Network Working Group
Internet-Draft
draft-riikonen-silc-pp-09.txt
Expires: 15 July 2007

P. Riikonen

15 January 2007

SILC Packet Protocol
<draft-riikonen-silc-pp-09.txt>

Status of this Draft

Abstract

This memo describes a Packet Protocol used in the Secure Internet Live Conferencing (SILC) protocol, specified in the Secure Internet Live Conferencing, Protocol Specification [SILC1]. This protocol describes the packet types and packet payloads which defines the contents of the packets. The protocol provides secure binary packet protocol that assures that the contents of the packets are secured and authenticated.

Table of Contents

1 Introduction 3
1.1 Requirements Terminology 4
2 SILC Packet Protocol 4
2.1 SILC Packet 4
2.2 SILC Packet Header 5
2.3 SILC Packet Types 8
2.3.1 SILC Packet Payloads 15
2.3.2 Generic payloads 16
2.3.2.1 ID Payload 16
2.3.2.2 Argument Payload 17
2.3.2.3 Argument List Payload 17
2.3.2.4 Channel Payload 18
2.3.2.5 Public Key Payload 19
2.3.2.6 Message Payload 20
2.3.3 Disconnect Payload 23
2.3.4 Success Payload 24
2.3.5 Failure Payload 25
2.3.6 Reject Payload 25
2.3.7 Notify Payload 26
2.3.8 Error Payload 35
2.3.9 Channel Message Payload 35
2.3.10 Channel Key Payload 36
2.3.11 Private Message Payload 38
2.3.12 Private Message Key Payload 38
2.3.13 Command Payload 40
2.3.14 Command Reply Payload 41
2.3.15 Connection Auth Request Payload 41
2.3.16 New ID Payload 42
2.3.17 New Client Payload 43
2.3.18 New Server Payload 44
2.3.19 New Channel Payload 45
2.3.20 Key Agreement Payload 45
2.3.21 Resume Router Payload 47
2.3.22 File Transfer Payload 47
2.3.23 Resume Client Payload 48
2.3.24 Acknowledgement Payload 50
2.4 SILC ID Types 50
2.5 Packet Encryption And Decryption 51
2.5.1 Normal Packet Encryption And Decryption 51
2.5.2 Channel Message Encryption And Decryption 52
2.5.3 Private Message Encryption And Decryption 53
2.6 Packet MAC Generation 53
2.7 Packet Padding Generation 54
2.8 Packet Compression 54
2.9 Packet Sending 55
2.10 Packet Reception 55

2.11 Packet Routing 55
2.12 Packet Broadcasting 57
3 Security Considerations 57
4 References 57
5 Author's Address 59
6 Full Copyright Statement 59

List of Figures

Figure 1: Typical SILC Packet
Figure 2: SILC Packet Header
Figure 3: ID Payload
Figure 4: Argument Payload
Figure 5: Argument List Payload
Figure 6: Channel Payload
Figure 7: Public Key Payload
Figure 8: Message Payload
Figure 9: Disconnect Payload
Figure 10: Success Payload
Figure 11: Failure Payload
Figure 12: Reject Payload
Figure 13: Notify Payload
Figure 14: Error Payload
Figure 15: Channel Key Payload
Figure 16: Private Message Key Payload
Figure 17: Command Payload
Figure 18: Connection Auth Request Payload
Figure 19: New Client Payload
Figure 20: New Server Payload
Figure 21: Key Agreement Payload
Figure 22: Resume Router Payload
Figure 23: File Transfer Payload
Figure 24: Resume Client Payload

1. Introduction

This document describes a Packet Protocol used in the Secure Internet Live Conferencing (SILC) protocol specified in the Secure Internet Live Conferencing, Protocol Specification [SILC1]. This protocol describes the packet types and packet payloads which defines the contents of the packets. The protocol provides secure binary packet protocol that assures that the contents of the packets are secured and authenticated. The packet protocol is designed to be compact to avoid unnecessary overhead as much as possible. This makes the SILC suitable also in environment of low bandwidth requirements such as mobile networks. All packet payloads can also be compressed to further reduce the size of the packets.

All packets in SILC network are always encrypted and their integrity is assured by computed MACs. The protocol defines several packet types and packet payloads. Each packet type usually has a specific packet payload that actually defines the contents of the packet. Each packet also includes a default SILC Packet Header that provides sufficient information about the origin and the destination of the packet.

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 Packet Protocol

2.1 SILC Packet

SILC packets deliver messages from sender to receiver securely by encrypting important fields of the packet. The packet consists of default SILC Packet Header, Padding, Packet Payload data, and, packet MAC.

The following diagram illustrates typical SILC packet.

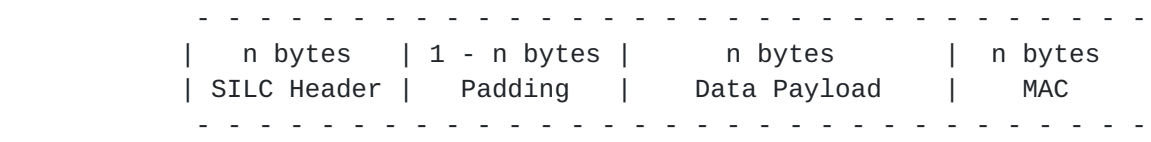


Figure 1: Typical SILC Packet

SILC Header is always the first part of the packet and its purpose is to provide information about the packet. It provides for example the packet type, origin of the packet and the destination of the packet. The header is variable in length. See the following section for description of SILC Packet header. Packets without SILC header or with malformed SILC header MUST be dropped.

Padding follows the packet header. The purpose of the padding is to make the packet multiple by eight (8) or by the block size of the cipher used in the encryption, which ever is larger. The maximum length of padding is currently 128 bytes. The padding is always encrypted. The padding is applied always, even if the packet is not encrypted. See the section 2.7 Padding Generation for more detailed information.

Data payload area follows padding and it is the actual data of the packet. The packet data is the packet payloads defined in this protocol. The data payload area is always encrypted.

The last part of SILC packet is the packet MAC that assures the integrity of the packet. See the section 2.6 Packet MAC Generation for more information. If compression is used the compression is always applied before encryption.

All fields in all packet payloads are always in MSB (most significant byte first) order.

2.2 SILC Packet Header

The SILC packet header is applied to all SILC packets and it is variable in length. The purpose of SILC Packet header is to provide detailed information about the packet. The receiver of the packet uses the packet header to parse the packet and gain other relevant parameters of the packet.

The following diagram represents the SILC packet header.

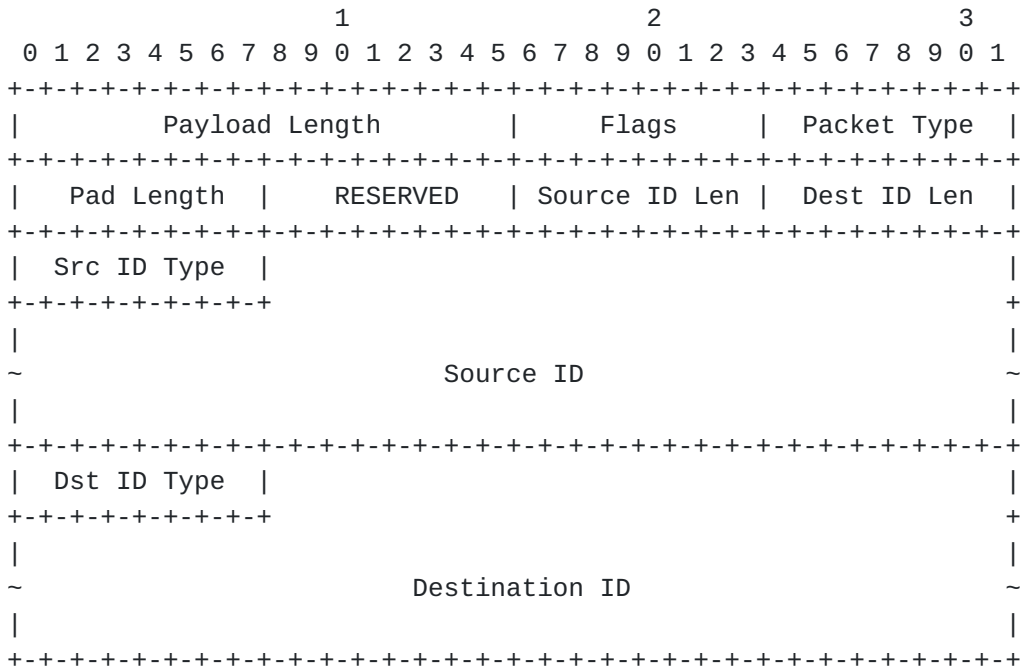


Figure 2: SILC Packet Header

o Payload Length (2 bytes) - Indicates the length of the packet not including the padding of the packet.

o Flags (1 byte) - Indicates flags to be used in packet processing. Several flags may be set by ORing the flags together.

The following flags are reserved for this field:

No flags 0x00

In this case the field is ignored.

Private Message Key 0x01

Indicates that the packet data MUST include private message that is encrypted using private key set by client. Servers does not know this key and cannot decrypt the payload, but simply passes it along. See section 2.5.3 Private Message Encryption And Decryption for more information.

List 0x02

Indicates that the packet consists of list of packet payloads indicated by the Packet Type field. The payloads are added one after the other. Note that there are packet types that must not be used as list. Parsing of list packet is done by calculating the length of each payload and parsing them one by one.

Broadcast 0x04

Marks the packet to be broadcasted. Client and normal server cannot send broadcast packets. Only router server may send broadcast packet. The router receiving of packet with this flag set MUST send (broadcast) the packet to its primary route. If router has several router connections the packet may be sent only to the primary route. See section 2.12 Packet Broadcasting for description of packet broadcasting.

Compressed 0x08

Marks that the payload of the packet is compressed.

The sender of the packet marks this flag when it compresses the payload, and any server or router en route to the recipient MUST NOT unset this flag. See section 2.8 Packet Compression for description of packet compressing.

Acknowledgement 0x10

Marks that the packet needs to be acknowledged by the recipient. The ACK packet MUST NOT have this flag set. The acknowledgement packet is SILC_PACKET_ACK packet. If the packet is not acknowledged the packet may be retransmitted. This flag is especially useful when using UDP/IP and SHOULD NOT be used with TCP/IP. The flag MUST NOT be used with message packets. The SILC_MESSAGE_FLAG_ACK can be used instead. Broadcast packets MUST NOT set this flag. Retransmission may use for example exponential backoff algorithm.

- Packet Type (1 byte) - Indicates the type of the packet. Receiver uses this field to parse the packet. See section 2.3 SILC Packets for list of defined packet types.

- Pad Length (1 byte) - Indicates the length of the padding applied after the SILC Packet header. Maximum length for padding is 128 bytes.

- RESERVED (1 byte) - Reserved field and must include a zero (0) value.

- Source ID Length (1 byte) - Indicates the length of the Source ID field in the header, not including this or any other fields.

- Destination ID Length (1 byte) - Indicates the length of the Destination ID field in the header, not including this or any other fields.

- Src ID Type (1 byte) - Indicates the type of ID in the Source ID field. See section 2.4 SILC ID Types for defined ID types.

- Source ID (variable length) - The actual source ID that indicates which is the original sender of the packet.

- Dst ID Type (1 byte) - Indicates the type of ID in the

Destination ID field. See section 2.4 SILC ID Types for defined ID types.

o Destination ID (variable length) - The actual destination ID that indicates which is the end receiver of the packet.

2.3 SILC Packet Types

SILC packet types defines the contents of the packet and it is used by the receiver to parse the packet. The packet type is 8 bits in length. The range for the packet types are from 0 - 255, where 0 is never sent and 255 is currently reserved for future extensions and MUST NOT be defined to any other purpose. Every SILC specification compliant implementation SHOULD support all the following packet types.

The below list of the SILC Packet types includes reference to the packet payload as well. Packet payloads are the actual packet data area. Each packet type defines packet payload which usually may only be sent with the specific packet type.

Most of the packets are packets that must be destined directly to entity that is connected to the sender. It is not allowed, for example, for a router to send SILC_PACKET_DISCONNECT packet to client that is not directly connected to the router. However, there are some special packet types that may be destined to some entity that the sender does not have direct connection with. These packets are for example private message packets, channel message packets, command packets and some other packets that may be broadcasted in the SILC network. The following packet desription list will define it separately if a packet is allowed to be sent to indirectly connected entity. Other packets MUST NOT be sent or accepted, if sent, to indirectly connected entities.

Some packets MAY be sent as lists by adding the List flag to the Packet Header and constructing multiple packet payloads one after the other. When this is allowed it is separately defined in the following list. Other packets MUST NOT be sent as list and the List flag MUST NOT be set.

List of SILC Packet types are defined as follows.

0 SILC_PACKET_NONE

This type is reserved and it is never sent.

1 SILC_PACKET_DISCONNECT

This packet is sent to disconnect the remote end. Reason of the disconnection is sent inside the packet payload.

Payload of the packet: See section 2.3.3 Disconnect Payload

2 SILC_PACKET_SUCCESS

This packet is sent upon successful execution of a protocol. The status of the success is sent in the packet payload.

Payload of the packet: See section 2.3.4 Success Payload

3 SILC_PACKET_FAILURE

This packet is sent upon failure of a protocol. The status of the failure is sent in the packet payload.

