Network Working Group P. Riikonen
Internet-Draft
draft-riikonen-flags-payloads-04.txt 2 December 2003
Expires: 2 June 2004

SILC Message Flag Payloads
<draft-riikonen-flags-payloads-04.txt>

Status of this Memo

Abstract

This memo describes the data payloads associated with the SILC Message Flags, as defined in the SILC Packet Protocol specification [SILC2]. The purpose of the Message Flags is to augment the function of the Message Payload used to send both private and channel messages, by allowing the sender to tell the receiver what type of data the payload includes, and how the data should be processed. Some of the Message Flags may define additional payloads to be associated with the flag, and this memo describes these payloads.

Table of Contents

1. Introduction

The Secure Internet Live Conferencing [SILC1] supports sending binary messages between users in the network. To make the data sending, and processing at the receiver's end as simple as possible the SILC defines Message Flags to the Message Payload [SILC2] that is used to send private and channel messages, which can help the receiver to decide how the data is encoded, and how it should be interpreted. Some of the Message Flags may define additional payloads to be associated with the flag, but the [SILC2] does not define them. This memo defines the payloads for those Message Flags that was marked to include additional payloads in [SILC2].

By defining the payloads for the Message Flags the Message Payload can be augmented to support any kind of data, which can be easily interpreted at the receiver end. For example, it would be possible to send audio stream, video stream, image files and HTML pages as messages, and the receiver can either choose to ignore the message or to process it, or to perhaps pass the message to some application for processing. Without specific payloads for Message Flags it is almost impossible for the receiver to interpret binary data from the payload.

1.1 Requirements Terminology

The keywords MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119].

2 SILC Message Flags

The Message Flags was added to the SILC protocol for the reason that SILC provides sending binary data as messages between users, and entities in the network, and interpreting pure binary data is almost impossible. With the Message Flags the purpose, the reason, and the method for how the message must be interpreted can be told to the recipient. Other conferencing protocols which are usually ASCII based protocols do not have such problems since they do not generally support sending of binary data at all, or require specific encoding of the data before it can be sent over the network.

The Message Payload in SILC can have flags that can augment the function of the payload. The flags can tell for example that the message is a request, or a reply to an earlier received request. They can tell that the message is some action that the sender is performing, or they can tell that the message is an auto reply, or that it is explicitly digitally signed by the sender.

The problem of Message Flags is that the space for flags mask is only 16 bits, so there is a limited number of flags available. For this reason having a flag that defines a generic way of sending any kind of data as a message, and can be easily interpreted at the receiver's end is important. For this reason the flag SILC_MESSAGE_FLAG_DATA was added to the protocol which can represent any data. This memo describe how this flag is used and how the associated payload is constructed and processed. This memo also describes payloads for all the other flags that can have associated payloads.

3 SILC Message Flag Payloads

The [SILC2] defines the flags which may have associated payloads. This section will list these flags and define the payloads.

3.1 SILC_MESSAGE_FLAG_REQUEST

Currently this flag can be used in the context of application specific, service specific or vendor specific requests, and the data payload type is dependent of this context. Therefore, payload is not defined for this flag in this memo. This flag may also be masked with some other flag in the message payload, including with some other flag that defines additional payload.

3.2 SILC_MESSAGE_FLAG_REPLY

Currently this flag can be used in the context of application specific, service specific or vendor specific replies, and the data payload type is dependent of this context. Therefore, payload is not defined for this flag in this memo. This flag may also be masked with some other flag in the message payload, including with some other flag that defines additional payload.

3.3 SILC_MESSAGE_FLAG_SIGNED

This flag is used to tell the recipient that the sent message is digitally signed by the sender, and that the recipient should verify the signature to verify the true authenticity of the received message. All message payloads in SILC provides message authentication code (MAC) which can be used to verify that the sender produced and sent the message. Even so, signing messages digitally can be used to verify the authenticity of the message when recipient trusts the sender and to provide non-repudiation.

This flag defines a payload which is used to deliver the actual message, sender's public key and the digital signature. The payload for SILC_MESSAGE_FLAG_SIGNED is as follows:

(*) indicates that the field is not encrypted.

