universidade de aveiro
theoria poiesis praxis

Safebox

Segurança

Miguel Vicente

63832

Joel Pinheiro

65151

5 de Janeiro de 2015

Conteúdo

Capítulo 1

1 Arquitectura

Neste primeiro capítulo mostramos a arquitectura da nossa solução.

1.1 Visão Funcional

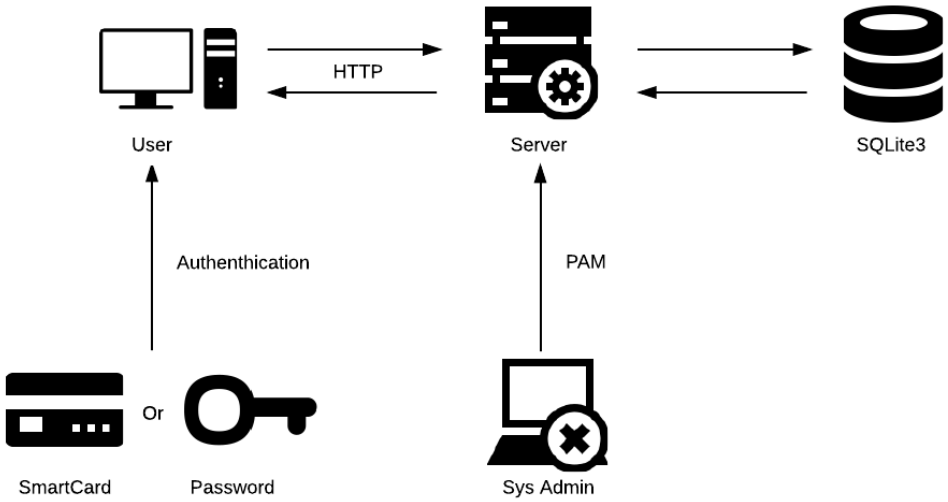

Figura 1.1: Arquitectura do Sistema

A arquitectura do sistema foi alterada de forma a permitir autênticação por SmartCard e por Password usando um módulo PAM.

Capítulo 2

2 Funcionalidades

No segundo capítulo mostramos as funcionalidades da solução.

2.1 Área pública

2.1.1 Funcionalidades

Este é o menu inicial da SafeBox. Nesta altura o utilizador não possui uma sessão iniciada na SafeBox, assim sendo são lhe apresentadas as seguintes opções:

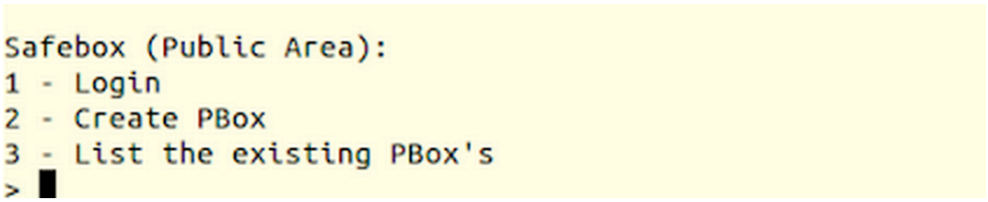

Figura 2.1: Funcionalidades da área pública

2.1.2 Login

Caso o utilizador já possua uma conta SafeBox, pode efectuar login na aplicação escolhendo a opção 1 e após inserir o smartcard ou, no caso de não ter, as credenciais de autênticação e estas verificadas terá acessos às restantes funcionalidades da SafeBox.

```
Safebox (Public Area):
1 - Login
2 - Create PBox
3 - List the existing PBox's
> 1
What's your username?
> client
What's your password?
>
```

