Transmitter and Receiver Processing Specification for a Unified ULE Security Extension

Michael Noisternig, Bernhard Collini-Nocker
Multimedia Comm. Group, Dept. of Computer Sciences
University of Salzburg
Salzburg, Austria
mnoist@cosy.sbg.ac.at, bnocker@cosy.sbg.ac.at

Prashant Pillai
School of Engineering, Design & Technology
University of Bradford
Bradford, United Kingdom
p.pillai@bradford.ac.uk

Lei Liang, Haitham Cruickshank
Centre for Communication Systems Research
University of Surrey
Guildford, United Kingdom
l.liang@surrey.ac.uk, h.cruickhank@surrey.ac.uk

***Abstract*—The Unidirectional Lightweight Encapsulation (ULE) protocol has been defined for efficient transport of IPv4/6 and other protocols over the MPEG-2 Transport Stream (TS). The proliferation of this technology on the mass market may benefit from a security solution protecting against potential threats such as eavesdropping, as well as masquerading, modification of messages, and replay attacks, similar to 802.11 security. A unified ULE security extension header format has been proposed previously by the authors. This paper discusses in detail the processing required for transmitters and receivers supporting this security extension for ULE.**

***Keywords*—security, ULE, satellite, processing**

I. INTRODUCTION

With the convergence of IP and satellite technology there is growing interest in re-using existing satellite networks for the transport of IP data. The Unidirectional Lightweight Encapsulation (ULE) [1] has been defined as an efficient and extensible encapsulation protocol for IPv4 and IPv6 and other protocols over the MPEG-2 Transport Stream (TS). However, the proliferation of such technology may benefit from a security solution protecting against various potential threats on the satellite broadcast link, similar to 802.11 security. Such a link may be susceptible to security threats like eavesdropping, masquerading, modification of messages, replay attacks, and denial of service [2]. The need for security for the ULE protocol and the various threat scenarios have been studied within the IETF ipdvb working group [3]. Two solutions to provide such security for ULE were presented in [4] and [5]. While both these solutions addressed the security requirements in different ways, each of them suffered from some drawbacks. A unified ULE security extension format was proposed earlier by the authors in [6], and is briefly discussed in this paper. The main content of this paper is a proposed specification for transmitter and receiver devices supporting this new security extension. The specification aims to be simple to implement within devices, yet powerful enough to support expected communication scenarios, including both unidirectional links as well as group communication. The security goal of identity protection, which requires the hiding of the destination address within a SubNetwork Data Unit (SNDU) from illegitimate receiver devices, introduces a new element in processing.

Section II briefly describes the unified security extension header format for the ULE protocol. Section III discusses general preliminaries for the transmitter and receiver processing specifications. The detailed description of the transmitter and receiver processing procedures are presented in Section IV and Section V respectively. Finally, the conclusions are presented Section VI.

II. ULE SECURITY EXTENSION HEADER FORMAT

A unified security extension header format has been defined that is based on earlier independent proposals. It aims to protect against the threats for the satellite link identified in RFC 5458 [2]. The format of the extension header format is outlined in blue in Figure 1. The individual fields are described briefly in the following:

- Security Identifier (SID): A 16-bit security identifier, similar to the SPI in IPsec, used to look up the security state for a connection.
- Encrypted Destination Address (optional): This field is only present if the identity protection service is selected. In that case, the 48-bit destination NPA address from the base header is omitted (i.e., the base header's D bit set to 1), and instead appears in the security extension header, where it is encrypted subsequently (along with the payload data).

- SA-dependent data: This variable-length field contains all the auxiliary information required for the security algorithms defined by the SA, such as sequence numbers, or initialization vectors.
- Encrypted Next Type: This 16-bit value denotes the type of a subsequent extension header, respectively the content type of the payload data. It is encrypted along with the payload.
- Message Authentication Code (optional): This field is present if the integrity protection and authentication service is selected.