Payload of the packet: See section 2.3.5 Failure Payload

4 SILC_PACKET_REJECT

This packet MAY be sent upon rejection of a protocol. The status of the rejection is sent in the packet payload.

Payload of the packet: See section 2.3.6 Reject Payload

5 SILC_PACKET_NOTIFY

This packet is used to send notify message. The packet is usually sent between server and client, but also between server and router. Client MUST NOT send this packet. Server MAY destine this packet to channel as well when the packet is distributed to all clients on the channel. This packet MAY be sent as list.

Payload of the packet: See section 2.3.7 Notify Payload.

6 SILC_PACKET_ERROR

This packet is sent when an error occurs. Server MAY send this packet. Client MUST NOT send this packet. The client MAY entirely ignore the packet, however, server is most likely to take action anyway. This packet MAY be sent

to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.8 Error Payload.

7 SILC_PACKET_CHANNEL_MESSAGE

This packet is used to send messages to channels. The packet includes Channel ID of the channel and the actual message to the channel. Messages sent to the channel are always protected by channel specific keys. This packet MAY be sent to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.9 Channel Message Payload

8 SILC_PACKET_CHANNEL_KEY

This packet is used to distribute new key for particular channel when server generates it. Each channel has their own independent keys that is used to protect the traffic on the channel. It is also possible to use channel private keys that are not server generated. In this case this packet is not used. Client MUST NOT send this packet. This packet MAY be sent to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.10 Channel Key Payload

9 SILC_PACKET_PRIVATE_MESSAGE

This packet is used to send private messages from client to another client. By default, private messages are protected by session keys established by normal key exchange protocol. However, it is possible to use specific key to protect private messages. See [SILC1] for private message key generation. This packet MAY be sent to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.11 Private Message Payload

10 SILC_PACKET_PRIVATE_MESSAGE_KEY

This packet is OPTIONAL and sender of the packet can indicate that a private message key should be used in private message

communication. The actual key material is not sent in this packet but must be either static or pre-shared key. The receiver of the packet is considered to be the responder when processing the static or pre-shared key material as defined in [SILC1] and [SILC3] for private message keys. This packet MAY be sent to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.12 Private Message Key Payload

11 SILC_PACKET_COMMAND

This packet is used to send commands from client to server. Server MAY send this packet to other servers as well. All commands are listed in their own section SILC Command Types in [SILC4]. The contents of this packet is command specific. This packet MAY be sent to entity that is indirectly connected to the sender.

Payload of the packet: See section 2.3.13 Command Payload

12 SILC_PACKET_COMMAND_REPLY

This packet is sent as reply to the SILC_PACKET_COMMAND packet. The contents of this packet is command specific. This packet MAY be sent to entity that is indirectly connected to the sender. This packet MAY be sent as list.

Payload of the packet: See section 2.3.14 Command Reply Payload and section 2.3.13 Command Payload

13 SILC_PACKET_KEY_EXCHANGE

This packet is used to start SILC Key Exchange Protocol, described in detail in [SILC3].

Payload of the packet: Payload of this packet is described in the section SILC Key Exchange Protocol and its sub sections in [SILC3].

14 SILC_PACKET_KEY_EXCHANGE_1

This packet is used as part of the SILC Key Exchange Protocol.

Payload of the packet: Payload of this packet is described in the section SILC Key Exchange Protocol and its sub sections in [SILC3].

15 SILC_PACKET_KEY_EXCHANGE_2

This packet is used as part of the SILC Key Exchange Protocol.

Payload of the packet: Payload of this packet is described in the section SILC Key Exchange Protocol and its sub sections in [SILC3].

16 SILC_PACKET_CONNECTION_AUTH_REQUEST

This packet is used to request an authentication method to be used in the SILC Connection Authentication Protocol. If initiator of the protocol does not know the mandatory authentication method this packet MAY be used to determine it. The party receiving this payload SHOULD respond with the same packet including the mandatory authentication method.

Payload of the packet: See section 2.3.15 Connection Auth Request Payload

17 SILC_PACKET_CONNECTION_AUTH

This packet is used to start and perform the SILC Connection Authentication Protocol. This protocol is used to authenticate the connecting party. The protocol is described in detail in [SILC3].

Payload of the packet: Payload of this packet is described in the section SILC Authentication Protocol and it sub sections in [SILC].

18 SILC_PACKET_NEW_ID

This packet is used to distribute new IDs from server to router and from router to all other routers in SILC network. This is used when for example new client is registered to

SILC network. The newly created IDs of these operations are distributed by this packet. Only server may send this packet, however, client MUST be able to receive this packet. This packet MAY be sent to entity that is indirectly connected to the sender. This packet MAY be sent as list.

Payload of the packet: See section 2.3.16 New ID Payload

19 SILC_PACKET_NEW_CLIENT

This packet is used by client to register itself to the SILC network. This is sent after key exchange and authentication protocols has been completed. Client sends various information about itself in this packet to the server.

Payload of the packet: See section 2.3.17 New Client Payload

20 SILC_PACKET_NEW_SERVER

This packet is used by server to register itself to the SILC network. This is sent after key exchange and authentication protocols has been completed. Server sends this to the router it connected to, or, if router was connecting, to the connected router. Server sends its Server ID and other information in this packet. The client MUST NOT send or receive this packet.

Payload of the packet: See section 2.3.18 New Server Payload

21 SILC_PACKET_NEW_CHANNEL

This packet is used to notify routers about newly created channel. Channels are always created by the router and it MUST notify other routers about the created channel. Router sends this packet to its primary route. Client MUST NOT send this packet. This packet MAY be sent to entity that is indirectly connected to the sender. This packet MAY be sent as list.

Payload of the packet: See section 2.3.19 New Channel Payload

22 SILC_PACKET_REKEY

This packet is used to indicate that re-key must be performed for session keys. See section Session Key Regeneration in

[SILC1] for more information. This packet does not have a payload.

23 SILC_PACKET_REKEY_DONE

This packet is used to indicate that re-key is performed and new keys must be used hereafter. This packet does not have a payload.

24 SILC_PACKET_HEARTBEAT

This packet is used by clients, servers and routers to keep the connection alive. It is RECOMMENDED that all servers implement keepalive actions and perform it to both direction in a link. This packet does not have a payload.

25 SILC_PACKET_KEY_AGREEMENT

This packet is used by clients to request key negotiation between another client in the SILC network. If the negotiation is started it is performed using the SKE protocol. The result of the negotiation, the secret key material, can be used for example as private message key. The server and router MUST NOT send this packet.

Payload of the packet: See section 2.3.20 Key Agreement Payload

26 SILC_PACKET_RESUME_ROUTER

This packet is used during backup router protocol when the original primary router of the cell comes back online and wishes to resume the position as being the primary router of the cell.

Payload of the packet: See section 2.3.21 Resume Router Payload

27 SILC_PACKET_FTP

This packet is used to perform an file transfer protocol in the SILC session with some entity in the network. The packet is multi purpose. The packet is used to tell other entity in the network that the sender wishes to perform an file transfer protocol. The packet is also used to actually tunnel the file transfer protocol stream. The file transfer protocol

stream is always protected with the SILC binary packet protocol.

Payload of the packet: See section 2.3.22 File Transfer Payload

28 SILC_PACKET_RESUME_CLIENT

This packet is used to resume a client back to the network after it has been detached. A client is able to detach from the network but the client is still valid client in the network. The client may then later resume its session back by sending this packet to a server. Routers also use this packet to notify other routers in the network that the detached client has resumed.

Payload of the packet: See section 2.3.23 Resume Client Payload

29 SILC_PACKET_ACK

This packet is used to acknowledge a packet that had the Acknowledgement packet flag set.

Payload of the packet: See section 2.3.24 Acknowledgement Payload

30 - 199

Currently undefined commands.

200 - 254

These packet types are reserved for private use and they will not be defined by this document.

255 SILC_PACKET_MAX

This type is reserved for future extensions and currently it MUST NOT be sent.

2.3.1 SILC Packet Payloads

All payloads resides in the main data area of the SILC packet. However all payloads MUST be at the start of the data area after the SILC packet header and padding. All fields in the packet payload are always

encrypted, as they reside in the data area of the packet which is always encrypted. Most of the payloads may only be sent with specific packet type which is defined in the description of the payload.

There are some other payloads in SILC as well. However, they are not common in the sense that they could be sent at any time. These payloads are not described in this section. These are payloads such as SILC Key Exchange payloads and so on. These are described in [SILC1], [SILC3] and [SILC4].

2.3.2 Generic payloads

This section describes generic payloads that are not associated to any specific packet type. They can be used for example inside some other packet payload.

2.3.2.1 ID Payload

This payload can be used to send an ID. ID's are variable in length thus this payload provides a way to send variable length ID.

The following diagram represents the ID Payload.

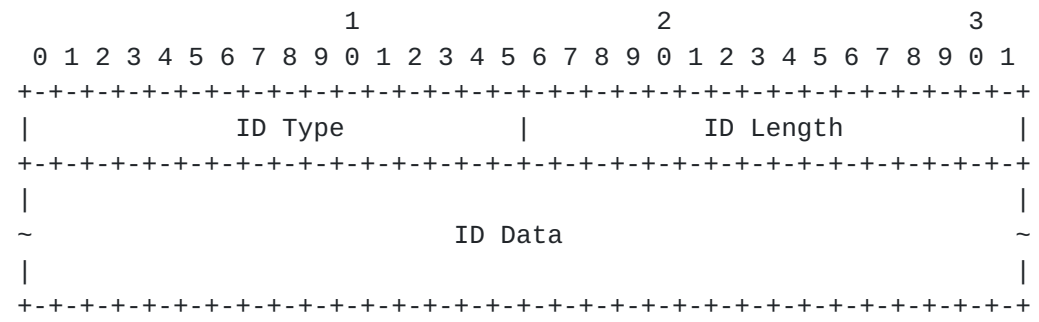


Figure 3: ID Payload

- ID Type (2 bytes) - Indicates the type of the ID. See section 2.4 SILC ID Types for list of defined ID types.
- ID Length (2 bytes) - Length of the ID Data area not including the length of any other fields in the payload.
- ID Data (variable length) - The actual ID data. The encoding of the ID data is defined in section 2.4 SILC ID Types.

2.3.2.2 Argument Payload

Argument Payload is used to set arguments for any packet payload that need and support arguments, such as commands. Number of arguments associated with a packet MUST be indicated by the packet payload which need the arguments. Argument Payloads MUST always reside right after the packet payload needing the arguments. Incorrect amount of argument payloads MUST cause rejection of the packet.

The following diagram represents the Argument Payload.

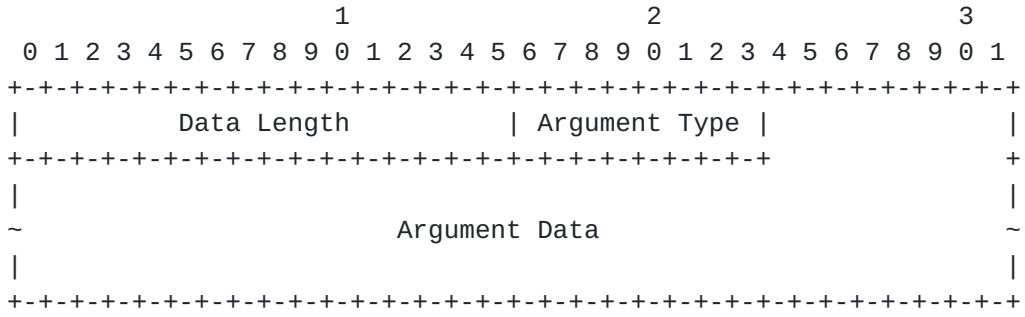


Figure 4: Argument Payload

- Data Length (2 bytes) - Length of the Argument Data field not including the length of any other field in the payload.

- Argument Type (1 byte) - Indicates the type of the argument. Every argument can have a specific type that are defined by the packet payload needing the argument. For example every command specify a number for each argument that may be associated with the command. By using this number the receiver of the packet knows what type of argument this is. If there is no specific argument type this field is set to zero (0) value.

- Argument Data (variable length) - Argument data.

2.3.2.3 Argument List Payload

Argument List Payload is a list of Argument Payloads appended one after the other. The number of arguments is indicated in the payload.

The following diagram represents the Argument List Payload.

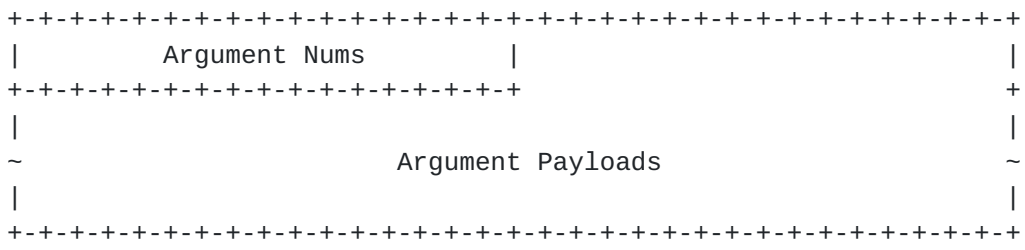


Figure 5: Argument List Payload

- Argument Nums (2 bytes) - Indicates the number of Argument Payloads. If zero (0) value is found in this field no arguments are present.

