TÉCNICO LISBOA

AISS Security Email Service

Sistema de Segurança de Emails

Manual Técnico

57701: Gonçalo Carito

68210: Dário Nascimento

MERC

IST-TAGUSPARK

2012/2013

Conteúdo

Capítulo 1

Introdução

Este documento descreve a implementação de um Serviço de Emails Seguro. Esta implementação teve como premissa o pedido de um **cliente** de uma proposta de plugin que se adaptasse, preferencialmente, ao cliente de emails *Mozilla Thunderbird* e que efectuasse as seguintes operações de segurança:

- Cifra/Decifra AES com chave gerada por caixa fornecida pelo Cliente
- Assinatura com Cartão de Cidadão da República Portuguesa
- Assinatura e Verificação de Timestamp Seguro

Desta forma o Serviço de Emails Seguro tem dois sujeitos:

- Sender - Tem como input uma pasta de dados que pode comprimir, cifrar, assinar e adicionar timestamp e gera um output para ser enviado ao destinatário (por exemplo por email)
- Receiver - Tem como input o email seguro do sender que irá decifrar, verificar assinatura e timestamp e descomprimir para receber o conteúdo original.

No **Capítulo 2** é apresentada a Arquitectura do Sistema constituido pelo módulo cliente onde são implementadas as funcionalidades de cifra, assinatura e compressão, e pelo servidor de timestamp seguro.
No **Capítulo 4** são tiradas as conclusões sobre o trabalho efectuado.

Capítulo 2

Arquitectura

Neste capítulo é apresentada a arquitectura do sistema, a interligação entre os vários módulos e uma explicação detalhada do funcionamento dos vários procedimentos como a cifra e assinatura.

2.1 Visão Geral

A solução desenvolvida permite a troca de mensagens de modo seguro e com garantias temporais. Esta solução é constituída por 3 módulos: Sender, Receiver e um servidor de Timestamp (**Figura 2.1**).

O Sender pretende enviar um conjunto de dados para o receiver mas com garantias de **confidencialidade**, **autenticidade**, **não repudiação** e instante temporal. Estas garantias são dadas por mecanismos externos: CipherBox, Cartão Cidadão e Servidor Timestamp, respectivamente.

Num cenário de utilização, o sender selecciona a pasta que pretende enviar e escolhe cada uma das seguintes acções:

- Sign - assinar o(s) ficheiro(s) com o seu Cartão de Cidadão da República Portuguesa
- Timestamp - adicionar um timestamp seguro assinado por uma Autoridade Certificada

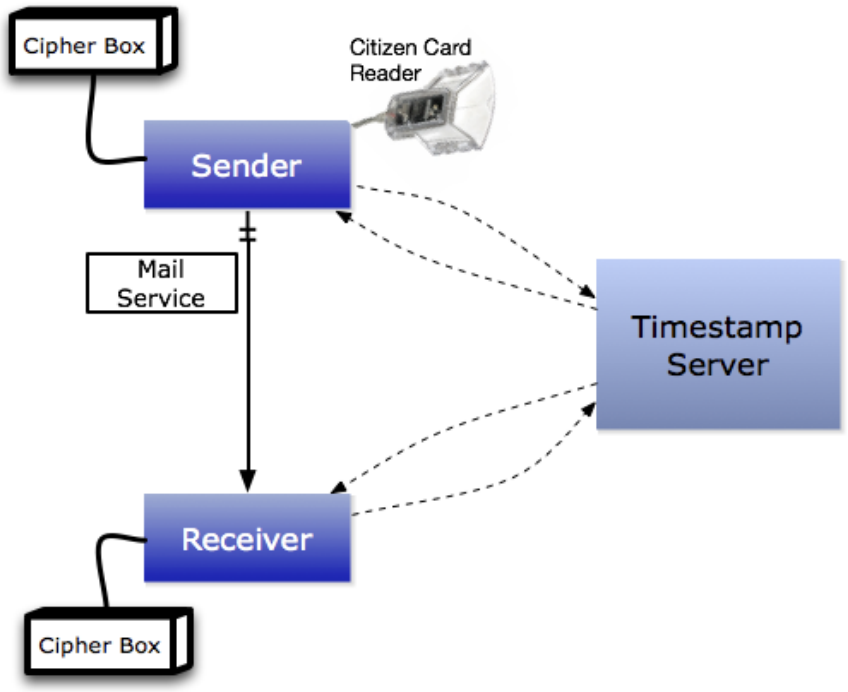

Figura 2.1: Arquitectura do Sistema

- Cipher - cifrar o(s) ficheiro(s)

Supondo que todas as opções foram seleccionadas é realizada a cadeia de acções descrita na **Figura** 2.2.

Os ficheiros seleccionados, são comprimidos num único. De seguida este ficheiro é assinado recorrendo ao Cartão de Cidadão e o seu *hash* é enviado ao servidor de timestamp seguro. O servidor cria um par associando um instante temporal ao hash. Este par é assinado pelo servidor afim de garantir que o servidor recebeu o hash no instante que lhe foi associado. Este certificado de Timestamp é devolvido ao sender. Mais detalhes na Secção 2.4.