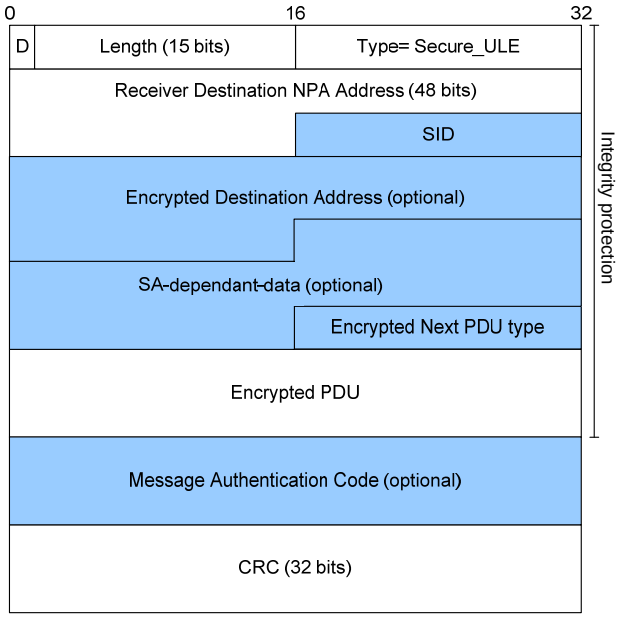


Figure 1. ULE Security Extension Header Format.

The security extension is a standard extension header as described in RFC 4326 [1]. This means that all ULE receivers must process this security header, or otherwise entirely discard the SNDU. The following sections discuss in detail how ULE transmitter and receiver devices are to process this security extension header.

III. PROCESSING PRELIMINARIES

The processing specification follows the IPsec [7] approach of defining a Security Policy Database (SPD) and a Security Association Database (SAD). The SPD contains an ordered list of Security Policies (SPs). These policies allow simple filtering of incoming or outgoing data based on address and other information, including assignment of different security services and keys to different connections and secure groups. The SAD refers to the set of security states, called Security Associations (SAs), which are referenced on the receiver side using the SID along with the address information of the received SNDU.

In the IPsec protocols, a receiver normally uses the SPI it has chosen itself for looking up SAs within its SAD. In our proposed extension the SPI is equivalent to the SID. However, this mechanism of using the SPI does not work for multicast settings, for other scenarios of group communication, and also for unidirectional links, where the SPI value has to be centrally selected by a group controller. A multicast IPsec implementation partially solves these issues by taking into account the source and multicast destination of a packet, and following a longest-match approach in determining the appropriate SA in the following way: First, an SA is looked up using the triple (SPI, destination, source) (1); if not successful, a search is done for (SPI, destination) (2); finally, only the SPI is taken for the lookup (3). While (1) aims to support source-specific multicast groups, (2) is meant for any-source multicast groups. This paper adapts that approach in the following way, using the Packet Identifier (PID) of the underlying MPEG-2 TS cell:

- For an SNDU with a multicast address present, a longest-match search on (SID, destination NPA, PID) is performed in the incoming SAD.
- Otherwise, a longest-match search is derived using (SID,PID) in the incoming SAD. This takes into account VPN-like settings with a single sender, as well as unidirectional links.

Support for shared SAs with multiple senders requires a coordinated solution for determining a unique SID value. In order to support shared SAs permitting bi-directional communication, a SAD should only store references to SAs, and reference bi-directional SAs in both the incoming and outgoing SAD.

Another difference to IPsec is the treatment of directionality for SPs and SAs. In a standard IPsec implementation, a match on a SP affects traffic in both directions, resulting in two separate unidirectional SAs to be created. This paper always requires separate SPs to be defined for incoming and outgoing data, and in turn allows SAs to be shared across several devices, supporting both unidirectional links and group communication.

For the rest of this paper, a Selector is defined as a pair of destination NPA address and PID.