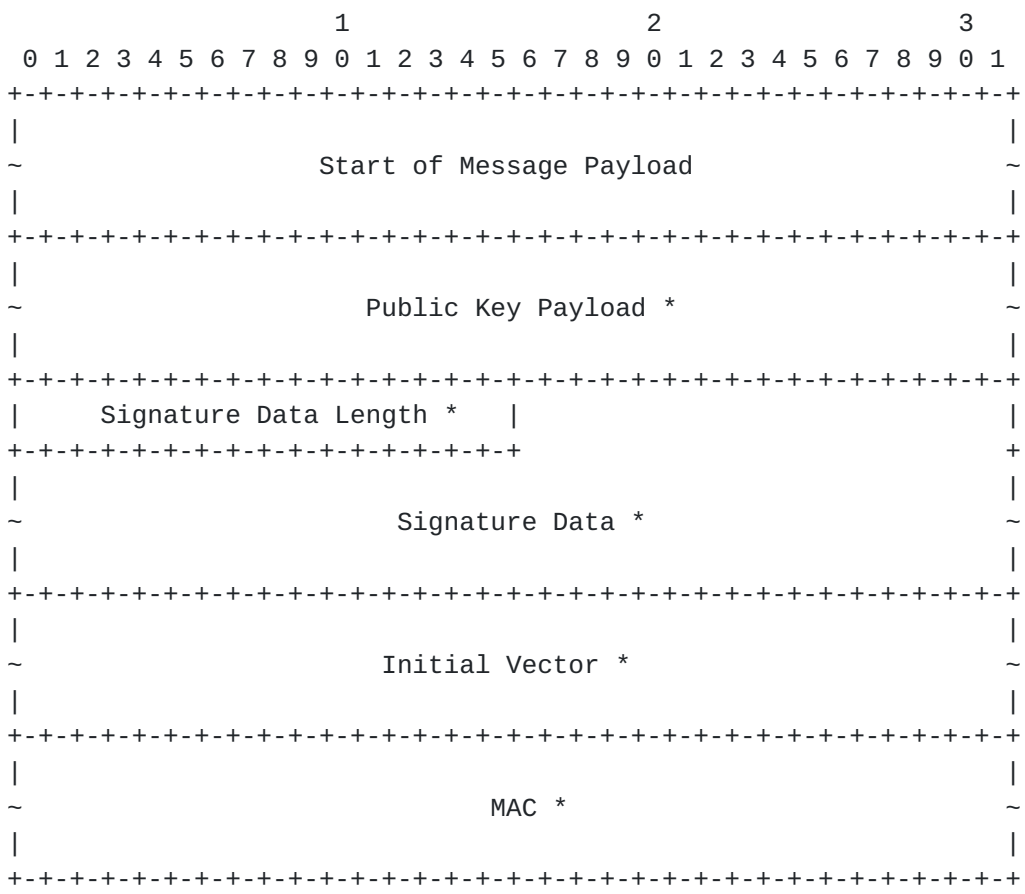

Figure 1: SILC_MESSAGE_FLAG_SIGNED Payload

o Start of Message Payload (variable length) - This is the start of the Message Payload without the IV and MAC fields, since those fields are appended at the end of this payload.

o Public Key Payload (variable length) - This includes the Public Key Payload [SILC2] which can be used to deliver the sender's public key (or certificate). It also indicates the type of the public key (or certificate) which the recipient use to identify how the signature must be verified. This payload must always be present but it is not required to include the public key data. The Public Key Type field in the Public Key Payload MUST be set to the correct type of the key, even if the actual public key data is not included. This field is not encrypted but is authenticated.

o Signature Data Length (2 bytes) - Indicates the length of the Signature Data field not including any other field. This field is not encrypted but is authenticated.

- Signature Data (variable length) - Includes the actual signature data. The signature computation and encoding is key type specific. See [SILC3] for all key types, and their respective references for how to compute and encode the signature. This field is not encrypted but is authenticated.

- Initial Vector (variable length) - the IV of the Message Payload as defined in [SILC2]. This field is not encrypted but is authenticated.

- MAC (variable length) - the MAC of the Message Payload as defined in [SILC2]. The MAC is computed after encryption and after signature computation. All data in the Message Payload and this payload, including the IV field are included in the MAC computation. This field is not encrypted.

How the data is processed before it is signed is key type specific. The actual data that to be signed MUST be the plaintext message payload before encryption. The data to be signed is concatenation of the Start of Message Payload field and the Public Key Payload, in that order. Any other fields are not included for signature data. Before signing, the data is always processed, usually hashed. The hash function to be used is defined in the key type specific definitions. See the key type specific references in [SILC3].

If the public key of the sender is included in the payload the recipient SHOULD verify it before accepting the public key. Recipient SHOULD verify the signature before accepting and caching the public key. With certificates the certificate verification may be done before verifying the signature. If the signature verification fails the message should still be displayed. The end user should also be notified about the result of the signature verification.

To make the packet size smaller implementations may not want to include the actual public key in all signed messages. Sending the public key in the first message is usually sufficient. Subsequent messages may include empty Public Key Payload with an indication of the public key type.

Implementations that do not support this flag can still process the message payload in normal manner. These implementations merely parse the decrypted payload in normal manner and ignore the extra data in the payload. They can do this by extracting the MAC and the IV from the end of the data buffer and thus ignoring the data between start of the Message Payload and the Initial Vector field.

This flag MAY be masked with any other Message Flag including those that define additional payloads. As long as the defined payload resides in the data area of the message payload this flag may be masked with the other flags.

3.4 SILC_MESSAGE_FLAG_DATA

This flag is used to represent any data as a message in the way that it can be easily interpreted by the recipient. This flag is used to send MIME objects as messages from the sender to the receiver. The MIME as defined in [RFC2045], [RFC2046], [RFC2047], [RFC2048] and [RFC2049] is well established protocol for sending different kind of data with many applications and protocols. It support dozens of different media types and encodings, and for this reason is ideal for sending data in SILC message payloads as well.

