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 manual descreve a nossa 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, tal como qualquer cliente de email tem dois principais sujeitos:

- Sender - quem envia o email e pode cifrar, assinar e adicionar timestamp a este
- Receiver - quem recebe o email seguro e pode decifrar, verificar assinatura e timestamp

A arquitectura e algoritmo foram pensados não só tendo em conta as exigências do cliente - seja a utilização de uma caixa de cifra ou do Cartão de Cidadão e existênca de portabilidade - mas, também, as melhores práticas em implementação de algoritmos de Segurança.

No **Capítulo 2** é apresentada toda a Arquitectura do Sistema constituída por um Cliente que ora pode receber ora enviar emails para outro Cliente e por um Servidor de Timestamps, iinterligação entre estes módulos e uma explicação mais detalhada do funcionamento dos vários procedimentos como, por exemplo, cifra e assinatura.

No **Capítulo 4** são tiradas algumas conclusões sobre o trabalho efectuado.

Capítulo 2

Arquitectura

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

2.1 Visão Geral

Numa utilização típica do Sistema de Emails, existe um cliente que envia e um que recebe, interagindo ambos com um Servidor de Timestamp, caso o cliente escolha utilizar esta opção. Assim existem três entidades distintas durante um processo de envio de email seguro.

- Sender - quem envia o email seguro
- Receiver - quem recebe o email seguro
- Servidor de Timestamp - para assegurar um envio de data assinado por uma autoridade externa.

O cliente que envia, Sender, pode, de entre as seguintes opções, escolher uma ou todas:

- 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

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

A ordem pela qual as operações são executadas pelo sistema encontra-se na **Figura 2.1**. De notar que a operação de zip, compressão, é sempre executada independemente da escolha das restantes.

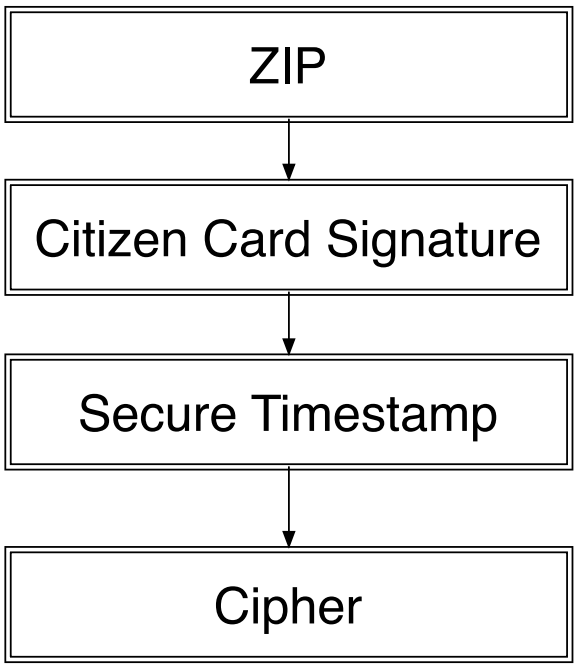

Figura 2.1: Ordem de execução

Supondo que o Sender irá executar todas as opções, observemos um cenário típico, através da ilustração presenta na **Figura 2.2**.

O Sender terá que ter uma Caixa de Cifra e o adaptador do Cartão de Cidadão e respectivo Cartão ligados ao seu PC. A primeira operação a ser executada, logo depois do zip, é a assinatura da mensagem. Para isso recorre ao seu Cartão de Cidadão e ao Certificado de Autenticação que se encontra dentro do mesmo. Depois de a mensagem estar assinada, a segunda operação a ser executada é a validação por timestamp. Tal como assinalado na figura **Figura 2.2**, o Sender envia um pedido ao Servidor de Timestamps e recebe um Timestamp seguro para juntar à sua mensagem: os detalhes desta operação serão explicados na Secção 2.4. Finalmente, a última operação a ser executada é a cifra com a Caixa de Cifra que foi fornecida pelo Cliente que requeriu este trabalho. Quando o Receiver recebe a mensagem protegida efectua as operações pela ordem inversa, começando pela decifra, verifica a validade do Timestamp através do servidor e, finalmente, verifica a Assinatura (fazendo sempre

