universidade de aveiro deti departamento de eletrónica, telecomunicações e informática

# Secure Messaging Repository System

**MIECT - Segurança 2017-2018 - p2g3**

**Realizado por:**
André Rodrigues, Cristiano Vagos

**Os Professores:**
André Zúquete, João Paulo Barraca

Aveiro, 9 December 2017

# Indice

# PROJECT OVERVIEW

## PROBLEMA

O objetivo deste projeto é desenvolver um sistema que permita aos utilizadores a troca de mensagens de forma assíncrona. As mensagens são enviadas e recebidas através de um repositório central não confiável, que guarda as mensagens dos utilizadores para futuros usos, como por exemplo ler conteúdo das mensagens.

O sistema é composto por um Rendezvous Point (Servidor) e por vários clientes que poderão utilizar o serviço após o seu registo.

## OBJECTIVOS

O sistema deve ser projectado para suportar os seguintes requisitos de segurança:

(dados do enunciado do projecto)

System must guarantee:

1. Confidentiality, integrity and authentication
2. Message delivery information
3. Identify preservation

To grade, system must:

1. Setup of a session key between a client and the server prior to exchange any command/response
2. Authentication (with the session key) and integrity control of all messages exchanged between client and server
3. Add to each server reply a genuineness warrant
4. Register relevant security-related data in a user creation process
5. Involve the Citizen Card in the user creation process
6. Encrypt messages delivered to other users
7. Signature of the messages delivered to other users (with the Citizen Card or another private key) and validation of those signatures
8. Encrypt messages saved in the receipt box
9. Send a secure receipt after reading a message
10. Check receipts of sent messages
11. Proper checking of public key certificates from Citizen Cards
12. Prevent a user from reading messages from other than their own message box
13. Prevent a user from sending a receipt for a message that they had not red

## REQUISITOS

Requisitos para correr o programa

Python2.7

pyopenssl

PyKCS11

…

Configurações

Chaves bits, cifras, modos cifra, rsa, simetricas

# PROJECT IMPLEMENTATION

## FUNÇÕES DISPONÍVEIS

O sistema implementado possui as funções básicas de um repositório de mensagens com features adicionais de modo a garantir os objectivos mencionados para o projecto.

### Registo

A função de registo permite que um novo utilizador seja capaz de criar um cliente no sistema, para tal é necessário fornecer dados tais como *username, password* e dados obtidos através do smartcard pertencente ao usuário (neste caso o Cartão de Cidadão) tais como *nome, serialnumber* e *certificados.*

O sistema possui várias restrições de modo a garantir consistência dos dados, por exemplo, um utilizador possui um *username* único, desta forma não existe mais do que um utilizador com o mesmo *username*.

Um utilizador pode registar mais do que um cliente, para tal é necessário que o seu *smartcard* seja válido e reconhecido pelo sistema.

### Conexão

Após o registo é possível efectuar uma nova conexão com o servidor, o utilizador deve introduz as suas credenciais de modo a garantir a sua identidade.

### Listagem dos Utilizadores Registados

É possível verificar quais os clientes registados no sistema e obter informações sobre estes tais como o *username, nome* (registado no smartcard) e *estado de ligação atual* (Online/Offline).

### Leitura, Envio e Estado das Mensagens (MessageBox)

Na zona MessageBox, um cliente pode verificar o estado da sua caixa de correio, poderá ler mensagens na sua caixa de entrada, mensagens recebidas, pode verificar a sua caixa de saída, consultar o estado das mensagens enviadas e o seu respectivo conteúdo e poderá enviar uma mensagem para outro cliente indicando o *username* deste.

### Emissão de um Recibo

Após a leitura de uma mensagem, o cliente poderá emitir um recibo da mensagem para o remetente, para tal é necessário o uso do seu smartcard. Após a emissão do recibo, o remetente poderá verificar o estado da mensagem, garantido que a mensagem foi lida correctamente e pela pessoa certa.

Conexão, estabelecer nova sessão
- autenticacao x
  - validação das credenciais x
  - validação do certificados x
  - challenge x
- estabelecimento de uma sessão x
  - diffie hellman x
- sessão estabelecida x
- sincronização dos dados x
  - atributos do cliente x
  - outros clientes (chaves, certificados, etc) x

Mensagens cliente-servidor
- encapsulamento
  - cifragem simétrica
  - hmac
  - envio do certificado
- validação do certificado

Mensagens cliente-cliente
- encapsulamento no servidor
  - mensagem
    - mensagem normal
    - cifragem mista / híbrida
    - receipt
    - hash
    - timestamp
  - receipt
    - assinatura
    - timestamp
    - mensagem, conteúdo

Captura wireshark dos pacotes

Organização do servidor
- tradução msgbox id <-> username

Load e save das chaves

Mensagens com as chaves após refreshing

Id das message box no servidor

# IMPLEMENTAÇÃO

## Registo

Numa primeira abordagem, um utilizador executa o cliente e irá registar a sua primeira conta, para tal, o cliente executa o comando **/create**, o servidor ao receber uma mensagem do tipo **create** irá fornecer a sua chave pública de modo a que o utilizador posso enviar as suas informações de registo de forma segura realizando uma cifra híbrida / mista, recebida a mensagem no cliente, é necessário agora fornecer um *username* não utilizado, uma *password* que irá ser utilizada para complementar a autenticidade e manter o seu par de chaves assimétricas guardado de forma cifrada (com a utilização da *password* como uma *passphrase*), a sua chave publica, RSA 2048 bits gerada pelo cliente, e os certificados das chaves públicas. Ao receber os dados necessários para o registo, o servidor procede à validação das credenciais (existência do *username)* e validação dos certificados (verificação da cadeia total de confiança de modo a garantir que os certificados não foram revogados.