Figura 2.2: Antigo Login

2.1.2.1. Novo Login

Figura 2.3: Novo Login

Figura 2.4: Validação do cartão de cidadão

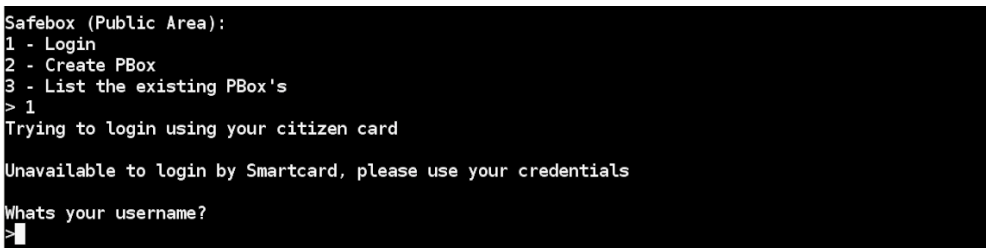

Figura 2.5: Cartão de cidadão não disponível. Autênticação por credenciais.

2.1.3 Nova Pbox

Esta opção permite a um novo utilizador criar uma PBox. Nesta fase da criação de utilizador, são geradas as suas chaves RSA pública e privada para futura cifragem e decifragem de ficheiros.

A chave privada ficará guardada no directório da aplicação de cliente e todos os restantes dados do utilizador são guardados a nível da base de dados. A sua password passa por um processo de hashing com a biblioteca bcrypt e ficará com esse formato na base de dados.

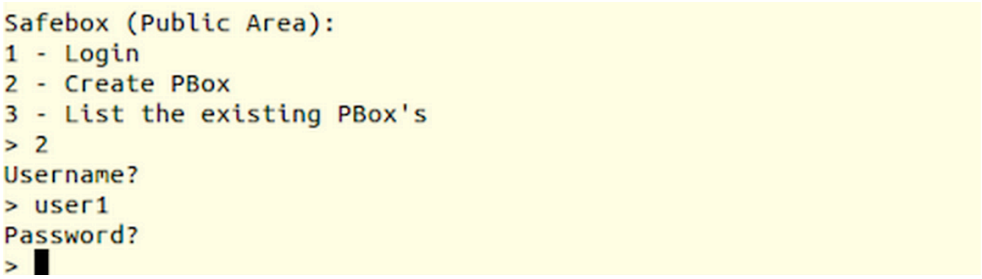

Figura 2.6: Nova Pbox

2.1.4 Logout

Termina a sessão da aplicação.

2.2 Área Privada

2.2.1 Funcionalidades

Funcionalidades disponíveis para utilizadores registados.

```
What's your username?
> client
What's your password?
>
Hi client
1 - Add a protected file to my PBox
2 - Get the file contents of a protected file in my PBox
3 - List my PBox files
4 - List files shared
5 - Share a file in my PBox with other PBoxes
6 - Delete a file from my PBox
7 - Unshare a file with other PBox
8 - Logout
> 
```

Figura 2.7: Funcionalidades da área privada

2.2.2 Upload de ficheiro

A opção de upload possibilita ao utilizador enviar um ficheiro para a sua Pbox. Caso o utilizador tente enviar um ficheiro com o mesmo nome de um ficheiro já existente na sua Pbox é lhe perguntado se deseja esmagar o ficheiro já existente. Se aceitar, o ficheiro e todos a meta-informação relativa a este é reescrita. No upload do ficheiro é gerada a chave da cifragem simétrica AES. Gerado o IV e Access e guardado na base de dados. É também adicionada uma entrada na tabela file e sharing (como owner).