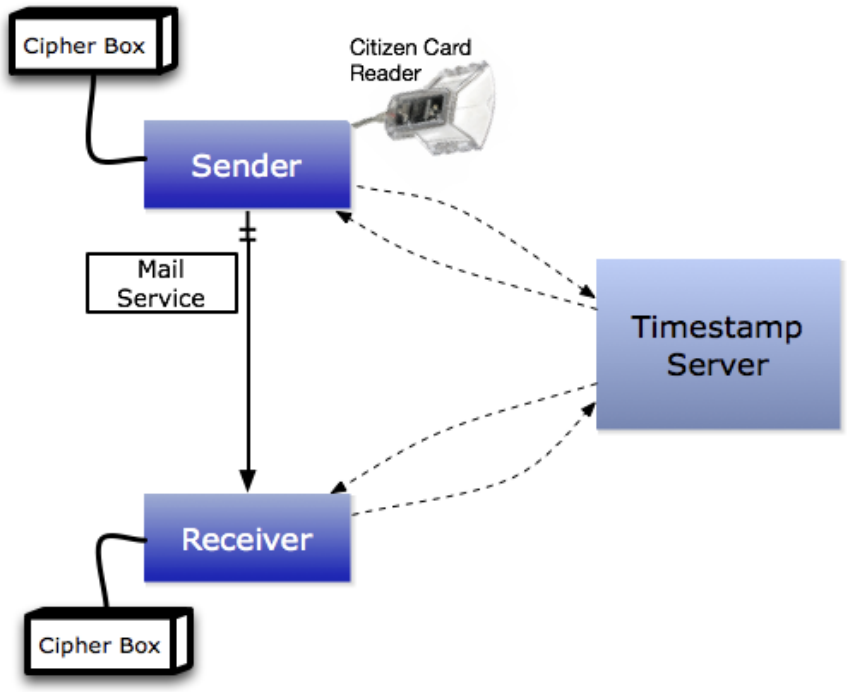

Figura 2.2: Arquitectura do Sistema

depois o unzip).

2.2 Zipping/Unzipping

Sempre que o Sender envia uma mensagem esta é, em primeiro lugar, zipada. O zip é efectuado em um ou mais ficheiros através da selecção da pasta com os dados a enviar, para isso é criado um Array com os ficheiros contidos na pasta, que são depois zipados. O output da operação de zip é um único ficheiro comprimido. A operação de unzip é feita de forma inversa. Para efectuar a operação, quer de zip quer de unzip, foi utilizada a biblioteca *zip* nativa no Java 6.

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 unzip). Para assinar/validar os ficheiros é utilizado o Cartão do Cidadão e a lib pkcs11. É utilizado o Certificado de Autenticação presente no cartão e a mensagem é assinada com a

respectiva chave privada. Juntamente com a mensagem assinada é também enviado o Certificado com a chave pública.

Do lado do Receiver, para este verificar se a mensagem é valida, utiliza a chave pública que foi enviada através do certificado.

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). Estre cria um Timestamp com a data actual e devolve para o Sender todos os dados cifrados com a sua chave pública, ou seja, assinados por este enquanto ”CA”. A mensagem é assim retornada no seguinte formato: Kp(H(m),T),T Em que H(m) é a hash da mensagem, T o timestamp e Kp a chave pública do Servidor de Timestamp.

O Receiver para verificar a validade do timestamp, ou seja, para assegurar que a mensagem foi mesmo enviada àquela hora, envia a mensagem assinada para o Servidor de Timestamp. A mensagem foi cifrada com a chave pública deste para que este possa ter a certeza que foi mesmo ele que a assinou. Para fazer a verificação utiliza, então, a sua chave privada.

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 ethernet fornecida pelo Cliente. A interface desta caixa é em C e para que o serviço interaja com esta foi utilizado o JNI. A mensagem é cifrada por blocos, o que possibilita a cifra de ficheiros de tamanho elevado.

O Receiver utiliza também uma Caixa de Cifra ethernet para descifrar a mensagem, aplicando-se a utilização das mesmas bibliotecas.

2.6 Base64

Antes de salvar (ou enviar para o email) o ficheiro protegido é feito do lado do Sender a codificação para base64 de forma a garantir compatibilidade com vários protocolos e evitar envio de cabeçalhos pois permite a codificação de 8 bits de data em 6 bits de transmissão sendo assim compativel com a maioria dos serviços de email que utilizem o protocolo standard.

O receiver faz a descodificação de base64 mesmo antes de proceder à decifra 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