Foi optado uma cifra híbrida inicial devido a ausência de uma sessão (e respectiva chave de sessão) e com o objetivo de garantir a confidencialidade dos dados relativos à criação da conta, assim, não é possível obter os dados tão facilmente através de uma intercepção.

## Conexão (*Challenge e Sessão)*

Quando um utilizador pretende conectar-se ao seu cliente, comando **/connect**, é feito novamente um pedido, desta vez de conexão ao servidor e a sua resposta vem acompanhada de um *challenge*, é necessário fornecer novamente as suas credenciais (no cliente), esta ação irá resolver o *challenge* e submete-lo para o servidor através do canal seguro inicial estabelecido com o uso de cifras hibridas, caso seja provada a identidade, isto é, o *challenge* foi bem "resolvido", o cliente pode passar para próxima fase, o estabelecimento de uma sessão com o servidor.

O *challenge* é um método de autenticação tradicional onde o servidor irá fornecer um conteúdo aleatório que terá que ser assinado pelo cliente (chave de autenticação), devolvido ao servidor e finalmente o servidor terá de verificar a assinatura e avaliar o estado de autenticação.

No estabelecimento de uma nova sessão o par cliente-servidor procede com o algoritmo de Diffie-Hellman, são acordados os valores necessários e em apenas num par de mensagens (cliente-servidor e servidor-cliente) é estabelecido uma chave de sessão. Apartir deste momento, o canal utilizado previamente (crifras híbridas) é abandonado e é criado um novo canal de comunicação seguro onde o conteúdo é cifrado recorrendo a uma chave simétrica derivada da chave de sessão, facilitando a "compressão" dos dados graças ao uso da cifragem simétrica, de notar que para cada pedido/mensagem trocada, é obtida uma nova chave derivada da chave de sessão.

De modo a garantir a segurança do canal ao longo do tempo, no final de um número aleatório pré-acordado, o cliente e o servidor vão acordar uma nova chave de sessão e também gerar um novo par de chaves assimétricas, conseguindo assim o *refrescamento das chaves.*

Estabelecida a sessão, todos os pedidos serão encapsulados e enviados neste novo canal de comunicação, para garantir a integridade, é utilizado o HMAC resultante da mensagem encapsulada e da chave derivada utilizada em cada mensagem.

Após o válido 'carregamento' do par de chaves assimétricas e conectado com o sistema, o cliente irá fazer uma sincronização com o servidor obtendo todos os dados relativos aos demais clientes (certificados, chave publica) permitindo também a utilização de mecanismos de segurança com outros clientes (cifra, decifra).

## Listagem dos Utilizadores Registados

De modo a efectuar a listagem dos utilizadores registados no sistema, o cliente terá que executar o comando **/list**, uma lista de utilizadores será disponibilizada indicando o *username, nome* e *estado de conexão,* os restantes dados relativos aos mecanismos de segurança de cada utilizador já foram previamente distribuídos na sincronização com o sistema, feito o login.

## Leitura, Envio e Estado das Mensagens (MessageBox)

É possível aceder a caixa de correio, *Message Box*, através do comando /all.

A *Message Box*, é constituída por dois dicionários com key auto-incrementado, o primeiro é respectivo à caixa de entrada e o segundo à caixa de saída. Na caixa de entrada, as mensagens são ordenadas por ordem de chegada, dando prioridade às mensagens não lidas sendo estas sinalizadas no topo, já na caixa de saída estas são apenas ordenadas pela ordem de envio.

É possível o envio de uma mensagem através do comando **/send <username> <mensagem>**, onde será feita uma verificação da existência do *username* indicado, o conteúdo da mensagem será cifrado com a utilização de cifras híbridas (utilização da chave publica do cliente destino, garantindo a *confidencialidade*) e a copia da mensagem, para uma futura verificação do estado da mensagem, será cifrada também com o uso de cifras híbridas (utilização da chave publica do próprio cliente, garantido a *confidencialidade*), juntamente com a mensagem será enviado a assinatura (assinada com a chave de assinatura) desta de modo a garantir a *autenticidade* da mesma.

A leitura da mensagem é feita recorrendo ao comando **/recv <id>**, onde o **id** seleciona a mensagem do dicionário correspondente à caixa de entrada. Para uma correcta leitura, é necessário que o cliente possua a sua chave privada carregada, ou seja, será usada a *password* como *passphrase* para fazer o seu "carregamento" e este terá que ser válido, assim, apenas quem a possui é capaz de a ler, *confidencialidade.*

No processo de leitura, são verificadas as cadeias de confiança dos certificados, a validade da chave de assinatura e a assinatura em si.

### Emissão de um Recibo

Durante a leitura, caso o cliente necessite, poderá enviar um *recibo* da leitura da mensagem informando quando é que a mensagem foi lida e que foi lida por si, para tal terá que o assinar com a sua chave de assinatura, relembrando que o seu conteúdo foi previamente cifrado recorrendo a uma cifra híbrida utilizando a chave publica do cliente que enviou a mensagem.

# MANUAL DE INSTRUÇÕES

# PROJECT CONCLUSION

## REFERÊNCIAS & CODE SNIPPETS

## CONCLUSÃO