Verteilte Systeme Labor

2.3

Generated by Doxygen 1.8.9.1

Tue Apr 19 2016 23:53:42

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 dat_broadcast_response Struct Reference

Broadcast response Each server responds to a broadcast sending its IP address.

```
#include <PacketLib.h>
```

Data Fields

- uint8_t serverIP [4]

  4 times 1 Byte IP V4 address

3.1.1 Detailed Description

Broadcast response Each server responds to a broadcast sending its IP address.

3.1.2 Field Documentation

3.1.2.1 uint8_t dat_broadcast_response::serverIP[4]

4 times 1 Byte IP V4 address

The documentation for this struct was generated from the following file:

- PacketLib.h

3.2 dat_decrypt_request Struct Reference

Decrypt data Request to decrypt data. Polynome has to be set first.

```
#include <PacketLib.h>
```

Data Fields

- int16_t clientID

  The ID of the requesting client.
- uint16_t blockID:14

  A (random) Block ID to tell the packets apart.

- uint16_t fill:2

  The amount of bytes at the end of the data field to ignore.
- uint16_t firstElement

  First element of the data structure (16 Bit Chunks)

3.2.1 Detailed Description

Decrypt data Request to decrypt data. Polynome has to be set first.

See also

dat_polynom_request

3.2.2 Field Documentation

3.2.2.1 uint16_t dat_decrypt_request::blockID

A (random) Block ID to tell the packets apart.

3.2.2.2 int16_t dat_decrypt_request::clientID

The ID of the requesting client.

3.2.2.3 uint16_t dat_decrypt_request::fill

The amount of bytes at the end of the data field to ignore.

3.2.2.4 uint16_t dat_decrypt_request::firstElement

First element of the data structure (16 Bit Chunks)

The documentation for this struct was generated from the following file:

- PacketLib.h

3.3 dat_decrypt_response Struct Reference

Return decrypted data Returns the data from successfull decryption.

```
#include <PacketLib.h>
```

Data Fields

- uint16_t blockID:14

  The Block ID set by the client.
- uint16_t fill:2

  The amount of bytes at the end of the data field to ignore.
- uint16_t reserved

  Reserved.
- uint8_t firstElement

  First element of the data structure (8 Bit Chunks)

3.3.1 Detailed Description

Return decrypted data Returns the data from successfull decryption.

3.3.2 Field Documentation

3.3.2.1 uint16_t dat_decrypt_response::blockID

The Block ID set by the client.

3.3.2.2 uint16_t dat_decrypt_response::fill

The amount of bytes at the end of the data field to ignore.

3.3.2.3 uint8_t dat_decrypt_response::firstElement

First element of the data structure (8 Bit Chunks)

3.3.2.4 uint16_t dat_decrypt_response::reserved

Reserved.

See also

VALUE_RESERVED

The documentation for this struct was generated from the following file:

- PacketLib.h

3.4 dat_polynom_request Struct Reference

Set polynome request This function tries to set the polynome if the server is free or the current client has lower priority.

```
#include <PacketLib.h>
```

Data Fields

- int16_t clientID

  The ID of the requesting client.
- uint16_t generator

  The generator polynome.

3.4.1 Detailed Description

Set polynome request This function tries to set the polynome if the server is free or the current client has lower priority.

3.4.2 Field Documentation

3.4.2.1 int16_t dat_polynom_request::clientID

The ID of the requesting client.

3.4.2.2 uint16_t dat_polynom_request::generator

The generator polynome.

The documentation for this struct was generated from the following file:

- PacketLib.h

3.5 dat_status_response Struct Reference

Response to a status request Servers respond with their current status.

```
#include <PacketLib.h>
```

Data Fields

- int16_t clientID
  The ID of the currently connected client.
- uint16_t reserved
  Reserved.
- uint32_t wordCount
  Amount of Decrypted data words for this client.

3.5.1 Detailed Description

Response to a status request Servers respond with their current status.

3.5.2 Field Documentation

3.5.2.1 int16_t dat_status_response::clientID

The ID of the currently connected client.

3.5.2.2 uint16_t dat_status_response::reserved

Reserved.

See also

VALUE_RESERVED

3.5.2.3 uint32_t dat_status_response::wordCount

Amount of Decrypted data words for this client.