- Argument Payloads (variable length) - The Argument Payloads appended one after the other. The payloads can be decoded since the length of the payload is indicated in each of the Argument Payload.

2.3.2.4 Channel Payload

Generic Channel Payload may be used to send information about a channel, its name, the Channel ID and a mode.

The following diagram represents the Channel Payload.

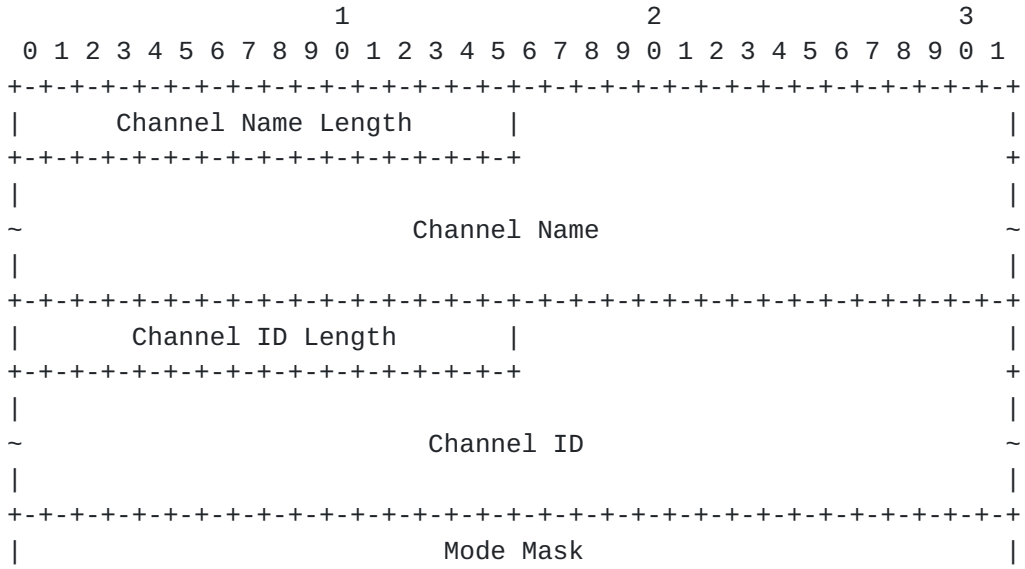


Figure 6: New Channel Payload

- Channel Name Length (2 bytes) - Length of the Channel Name field.
- Channel Name (variable length) - The name of the channel.
- Channel ID Length (2 bytes) - Length of the Channel ID field.
- Channel ID (variable length) - The encoded Channel ID.
- Mode Mask (4 bytes) - A mode. This can be the mode of the channel but it can also be the mode of a client on the channel. The contents of this field is dependent of the usage of this payload. The usage is defined separately when this payload is used. This is a 32 bit MSB first value.

2.3.2.5 Public Key Payload

Generic Public Key Payload may be used to send different type of public keys and certificates.

The following diagram represents the Public Key Payload.

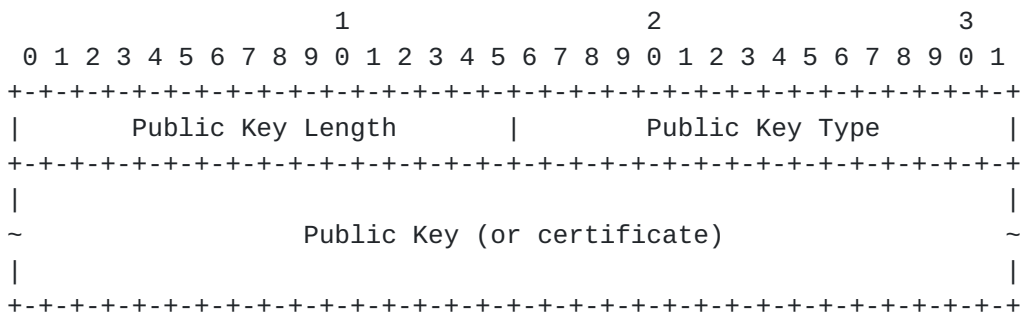


Figure 7: Public Key Payload

- Public Key Length (2 bytes) - The length of the Public Key (or certificate) field, not including any other field.

o Public Key Type (2 bytes) - The public key (or certificate) type. This field indicates the type of the public key in the packet. See the [SILC3] for defined public key types.

o Public Key (or certificate) (variable length) - The encoded public key or certificate data.

2.3.2.6 Message Payload

Generic Message Payload can be used to send messages in SILC. It is used to send channel messages and private messages.

The following diagram represents the Message Payload.

(*) indicates that the field is not encrypted.

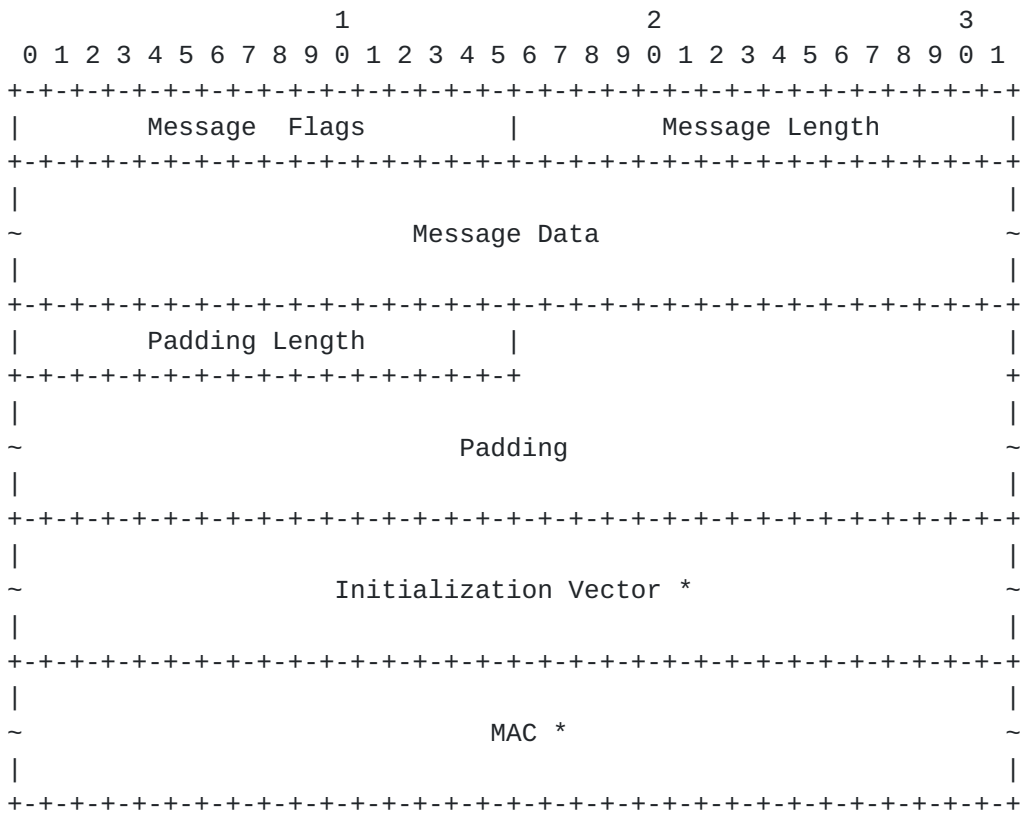


Figure 8: Message Payload

o Message Flags (2 bytes) - Includes the Message Flags of the message. The flags can indicate a reason or a purpose for the message. The following Message Flags are defined:

0x0000 SILC_MESSAGE_FLAG_NONE

No specific flags set.

0x0001 SILC_MESSAGE_FLAG_AUTOREPLY

This message is an automatic reply to an earlier received message.

0x0002 SILC_MESSAGE_FLAG_NOREPLY

There should not be reply messages to this message.

0x0004 SILC_MESSAGE_FLAG_ACTION

The sender is performing an action and the message is the indication of the action.

0x0008 SILC_MESSAGE_FLAG_NOTICE

The message is for example an informational notice type message.

0x0010 SILC_MESSAGE_FLAG_REQUEST

This is a generic request flag to send request messages. A separate document should define any payloads associated to this flag.

0x0020 SILC_MESSAGE_FLAG_SIGNED

This flag indicates that the message is signed with sender's private key and thus can be verified by the receiver using the sender's public key. A separate document should define the detailed procedure of the signing process and any associated payloads for this flag.

0x0040 SILC_MESSAGE_FLAG_REPLY

This is a generic reply flag to send a reply to previously received request. A separate document should define any payloads associated to this flag.

0x0080 SILC_MESSAGE_FLAG_DATA

This is a generic data flag, indicating that the message includes some data which can be interpreted in a specific way. Using this flag any kind of data can be delivered inside message payload. A separate document should define how this flag is interpreted and define any associated payloads.

0x0100 SILC_MESSAGE_FLAG_UTF8

This flag indicates that the message is UTF-8 encoded textual message. When sending text messages in SILC this flag SHOULD be used. When this flag is used the text sent as message MUST be UTF-8 encoded.

0x0200 SILC_MESSAGE_FLAG_ACK

This flag indicates the sender requires the recpipient to acknowledge the received message. This same flag is used in the acknowledgement. A separate document should define how the acknowledgement is performed.

0x0400 - 0x1000 RESERVED

Reserved for future flags.

0x2000 - 0x8000 PRIVATE RANGE

Private range for free use.

- Message Length (2 bytes) - Indicates the length of the Message Data field in the payload, not including any other field.

- Message Data (variable length) - The actual message data.

- Padding Length (2 bytes) - Indicates the length of the Padding field in the payload, not including any other field.

- Padding (variable length) - If this payload is used as channel messages, the padding MUST be applied because this payload is encrypted separately from other parts

of the packet. If this payload is used as private messages, the padding is present only when the payload is encrypted with private message key. If encrypted with session keys this field MUST NOT be present and the Padding Length field includes a zero (0) value. The padding SHOULD be random data.

- Initialization Vector (variable length) - This field MUST be present when this payload is used as channel messages. The IV SHOULD be random data for each channel message.

  When encrypting private messages with session keys this field MUST NOT be present. For private messages this field is present only when encrypting with a static private message key (pre-shared key). If randomly generated key material is used this field MUST NOT be present. Also, If Key Agreement (SKE) was used to negotiate fresh key material for private message key this field MUST NOT be present. See the section 4.6 in [SILC1] for more information about IVs when encrypting private messages.

  This field includes the initialization vector used in message encryption. It need to be used in the packet decryption as well. Contents of this field depends on the encryption algorithm and encryption mode. This field is not encrypted, is not included in padding calculation and its length equals to cipher's block size. This field is authenticated by the message MAC.

- MAC (variable length) - The MAC computed from the Message Flags, Message Length, Message Data, Padding Length, Padding and Initialization Vector fields in that order. The MAC is computed after the payload is encrypted. This is so called Encrypt-Then-MAC order; first encrypt, then compute MAC from ciphertext. The MAC protects the integrity of the Message Payload. Also, when used as channel messages it is possible to have multiple private channel keys set, and receiver can use the MAC to verify which of the keys must be used in decryption. This field is not present when encrypting private messages with session key. This field is not encrypted. This field is authenticated by the SILC packet MAC.

2.3.3 Disconnect Payload

Disconnect payload is sent upon disconnection. Reason of the disconnection is sent to the disconnected party in the payload.

The payload may only be sent with SILC_PACKET_DISCONNECT packet. It MUST NOT be sent in any other packet type. The following diagram represents the Disconnect Payload.

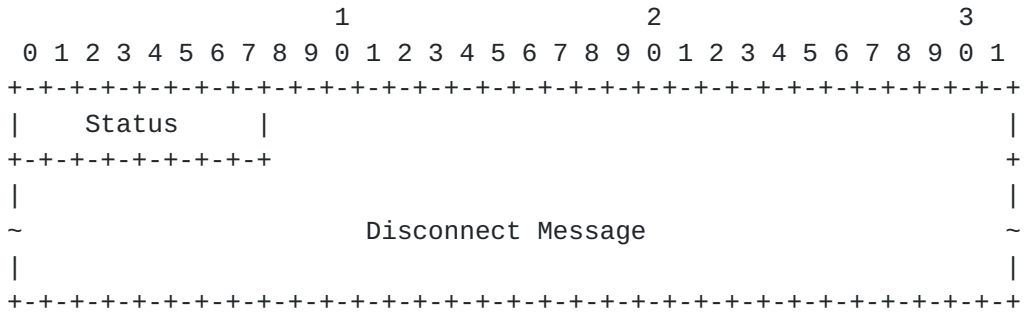


Figure 9: Disconnect Payload

- Status (1 byte) - Indicates the Status Type, defined in [SILC3] for the reason of disconnection.

- Disconnect Message (variable length) - Human readable UTF-8 encoded string indicating reason of the disconnection. This field MAY be omitted.

2.3.4 Success Payload

Success payload is sent when some protocol execution is successfully completed. The payload is simple; indication of the success is sent. This may be any data, including binary or human readable data, and it is protocol dependent.