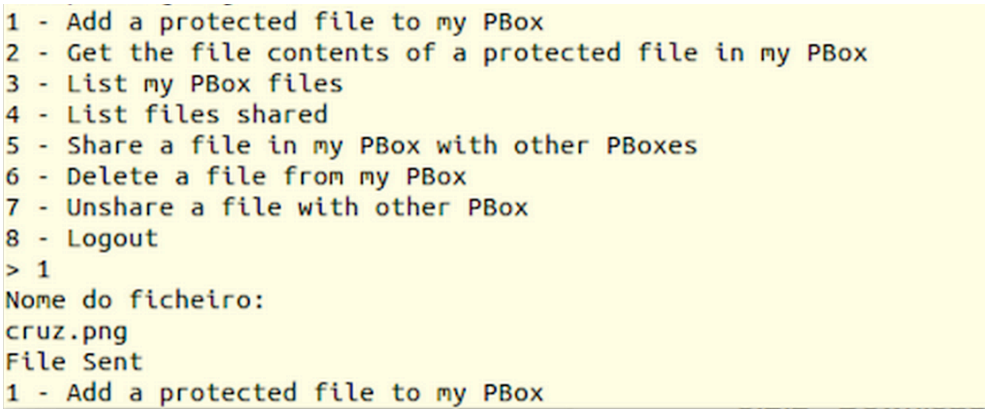

Figura 2.8: Upload de ficheiro

2.2.3 Download de ficheiro

Na opção download é imprimida uma lista de ficheiros cuja Pbox tem acesso. De seguida o utilizador seleciona qual o ficheiro que pretende descarregar. É validado se o ficheiro que o utilizador quer existe.

O ficheiro cifrado é descarregado aos chunks. É decifrado o IV e Access com a chave privada do cliente que está guardada localmente e temos a cifra do AES. Com a cifra de AES deciframos o ficheiro cifrado e gravamos o ficheiro no cliente na mesma pasta onde o cliente está a correr o programa.

2.2.4 Listagem de ficheiros na Pbox

É apresentada ao utilizador uma lista de ficheiros a que ele tem acesso na sua PBox, originais ou partilhados.

```
1 - Add a protected file to my PBox
2 - Get the file contents of a protected file in my PBox
3 - List my PBox files
4 - List files shared
5 - Share a file in my PBox with other PBoxes
6 - Delete a file from my PBox
7 - Unshare a file with other PBox
8 - Logout
> 3
7 cruz.png
```

Figura 2.9: Listagem de ficheiros na PBox do utilizador

2.2.5 Listagem de ficheiros partilhados com outros utilizadores

É apresentada ao utilizador uma lista com os ficheiros que o utilizador partilhou com outros utilizadores e o respectivo nome de utilizador.

```
Type username to share file with: client
File was shared
None
1 - Add a protected file to my PBox
2 - Get the file contents of a protected file in my PBox
3 - List my PBox files
4 - List files shared
5 - Share a file in my PBox with other PBoxes
6 - Delete a file from my PBox
7 - Unshare a file with other PBox
8 - Logout
> 4
7 cruz.png client
```

Figura 2.10: Listagem de ficheiros partilhados com outros utilizadores

2.2.6 Partilhar um ficheiro com outro utilizador

Permite ao utilizador partilhar um ficheiro com outro utilizador. É apresentada ao utilizador a sua lista de ficheiros, da qual ele deve escolher o que quer partilhar e de seguida inserir o nome de utilizador com quem pretende partilhar.

A lógica de partilha segura do ficheiro passa por obter do servidor a chave pública do utilizador ao qual se vai dar acesso e a chave AES e o vector de inicialização (cifrados) do respectivo ficheiro.

No aplicação de cliente é decifrada a chave AES e o vector de inicialização com a chave privada do utilizador e recifrados com a chave pública do novo utilizador que irá ter acesso ao ficheiro.

É adicionado um novo registo na base dados na tabela de sharings, com a identificação do ficheiro, a recifra dos dados de acesso ao ficheiro, a identificação do utilizador que obteve acesso ao ficheiro e a identificação do utilizador que lhe deu acesso ao ficheiro.

