key;

* It is necessary to ensure that IVs are not reused. Under the same data key, the construction of IVs must guarantee that they are not repeated;

* The update cycle of the master key should be determined based on the actual number of derived data keys to be generated.

6. IANA Considerations

TBD.

7. Informative References

[PSP] "PSP Architecture Specification", n.d., <https://github.com/google/psp/blob/main/doc/PSP_Arch_Spec.pdf>.

[UEC_TSS] "Ultra Ethernet Specification v1.0", n.d., <https://ultraethernet.org/wp-content/uploads/sites/20/2025/06/UE-Specification-6.11.25.pdf>.

[I-D.ietf-ipsecme-eesp] Klassert, S., Antony, A., and C. Hopps, "Enhanced Encapsulating Security Payload (EESP)", Work in Progress, Internet-Draft, draft-ietf-ipsecme-eesp-02, 19 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ipsecme-eesp-02>.

[I-D.ietf-ipsecme-eesp-ikev2] Klassert, S., Antony, A., Brunner, T., and V. Smyslov, "IKEv2 negotiation for Enhanced Encapsulating Security Payload (EESP)", Work in Progress, Internet-Draft, draft-ietf-ipsecme-eesp-ikev2-01, 16 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ipsecme-eesp-ikev2-01>.

Authors' Addresses

Liang Xia
Huawei Technologies
Email: frank.xialiang@huawei.com

Weiyu Jiang
Huawei Technologies
Email: jiangweiyu1@huawei.com