A. Security Policy Database (SPD)

A SPD contains an ordered list of SPs, similar to Access Control Lists (ACLs). Each transmitter and receiver device defines one SPD for outgoing data, and one for incoming SNDUs. For both databases, a SP must provide the following information:

- A SP Selector, together with a SP Selector mask. This provides a simple and basic way to filter based on address information. For a transmitter SP, the Selector may be extended to include additional filtering information, such as higher-layer addresses, and port numbers.
- Information about the SA(s) to be instantiated by this SP, when a match is found based on filtering. This contains:
  - A Selector mask, denoted SA Selector mask, which specifies the set of SAs derived from the policy. This provides a simple way to instantiate secure unicast connections, as well as shared SAs for secure group communication. It is similar to the Populate-From-Packet (PFP) flags in the IPsec specification, but slightly more general.
  - An optional SID value. If not defined, Group Controller and Key Server (GCKS) data must be present for the SID to be selected dynamically.
  - Optional GCKS data. The GCKS must be contacted by a device which cannot find a SA for a matching SP, and when the SP does not define a static SID and default key data in its first set of Security Parameters.
  - An ordered list of Security Parameter sets used for instantiating a SA, sorted according to preference. On creating a SA, devices must default to the first entry in the list, unless a key management protocol permits negotiation (e.g., for unicast, bidirectional settings), and the device contacts the GCKS to request another set of Security Parameters from the list.

  Each set of Security Parameters contains information about:

  - The cryptographic algorithms used.
  - The cryptographic parameters required by these algorithms (e.g., sequence number length, MAC length).
  - Optional key data for manual keying.

  A Security Parameter set may select no security services at all, by which it is to be interpreted requesting processing without the security extension header.

Each SPD may be manually constructed by a device or network operator, but it may also be dynamically set up via a GCKS. Note that we do not describe how to create such databases, or how to store, manage, and look up SPs within the SPD, as this is regarded implementation specific details.

B. Security Association Database (SAD)

A Security Association (SA) is an instantiation of a SP. It describes the current state of a secure connection between two or more devices. Each SA within the SAD must contain the following information:

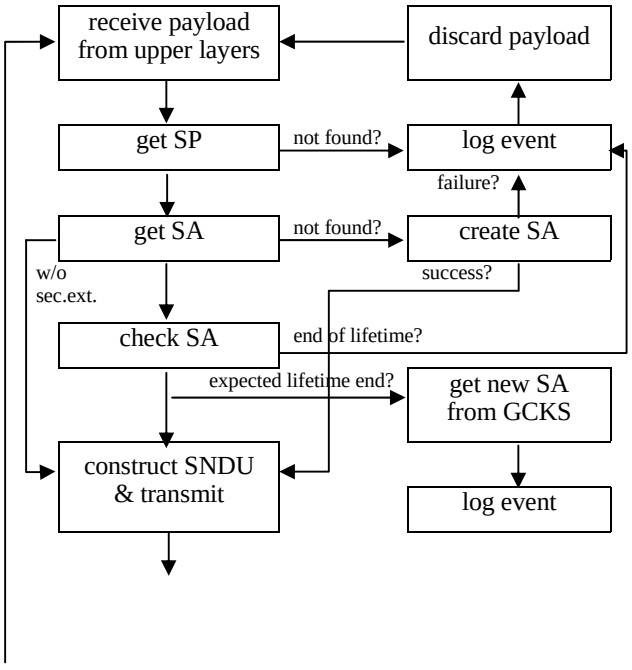
- The SA Selector derived from the instantiating SP.
- The SID defined by the SP or the GCKS.
- Static security parameters defined by the SP (cryptographic algorithms, MAC length, Sequence Number length, etc.).
- Dynamic security parameters (keys, sequence number, SA lifetime, etc.), initially defined by the SP or the GCKS, and updated during processing.
- GCKS specific data defined by the SP or the GCKS, including GCKS contact information.

As with the SPD, the description of the implementation-specific details of the SAD is out of scope of this paper.

