15/03/2015

Secure Group Communication

Distributed Systems course project

Mattia Brusamento, Federico Maria Ceruti, Gianmarco Donetti

POLITECNICO DI MILANO

SUMMARY

1 SERVER

1.1 Server

It calls directly startServer:

1. Initialization: FlatTable, SyncQueue, RequestManager
2. It starts a new ExecutorService and it remains listening on a new ServerSocket
3. It assigns the executor to the RequestManager
4. Loop in order to accept new connections: each of them are assigned to the executor.

1.2 Request Manager

Never-ending loop: it manages the requests from clients in the queue one after the other. These requests can be of two types: join or leave.

1.2.1 Handle Join

1. It checks whether there is space for a new member or not
2. It manages, when it's possible, the new join:
    a. It sends the new keys to the group members
    b. It waits for all the acks
    c. It confirms the update of the group to the members and it let the new one come in

1.2.2 Handle Leave

1. It manages the leave request:
    a. It changes all the keys and it sends the update to all the other members of the group
    b. All the participants stop themselves, update their keys and return an ack to the server
    c. Received all the acks, the server confirms the leaving
    d. The communication restarts

1.3 Socket Connession

A socket connection between server and a single client (group member). It controls the communication between a client and the manager of the group communication service, until it is open, so it has to read messages like join request, leave request, ack.

1.3.1 Send

Used by the server in order to send messages to the client linked with the proper connection.

2 GUI

2.1 START

1. It starts a new chat interface
2. It initializes the tools for the multicast communication (KeyManager and Crypter)
3. It initializes the tools for the server communication (ServerDispatcher)

3 COMMUNICATION

3.1 Server Dispatcher

It manages the communication between a group member and the server from the point of view of the client.

3.1.1 Request Join

1. It is the first action done by a client, so it opens a new socket towards the default server
2. It initializes ServerSender and ServerReceiver
3. It sends a join request to the server, encrypted with public key

3.1.2 Request Leave

It sends a leave request to the server with a proper message.

3.1.3 Handle Message

It checks whether the received object is acceptable or not, then it calls the proper methods, based on the received message type (see below).

3.1.4 Accepted Join

1. It takes the whole body of the message
2. From the body, it gets the keks and the dek

3.1.5 Someone Joining

1. It stops the user from sending further messages
3. It takes the whole body of the message
2. It updates the keks and the dek
3. It returns an ack to the server

3.1.6 Someone Leaving

4. It stops the user from sending further messages
4. It takes the whole body of the message
5. It updates the keks and the dek
6. It returns an ack to the server

3.2 Group Receiver

It catches the multicast communication datagrams and it sends them to the message handler.

3.3 Server Receiver

It catches the client-server communication messages and it sends them to the message handler.

3.4 Group Sender

It sends the datagrams on the multicast group channel.

3.5 Server Sender

It sends the messages on the client-server communication channel.

4 SECURITY

4.1 Key Manager

4.1.1 Confirm Update

It initializes the encrypter and the two decrypters.

4.1.2 Update On Join

It updates all the keys after a join in the proper way

4.1.3 Update On Leave

It updates all the keys after a leave in the proper way.

4.2 Crypter

It has to encrypt and decrypt the given messages.

4.2.1 Send Message

It encrypts the message and it sends it using the MessageSender.

4.2.2 Handle Message

It tries to decrypt the message with the two saved deks and it returns the result to the GUI.