universidade de aveiro deti departamento de eletrónica, telecomunicações e informática

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

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 pedindo a sua chave pública de modo a conseguir aplicar no próximo pedido uma cifra híbrida, uma vez que não possui nenhum registo nem chave secreta previamente estabelecida, o cliente vai gerar uma simétrica aleatória e cifrar com a chave pública do servidor para efectuar o registo, no segundo caso, após o registo (criação da ***message box***), quando tenta autenticar-se, isto é, ligar-se à sua ***message box***, o cliente e o servidor entrarão numa fase de autenticação composta por um challenge que iremos descrever mais a frente e um estado de negociação de uma chave secreta de sessão através do algoritmo de Diffie-Hellman e irão usar esta chave na troca de mensagens.

Esta chave de sessão secreta será utilizada para gerar uma chave derivada (***key derivation functions***) com um ***salt*** acordado para cifrar mensagens seguras entre cliente e o servidor e para cada pedido entre os peers.

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 total, derivada(chave secreta)) para garantir a integridade da informação trocada entre as duas entidades (cliente-servidor), assim o servidor tem a certeza que a mensagem não foi ***forged***.

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 comparado com a cifra assimétrica por esta não utilizar o princípio da difusão de Shannon.

Garantindo a confidencialidade, iremos garantir a autenticidade e integridade 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',
    'messageCiphered' : {
                                'type' : 'list',
                                    ...

                            }
    'HMAC': ...
}
```

Fig.1 - Encapsulamento

Numa primeira tentativa de conexão de um cliente à sua ***message-box*** inicia-se uma fase de autenticação onde o cliente demonstra interesse em conectar-se e o servidor produz um ***challenge*** (conteúdo random), o cliente ao receber este conteúdo terá que o **cifrar** com a sua **chave privada** e enviar de volta para o servidor, o servidor **decifra** com a **chave pública do cliente** (previamente registada na criação) e compara com o valor random inicial, caso seja igual, o challenge foi bem sucedido e o cliente é autenticado e segue para o próximo passo onde 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 **toda**) e da chave de sessão.

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

No envio de uma mensagem, é derivada uma chave da chave de sessão com um determinado **salt**, este **salt** será fornecido à entidade destinatária juntamente com a mensagem de modo ser possível derivar outra chave igual no destino.

Quando recebida uma mensagem, o servidor ou o cliente irá analisar, caso seja do tipo ***'secure'*** irá criar a chave derivada utilizando o **salt** fornecido e irá decifrar a mensagem com esta mesma chave, obtendo a mensagem que estava encapsulada e verificando a sua integridade, 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, outra abordagem seria o envio de um ***acknowledgement*** o que tornaria este processo mais lento.

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), serial number (CC), a chave pública e outras informações complementares tais como a **chave de sessão que será renovada a cada x pedidos**, o estado de conexao (connected, not connected), o numero privado do servidor que será diferente para cada cliente de modo a gerar uma chave de sessão unica 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 este 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 publica deste mesmo cliente source, garantindo a confidencialidade na execução de um status da mensagem, só quem mandou pode ler o status de uma mensagem

enviada. 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 red.**

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 (Fig.3), o mesmo para os receipts, apenas o cliente que enviou consegue ler os seus receipts (campo ***copy*** cifrado com a sua publica).

Os receipts são feitos de forma automática, ou seja, quando um utilizador lê correctamente uma mensagem nova (Fig.4) é enviado um receipt para o remetente cifrado com a chave pública deste.

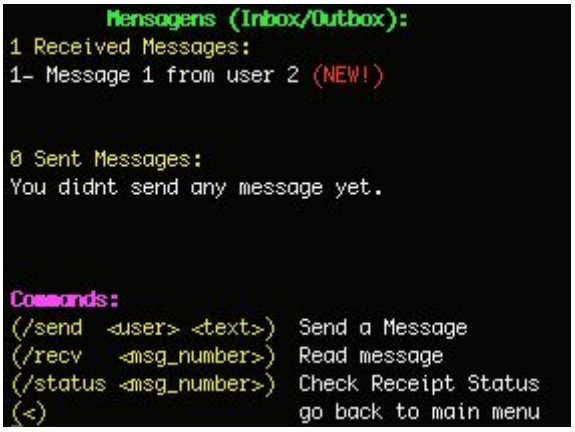

Fig.3 - Mensagens (comando /all)

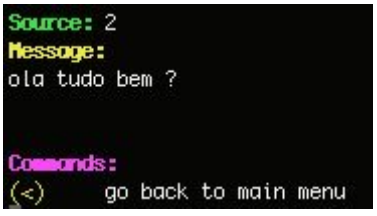

Fig.4 - Leitura de uma mensagem (comando /recv <id msg>)