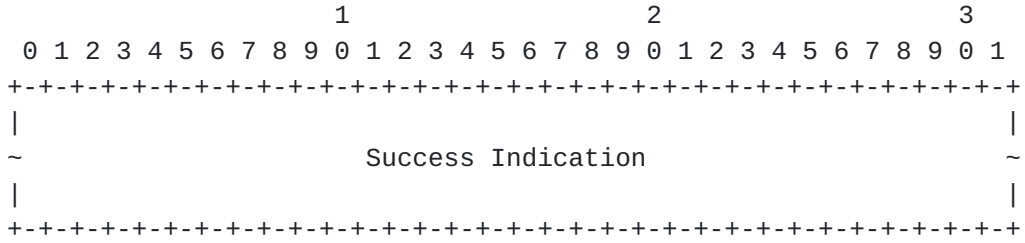


Figure 10: Success Payload

- Success Indication (variable length) - Indication of the success. This may be for example some flag that indicates the protocol and the success status or human readable success message. The true length of this payload is available by calculating it from the SILC

Packet Header.

2.3.5 Failure Payload

This is opposite of Success Payload. Indication of failure of some protocol is sent in the payload.

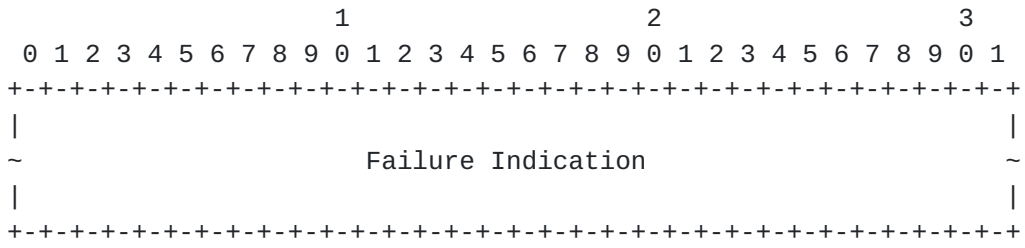


Figure 11: Failure Payload

- Failure Indication (variable length) - Indication of the failure. This may be for example some flag that indicates the protocol and the failure status or human readable failure message. The true length of this payload is available by calculating it from the SILC Packet Header.

2.3.6 Reject Payload

This payload is sent when some protocol is rejected to be executed. Other operations MAY send this as well that was rejected. The indication of the rejection is sent in the payload. The indication may be binary or human readable data and is protocol dependent.

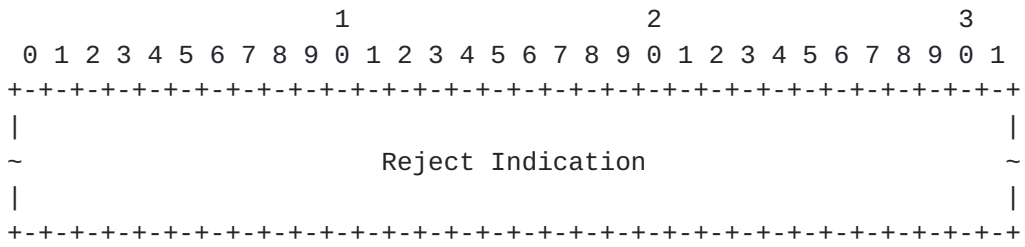


Figure 12: Reject Payload

- Reject Indication (variable length) - Indication of the rejection. This maybe for example some flag that

indicates the protocol and the rejection status or human readable rejection message. The true length of this payload is available by calculating it from the SILC Packet Header.

2.3.7 Notify Payload

Notify payload is used to send notify messages. The payload is usually sent from server to client and from server to router. It is also used by routers to notify other routers in the network. This payload MAY also be sent to a channel. Client MUST NOT send this payload. When this packet is received by client it SHOULD process it. Servers and routers MUST process notify packets.

The payload may only be sent with SILC_PACKET_NOTIFY packet. It MUST NOT be sent in any other packet type. The following diagram represents the Notify Payload.

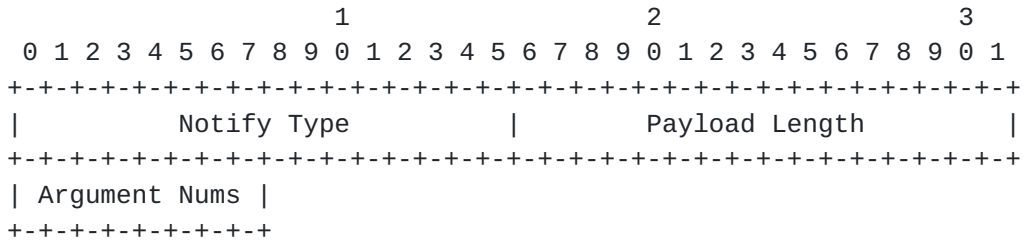


Figure 13: Notify Payload

- Notify Type (2 bytes) - Indicates the type of the notify message.
- Payload Length (2 bytes) - Length of the entire Notify Payload including any associated Argument Payloads.
- Argument Nums (1 byte) - Indicates the number of Argument Payloads associated to this payload. Notify types may define arguments to be sent along the notify message.

Following the list of currently defined notify types. The format for notify arguments is same as in SILC commands described in [SILC4]. Note that all IDs sent in arguments are sent inside ID Payload. Also note that all strings sent as arguments MUST be UTF-8 [RFC3629] encoded, unless otherwise defined. Also note that all public keys or

certificates sent inside arguments are actually Public Key Payloads.

0 SILC_NOTIFY_TYPE_NONE

If no specific notify type apply for the notify message this type MAY be used.

```
Max Arguments:  1
    Arguments:  (1) <message>
```

The <message> is implementation specific free text string. Receiver MAY ignore this message.

1 SILC_NOTIFY_TYPE_INVITE

Sent when an client is invited to a channel. This is also sent when the invite list of the channel is changed. This notify type is sent to local servers on the channel, but MUST NOT be sent to clients on the channel. Router MUST broadcast this to its primary router and to local servers on the channel. When a client was directly invited to the channel this is also sent to that client. In this case the packet is destined to the client.

```
Max Arguments:  5
    Arguments:  (1) <Channel ID>          (2) <channel name>
                (3) [<sender Client ID>]  (4) [<add | del>]
                (5) [<invite list>]
```

The <Channel ID> is the channel. The <channel name> is the name of the channel and is provided because the client which receives this notify packet may not have a way to resolve the name of the channel from the <Channel ID>. The <sender Client ID> is the Client ID which invited the client to the channel. The <add | del> is an argument of size of 1 byte where 0x00 means adding a client to invite list, and 0x01 means deleting a client from invite list. The <invite list>, if present, indicates the information to be added to or removed from the invite list. The <invite list> format is defined in [SILC4] with SILC_COMMAND_INVITE command. When this notify is destined to a client the <add | del> and <invite list> MUST NOT be sent. When <add | del> is used to announce information during server connecting phase the argument type MUST be 0x03. See section 4.2.1 in [SILC1] for more information.

2 SILC_NOTIFY_TYPE_JOIN

Sent when client has joined to a channel. The server MUST distribute this type to the local clients on the channel and then send it to its primary router. Note that, when router is joining the client on behalf of normal server then router MUST send this notify type locally and globally. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network. This notify is sent also to the client that joined the channel.

Max Arguments: 2
Arguments: (1) [<Client ID>] (2) <Channel ID>

The <Client ID> is the client that joined to the channel indicated by the <Channel ID>.

3 SILC_NOTIFY_TYPE_LEAVE

Sent when client has left a channel. The server must distribute this type to the local clients on the channel and then send it to its primary router. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network. This notify MUST NOT be sent to the leaving client.

Max Arguments: 1
Arguments: (1) <Client ID>

The <Client ID> is the client which left the channel.

4 SILC_NOTIFY_TYPE_SIGNOFF

Sent when client signoff from SILC network. The server MUST distribute this type to the local clients on the channel and then send it to its primary router. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network. This notify MUST NOT be sent to the quitting client. The Destination ID in the packet may be any ID depending to who it is destined.

Max Arguments: 2
Arguments: (1) <Client ID> (2) <message>

The <Client ID> is the client which left SILC network. The <message> is free text string indicating the reason of the signoff.

5 SILC_NOTIFY_TYPE_TOPIC_SET

Sent when topic is set/changed on a channel. This type may be sent only to the clients which are joined on the channel which topic was just set or changed. The packet is destined to the channel.

Max Arguments: 2
Arguments: (1) <ID Payload> (2) <topic>

The <ID Payload> is the ID of the entity who set the topic. It usually is Client ID but it can be Server ID and Channel ID as well.

6 SILC_NOTIFY_TYPE_NICK_CHANGE

Sent when client changes nick on a channel. The server MUST distribute this type only to the local clients on the channel and then send it to its primary router. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network. This packet is destined directly to the sent entity. This MUST be sent to those clients that are joined on same channels as the client that changed the nickname. This notify MUST NOT be sent multiple times to the same recipient. This notify MUST be sent also to the client that changed the nickname.

Max Arguments: 3
Arguments: (1) <Old Client ID> (2) <New Client ID>
(3) <nickname>

The <Old Client ID> is the old ID of the client which changed the nickname. The <New Client ID> is the new ID generated by the change of the nickname. The <nickname> is the new nickname. Note that it is possible to send this notify even if the nickname has not changed, but client ID was changed.

7 SILC_NOTIFY_TYPE_CMODE_CHANGE

Sent when channel mode has changed. This type MUST be sent only to the clients which are joined on the channel which mode was changed. This packet is destined to the channel.

Max Arguments: 8
Arguments: (1) <ID Payload> (2) <mode mask>
(3) [<cipher>] (4) <[hmac>]

(5) [<passphrase>] (6) [<founder public key>]
(7) [<channel pubkey>] (8) [<user limit>]

The <ID Payload> is the ID (usually Client ID but it can be Server ID as well when the router is enforcing channel mode change) of the entity which changed the mode. The <mode mask> is the new mode mask of the channel. The client can safely ignore the <cipher> argument since the SILC_PACKET_CHANNEL_KEY packet will force the new channel key change anyway. The <hmac> argument is important since the client is responsible of setting the new HMAC and the hmac key into use. The <passphrase> is the passphrase of the channel, if it was now set. The <founder public key> argument is sent when the founder mode on the channel was set. All routers and servers that receive the packet MUST save the founder's public key so that the founder can reclaim the channel founder rights back for the channel on any server in the network. The <user limit> argument is present when the user limit was set or changed on the channel.

The <channel pubkey> is an Argument List Payload and it is used to add and/or remove channel public keys from the channel. Also, when announcing channel information between servers and routers during connecting phase this argument includes the list of channel public keys. To add a public key to channel public key list the SILC_CMODE_CHANNEL_AUTH mode is set and the argument type is 0x00, and the argument is the public key. To remove a public key from the channel public key list the argument type is 0x01, and the argument is the public key to be removed. If the mode SILC_CMODE_CHANNEL_AUTH is unset (and was set earlier) all public keys are removed at once. Implementation MAY add and remove multiple public keys at the same time by including multiple arguments to the <channel pubkey> Argument List Payload where each argument is one Public Key Payload. When <channel pubkey> is used to announce information during server connecting phase the argument type MUST be 0x03. See section 4.2.1 in [SILC1] for more information.

8 SILC_NOTIFY_TYPE_CUMODE_CHANGE

Sent when user mode on channel has changed. This type MUST be sent only to the clients which are joined on the channel where the target client is on. This packet is destined to the channel.

Max Arguments: 4
Arguments: (1) <ID Payload> (2) <mode mask>
(3) <Target Client ID> (4) [<founder pubkey>]

The <ID Payload> is the ID (usually Client ID but it can be Server ID as well when the router is enforcing user's mode change) of the entity which changed the mode. The <mode mask> is the new mode mask of the channel. The <Target Client ID> is the client which mode was changed. The <founder pubkey> is the public key of the channel founder and may be sent only when first time setting the channel founder mode using the SILC_COMMAND_CUMODE command, and when sending this notify.

9 SILC_NOTIFY_TYPE_MOTD

Sent when Message of the Day (motd) is sent to a client.

Max Arguments: 1
Arguments: (1) <motd>

The <motd> is the Message of the Day. This notify MAY be ignored and is OPTIONAL.

10 SILC_NOTIFY_TYPE_CHANNEL_CHANGE

Sent when channel's ID has changed for a reason or another. This is sent by normal server to the client. This can also be sent by router to other server to force the Channel ID change. The Channel ID MUST be changed to use the new one. When sent to clients, this type MUST be sent only to the clients which are joined on the channel. This packet is destined to the sent entity.

Max Arguments: 2
Arguments: (1) <Old Channel ID> (2) <New Channel ID>

The <Old Channel ID> is the channel's old ID and the <New Channel ID> is the new one that MUST replace the old one. Server which receives this from router MUST re-announce the channel to the router by sending SILC_PACKET_NEW_CHANNEL packet with the new Channel ID.

11 SILC_NOTIFY_TYPE_SERVER_SIGNOFF

Sent when server quits SILC network. Those clients from this server that are on channels must be removed from the channel. This packet is destined to the sent entity.

Max Arguments: 256

```
    Arguments:  (1) <Server ID>   (n) [<Client ID>]   [...]
```

The <Server ID> is the server's ID. The rest of the arguments are the Client IDs of the clients which are coming from this server and are thus quitting the SILC network also. If the maximum number of arguments are reached another SILC_NOTIFY_TYPE_SERVER_SIGNOFF notify packet MUST be sent. When this notify packet is sent between routers the Client ID's MAY be omitted. Server receiving the Client ID's in the payload may use them directly to remove the client.