The documentation for this struct was generated from the following file:

- PacketLib.h

3.6 dat_unlock_request Struct Reference

Unlocks the server After the server is not needed anymore you can unlock it with this function. The server can then be used by another slave.

```
#include <PacketLib.h>
```

Data Fields

- int16_t clientID
  The ID of the current client.
- uint16_t reserved
  Reserved.

3.6.1 Detailed Description

Unlocks the server After the server is not needed anymore you can unlock it with this function. The server can then be used by another slave.

See also

dat_polynom_request

3.6.2 Field Documentation

3.6.2.1 int16_t dat_unlock_request::clientID

The ID of the current client.

3.6.2.2 uint16_t dat_unlock_request::reserved

Reserved.

See also

VALUE_RESERVED

The documentation for this struct was generated from the following file:

- PacketLib.h

3.7 error Struct Reference

Error frame.

```
#include <PacketLib.h>
```

Data Fields

- uint8_t errCode
  The error code of the occurring error.
- uint16_t blockID:14
  Block ID where the error occurred (for ERR_DECRYPT and ERR_SERVERINUSE). Else 0.
- uint16_t reserved:10
  Reserved.

3.7.1 Detailed Description

Error frame.

3.7.2 Field Documentation

3.7.2.1 uint16_t error::blockID

Block ID where the error occurred (for ERR_DECRYPT and ERR_SERVERINUSE). Else 0.

3.7.2.2 uint8_t error::errCode

The error code of the occurring error.

See also

ERR_PACKETLENGTH
ERR_INVALIDVERSION
ERR_INVALIDMODE
ERR_NOSUCHFUNCTION
ERR_INVALIDTYPE
ERR_DATA
ERR_SERVERINUSE
ERR_FUNCTIONTIMEOUT
ERR_FUNCTIONEXEC
ERR_DECRYPT
ERR_ALLOC
ERR_UNKNOWN

3.7.2.3 uint16_t error::reserved

Reserved.

See also

VALUE_RESERVED

The documentation for this struct was generated from the following file:

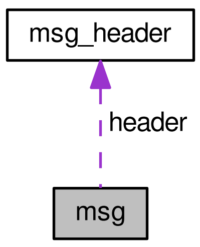
- PacketLib.h

3.8 msg Struct Reference

Structure for a message This structure holds pointers for the message header and the data structure.

```
#include <PacketLib.h>
```

Collaboration diagram for msg:



Data Fields

- msg_header ∗ header

  A pointer to the header of the structure.
- void ∗ data

  A pounter to the data field of the structure. Nullpointer for no data field.

3.8.1 Detailed Description

Structure for a message This structure holds pointers for the message header and the data structure.

3.8.2 Field Documentation

3.8.2.1 void∗ msg::data

A pounter to the data field of the structure. Nullpointer for no data field.

See also

dat_polynom_request
dat_decrypt_request
dat_decrypt_response
dat_unlock_request
dat_broadcast_response
dat_status_response
error

3.8.2.2 msg_header∗ msg::header

A pointer to the header of the structure.

See also

msg_header

The documentation for this struct was generated from the following file:

- PacketLib.h

3.9 msg_header Struct Reference

A structure for the message header You can easily type cast the first 8 Byte of a message to this struct. The internal structure holds all the values then.

```
#include <PacketLib.h>
```

Data Fields

- uint8_t priority

    The priority of the message (0 = HIGH, 255 = LOW)
- uint8_t version

    The current version of the script.
- uint8_t mode

    The mode of the message (sender type)
- uint8_t func:4

    The called function of this message.
- uint8_t type:4

    The message Type.
- uint16_t length

    The Length of the message data field.
- uint16_t reserved

    Reserved.

3.9.1 Detailed Description

A structure for the message header You can easily type cast the first 8 Byte of a message to this struct. The internal structure holds all the values then.

3.9.2 Field Documentation

3.9.2.1 uint8_t msg_header::func

The called function of this message.

See also

FNC_POLYNOME
FNC_DECRYPT
FNC_UNLOCK
FNC_BROADCAST
FNC_STATUS

3.9.2.2 uint16_t msg_header::length

The Length of the message data field.

3.9.2.3 uint8_t msg_header::mode

The mode of the message (sender type)

See also

MODE_STATUS
MODE_SERVER
MODE_CLIENT

3.9.2.4 uint8_t msg_header::priority

The priority of the message (0 = HIGH, 255 = LOW)

3.9.2.5 uint16_t msg_header::reserved

Reserved.

See also

VALUE_RESERVED

3.9.2.6 uint8_t msg_header::type

The message Type.