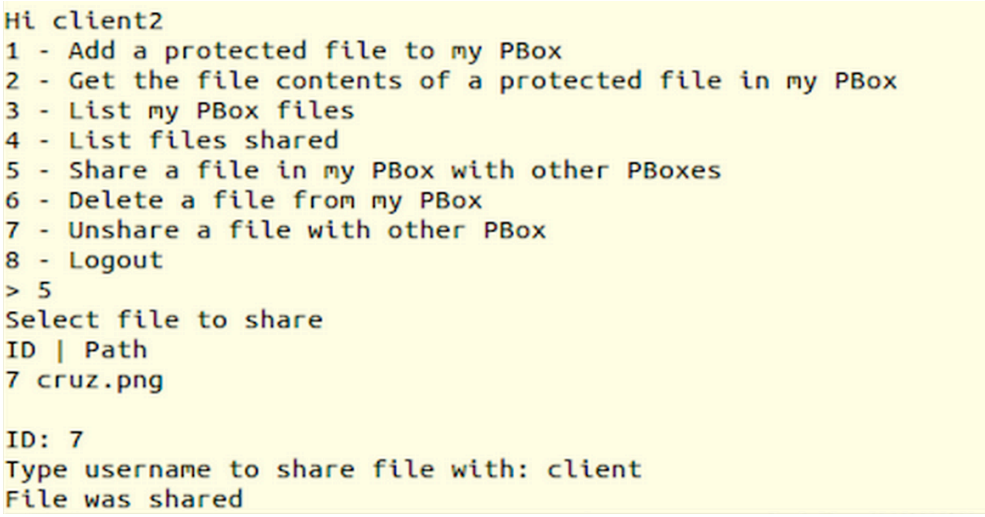

Figura 2.11: Partilha de um ficheiro com outro utilizador

2.2.7 Apagar um ficheiro da Pbox

Na opção apagar um ficheiro é imprimido uma lista de ficheiros que aquele utilizador é owner. É validado se o utilizador selecionou um dos ficheiros da lista. Se sim, são eliminadas todas as entradas da tabela sharing que estejam associadas aquele ficheiro e também a entrada na tabela file do ficheiro que se está a remover.

No final, o ficheiro é efectivamente removido da safebox para sempre.

Capítulo 3

3 Abordagem

Neste terceiro capítulo é explicada a abordagem mais orientada à segunda fase do projecto.

3.1 Registo de Utilizador

Visto que o principal objectivo para esta fase do projecto seria introduzir na Safebox uma lógica de autenticação por smartcard, tomámos a decisão de apenas permitir ao utilizador registar-se na aplicação na presença do seu cartão de cidação ligado ao computador em questão. Assim, garantimos a componente humana em cada registo e criação de uma PBox e limitamos a existência de uma PBox por Cartão de Cidadão.

Aquando o registo, são extraídos os dados do smartcard necessários à gestão da PBox mais especificamente o expoente e o módulo da chave pública do cartão e o identificador, que são armazenados na base de dados do sistema. Numa perspectiva de validação, é verificada toda a cadeia de certificados do smartcard em questão quando à sua data de validade, validade da sua chave pública e revogação.

É também pedido ao utilizador para escolher um nome de utilizador e uma password, visto que a autenticação por password também é permitida no nosso sistema, no caso da ausência de um cartão para ler.

A password de utilizador não pode ser inferior a 6 caracteres e após ser introduzida, é submetida a uma função de hash do tipo bcrypt e enviada para o servidor.

3.2 Autênticação de utilizador registado

A autenticação de utilizador na aplicação baseia-se no esquema CHAP, utilizando as capacidades de assinatura do Cartão de Cidadão.

O servidor ao receber um pedido de autenticação gera uma sequência de números aleatórios. Essa sequência é enviada para o cliente, que deverá assiná-la com as propriedades criptográficas do seu cartão de cidadão. A assinatura é enviada para o servidor que a envia para a PAM que terá de verificar a assinatura com os dados da chave pública do utilizador guardados no sistema. Se a assinatura for correctamente verificada o utilizador encontra-se autenticado. Para podermos identificar qual o utilizador que pretende autenticar-se e a chave pública que deverá verificar a assinatura, é utilizado o identificador do Cartão de Cidadão.

Mais uma vez, a cada tentativa de autenticação é efectuada a validação da cadeia de certificados do smartcard em questão.

Na impossibilidade de este processo ser feito através de smartcard, quer por inexistência de um ou caso este não esteja válido, é dada a possibilidade ao utilizador de se autenticar com as credenciais criadas no seu registo. Neste processo o cliente gera uma nova hash, com base naquela guardada na base de dados para o username que inseriu. A nova hash é enviada para o servidor, que a envia para a PAM que terá de verificar se as duas hashes são idênticas.