IV. TRANSMITTER PROCESSING

The following list describes in detail the processing steps required for a ULE encapsulator implementing the unified ULE security extension:

1. *Get SP*: Upon constructing an SNDU for transmission over the ULE link, an encapsulator must scan its outgoing SPD for a matching policy. If no such policy can be found, the SNDU data must be discarded, and this event should be logged as an invalid transmission attempt.
2. *Get SA*: Given a matching SP, a SA is searched within the outgoing SAD using the SNDU's Selector information and the policy's SA Selector masks, including the SP's SID value if defined. If no SA could be found, it must be set up as follows: If the SP's first Security Parameter set contains default key data, or defines data to be sent without protection, the SA is immediately created and initialized according to these settings. Otherwise, if the SP defines GCKS contact information, it must be queried for obtaining key material. During that attempt the device should postpone transmission, or discard the data. Any case of failure must result in the data being discarded, and this event should be logged accordingly (e.g., as a user authentication failure in the case of membership denial by the GCKS). If the SA allows passing data unprotected, processing continues as usually.
3. *Check SA*: The SA may have a pre-defined lifetime (e.g., maximum number of encryptions, sequence number state, seconds since instantiation) after which it may no longer be used. To allow for a transient switch-over to a new SA, the SA must define a point prior to its lifetime end at which transmitters switch to the new SA. If that point is reached, a device may proactively request a new SA from a GCKS. In general, it is the responsibility of the operator or the GCKS to create new SAs, and distributing them to legitimate transmitter and receiver devices in time. If no new SA is available, a transmitter may still use the current SA during its full lifetime. After that, it must discard the data, and this event should be logged.
4. *Construct SNDU*: A protected SNDU is built as follows:
    a. First, the ULE base header and any extension headers preceding the security extension header are written. If the SA requests identity protection, the destination NPA address is omitted from the base header, setting the base header's D bit to 1. The last next-header-type field within the extension header chain contains the type code for the security extension.
    b. The SID value is written as the first field of the security extension header.
    c. If identity protection is used, the SNDU's destination NPA address follows. It is encrypted along with the payload.
    d. Any SA-dependent data, such as sequence numbers and initialization vectors, are written subsequently.
    e. Then, the next-header-type field, any further extension headers, and the payload data are encoded as defined by the SA's data confidentiality algorithm (together with the encrypted destination NPA address).
    f. For authentication and integrity protection, a MAC of length as defined by the SA is appended. The MAC is computed over all the data encoded so far, i.e., from the start of the SNDU to the end of the payload data.
    g. Finally, the CRC is calculated and appended, and the SNDU further processed according to RFC 4326.
5. *Update SA*: After transmission of the SNDU, the SA must be updated accordingly (e.g., the sequence number incremented by one).



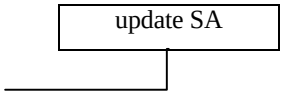


Figure 2. Transmitter activity diagram.

V. RECEIVER PROCESSING

For a receiver device to process SNDUs containing the security extension, the following steps must be adhered to:

1. *Decode SNDU (1)*: First, the CRC of an SNDU received is verified, and the base header and extension headers preceding a security extension header or the payload are evaluated.
2. *Get SA*: If a security extension header is found, a longest-match search within the incoming SAD is performed as described in section III. If it contains a matching SA, processing continues at step 4.
3. *Get SP*: The SPD is scanned similar to step 1 in the transmitter processing specification. However, the destination NPA address may be hidden, and consequently the SP must define whether the address must be matched or not. When the SNDU is received without a security extension header but the SP does not permit data to pass unprotected, the SNDU must be discarded immediately, and this event should be logged accordingly. Likewise, if there is a security extension header, but the policy allows only for unprotected data, the SNDU must be discarded, and the event should be logged. When permitted, an unprotected SNDU is processed as usually. Otherwise, a SA is created as described in step 2 of the transmitter processing specification.
4. *Check SA*: If the SA's lifetime has expired, the SNDU must be discarded, and this should be logged accordingly. If the extension header contains an encrypted destination NPA address, it is first decrypted, using the SA-dependent data, and checked for a valid address for that SA. If the decoded address is not accepted by the receiver device and the SA, the SNDU is discarded silently.
5. *Decode SNDU (2)*: This step includes verification of the MAC, protection against replay attacks, and decoding of the payload. In any case of failure, the SNDU must be discarded, and this event should be logged accordingly.
6. *Update SA*: After successful reception of an SNDU, the SA must be updated accordingly (e.g., the sequence number set to the one found within the SNDU, incremented by one).