When the receiver has checked that the message payload includes the SILC_MESSAGE_FLAG_DATA flag, it may then start parsing the MIME header. It would also be possible to pass the message to some application which can already interpret MIME objects. If the receiver does not support the media type received in the MIME header, it SHOULD be treated as "application/octet-stream". The receiver MAY also ignore and discard messages that it does not support.

The MIME header MUST be at the start of the data area of the Message Payload. The MIME header received in the data area of the payload SHOULD have the MIME-Version field at first and then Content-Type field. The MIME-Version field is not required to be present in each body part of multipart entity. Additionally the header MAY also include any other MIME compliant headers. The character encoding for the MIME Header strings inside the message payload is US-ASCII, as defined in [RFC2045]. The actual MIME object may define additional character sets or encodings for the data it delivers.

Hence, the MIME Header in the message payload may be as follows:

```
MIME-Version: 1.0\r\n
Content-Type: discrete/composite\r\n
Content-Transfer-Encoding: binary\r\n
\r\n
```

The Content-Transfer-Encoding field behaves as defined in [RFC2045] and defines the encoding of the data in the MIME object. The preferred data encoding with SILC is "binary". However, many MIME media types defines their preferred encoding and they may be used if binary encoding is not suitable.

When sending large amounts of traffic or large files as MIME objects the limits of the SILC Packet needs to be taken into consideration. The maximum length of SILC Packet is 2^16 bytes, and larger messages would need to be fragmented. MIME provides way of fragmenting and reassembling messages, and it is to be done with SILC as defined in [RFC2046]. The MIME fragmentation is defined for gateway usage, but in case of SILC the sender (for example, a client) may also start sending fragmented MIME objects.

This flag SHOULD NOT be masked with some other Message Flag that defines payloads for message data. Generally this sort of setting would be impossible for the receiver to interpret. However, flags that does not define any specific payloads MAY be masked with this flag as well. For example, this flag could be masked also with SILC_MESSAGE_FLAG_REQUEST flag. It also can be masked with SILC_MESSAGE_FLAG_SIGNED flag since it does not define data specific payload.

3.5 SILC_MESSAGE_FLAG_ACK

This flag is used to send acknowledgement messages. When sender of a message requires the recipient to acknowledge the received message, the sender MUST set the SILC_MESSAGE_FLAG_ACK and MUST NOT set the SILC_MESSAGE_FLAG_NOREPLY. When a message with this flag set is received an acknowledgement message MUST be sent back. In the acknowledgement message the sender MUST set the SILC_MESSAGE_FLAG_ACK, SILC_MESSAGE_FLAG_AUTOREPLY and SILC_MESSAGE_FLAG_NOREPLY flags. The receiver MUST NOT acknowledge the acknowledgement message. This flag MUST NOT be used with channel messages, and MUST be ignored if received in a channel message.

The construction of the acknowledgement reply message is normal Message Payload where the Message Data field includes a computed MAC of the original received Message Payload MAC. Hence, the MAC is computed as follows:

```
ack_mac = mac(key, MAC);
```

Where the 'key' is the MAC key used to compute MACs for the Message Payload, and the 'MAC' is the MAC taken from the received Message Payload. The 'ack_mac' is placed in the Message Data field in a new Message Payload, and the payload is encrypted in normal manner. After this the message is sent back to the original sender of the message.

The receiver of the acknowledgement reply message SHOULD verify the MAC from the Message Data field to assure that acknowledgement was received to an earlier sent message. Implementation needs to keep the old message MACs stored until acknowledgement is received. It is left for

implementation to decide any possible retransmission strategy if acknowledgement messages are not received.

4 Security Considerations

In case of SILC_MESSAGE_FLAG_DATA the implementors should pay special attention to the security implications of any media type that can cause the remote execution of any actions in the receiver's environment. The [RFC2046] and [RFC2048] discusses more MIME specific security considerations. Even though SILC provides secured messages, in case of MIME which can be used to transfer files and documents which are stored in the receiver's local environment, securing separately the MIME object may be desired. For example, augmenting the MIME support in SILC messages to support S/MIME may be desired in some implementations.

5 References

[SILC1] Riikonen, P., "Secure Internet Live Conferencing (SILC), Protocol Specification", Internet Draft, June 2003.

[SILC2] Riikonen, P., "SILC Packet Protocol", Internet Draft, June 2003.

[SILC3] Riikonen, P., "SILC Key Exchange and Authentication Protocols", Internet Draft, June 2003.

[RFC2045] Freed, N., et al., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", Standards Track, RFC 2045, November 1996.

[RFC2046] Freed, N., et al., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", Standards Track, RFC 2045, November 1996.

[RFC2047] Moore K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text" Standards Track, RFC 2047, November 1996.

[RFC2048] Freed, N., et al., "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", Standards Track, RFC 2048, November 1996.

[RFC2049] Freed, N., et al., "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples", Standards Track, RFC 2049, November 1996.

[RFC2119] Bradner, S., "Key Words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

6 Author's Address

Pekka Riikonen
Snellmaninkatu 34 A 15
70100 Kuopio
Finland

EMail: priikone@iki.fi

7 Full Copyright Statement