3.3 Gestão de sessão

A gestão de sessão de um utilizador na aplicação está relacionada com o processo de autenticação na Safebox. Quando o processo de autenticação estiver concluído, é gerado aleatóriamente um identificador de sessão que é trocado nas comunicações entre cliente e servidor.

Para evitar que este seja apanhado por terceiros que tentem capturar uma conversação, todos os dados a comunicar estão cifrados com as respectivas chaves publicas de cliente ou servidor e são decifrados no respectivo destino com a chave privada do destinatário.

3.4 PAM

Na perspectiva de tornar a autenticação na Safebox expansível e administrável, o processo de autenticação é da responsabilidade de dois módulos PAM, um para cada processo de autenticação (smartcard ou password).

A PAM é composta por duas bibliotecas partilhadas (pam_safebox.so e pam_safebox_pw.so) sendo o primeiro responsável pela autenticação através de smartcard e o segundo por password.

A comunicação com o módulo PAM é feita exclusivamente do lado do servidor, visto que as limitações impostas pelo CherryPy tornam uma interacção PAM-Cliente algo bastante complexo. Assim, decidimos implementar um módulo python de comunicação com os módulos PAM descritos acima, que é responsável pela conversação entre ambos.

Idealmente a PAM seria composta por um único serviço em que a pilha de módulos seria composta pelos dois módulos descritos acima, ambos suficientes para autenticar um utilizador no sistema, sendo que no topo da pilha encontrar-se-ia o módulo de autenticação por smartcard. No entanto, devido a um problema que para nós foi impossível resolver, a conversação com segundo módulo da pilha não funcionava correctamente. Assim, para contornar o problema, decidimos criar dois serviços, um para autenticação por smartcard e outro por password sendo o servidor responsável por decidir qual invocar.

3.5 SQL Injection

Uma vez que estamos a usar a framework CherryPy, não temos problemas de SQL Injection no nosso trabalho. O CherryPy usa uma query parameterizável que impossibilita este tipo de ataques.

Fizemos vários testes na aplicação e podemos concluir isto mesmo.

Capítulo 4

4 Limitações

4.1 Fase 1

Na criação de uma SafeBox, o cliente deverá correr num directório que não possua nenhuma chave privada, pois a criação de uma PBox irá gerar uma chave privada nesse mesmo directório, podendo-se perder a chave original

O upload de um ficheiro, requer que o ficheiro esteja no directório onde a aplicação de cliente está a correr.

O download de um ficheiro não irá funcionar correctamente caso já exista um ficheiro com o mesmo nome nesse directório.

O controlo de integridade de ficheiros não obtém um resultado correcto no download de um ficheiro partilhado, pois a aplicação não obtém correctamente a chave HMAC relativa a esse ficheiro.

O retirar acesso de um ficheiro foi pensado com uma função recursiva que remove os sharings às pbox's que esta deu acesso e por aí diante recursivamente, no entanto, o algoritmo não funciona para todos os casos.

4.2 Fase 2

Limitações da segunda fase da SafeBox.

A maior parte dos problemas que nos surgiram giraram em torno da integração de autenticação por PAM na nossa aplicação. Sendo o nosso projecto implementado em python e os nossos módulos PAM em C, a comunicação entre os dois revelou-se uma tarefa algo complexa com já foi descrito acima. A solução que obtivemos não é perfeita, pois idealmente a autenticação seria exclusiva responsabilidade da PAM e na nossa implementação existem dependências PAM-Servidor. No entanto achamos que o resultado final mostra termos assimilado qual o papel e a importância da utilização deste tipo de serviço.

Também tivemos alguma dificuldade ao lidar com o cartão de cidadão, devido à única API de comunicação com smartcards para python (PyKCS11) ter uma documentação pouco extensa, sendo formada apenas por alguns exemplos implementados pelos autores.