

# SEGURANÇA

## ***Secure Messaging Repository System*** *Security Overview*

**Trabalho realizado por:**  
André Rodrigues 73152  
Cristiano Vagos 65169

**Professor:**  
André Zúquete

Aveiro, 9 de Novembro de 2017

# Índice

<b>Introdução</b>	<b>2</b>
Conexão Cliente-Servidor	3
Mensagem Cliente-Cliente	3
Objectivos	4
Conexão	4
Criação	4
Envio de Mensagens e Receipts	5

# Introdução

A comunicação através de mensagens online é algo que é utilizado diariamente pela maioria dos utilizadores. Esta comunicação deve ser segura e confiável, para que o sistema seja capaz de assegurar que as mensagens são entregues e de maneira a que um atacante não descubra o seu conteúdo, nem seja capaz de atacar o sistema de outras formas que comprometam a sua segurança.

Este projeto trata-se disso mesmo, de um sistema de mensagens seguro que permita ter vários aspectos abordados no âmbito da unidade curricular de forma à troca de mensagens e autenticação de cada utilizador ser segura.

Com este primeiro relatório pretendemos indicar qual a nossa abordagem ao problema, bem como a discussão das soluções encontradas para resolver os problemas e sobretudo defendermo-nos de um possível ataque.

## Conexão Cliente-Servidor

De forma a podermos comunicar, precisamos de um servidor que lide com vários pedidos, entre eles a conexão ao sistema com o uso de sockets, bem como o tratamento dos pedidos feitos por cada cliente do sistema de mensagens (criação de nova mensagem, envio, recepção, etc.).

Numa primeira abordagem, quando um cliente conecta-se ao servidor têm a opção de registar uma **message box** ou de **autenticar-se**. No primeiro caso, o cliente envia um pedido ao servidor sem qualquer tipo de segurança, uma vez que não possui nenhum registo, no segundo caso, quando tenta autenticar-se, isto é, ligar-se à sua **message box**, o cliente e o servidor entrarão num estado de negociação de uma chave secreta de sessão através do algoritmo de Diffie-Hellman.

Esta chave de sessão secreta será utilizada para cifrar mensagens seguras entre cliente e o servidor e para cada pedido entre os peers, será derivada uma chave da chave secreta (key derivation function) de modo que a cada pedido seja utilizada uma chave diferente, de notar que é possível verificar se esta chave derivada realmente deriva da chave secreta pré-estabelecida.

De modo a aumentar a segurança entre a comunicação cliente-servidor, optamos por fazer um refrescamento da chave de sessão quando atingirmos um determinado número de pedidos pré-estabelecido.

Também são feitas verificações ao longo das comunicações, para tal usamos HMAC (mensagem, chave derivada(chave secreta)) para garantir a integridade da informação trocada entre as duas entidades.

Optámos por este método pois uma cifra simétrica tem maior desempenho que uma assimétrica e através deste conseguimos obter uma comunicação rápida e segura.

## Mensagem Cliente-Cliente

Antes de abordar a comunicação cliente-cliente é necessário focar que não existirá uma comunicação directa entre eles, mas sim uma indirecta, ou seja, o servidor será sempre o intermediário e irá garantir a comunicação segura entre estes e a sua integridade.

No envio de uma mensagem em claro para outro cliente, optamos por usar o método de cifra híbrida, ou seja, o conteúdo da mensagem é cifrado por uma chave simétrica uma vez que como podemos lidar com várias mensagens de tamanho indefinido este método é mais eficiente, comparando com a cifra assimétrica por esta não utilizar o princípio da difusão de Shannon.

Garantindo a confidencialidade, iremos garantir a autenticidade através de assinaturas digitais entre os dois clientes.

## Objectivos

### Conexão

- Setup of a session key between a client and the server prior to exchange any command/response;
- Authentication (with the session key) and integrity control of all messages exchanged between client and server;
- Add to each server reply a genuineness warrant (i.e., something proving that the reply is the correct one for the client's request, and not for any other request);

```
{
  'type' : 'secure',
  'messageCiphared' : {
    'type' : 'list',
    ...
  }
  'HMAC' : ...
}
```

Fig.1 - Encapsulamento

Numa primeira tentativa de conexão de um cliente à sua *message-box*, é acordado uma chave de sessão com o servidor com o algoritmo de Diffie-Hellman.