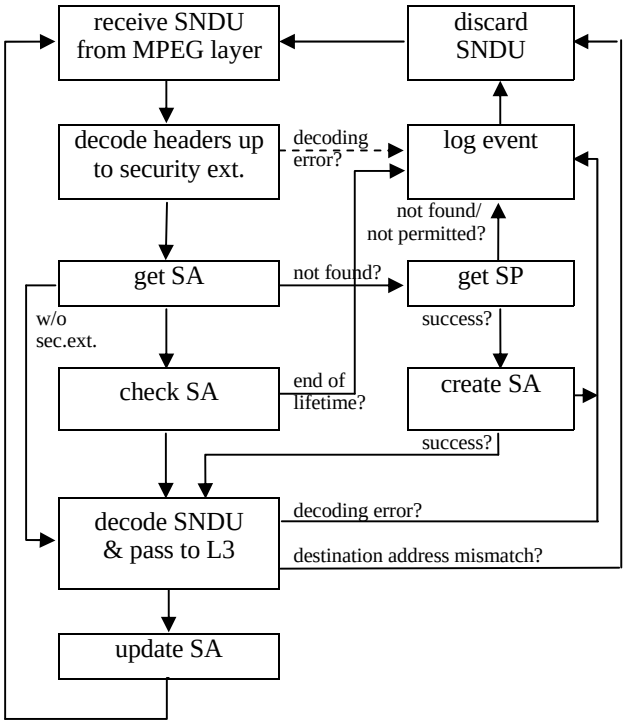


Figure 3. Receiver activity diagram.

VI. Conclusion

In this paper we have described the processing specification and procedure flow for transmitter and receiver devices supporting the new ULE security extension. The design and specifications have been defined general enough to allow flexible replacement of cryptographic algorithms, and the use of any key management protocol for the dynamic creation of Security Associations and Policies.

Acknowledgment

This work is performed as part of the European Union (EU) Sixth Framework Programme (FP6) Satellite Communications Networks of Excellence (SatNEx) project, Phase II.

References

[1] G. Fairhurst & B. Collini-Nocker. “Unidirectional Lightweight Encapsulation (ULE) for transmission of IP datagrams over MPEG-2 Transport Stream (TS)”, IETF RFC 4326.

[2] H. Cruickshank, P. Pillai, M. Noisternig, S. Iyengar. “Security requirements for the Unidirectional Lightweight Encapsulation (ULE) Protocol”, IETF, RFC5458, 2009.

[3] IETF IP over DVB WG, http://www.ietf.org/html. charters/ipdvb-charter.html.

[4] H. Cruickshank, P. Pillai, S. Iyengar. “Security Extension for Unidirectional Lightweight Encapsulation Protocol”, IETF, draft-cruickshank-ipdvb-sec-05 (expired), (2008).

[5] M. Noisternig, B. Collini-Nocker. “A lightweight security extension for the Unidirectional Lightweight Encapsulation (ULE) protocol”, IETF, draft-noisternig-ipdvb-ulesec-01 (expired), (2008).

[6] H. Cruickshank, L. Liang, P. Pillai, M. Noisternig, B. Collini-Nocker, G. Fairhurst, “Unified Link Layer Security Design for IP Encapsulation using Unidirectional Lightweight Encapsulation over Satellites”, 27th AIAA International Communications Satellite Systems Conference, in press, 2009.

[7] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.