12 SILC_NOTIFY_TYPE_KICKED

Sent when a client has been kicked from a channel. This MUST also be sent to the client which was kicked from the channel. The client which was kicked from the channel MUST be removed from the channel. The client MUST also be removed from channel's invite list if it is explicitly added in the list. This packet is destined to the channel. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network.

```
Max Arguments:  3
    Arguments:  (1) <Client ID>          (2) [<comment>]
                (3) <Kicker's Client ID>
```

The <Client ID> is the client which was kicked from the channel. The kicker may have set the <comment> string to indicate the reason for the kicking. The <Kicker's Client ID> is the kicker.

13 SILC_NOTIFY_TYPE_KILLED

Sent when a client has been killed from the network. This MUST also be sent to the client which was killed from the network. This notify MUST be sent to those clients which are joined on same channels as the killed client. The client which was killed MUST be removed from the network. This packet is destined directly to the sent entity. The router or server receiving the packet distributes this type to the local clients on the channel and broadcast it to the network. The client MUST also be removed from joined channels invite list if it is explicitly added in the lists. This notify MUST NOT be sent multiple times to same recipient.

```
Max Arguments:  3
    Arguments:  (1) <Client ID>          (2) [<comment>]
```

(3) <Killer's ID>

The <Client ID> is the client which was killed from the network. The killer may have set the <comment> string to indicate the reason for the killing. The <Killer's ID> is the killer, which may be client but also router server.

14 SILC_NOTIFY_TYPE_UMODE_CHANGE

Sent when user's mode in the SILC changes. This type is sent only between routers as broadcast packet.

Max Arguments: 2
Arguments: (1) <Client ID> (2) <mode mask>

The <Client ID> is the client which mode was changed. The <mode mask> is the new mode mask.

15 SILC_NOTIFY_TYPE_BAN

Sent when the ban list of the channel is changed. This notify type is sent to local servers on the channel, but MUST NOT be sent to clients on the channel. Router MUST broadcast this to its primary router and to local servers on the channel.

Max Arguments: 3
Arguments: (1) <Channel ID> (2) [<add | del>]
(3) [<ban list>]

The <Channel ID> is the channel which ban list was changed. The <add | del> is an argument of size of 1 byte where 0x00 means adding a client to ban list, and 0x01 means deleting a client from ban list. The <ban list> indicates the information to be added to or removed from the ban list. The <ban list> format format is defined in [SILC4] with SILC_COMMAND_BAN command. When <add | del> is used to announce information during server connecting phase the argument type MUST be 0x03. See section 4.2.1 in [SILC1] for more information.

16 SILC_NOTIFY_TYPE_ERROR

Sent when an error occurs during processing some SILC procedure. This is not used when error occurs during command processing, see [SILC4] for more information about commands and command replies. This type is sent directly to the sender of the packet whose

packet caused the error. See [SILC1] for definition when this type can be sent.

Max Arguments: 256
Arguments: (1) <Status Type> (n) [...]

The <Status Type> is the error type defined in [SILC4]. Note that same types are also used with command replies to indicate the status of a command. Both commands and this notify type share same status types. Rest of the arguments are status type dependent and are specified with those status types that can be sent currently inside this notify type in [SILC4]. The <Status Type> is size of 1 byte.

17 SILC_NOTIFY_TYPE_WATCH

Sent to indicate change in a watched user. Client can set nicknames to be watched with SILC_COMMAND_WATCH command, and receive notifications when they login to network, signoff from the network or their user mode is changed. This notify type is used to deliver these notifications. The notify type is sent directly to the watching client.

Max Arguments: 5
Arguments: (1) <Client ID> (2) [<nickname>]
(3) <user mode> (4) [<Notify Type>]
(5) [<public key>]

The <Client ID> is the user's Client ID which is being watched, and the <nickname> is its nickname. If the client just changed the nickname, then <nickname> is the new nickname, but the <Client ID> is the old client ID. The <user mode> is the user's current user mode. The <Notify Type> can be same as the Notify Payload's Notify Type, and is 16 bit MSB first order value. If provided it may indicate the notify that occurred for the client. If client logged in to the network the <Notify Type> MUST NOT be present. The <public key> MAY be present, and it is the public key of the client being watched.

Notify types starting from 16384 are reserved for private notify message types.

Router server which receives SILC_NOTIFY_TYPE_SIGNOFF, SILC_NOTIFY_TYPE_SERVER_SIGNOFF, SILC_NOTIFY_TYPE_KILLED, SILC_NOTIFY_TYPE_NICK_CHANGE and SILC_NOTIFY_TYPE_UMODE_CHANGE MUST check whether someone in the local cell is watching the nickname the client has, and send the SILC_NOTIFY_TYPE_WATCH notify to the

watcher, unless the watched client in case has the user mode SILC_UMODE_REJECT_WATCHING set. If the watcher client and the client that was watched is same the notify SHOULD NOT be sent.

2.3.8 Error Payload

Error payload is sent upon error in protocol. Error may occur in various conditions when server sends this packet. Client MUST NOT send this payload but MUST be able to accept it. However, client MAY ignore the contents of the packet as server is going to take action on the error anyway. However, it is recommended that the client takes error packet seriously.

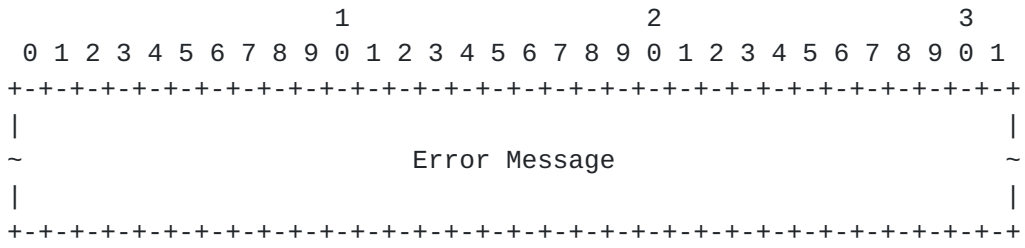


Figure 14: Error Payload

o Error Message (variable length) - Human readable error message.

2.3.9 Channel Message Payload

Channel Message Payload is used to send message to channels, a group of users. These messages can only be sent if client has joined to some channel. Even though this packet is very common in SILC it is still special packet. Some special handling on sending and reception of channel message is required.

Padding MUST be applied into this payload since the payload is encrypted separately from other parts of the packet with the channel specific key. Hence the requirement of the padding. The packet MUST be made multiple by eight (8) or by the block size of the cipher, which ever is larger.

The SILC header in this packet is encrypted with the session key of the next receiver of the packet. Nothing else is encrypted with that key. Thus, the actual packet and padding to be encrypted with the session key is SILC Header plus padding to it.

Receiver of the the channel message packet is able to determine the channel the message is destined to by checking the Destination ID from the SILC Packet header which tells the destination channel. The original sender of the packet is also determined by checking the source ID from the header which tells the client which sent the message. The Destination ID MUST be Channel ID in the SILC Packet header.

This packet use generic Message Payload as Channel Message Payload. See section 2.3.2.6 for generic Message Payload.

2.3.10 Channel Key Payload

All traffic in channels are protected by channel specific keys. Channel Key Payload is used to distribute channel keys to all clients on the particular channel. Channel keys are sent when the channel is created, when new user joins to the channel and whenever a user has left a channel. Server creates the new channel key and distributes it to the clients by encrypting this payload with the session key shared between the server and the client. After that, client MUST start using the key received in this payload to protect the traffic on the channel.

The client which is joining to the channel receives its key in the SILC_COMMAND_JOIN command reply message thus it is not necessary to send this payload to the entity which sent the SILC_COMMAND_JOIN command.

Channel keys are cell specific thus every router in the cell have to create a channel key and distribute it if any client in the cell has joined to a channel. Channel traffic between cell's are not encrypted using channel keys, they are encrypted using normal session keys between two routers. Inside a cell, all channel traffic is encrypted with the specified channel key. Channel key SHOULD expire periodically, say, in one hour, in which case new channel key is created and distributed.

Note that, this packet is not used if SILC_CMODE_PRIVKEY mode is set on channel. This means that channel uses channel private keys which are not server generated. For this reason server cannot send this packet as it does not know the key.

The destination ID in the packet SHOULD be the entity to whom the packet is sent. Using Channel ID as destination ID is not necessary as the Channel ID is included in the Channel Key Payload.

The payload may only be sent with SILC_PACKET_CHANNEL_KEY packet.

It MUST NOT be sent in any other packet type. The following diagram represents the Channel Key Payload.

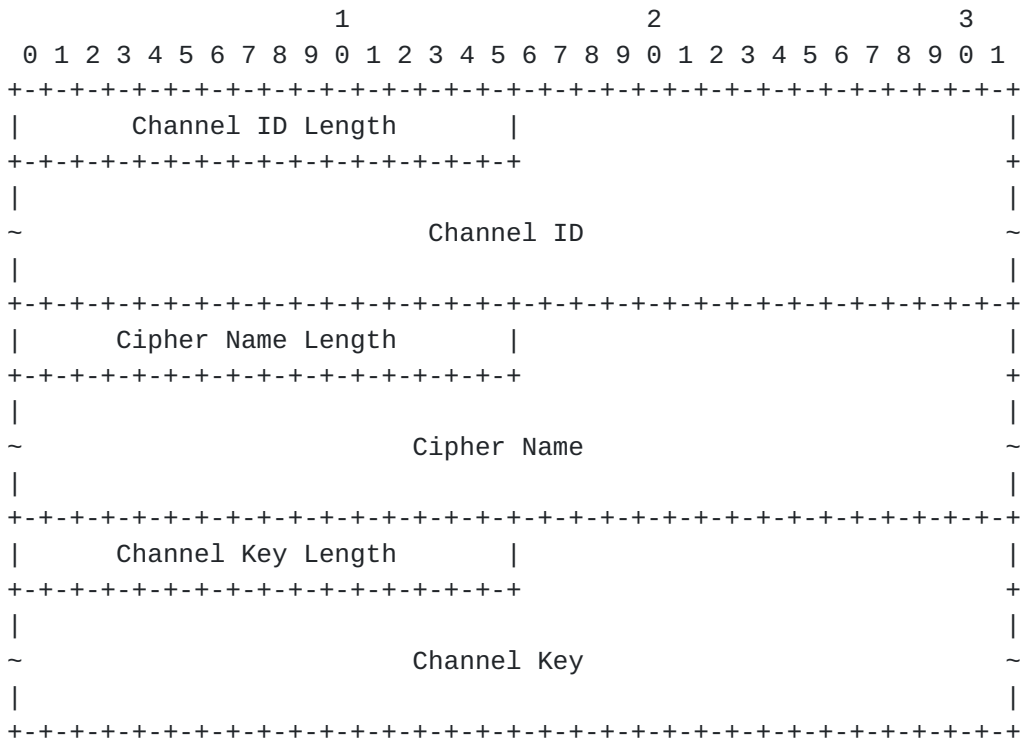


Figure 15: Channel Key Payload

- Channel ID Length (2 bytes) - Indicates the length of the Channel ID field in the payload, not including any other field.

- Channel ID (variable length) - The Channel ID of the channel.

- Cipher Name Length (2 bytes) - Indicates the length of the Cipher name field in the payload, not including any other field.

- Cipher Name (variable length) - Name of the cipher used in the protection of channel traffic. This name is initially decided by the creator of the channel but it may change during the life time of the channel as well.

- Channel Key Length (2 bytes) - Indicates the length of the

Channel Key field in the payload, not including any other field.

o Channel Key (variable length) - The actual channel key material. See [SILC1] on how to start using the key.

2.3.11 Private Message Payload

Private Message Payload is used to send private message between two clients. The messages are sent only to the specified user and no other user inside SILC network is able to see the message.

The message can be protected by the session key established by the SILC Key Exchange Protocol. However, it is also possible to agree to use a private message key to protect just the private messages. It is for example possible to perform Key Agreement between two clients. See section 2.3.20 Key Agreement Payload how to perform key agreement. It is also possible to use static or pre-shared keys to protect private messages. See the 2.3.12 Private Message Key Payload and [SILC1] section 4.6 for detailed description for private message key generation.

If normal session key is used to protect the message, every server between the sender client and the receiving client MUST decrypt the packet and always re-encrypt it with the session key of the next receiver of the packet. See section Client To Client in [SILC1].

When the private message key is used, and the Private Message Key flag was set in the SILC Packet header no server or router en route is able to decrypt or re-encrypt the packet. In this case only the SILC Packet header is processed by the servers and routers en route. Section Client To Client in [SILC1] gives example of this scheme.

This packet use generic Message Payload as Private Message Payload. See section 2.3.2.6 for generic Message Payload.

2.3.12 Private Message Key Payload

This payload is OPTIONAL and can be used to indicate that a static or pre-shared key should be used in the private message communication to protect the messages. The actual key material has to be sent outside the SILC network, or it has to be a static or pre-shared key. The sender of this packet is considered to be the initiator and the receiver the responder when processing the raw key material as described in the section 4.6 in [SILC1] and in the section 2.3 in [SILC3].

Note that it is also possible to use static or pre-shared keys in client implementations without sending this packet. Clients may naturally agree to use a key without sending any kind of indication to each other. The key may be for example a long-living static key that the clients has agreed to use at all times. Note that it is also possible to agree to use private message key by performing a Key Agreement. See the section 2.3.20 Key Agreement Payload.