Por fim, a assinatura de cartão de cidadão, o certificado de Timestamp e os dados são organizados numa estrutura e esta estrutura é cifrada. Os dados cifrados são colocados dentro de uma nova estrutura que descreve as operações realizadas e permite inverter o processo.
Por fim, esta estrutura é codificada e escrita em disco em formato de ficheiro para que o Sender possa enviar.

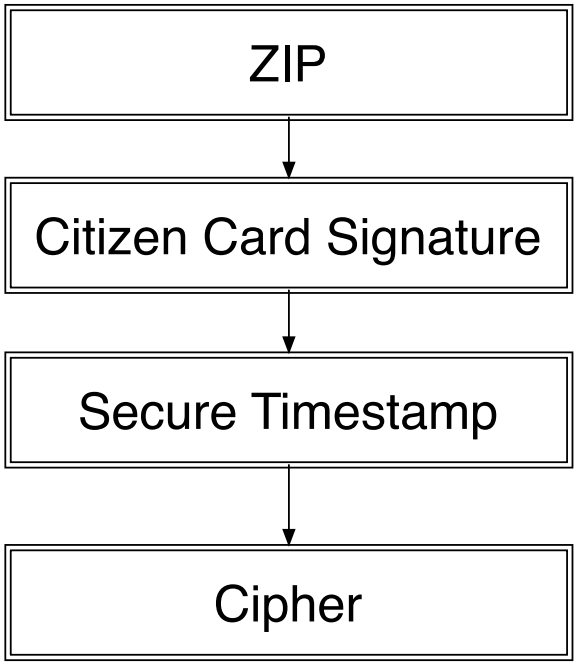

Figura 2.2: Ordem de execução

Quando o Receiver recebe este ficheiro protegido efectua as operações pela ordem inversa, começando pela decifra, verificação da validade do Timestamp através do servidor, verificação a Assinatura e descomprimindo os dados.

2.2 Zipping/Unzipping

Sempre que o Sender envia uma mensagem, seleciona a pasta que pretende enviar. Esta pasta é comprimida utilizando a biblioteca *Zip* nativa do Java 6. O output desta operação é um ficheiro único que contém toda a pasta comprimida e por isso reduz a dimensão do anexo a cifrar, assinar e enviar.
É então criada uma estrutura que será preenchida nos próximos passos com os metadados.

2.3 Sign/Verify

A segunda operação a ser executada pelo Sender é a operação de Assinatura; no caso do Receiver esta é a penultima última operação a ser executada (antes do un-

zip).
Para garantir a não repudiação e autenticidade do ficheiro comprimido é utilizado o Cartão do Cidadão através da biblioteca PKCS11. O ficheiro é *hashed* com SHA-1 e assinados pela chave RSA privada contida no cartão de cidadão. A assinatura e o Certificado de chave pública do cartão de cidadão são colocados na estrutura de metadados.

Do lado do Receiver, a assinatura é verificada em 2 passos. Primeiro verificamos se o certificado enviado é válido, verificando a assinatura do certificado contra os *root certificate* do Estado Português que é incluído na solução. De seguida, verifica se a mensagem foi assinada pelo detentor do certificado. Caso a mensagem seja válida, é apresentado o nome do detentor do certificado.
Este mecanismo possui uma optimização por assinar apenas o contéudo comprimido. Não obstante, **garantimos apenas a não repudiação do contéudo comprimido e não do contéudo original.**

2.4 Timestamp Seguro

Para utilizar um mecanismo de Timestamp seguro, o Sender envia ao Servidor de Timestamp um pedido com o hash do(s) ficheiro(s).
O servicor cria um Timestamp com a data actual e anexa-o ao hash recebido. Este par é assinado com a chave privada do servidor, criando uma mensagem com o formato: *Kp(H(m),T),T* Em que H(m) é a hash da mensagem, T o timestamp e Kp a chave privada do Servidor de Timestamp.
Este serviço apenas garante que **este hash foi assinado pelo servidor na data definida.**

O Receiver verifica a validade do timestamp, ou seja, o momento em que o servidor de confiança associou o hash da mensagem. Para tal, é utilizado o certificado do Servidor de Timestamp seguro que está incluido no pacote da solução.
Não é dada qualquer garantia da data de envio de email porque o utilizador pode enviar o mail à-posteriori.

2.5 Cipher/Decipher

A última operação a ser executada pelo Sender e a primeira a ser executada pelo Receiver é a operação de cifra (e decifra, respectivamente). Para efectuar a cifra, o Sender utiliza a Caixa de Cifra via ethernet fornecida pelo Cliente. O software base para connecção com a caixa foi disponibilizado em C. As invocações Java foram convertidas para invocações C através do JNI. A mensagem é cifrada por blocos, o que possibilita a cifra de ficheiros de grandes dimensões. O Receiver utiliza também uma Caixa de Cifra ethernet para decifrar a mensagem.
Esta caixa realiza cifra e decifra AES. A chave utilizada está contida na caixa por isso esta caixa constitui um mecanismo de segredo partilhado, ou seja, apenas quem tem a caixa pode decifrar o contéudo.