Estabelecida a chave de sessão (ou chave secreta), todos os pedidos serão encapsulados (Fig.1), a mensagem encapsulada é cifrada com a chave de sessão acordada, para verificar a integridade é enviado o HMAC resultante do hash (mensagem) e da chave de sessão.

O servidor terá uma chave de sessão diferente associada a cada cliente identificando assim de forma segura cada pedido.

Quando recebida uma mensagem, o servidor ou o cliente irá analisar, caso seja do tipo 'secure' irá decifrar a mensagem com a chave de sessão, obtendo a mensagem que estava encapsulada, no caso da Fig. 1 seria uma mensagem de tipo '*list*' e a mensagem resultante para o cliente seria do tipo '*resultList*', tornando assim possível a distinção das respostas para cada pedido do cliente ao servidor.

### Criação

- Register relevant security-related data in a user creation process;
- Involve the Citizen Card in the user creation process;

Na criação de um user, é registado no servidor o id, o uuid, o nome (CC), a chave pública e outras informações complementares tais como a chave de sessão que será renovada a cada x pedidos, o estado de conexão (connected, not connected), o número privado do servidor que será diferente para cada cliente de modo a gerar uma chave de sessão única para cada cliente.

## Envio de Mensagens e Receipts

- Encrypt messages delivered to other users;
- Signature of the messages delivered to other users (with the Citizen Card or another private key) and validation of those signatures;
- Encrypt messages saved in the receipt box;
- Send a secure receipt after reading a message;
- Check receipts of sent messages;
- Proper checking of public key certificates from Citizen Cards;

```
{
  'type' : 'secure',
  'messageCIPHERED' : {
    'type' : 'send',
    'src' : 'srcUser',
    'dst' : 'dstUser',
    'msg' : {
      'messageCIPHERED' : messageCIPHERED,
      'symetricKeyCIPHERED' : symetricKeyCIPHERED,
    },
    'copy' : {
      'copyCIPHERED' : copyCIPHERED,
      'CopySymKeyCIPHERED' : CopySymKeyCIPHERED,
    },
    'sec-data' : {
      'signedMessage' : ... ,
      'certificate' : ... ,
    },
  },
  'HMAC' : ...
}
```

Fig.2 - Mensagem tipo 'send' encapsulada

No envio de mensagens, o texto a enviar será cifrado com a utilização de cifras híbridas, isto é, uma vez que o texto pode ter tamanhos variados, é aplicado uma chave simétrica de modo a acelerar o processo de cifra e esta chave simétrica é cifrada pela chave pública do cliente destino, este conteúdo será colocado na **message-box** do cliente destino enquanto que na **receipt-box** do cliente source será colocado o mesmo texto só que desta vez, a chave simétrica será cifrada com a pública deste mesmo cliente source, garantindo a confidencialidade na execução de um status da mensagem.

Igualmente, os receipts são enviados utilizando o mesmo método descrito, com cifras híbridas.

Adicionalmente será enviado a assinatura da mensagem e o certificado da chave pública de modo a que seja possível validar a assinatura e autenticidade da mensagem, será feita uma pré distribuição dos certificados e das roots de modo a possibilitar a verificação das cadeias de confiança.

- Prevent a user from reading messages from other than their own message box;
- Prevent a user from sending a receipt for a message that they had not read.

Só o utilizador com a chave privada é que consegue ler as mensagens que lhe foram destinadas, no entanto é feita uma triagem das mensagens, a sua seleção é feita através de um dicionário auto-incrementado.

Os receipts são feitos de forma automática, ou seja, quando um utilizador lê correctamente uma mensagem nova é enviado um receipt para o remetente cifrado com a chave pública deste

```

Mensagens (Inbox/Outbox):
1 Received Messages:
1- Message 1 from user 2 (NEW!)

0 Sent Messages:
You didnt send any message yet.

Commands:
(/send <user> <text>) Send a Message
(/recv <msg_number>) Read message
(/status <msg_number>) Check Receipt Status
(<) go back to main menu

```

Fig.3 - Mensagens (comando /all)

```

Source: 2
Message:
ola tudo bem ?

Commands:
(<) go back to main menu

```

Fig.4 - Leitura de uma mensagem (comando /recv <id msg>)