This payload may only be sent by client to another client. Server MUST NOT send this payload. After sending this payload and setting the key into use this payload the sender of private messages MUST set the Private Message Key flag into the SILC Packet Header.

The payload may only be sent with SILC_PACKET_PRIVATE_MESSAGE_KEY packet. It MUST NOT be sent in any other packet type. The following diagram represents the Private Message Key Payload.

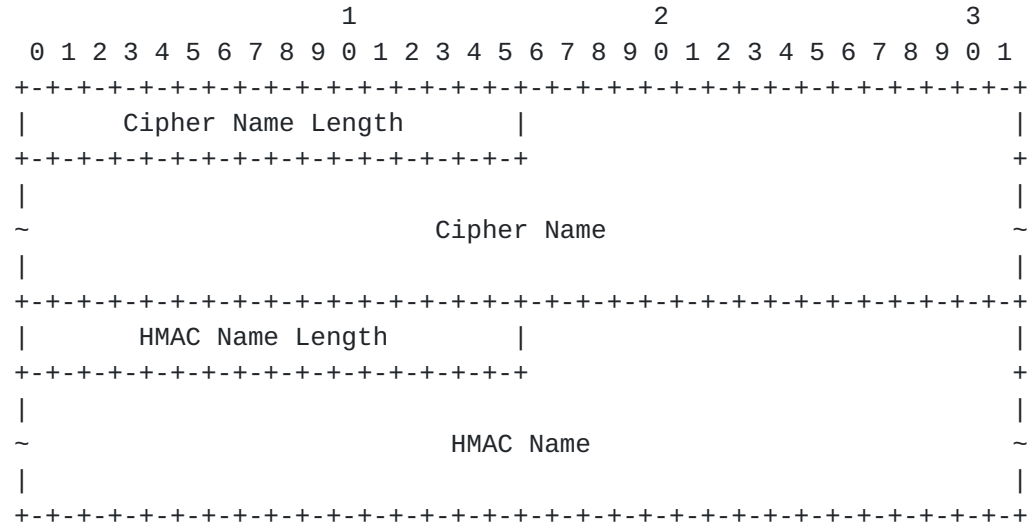


Figure 16: Private Message Key Payload

- Cipher Name Length (2 bytes) - Indicates the length of the Cipher Name field in the payload, not including any other field.

- Cipher Name (variable length) - Name of the cipher to use in the private message encryption. If this field does not exist then the default cipher of the SILC protocol is used. See the [SILC1] for defined ciphers.

- HMAC Name Length (2 bytes) - Indicates the length of the

HMAC Name field in the payload, not including any other field.

o HMAC Name (variable length) - Name of the HMAC to use in the private message MAC computation. If this field does not exist then the default HMAC of the SILC protocol is used. See the [SILC1] for defined HMACs.

2.3.13 Command Payload

Command Payload is used to send SILC commands from client to server. Also server MAY send commands to other servers. The following diagram represents the Command Payload.

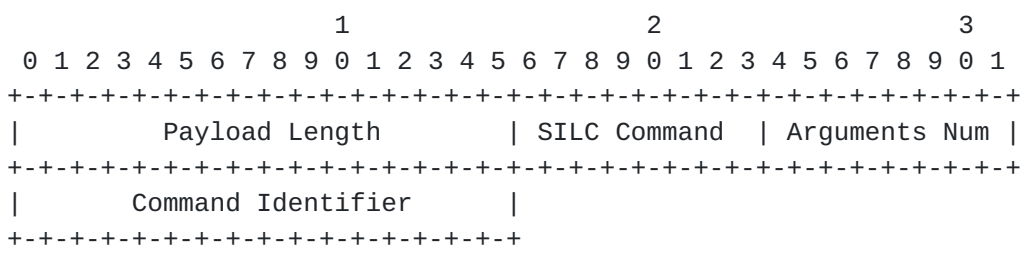


Figure 17: Command Payload

o Payload Length (2 bytes) - Length of the entire command payload including any command argument payloads associated with this payload.

o SILC Command (1 byte) - Indicates the SILC command. This MUST be set to non-zero value. If zero (0) value is found in this field the packet MUST be discarded.

o Arguments Num (1 byte) - Indicates the number of arguments associated with the command. If there are no arguments this field is set to zero (0). The arguments MUST follow the Command Payload. See section 2.3.2.2 for definition of the Argument Payload.

o Command Identifier (2 bytes) - Identifies this command at the sender's end. The entity which replies to this command MUST set the value found from this field into the Command Payload used to send the reply to the sender. This way the sender can identify which command reply belongs to which originally sent command. What this field includes is implementation issue but it is RECOMMENDED that wrapping counter value is

used in the field.

See [SILC4] for detailed description of different SILC commands, their arguments and their reply messages.

2.3.14 Command Reply Payload

Command Reply Payload is used to send replies to the commands. The Command Reply Payload is identical to the Command Payload thus see the 2.3.13 section for the payload specification.

The entity which sends the reply packet MUST set the Command Identifier field in the reply packet's Command Payload to the value it received in the original command packet.

See SILC Commands in [SILC4] for detailed description of different SILC commands, their arguments and their reply messages.

2.3.15 Connection Auth Request Payload

Client MAY send this payload to server to request the authentication method that must be used in authentication protocol. If client knows this information beforehand this payload is not necessary to be sent. Server performing authentication with another server MAY also send this payload to request the authentication method. If the connecting server already knows this information this payload is not necessary to be sent.

Server receiving this request SHOULD reply with same payload sending the mandatory authentication method. Algorithms that may be required to be used by the authentication method are the ones already established by the SILC Key Exchange protocol. See section Key Exchange Start Payload in [SILC3] for detailed information.

The payload may only be sent with SILC_PACKET_CONNECTION_AUTH_REQUEST packet. It MUST NOT be sent in any other packet type. The following diagram represents the Connection Auth Request Payload.

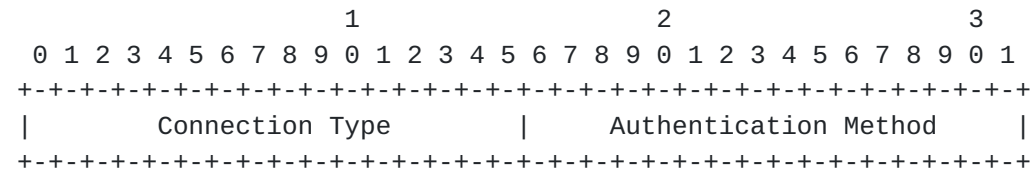


Figure 18: Connection Auth Request Payload

- Connection Type (2 bytes) - Indicates the type of the connection. The following connection types are defined:

  1 Client connection
  2 Server connection
  3 Router connection

  If any other type is found in this field the packet MUST be discarded and the authentication MUST be failed.

- Authentication Method (2 bytes) - Indicates the authentication method to be used in the authentication protocol. The following authentication methods are defined:

  0 NONE (mandatory)
  1 password (mandatory)
  2 public key (mandatory)

  If any other type is found in this field the packet MUST be discarded and the authentication MUST be failed. If this payload is sent as request to receive the mandatory authentication method this field MUST be set to zero (0), indicating that receiver should send the mandatory authentication method. The receiver sending this payload to the requesting party, MAY also set this field to zero (0) to indicate that authentication is not required. In this case authentication protocol still MUST be started but server is most likely to respond with SILC_PACKET_SUCCESS immediately.

2.3.16 New ID Payload

New ID Payload is a multipurpose payload. It is used to send newly created ID's from clients and servers. When client connects to server and registers itself to the server by sending SILC_PACKET_NEW_CLIENT packet, server replies with this packet by sending the created ID for the client. Server always creates the ID for the client.

This payload is also used when server tells its router that new client has registered to the SILC network. In this case the server sends the Client ID of the client to the router. Similarly when router distributes information to other routers about the client in the SILC network this payload is used.

Also, when server connects to router, router use this payload to inform other routers about new server in the SILC network. However, every

server (or router) creates their own ID's thus the ID distributed by this payload is not created by the distributor in this case. Servers create their own ID's. Server registers itself to the network by sending SILC_PACKET_NEW_SERVER to the router it connected to. The case is same when router connects to another router.

This payload MUST NOT be used to send information about new channels. New channels are always distributed by sending the dedicated SILC_PACKET_NEW_CHANNEL packet. Client MUST NOT send this payload. Both client and server (and router) MAY receive this payload.

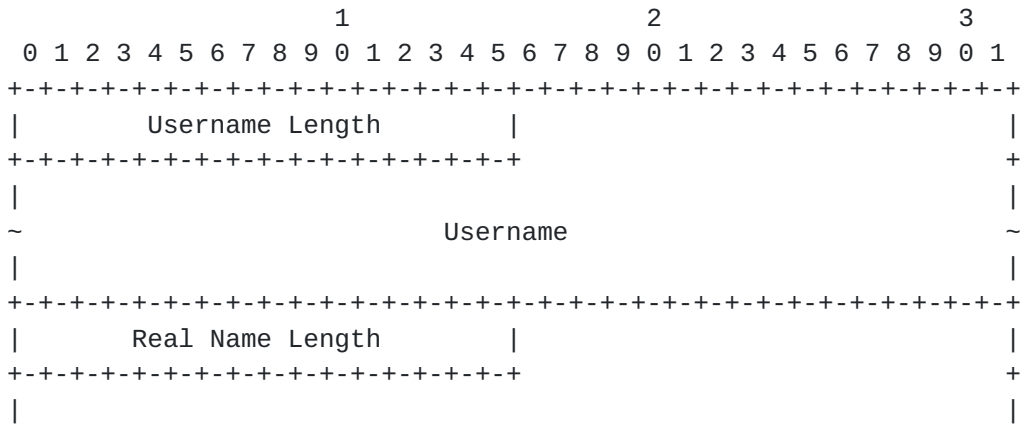
The packet use generic ID Payload as New ID Payload. See section 2.3.2.1 for generic ID Payload.

2.3.17 New Client Payload

When client is connected to the server, keys has been exchanged and connection has been authenticated, client MUST register itself to the server. Client's first packet after key exchange and authentication protocols MUST be SILC_PACKET_NEW_CLIENT. This payload tells server all the relevant information about the connected user. Server creates a new client ID for the client when received this payload and sends it to the client in New ID Payload.

This payload sends username and real name of the user on the remote host which is connected to the SILC server with SILC client. The server creates the client ID according the information sent in this payload. The nickname of the user becomes the nickname sent in this payload.

The payload may only be sent with SILC_PACKET_NEW_CLIENT packet. It MUST NOT be sent in any other packet type. The following diagram represents the New Client Payload.



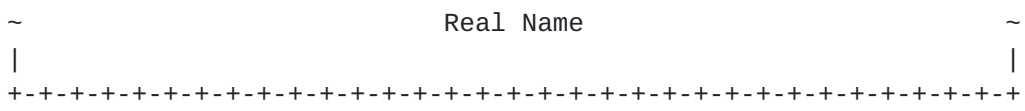


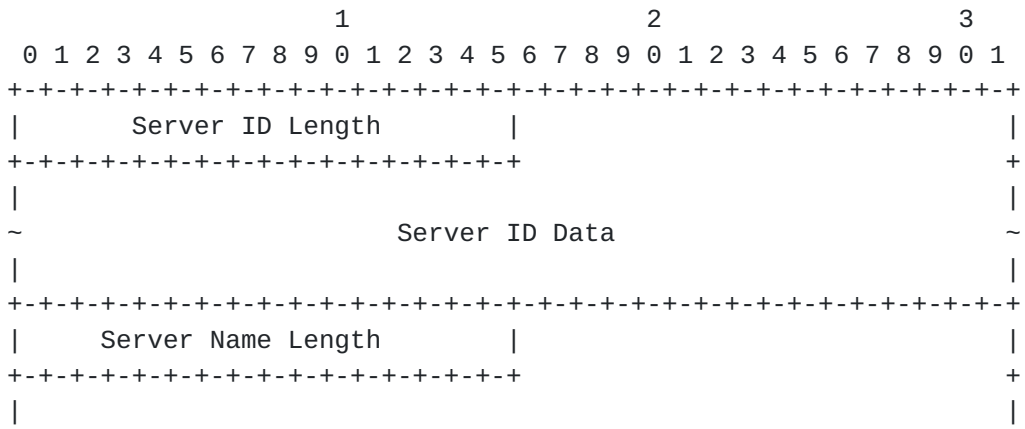
Figure 19: New Client Payload

- o Username Length (2 bytes) - Length of the Username field.
- o Username (variable length) - The username of the user on the host where connecting to the SILC server.
- o Real Name Length (2 bytes) - Length of the Real Name field.
- o Real Name (variable length) - The real name of the user on the host where connecting to the SILC server.

2.3.18 New Server Payload

This payload is sent by server when it has completed successfully both key exchange and connection authentication protocols. The server MUST register itself to the SILC Network by sending this payload. The first packet after these key exchange and authentication protocols is SILC_PACKET_NEW_SERVER packet. The payload includes the Server ID of the server that it has created by itself. It also includes a name of the server that is associated to the Server ID.

The payload may only be sent with SILC_PACKET_NEW_SERVER packet. It MUST NOT be sent in any other packet type. The following diagram represents the New Server Payload.