See also

MSG_REQUEST
MSG_RESPONSE
MSG_ERROR

3.9.2.7 uint8_t msg_header::version

The current version of the script.

See also

PROTOCOL_VERSION

The documentation for this struct was generated from the following file:

- PacketLib.h

Chapter 4

File Documentation

4.1 Macros.h File Reference

Macros

- #define HEADER_LENGTH 8

  The header length in bytes.
- #define VALUE_RESERVED 0

  Standard value for reserved fields.
- #define PROTOCOL_VERSION 14

  The version of the protocol.
- #define MODE_STATUS 1

  The status script is the message source.
- #define MODE_SERVER 2

  The message originated from a server.
- #define MODE_CLIENT 3

  A client sent the message.
- #define FNC_POLYNOME 0

  Sets the polynome in the server.
- #define FNC_DECRYPT 1

  Decrypts a chunk of the file.
- #define FNC_UNLOCK 2

  Unlocks the server to make it available for other clients.
- #define FNC_BROADCAST 5

  Broadcast to discover all available servers.
- #define FNC_STATUS 6

  Status request for that node.
- #define MSG_REQUEST 3

  Request the specified function.
- #define MSG_RESPONSE 4

  Response to an earlier request.
- #define MSG_ERROR 15

  An error occurred decoding or executing.
- #define NO_ERROR 0

  No error detected.
- #define ERR_PACKETLENGTH 1

  The packet length is invalid or does not match the actual length.

- #define ERR_INVALIDVERSION 2

    The version does not match the one defined in PACKET_LENGTH.
- #define ERR_INVALIDMODE 3

    The mode does not exist.
- #define ERR_NOSUCHFUNCTION 4

    The requested function does not exist (on this node)
- #define ERR_INVALIDTYPE 5

    The type is not specified.
- #define ERR_DATA 6

    Error in the data field detected.
- #define ERR_SERVERINUSE 16

    The server is currently used by another client.
- #define ERR_FUNCTIONTIMEOUT 32

    The called function timed out.
- #define ERR_FUNCTIONEXEC 33

    An error executing this function was detected.
- #define ERR_DECRYPT 64

    The data could not be decrypted due to an error.
- #define ERR_ALLOC 128

    Not enough free space to allocate data.
- #define ERR_UNKNOWN 254

    An error occurred that does not match any of the other ones (this should never happen)

4.1.1 Macro Definition Documentation

4.1.1.1 #define ERR_ALLOC 128

Not enough free space to allocate data.

4.1.1.2 #define ERR_DATA 6

Error in the data field detected.

4.1.1.3 #define ERR_DECRYPT 64

The data could not be decrypted due to an error.

4.1.1.4 #define ERR_FUNCTIONEXEC 33

An error executing this function was detected.

4.1.1.5 #define ERR_FUNCTIONTIMEOUT 32

The called function timed out.

4.1.1.6 #define ERR_INVALIDMODE 3

The mode does not exist.

4.1.1.7 #define ERR_INVALIDTYPE 5

The type is not specified.

4.1.1.8 #define ERR_INVALIDVERSION 2

The version does not match the one defined in PACKET_LENGTH.

4.1.1.9 #define ERR_NOSUCHFUNCTION 4

The requested function does not exist (on this node)

4.1.1.10 #define ERR_PACKETLENGTH 1

The packet length is invalid or does not match the actual length.

4.1.1.11 #define ERR_SERVERINUSE 16

The server is currently used by another client.

4.1.1.12 #define ERR_UNKNOWN 254

An error occurred that does not match any of the other ones (this should never happen)

4.1.1.13 #define FNC_BROADCAST 5

Broadcast to discover all available servers.

4.1.1.14 #define FNC_DECRYPT 1

Decrypts a chunk of the file.

4.1.1.15 #define FNC_POLYNOME 0

Sets the polynome in the server.

4.1.1.16 #define FNC_STATUS 6

Status request for that node.

4.1.1.17 #define FNC_UNLOCK 2

Unlocks the server to make it available for other clients.

4.1.1.18 #define HEADER_LENGTH 8

The header length in bytes.

4.1.1.19 #define MODE_CLIENT 3

A client sent the message.

4.1.1.20 #define MODE_SERVER 2

The message originated from a server.

4.1.1.21 #define MODE_STATUS 1

The status script is the message source.

