DEPARTAMENTO DE ELETRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA,
UNIVERSIDADE DE AVEIRO

Segurança

Safe Comunication

João Branquinho, 76543
Tiago Ramalho, 76718
10 de Novembro de 2017

CONTEÚDO

1 FUNCIONALIDADES

1.1 SETUP OF A SESSION KEY BETWEEN A CLIENT AND THE SERVER PRIOR TO EXCHANGE ANY COMMAND/RESPONSE

Para criar uma sessão segura será utilizado uma chave simétrica para cifrar as mensagens trocadas entre o servidor e o cliente. Para estabelecer a chave simétrica será utilizado o método *Diffie-Hellman*.

1.2 AUTHENTICATION (WITH THE SESSION KEY) AND INTEGRITY CONTROL OF ALL MESSAGES EXCHANGED BETWEEN CLIENT AND SERVER

Depois de alguma consideração, decidiu-se que a melhor escolha para que fosse garantida a autenticidade e integridade seria acrescentar um *MAC* através do método *HMAC (Hash-based Message Authentication Code)* em todas as mensagens trocadas entre o cliente e o servidor, garantindo integridade e autenticidade.

1.3 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)

Para garantir o sucesso desta funcionalidade a solução pensada foi simples e envolve os seguintes passos:

- Quando o utilizador é criado (funcionalidade descrita mais abaixo) deverá enviar vários campos para além do seu *UUID*. Um dos quais deverá ser o valor inteiro **1** assinado com o seu cartão de cidadão. A resposta à criação do utilizador vem com o mesmo valor assinado pelo utilizador.
- Para cada mensagem que o utilizador enviar deve incrementar em uma unidade este valor, assinar o mesmo e enviar. O servidor responderá com este mesmo valor identificando a resposta da mensagem
- Este valor deverá ser armazenado tanto pelo servidor como pelo utilizador. Assim o utilizador sabe que valor tem de assinar para realizar o pedido seguinte e o servidor sabe qual o valor a receber

Desta forma cada mensagem Cliente/Servidor é acompanhada por um identificador de mensagem bem como a sua resposta, identificando o par *request/reply*. Este método garante também que o servidor autentica o utilizador a cada mensagem que recebe, dado que para cada pedido só um utilizador conseguiria assinar aquela mensagem, garantindo o não repudio.

1.4 REGISTER RELEVANT SECURITY-RELATED DATA IN A USER CREATION PROCESS

Na criação do cliente *CREATE* vai ser guardado o certificado do cliente, a sua chave pública que irá servir para encriptar mensagens enviadas para ele. Esta função vai ainda guardar

no *UUID* uma *Hash* da chave pública do cliente bem como o identificador de mensagem assinado referido anteriormente.

1.5 Involve the Citizen Card in the user creation process

O cartão de cidadão será utilizado para:

- Assinatura de mensagens pessoais entre clientes;
- Assinatura do identificador de mensagem; e
- Será usada na criação do *UUID* através do *diggest* da sua *public key*.

1.6 Encrypt messages delivered to other users

As mensagens enviadas vão sofrer uma encriptação mista, assim, o texto em branco é cifrado com uma chave simétrica e depois encriptado com a chave publica do cliente destinatário.

1.7 Signature of the messages delivered to other users (with the Citizen Card or another private key) and validation of those signatures

Quando um utilizador gera uma mensagem esta é imediatamente assinada e só depois é aplicada a cifra mista. Assim quando o recetor decifra a mensagem tem imediatamente acesso à assinatura e seu certificado que deverá verificar até à raiz.

1.8 Encrypt messages saved in the receipt box

As mensagens na *rbox* vão ser encriptadas com uma cifra mista que usará a chave publica do próprio cliente, assim só ele irá conseguir obter novamente a mensagem enviada.

1.9 Send a secure receipt after reading a message

Para confirmar a receção da mensagem, o destinatário, envia um *RECEIPT* para o servidor assinando o campo *receipt* que tem um hash da mensagem original.

1.10 Check receipts of sent messages

Para confirmar, o *receipt* enviado por outro cliente, vai ser verificada a assinatura, e depois vai ser comparada a hash que se recebeu com a hash criada a partir do texto original que está na rbox.

1.11 Proper checking of public key certificates from Citizen Cards

A validação da cadeia de certificação deverá ser feita mais do que uma vez:

- Quando um utilizador é criado o servidor vai armazenar o certificado que é enviado junto do *UUID*. Antes de armazenar estes dados deverá verificar a cadeia de certificação do utilizador. Isto permite que não seja criado um utilizador que fornecer uma cadeia de verificação inválida, obrigando à utilização do cartão desde o principio.
- Quando um utilizador recebe uma mensagem que lhe foi enviada por outro utilizador é do seu interesse verificar a entidade da pessoa que lhe enviou a mensagem. Para isso vê-se obrigado a confirmar a cadeia de certificação.

1.12 Prevent a user from reading messages from other than their own message box

Uma maneira de prevenir tal situação é, como já foi referido, as mensagens serem cifradas com a chave pública a quem a mensagem vai ser enviada e deste modo só esse poderá decifrar a mensagem e ler o seu conteúdo. Para além deste caso vai também ser implementado uma função no servidor para impedir a resposta caso o cliente esteja a consultar a *mbox* de outro, deste modo o servidor verifica quem está a fazer o pedido e caso não seja o dono da *mbox* envia uma mensagem a dizer que não pode ver certos conteúdos. Assim não vai ser acrescentado o *underscore* (_) à mensagem e então não será considerado que foi lida.

1.13 Prevent a user from sending a receipt for a message that they had not read.

Para cumprir este requisito é importante a funcionalidade anterior pois caso a mensagem ainda não tenha o *underscore*, o servidor não vai aceitará esta mensagem e será ignorada, o mesmo irá acontecer caso um cliente envie um *receipt* de uma mensagem que não lhe pertença.

2 Mecanismos e Métodos de cifra

Neste trabalho serão utilizados diferentes chaves e métodos de cifra. Nesta secção será explanado quais os considerados ideais para o efeito.

2.1 Chaves de sessão

Em cada sessão Cliente servidor serão utilizadas chaves simétricas. Para cifrar cada uma das mensagens de sessão será utilizada a cifra AES com o método CTR.

2.2 Cifras mistas

Para garantir que só a pessoa a quem é dirigida a mensagem a lê é necessário, ou pelo menos recomendável, o uso de chaves assimétricas. O inconveniente do seu uso é a velocidade de cifra/decifra ser reduzida. Posto isto, neste projeto, serão usadas cifras mistas. Uma chave simétrica cifra o *Plain Text*, depois disto uma chave assimétrica cifra todos os dados necessários para que o recetor da mensagem a decifre:

- *Initialization vector*
- Chave simétrica
- *Padding* (opcional)

Para este propósito serão usadas chaves assimétricas RSA de 2048 bits com recurso a *OAEP* e chaves simétricas AES com método CTR.

2.3 HMAC

Para cada chave de sessão trocada entre o cliente e o servidor, será acrescentado um *MAC* com o método *HMAC* que irá garantir autenticidade e integridade da mensagem.

2.4 SHA-256

Para a criação do *ID* de utilizador será utilizado o *SHA-256* como função de *diggest*.