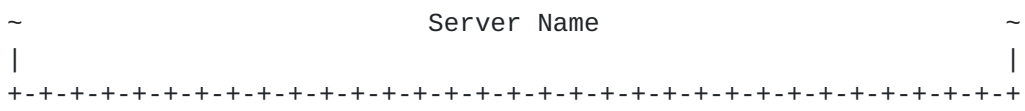


Figure 20: New Server Payload

- Server ID Length (2 bytes) - Length of the Server ID Data field.
- Server ID Data (variable length) - The encoded Server ID data.
- Server Name Length (2 bytes) - Length of the server name field.
- Server Name (variable length) - The server name string.

2.3.19 New Channel Payload

Information about newly created channel is broadcasted to all routers in the SILC network by sending this packet payload. Channels are created by router of the cell. Server never creates channels unless it is a standalone server and it does not have router connection, in this case server acts as router. Normal server send JOIN command to the router (after it has received JOIN command from client) which then processes the command and creates the channel. Client MUST NOT send this packet. Server MAY send this packet to a router when it is announcing its existing channels to the router after it has connected to the router.

The packet use generic Channel Payload as New Channel Payload. See section 2.3.2.3 for generic Channel Payload. The Mode Mask field in the Channel Payload is the mode of the channel.

2.3.20 Key Agreement Payload

This payload is used by clients to request key negotiation between another client in the SILC Network. The key agreement protocol used is the SKE protocol. The result of the protocol, the secret key material, can be used for example as private message key between the two clients. This significantly adds security as the clients agree about the key without any server interaction. The protocol is executed peer to peer. The server and router MUST NOT send this payload.

The sender MAY tell the receiver of this payload the hostname and the

port where the SKE protocol is running in the sender's end. The receiver MAY then initiate the SKE negotiation with the sender. The sender MAY also optionally not to include the hostname and the port of its SKE protocol. In this case the receiver MAY reply to the request by sending the same payload filled with the receiver's hostname and the port where the SKE protocol is running. The sender MAY then initiate the SKE negotiation with the receiver.

This payload may be sent with SILC_PACKET_KEY_AGREEMENT and SILC_PACKET_FTP packet types. It MUST NOT be sent in any other packet types. The following diagram represents the Key Agreement Payload.

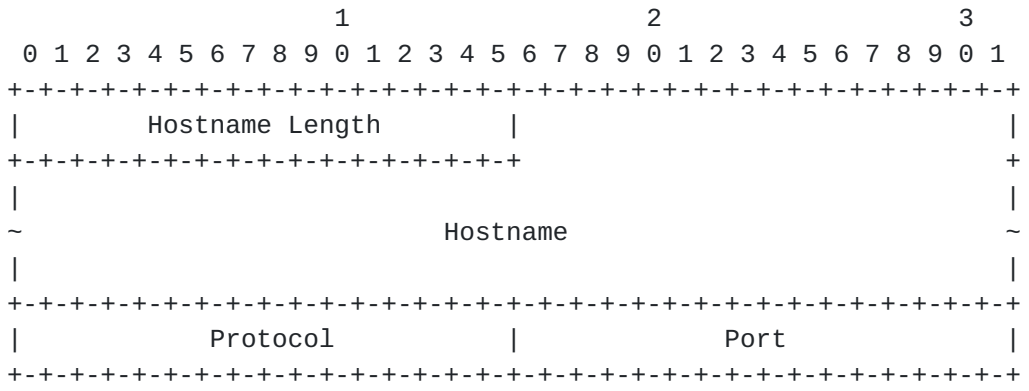


Figure 21: Key Agreement Payload

- Hostname Length (2 bytes) - Indicates the length of the Hostname field.

- Hostname (variable length) - The hostname or IP address where the SKE protocol is running, as UTF-8 encoded string. The sender MAY fill this field when sending the payload. If the receiver sends this payload as reply to the request it MUST fill this field.

- Protocol (2 bytes) - The internet protocol used for the key agreement connection. Possible values are 0 for TCP and 1 for UDP. Other values are unsupported. This is a 16 bit MSB first order value. If Hostname field is not present, the value in this field is ignored.

- Port (2 bytes) - The port where the SKE protocol is bound. The sender MAY fill this field when sending the payload. If the receiver sends this payload as reply to the request it MUST fill this field. This is a 16 bit MSB first order value.

After the key material has been received from the SKE protocol it is processed as the [SILC3] describes. If the key material is used as channel private key then the Sending Encryption Key, as defined in [SILC3] is used as the channel private key. Other key material must be discarded. The [SILC1] in section 4.6 defines the way to use the key material if it is intended to be used as private message keys. Any other use for the key material is undefined.

2.3.21 Resume Router Payload

See the [SILC1] for Resume Router protocol where this payload is used. The payload may only be sent with SILC_PACKET_RESUME_ROUTER packet. It MUST NOT be sent in any other packet type. The following diagram represents the Resume Router Payload.

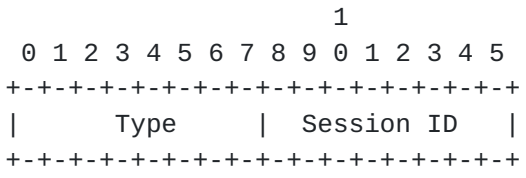


Figure 22: Resume Router Payload

- Type (1 byte) - Indicates the type of the backup resume protocol packet. The type values are defined in [SILC1].

- Session ID (1 bytes) - Indicates the session ID for the backup resume protocol. The sender of the packet sets this value and the receiver MUST set the same value in subsequent reply packet.

2.3.22 File Transfer Payload

File Transfer Payload is used to perform file transfer protocol between two entities in the network. The actual file transfer protocol is always encapsulated inside the SILC Packet. The actual data stream is also sent peer to peer outside SILC network.

When an entity, usually a client wishes to perform file transfer protocol with another client in the network, they perform Key Agreement protocol as described in the section 2.3.20 Key Agreement Payload and in [SILC3], inside File Transfer Payload. After the Key Agreement protocol has been

performed the subsequent packets in the data stream will be protected using the new key material. The actual file transfer protocol is also initialized in this stage. All file transfer protocol packets are always encapsulated in the File Transfer Payload and protected with the negotiated key material.

The payload may only be sent with SILC_PACKET_FTP packet. It MUST NOT be sent in any other packet type. The following diagram represents the File Transfer Payload.

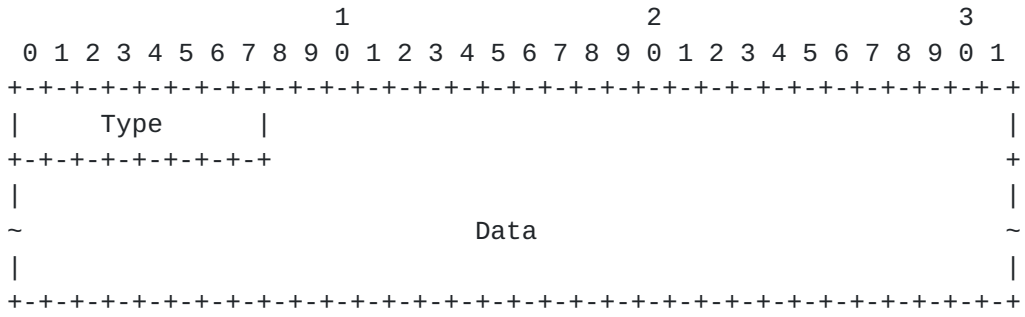


Figure 23: File Transfer Payload

o Type (1 byte) - Indicates the type of the file transfer protocol. The following file transfer protocols has been defined:

1 Secure File Transfer Protocol (SFTP) (mandatory)

If zero (0) value or any unsupported file transfer protocol type is found in this field the packet MUST be discarded. The currently mandatory file transfer protocol is SFTP. The SFTP protocol is defined in [SFTP].

o Data (variable length) - Arbitrary file transfer data. The contents and encoding of this field is dependent of the usage of this payload and the type of the file transfer protocol. When this payload is used to perform the Key Agreement protocol, this field include the Key Agreement Payload, as defined in the section 2.3.20 Key Agreement Payload. When this payload is used to send the actual file transfer protocol data, the encoding is defined in the corresponding file transfer protocol.

2.3.23 Resume Client Payload

This payload is used by client to resume its detached session in the SILC Network. A client is able to detach itself from the network by sending SILC_COMMAND_DETACH command to its server. The network connection to the client is lost but the client remains as valid client in the network. The client is able to resume the session back by sending this packet and including the old Client ID, and an Authentication Payload [SILC1] which the server use to verify with the detached client's public key. This also implies that the mandatory authentication method is public key authentication.

Server or router that receives this from the client also sends this, without the Authentication Payload, to routers in the network so that they know the detached client has resumed. Refer to the [SILC1] for detailed description how the detaching and resuming procedure is performed.

The payload may only be sent with SILC_PACKET_RESUME_CLIENT packet. It MUST NOT be sent in any other packet type. The following diagram represents the Resume Client Payload.

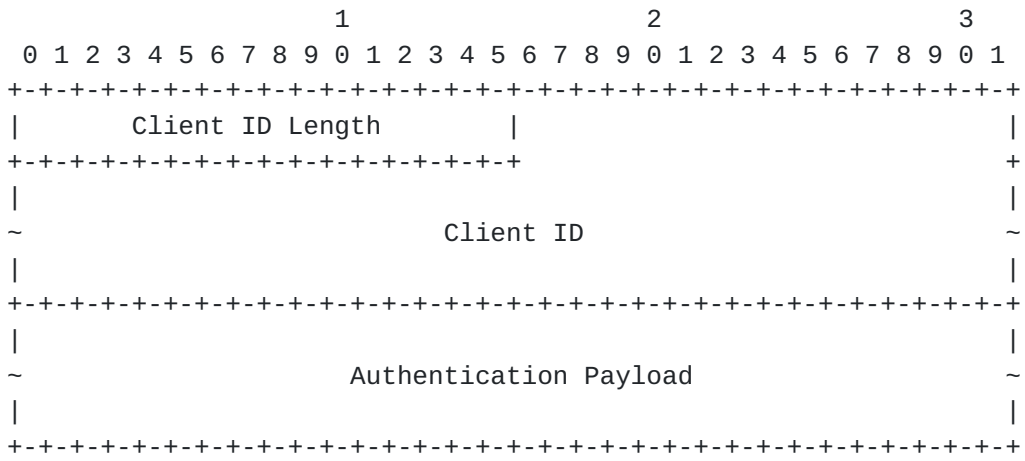


Figure 24: Resume Client Payload

- Client ID Length (1 byte) - The length of the Client ID field not including any other field.

- Client ID (variable length) - The detached client's Client ID. The client that sends this payload must know the Client ID.

- Authentication Payload (variable length) - The authentication payload that the server will verify with the detached client's public key. If the server doesn't know the public key, it must retrieve it for example with SILC_COMMAND_GETKEY command.

2.3.24 Acknowledgement Payload

This payload is used to acknowledge a packet that had the Acknowledgement packet flag set. The payload includes the sequence number of the packet that had the flag set, which the recipient can use to identify that the packet was acknowledged.

The payload may only be sent with SILC_PACKET_ACK packet. It MUST NOT be sent in any other packet type. The following diagram represents the Acknowledgement Payload.

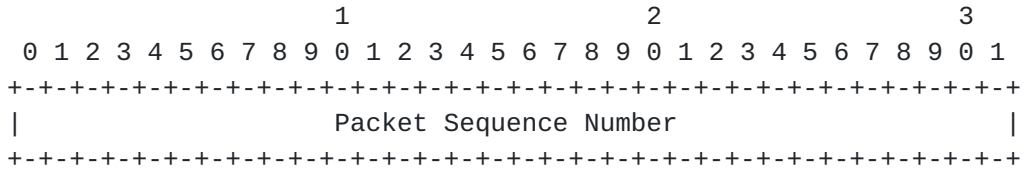


Figure 24: Resume Client Payload

o Packet Sequence Number (4 bytes) - The packet sequence number of the packet that had the Acknowledgement flag set.

2.4 SILC ID Types

ID's are used in the SILC network to associate different entities. The following ID's has been defined to be used in the SILC network.

0 No ID

This is used when other ID type is available at the time.

1 Server ID

Server ID to associate servers. See the format of this ID in [SILC1].

2 Client ID

Client ID to associate clients. See the format of this ID in [SILC1].

3 Channel ID

Channel ID to associate channels. See the format of this ID in [SILC1].

When encoding different IDs into the ID Payload, all fields are always in MSB first order. The IP address, port, and/or the random number are encoded in the MSB first order.

2.5 Packet Encryption And Decryption

SILC packets are encrypted almost entirely. Only the MAC at the end of the packet is never encrypted. The SILC Packet header is the first part of a packet to be encrypted and it is always encrypted with the key of the next receiver of the packet. The data payload area of the packet is always entirely encrypted and it is usually encrypted with the next receiver's key. However, there are some special packet types and packet payloads that require special encryption process. These special cases are described in the next sections. First is described the normal packet encryption process.

2.5.1 Normal Packet Encryption And Decryption

Normal SILC packets are encrypted with the session key of the next receiver of the packet. The entire SILC Packet header and the packet data payload is is encrypted with the same key. Padding of the packet is also encrypted always with the session key, also in special cases. Computed MAC of the packet MUST NOT be encrypted.