4.1.1.22 #define MSG_ERROR 15

An error occurred decoding or executing.

4.1.1.23 #define MSG_REQUEST 3

Request the specified function.

4.1.1.24 #define MSG_RESPONSE 4

Response to an earlier request.

4.1.1.25 #define NO_ERROR 0

No error detected.

4.1.1.26 #define PROTOCOL_VERSION 14

The version of the protocol.

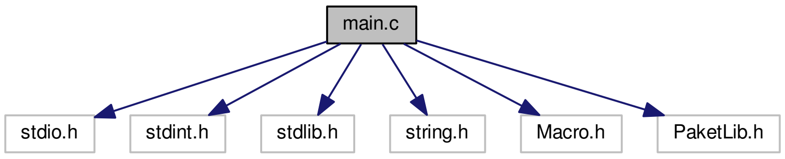
4.1.1.27 #define VALUE_RESERVED 0

Standard value for reserved fields.

4.2 main.c File Reference

```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>
#include "Macro.h"
#include "PaketLib.h"
```

Include dependency graph for main.c:



Functions

- int main ()

4.2.1 Function Documentation

4.2.1.1 int main ( )

4.3 PacketLib.h File Reference

Data Structures

- struct msg_header

  A structure for the message header You can easily type cast the first 8 Byte of a message to this struct. The internal structure holds all the values then.
- struct msg

  Structure for a message This structure holds pointers for the message header and the data structure.
- struct dat_polynom_request

  Set polynome request This function tries to set the polynome if the server is free or the current client has lower priority.
- struct dat_decrypt_request

  Decrypt data Request to decrypt data. Polynome has to be set first.
- struct dat_decrypt_response

  Return decrypted data Returns the data from successfull decryption.
- struct dat_unlock_request

  Unlocks the server After the server is not needed anymore you can unlock it with this function. The server can then be used by another slave.
- struct dat_broadcast_response

  Broadcast response Each server responds to a broadcast sending its IP address.
- struct dat_status_response

  Response to a status request Servers respond with their current status.
- struct error

  Error frame.

Typedefs

- typedef struct msg_header msg_header

  A structure for the message header You can easily type cast the first 8 Byte of a message to this struct. The internal structure holds all the values then.

- typedef struct msg msg

    Structure for a message This structure holds pointers for the message header and the data structure.
- typedef struct dat_polynom_request dat_polynom_request

    Set polynome request This function tries to set the polynome if the server is free or the current client has lower priority.
- typedef struct dat_decrypt_request dat_decrypt_request

    Decrypt data Request to decrypt data. Polynome has to be set first.
- typedef struct dat_decrypt_response dat_decrypt_response

    Return decrypted data Returns the data from successfull decryption.
- typedef struct dat_unlock_request dat_unlock_request

    Unlocks the server After the server is not needed anymore you can unlock it with this function. The server can then be used by another slave.
- typedef struct dat_broadcast_response dat_broadcast_response

    Broadcast response Each server responds to a broadcast sending its IP address.
- typedef struct dat_status_response dat_status_response

    Response to a status request Servers respond with their current status.
- typedef struct error error

    Error frame.

4.3.1 Typedef Documentation

4.3.1.1 typedef struct **dat_broadcast_response dat_broadcast_response**

Broadcast response Each server responds to a broadcast sending its IP address.

4.3.1.2 typedef struct **dat_decrypt_request dat_decrypt_request**

Decrypt data Request to decrypt data. Polynome has to be set first.

See also

dat_polynom_request

4.3.1.3 typedef struct **dat_decrypt_response dat_decrypt_response**

Return decrypted data Returns the data from successfull decryption.

4.3.1.4 typedef struct **dat_polynom_request dat_polynom_request**

Set polynome request This function tries to set the polynome if the server is free or the current client has lower priority.

4.3.1.5 typedef struct **dat_status_response dat_status_response**

Response to a status request Servers respond with their current status.

4.3.1.6 typedef struct **dat_unlock_request dat_unlock_request**

Unlocks the server After the server is not needed anymore you can unlock it with this function. The server can then be used by another slave.

See also

dat_polynom_request

4.3.1.7 typedef struct error error

Error frame.

4.3.1.8 typedef struct msg msg

Structure for a message This structure holds pointers for the message header and the data structure.

4.3.1.9 typedef struct msg_header msg_header

A structure for the message header You can easily type cast the first 8 Byte of a message to this struct. The internal structure holds all the values then.