2.6 Base64

Antes de salvar (ou enviar para o email) o ficheiro gerado é codificado em base64. Este sistema codifica cada byte em 6 bits. Estes 6 bits são caracteres conhecidos pela norma ASCII e por isso o ficheiro pode ser enviado em formato de *plain-text* sem que alguns sistemas de codificação e *sanitize* removam caracteres desconhecidos e deste modo adulterem o contéudo da mensagem.

Capítulo 3

Avaliação

De modo a confirmar a segurança do nosso sistema, adulteramos as mensagens e assinaturas realizadas. Confirmarmos que o sistema detecta corretamente a adulteração de mensagens.
Foram realizados vários testes de desempenho do sistema utilizando um computador de utilizador comum MacBook Pro com processador 2.9 GHz Intel Core i7, memória 8 GB 1600 MHz DDR3 e sistema operativo MacOsX 10.8.3 e disco SSD (evidênciado na latência de escrita em disco) O servidor de timestamp foi executado localmente pelo que não são considerados atrasos na rede.
Foram realizadas 5 amostras por cada caso e determinados os níveis de confiança a 95% para os tempos obtidos. O directório enviado é constituído por 286 ficheiros PDF e Word que totalizam 100MBytes.

- **Compressão:** A compressão do ficheiro converte 100MB em 95.5MB. Esta taxa depende do formato e contéudo dos ficheiros de origem. Demorou em média 4484 com desvio padrão de 30ms.

- **Hashing do ficheiro:** A realização do hash do ficheiro comprimido para utilizar nas assinaturas demora em média 700ms com desvio padrão de 10ms.

- **Assinatura com cartão de cidadão:** A assinatura com cartão de cidadão demorou 936ms com desvio padrão de 24ms.

- **Timestamp:** O serviço de timestamp demora 70ms a decorrer com desvio padrão de 5ms.

- **Codificação e escrita em disco:** A conversão para base 64 e escrita em disco demorou 1335ms com desvio padrão de 35ms. A codificação de base 64 implica um aumento da dimensão do ficheiro ao factor de 8/6 pelo que o ficheiro final tem 130.7MB.

- **Descompressão e escrita em disco**A descompressão demorou 1380ms com desvio padrão 32ms.

- **Verificação timestamp:** A verificação do timestamp demorou 32ms com desvio padrão de 6ms.

- **Verificação Assinatura:** 6ms, desvio de 3ms.

- **Cifra e Decifra:** A especificação técnica da caixa fornecida pelo cliente indica 60 segundos para cada 100MB pelo que assumimos este valor na ausência da possibilidade de realização de testes experimentais.

No total o processo de envio demora (excluindo a cifra) 6 segundos enquanto que a decifra demora 3 segundos. Estes valores não são significativos para a experiência do utilizador quando comparados com o tempo de inserção do PIN do cartão de cidadão e da cifra pela caixa fornecida pelo cliente.
A excelente performance do nosso sistema deve-se em parte ao facto de os dados só serem lidos e escritos uma única vez no disco. O reduzido numero de acessos a disco reduz o atraso do processo. Por outro lado implica a utilização de mais memória RAM do utilizador. Este processo teve um pico de utilização máxima de memória de 300MB. Consideramos que este custo não é significativo dadas as especificações atuais dos computadores para utilizador final. Caso o utilizador tenha menos memória, a solução é aplicável mas para ficheiros de menores dimensões.
A análise de performance concluí portanto que a nossa solução é eficiente.

Capítulo 4

Conclusão

Tendo em conta os principais desafios que foram apresentados no ínicio do trabalho como:

- Possíveis problemas de memória
- Assegurar eficiência nas operações
- Cartão de Cidadão com uma biblioteca instável

Mas também os principais objectivos, como:

- Ter uma interface user-friendly
- Ser um sistema compatível com aplicações que acedem a ficheiros
- Manter uma boa performance

Podemos concluir que os resultados obtidos foram muito adequados. A grande maioria dos problemas que poderiam vir a existir foram resolvidos. Naturalmente que haveria sempre melhorias a executar como tornar o Sistema compatível com outros Sistemas Operativos assim como ser mais flexível e menos condicional a interacção com outros Clientes de Email para além do Thunderbird.

Apesar de ser uma biblioteca instável e de haver o relato de alguns problemas de utilização com o Mac OS, a biblioteca do Cartão de Cidadão demonstrou um bom desempenho ao longo do tempo de desenvolvimento do projecto, raramente falhando. A utilização da bibloteca gráfica swing do Java permitiu o desenho de uma interface