Decryption process in these cases are straightforward. The receiver of the packet MUST first decrypt the SILC Packet header, or some parts of it, usually first 16 bytes of it. Then the receiver checks the packet type from the decrypted part of the header and can determine how the rest of the packet must be decrypted. If the packet type is any of the special cases described in the following sections the packet decryption is special. If the packet type is not among those special packet types rest of the packet can be decrypted with the same key. At this point the receiver is also able to determine the length of the packet.

With out a doubt, this sort of decryption processing causes some overhead to packet decryption, but never the less, is required.

The MAC of the packet is also verified at this point. The MAC is computed from the ciphertext of the packet so it can be verified at this stage. The length of the packet need to be known to be able to verify the MAC from the ciphertext so the first 16 bytes need to be decrypted to determine the packet length. However, the MAC MUST be verified from the entire ciphertext.

2.5.2 Channel Message Encryption And Decryption

Channel Messages (Channel Message Payload) are always encrypted with the channel specific key. However, the SILC Packet header is not encrypted with that key. As in normal case, the header is encrypted with the key of the next receiver of the packet. Note that, in this case the encrypted data area is not touched at all; it MUST NOT be re-encrypted with the session key.

Receiver of a channel message, who ever that is, is REQUIRED to decrypt the SILC Packet header to be able to recognize the packet to be as channel message. This is same procedure as for normal SILC packets. As the receiver founds the packet to be channel message, rest of the packet processing is special. Rest of the SILC Packet header is decrypted with the same session key along with the padding of the packet. After that the packet is protected with the channel specific key and thus can be decrypted only if the receiver is the client on the channel. See section 2.7 Packet Padding Generation for more information about padding on special packets.

If the receiver of the channel message is router which is routing the message to another router then it MUST decrypt the Channel Message payload too. Between routers (that is, between cells) channel messages are protected with session keys shared between the routers. This causes another special packet processing for channel messages. If the channel message is received from another router then the entire packet, including Channel Message payload, MUST be encrypted with the session key shared between the routers. In this case the packet decryption process is as with normal SILC packets. Hence, if the router is sending channel message to another router the Channel Message payload MUST have been decrypted and MUST be re-encrypted with the session key shared between the another router. In this case the packet encryption is as with any normal SILC packet.

It must be noted that this is only when the channel messages are sent from router to another router. In all other cases the channel message encryption and decryption is as described before. This different processing of channel messages with router to router connection is because channel keys are cell specific. All cells have their own channel keys thus the channel message traveling from one cell to another MUST be protected as it would be any normal SILC packet.

If the SILC_CMODE_PRIVKEY channel mode has been set for the channel then the router cannot decrypt the packet as it does not know the private key. In this case the entire packet MUST be encrypted with the session key and sent to the router. The router receiving the packet MUST check the channel mode and decrypt the packet accordingly.

2.5.3 Private Message Encryption And Decryption

By default, private message in SILC are protected by session keys. In this case the private message encryption and decryption process is equivalent to normal packet encryption and decryption.

However, private messages MAY be protected with private message key which causes the packet to be special packet. The procedure in this case is very much alike to channel packets. The actual private message is encrypted with the private message key and other parts of the packet is encrypted with the session key. See 2.7 Packet Padding Generation for more information about padding on special packets.

The difference from channel message processing is that server or router en route never decrypts the actual private message, as it does not have the key to do that. Thus, when sending packets between router the processing is same as in any other case as well; the packet's header and padding is protected by the session key and the data area is not touched and is not re-encrypted.

The true receiver of the private message is able to decrypt the private message as it shares the key with the sender of the message.

2.6 Packet MAC Generation

Data integrity of a packet is protected by including a message authentication code (MAC) at the end of the packet. The MAC is computed from shared secret MAC key, that is established by the SILC Key Exchange protocol, from packet sequence number, and from the encrypted packet data. The MAC is always computed after packet is encrypted. This is so called Encrypt-Then-MAC order; packet is first encrypted, then MAC is computed from the encrypted data.

The MAC is computed from entire packet. Every bit of data in the packet, including SILC Packet Header is used in the MAC computing. This way the entire packet becomes authenticated.

Hence, packet's MAC generation is as follows:

```
mac = MAC(key, sequence number | Encrypted SILC packet)
```

The MAC key is negotiated during the SKE protocol. The sequence number is a 32 bit MSB first value starting from zero for first packet and increasing for subsequent packets, finally wrapping after 2^32 packets. The value is never reset, not even after rekey has been performed. However, rekey MUST be performed before the sequence number wraps and repeats from zero. Note that the sequence number is incremented only

when MAC is computed for a packet. If packet is not encrypted and MAC is not computed then the sequence number is not incremented. Hence, the sequence number is zero for the very first encrypted packet.

See [SILC1] for defined and allowed MAC algorithms.

2.7 Packet Padding Generation

Padding is needed in the packet because the packet is encrypted. It always MUST be multiple by eight (8) or multiple by the block size of the cipher, which ever is larger. The padding is always encrypted.

For normal packets the padding is added after the SILC Packet Header and between the Data Payload area. The padding for normal packets may be calculated as follows:

```
padding_length = 16 - (packet_length mod block_size)
if (padding_length < 8)
  padding_length += block_size
```

The 'block_size' is the block size of the cipher. The maximum padding length is 128 bytes, and minimum is 8 bytes. For example, packets that include a passphrase or a password for authentication purposes SHOULD pad the packet up to the maximum padding length. The maximum padding is calculated as follows:

```
padding_length = 128 - (packet_length mod block_size)
```

For special packets the padding calculation is different as special packets may be encrypted differently. In these cases the encrypted data area MUST already be multiple by the block size thus in this case the padding is calculated only for SILC Packet Header, not for any other area of the packet. The same algorithm works in this case as well, except that the 'packet length' is now the SILC Packet Header length.

The padding MUST be random data, preferably, generated by cryptographically strong random number generator for each packet separately.

2.8 Packet Compression

SILC Packets MAY be compressed. In this case the data payload area is compressed and all other areas of the packet MUST remain as they are. After compression is performed for the data area, the length field of Packet Header MUST be set to the compressed length of the

data.

The compression MUST always be applied before encryption. When the packet is received and decrypted the data area MUST be decompressed. Note that the true sender of the packet MUST apply the compression and the true receiver of the packet MUST apply the decompression. Any server or router en route SHOULD NOT decompress the packet.

2.9 Packet Sending

The sender of the packet MUST assemble the SILC Packet Header with correct values. It MUST set the Source ID of the header as its own ID, unless it is forwarding the packet. It MUST also set the Destination ID of the header to the true destination. If the destination is client it will be Client ID, if it is server it will be Server ID and if it is channel it will be Channel ID.

If the sender wants to compress the packet it MUST apply the compression now. Sender MUST also compute the padding as described in above sections. Then sender MUST encrypt the packet as has been described in above sections according whether the packet is normal packet or special packet. Then sender MUST compute the MAC of the packet. The computed MAC MUST NOT be encrypted.

2.10 Packet Reception

On packet reception the receiver MUST check that all fields in the SILC Packet Header are valid. It MUST check the flags of the header and act accordingly. It MUST also check the MAC of the packet and if it is to be failed the packet MUST be discarded. Also if the header of the packet includes any bad fields the packet MUST be discarded.

See above sections on the decryption process of the received packet.

The receiver MUST also check that the ID's in the header are valid ID's. Unsupported ID types or malformed ID's MUST cause packet rejection. The padding on the reception is always ignored.

The receiver MUST also check the packet type and start parsing the packet according to the type. However, note the above sections on special packet types and their parsing.

2.11 Packet Routing

Routers are the primary entities in the SILC network that takes care of packet routing. Normal servers performs packet forwarding, for example, when they are forwarding channel message to the local clients. Routing is quite simple as every packet tells the true origin and the true destination of the packet.

It is still RECOMMENDED for routers that has several routing connections to create route cache for those destinations that has faster route than the router's primary route. This information is available for the router when other router connects to the router. The connecting party then sends all of its locally connected clients, servers and channels. These informations helps to create the route cache. Also, when new channels are created to a cell its information is broadcasted to all routers in the network. Channel ID's are based on router's ID thus it is easy to create route cache based on these informations. If faster route for destination does not exist in router's route cache the packet MUST be routed to the primary route (default route).

However, there are some issues when routing channel messages to group of users. Routers are responsible of routing the channel message to other routers, local servers and local clients as well. Routers MUST send the channel message to only one router in the network, preferably to the shortest route to reach the channel users. The message can be routed into either upstream or downstream. After the message is sent to a router in the network it MUST NOT be sent to any other router in either same route or other route. The message MUST NOT be routed to the router it came from.

When routing for example private messages they should be routed to the shortest route always to reach the destination client as fast as possible.

For server which receives a packet to be forwarded to an entity that is indirectly connected to the sender, the server MUST check whether that particular packet type is allowed to be sent to that destination. Not all packets may be sent by some odd entity to for example a local client, or to some remote server or router, that is indirectly connected to the sender. See section 2.3 SILC Packet Types and paragraph about indirectly connected entities and sending packets to them. That section defines the packets that may be sent to indirectly connected entities. When a server or a router receives a packet that may be sent to indirectly connected entity and it is destined to other entity except that server, it MUST route it further either to shortest route or to the primary route to reach that destination.

Routers form a ring in the SILC network. However, routers may have other direct connections to other routers in the network too. This can cause interesting routing problems in the network. Since the network is a ring, the packets usually should be routed into clock-wise direction, or if it

cannot be used then always counter clock-wise (primary route) direction. Problems may arise when a faster direct route exists and router is routing a channel message. Currently channel messages must be routed either in upstream or downstream, they cannot be routed to other direct routes. The SILC protocol should have a shortest path discovery protocol, and some existing routing protocol, that can handle a ring network with other direct routes inside the ring (so called hybrid ring-mesh topology), MAY be defined to be used with the SILC protocol. Additional specifications MAY be written on the subject to permeate this specification.

2.12 Packet Broadcasting

SILC packets MAY be broadcasted in SILC network. However, only router server may send or receive broadcast packets. Client and normal server MUST NOT send broadcast packets and they MUST ignore broadcast packets if they receive them. Broadcast packets are sent by setting Broadcast flag to the SILC packet header.

Broadcasting packets means that the packet is sent to all routers in the SILC network, except to the router that sent the packet. The router receiving broadcast packet MUST send the packet to its primary route. The fact that SILC routers may have several router connections can cause problems, such as race conditions inside the SILC network, if care is not taken when broadcasting packets. Router MUST NOT send the broadcast packet to any other route except to its primary route.

If the primary route of the router is the original sender of the packet the packet MUST NOT be sent to the primary route. This may happen if router has several router connections and some other router uses the router as its primary route.

Routers use broadcast packets to broadcast for example information about newly registered clients, servers, channels etc. so that all the routers may keep these informations up to date.

3 Security Considerations

Security is central to the design of this protocol, and these security considerations permeate the specification. Common security considerations such as keeping private keys truly private and using adequate lengths for symmetric and asymmetric keys must be followed in order to maintain the security of this protocol.

4 References

[SILC1] Riikonen, P., "Secure Internet Live Conferencing (SILC), Protocol Specification", Internet Draft, January 2007.

[SILC3] Riikonen, P., "SILC Key Exchange and Authentication Protocols", Internet Draft, January 2007.

[SILC4] Riikonen, P., "SILC Commands", Internet Draft, January 2007.

[IRC] Oikarinen, J., and Reed D., "Internet Relay Chat Protocol", RFC 1459, May 1993.

[IRC-ARCH] Kalt, C., "Internet Relay Chat: Architecture", RFC 2810, April 2000.

[IRC-CHAN] Kalt, C., "Internet Relay Chat: Channel Management", RFC 2811, April 2000.

[IRC-CLIENT] Kalt, C., "Internet Relay Chat: Client Protocol", RFC 2812, April 2000.

[IRC-SERVER] Kalt, C., "Internet Relay Chat: Server Protocol", RFC 2813, April 2000.

[SSH-TRANS] Ylonen, T., et al, "SSH Transport Layer Protocol", Internet Draft.

[PGP] Callas, J., et al, "OpenPGP Message Format", RFC 2440, November 1998.

[SPKI] Ellison C., et al, "SPKI Certificate Theory", RFC 2693, September 1999.

[PKIX-Part1] Housley, R., et al, "Internet X.509 Public Key Infrastructure, Certificate and CRL Profile", RFC 2459, January 1999.

[Schneier] Schneier, B., "Applied Cryptography Second Edition", John Wiley & Sons, New York, NY, 1996.

[Menezes] Menezes, A., et al, "Handbook of Applied Cryptography", CRC Press 1997.

[OAKLEY] Orman, H., "The OAKLEY Key Determination Protocol", RFC 2412, November 1998.

[ISAKMP] Maughan D., et al, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[IKE] Harkins D., and Carrel D., "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[HMAC] Krawczyk, H., "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[PKCS1] Kalinski, B., and Staddon, J., "PKCS #1 RSA Cryptography Specifications, Version 2.0", RFC 2437, October 1998.

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

[SFTP] Ylonen T., and Lehtinen S., "Secure Shell File Transfer Protocol", Internet Draft, March 2001.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 3629, November 2003.

5 Author's Address

Pekka Riikonen
Helsinki
Finland

EMail: priikone@iki.fi

6 Full Copyright Statement