universidade de aveiro
Departamento de Eletrónica Telecomunicações e Informática

Segurança 2015/2016

IEDCS : A DIGITAL RIGHTS MANAGEMENT PLATFORM

André Ribeiro Lopes nº 67833
Luis Félix nº 67558

1. ESTRATÉGIA

Este projecto pode ser pensado como dividido em três módulos: um cliente, um servidor e uma API REST, para comunicação entre os dois. Tudo foi desenvolvido na linguagem de programação Java, com recurso a diversas frameworks.

Para o cliente, a interacção do utilizador com a loja é feita através da linha de comandos usando Cliche para implementação da Shell interactiva e utilizámos a library Epublib para visualização dos e-books.

Para o deployment do projecto usou-se Tomcat, com uma base de dados em mySQL, para a qual usámos ainda Hibernate, que providencia uma camada de abstracção sobre a base de dados e facilita a integração com o código Java.

Quanto à API, inicialmente pensámos em usar a framework Spring, no entanto concluímos que é uma framework demasiado pesada e trabalhosa para o que necessitávamos, pelo que optámos então por Jersey.

2. ESTRUTURA DO PROJECTO

O nosso código está dividido em dois projectos.

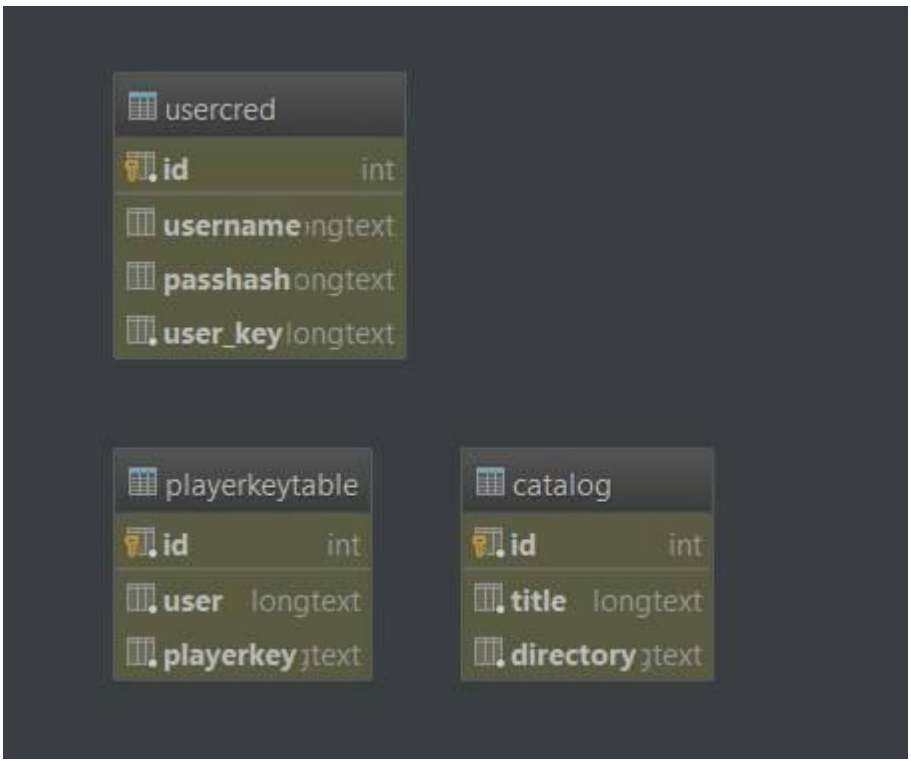
No projecto IEDCS-Jersey encontra-se a implementação da API.

No projecto security2015-p3g9, encontra-se a implementação do cliente, na package com.iedcs.player e todas as classes e métodos implementados para o âmbito de segurança estão na package com.iedcs.security. O Player em si está na package player e está implementado no ficheiro IEDCS_Player.java. Quase todas as classes, principalmente as de criptografia, têm um *main* para testes individuais.

2.1. BASE DE DADOS

A estrutura da base de dados manteve-se praticamente inalterada desde a apresentação inical. Na tabela playerketable são guardadas todas as player keys registadas no sistema. Esta tabela é usada na autenticação do Player no arranque. A tabela catalog mantém o título de todos os e-books que o sistema guarda, assim como o nome do ficheiro epub que o contém.

Cada utilizador eêm a si associado um username e password, assim como uma user_key.



3. FUNCIONALIDADES E TECNOLOGIAS

3.1. INICIALIZAÇÃO

```
choose working directory
C:\Users\Andre\Documents\ebooks\
```

Antes de usar o sistema é necessário especificar uma directoria. É nela que vai ser usada para todas as operações do programa e onde devem estar os ebooks em formato .epub.

```
"C:\Program ...

Sending 'GET' request to URL : http://localhost:8080/rest/hello/sendPK
Response Code : 200
rO0ABXNyABRqYXZhLnNlY3VyaXR5LktleVJlcL35T7OImqVDAgAETAAJYWxnb3JpdGhtdAASTGp

Sending 'POST' request to URL : http://localhost:8080/rest/hello/validate
Post parameters : xGxylay44YQVQlnYRWENRKMmT/QIh9sFIa+msiiLK3hhceAs5oqphcHI+
Response Code : 200
Client validated

IEDCS>
```

É usada criptografia assimétrica para validar o cliente:
1 - O cliente pede ao servidor a sua chave pública;
2 - O cliente, no arranque do Player envia a sua player key (hardcoded nesta primeira fase do projecto) para o servidor cifrada com a chave pública obtida;
3 - No servidor, é verificado se a player key enviada corresponde a uma player key guardada na tabela de player keys registadas na BD.
4- É enviada a resposta. Se a validação for concluída, o player é autorizado a continuar a sua operação.

3.2. LISTAR CATÁLOGO DO SERVIDOR

```
IEDCS> catalog
*****************************************************
**********AVAILABLE CATALOG ON IEDCS SERVER**********
*****************************************************
PrideAndPrejudice
AliceInWonderland
1984
GreatExpectations
OriginOfSpecies
```

Antes de comprar, o utilizador pode listar o catálogo disponível através do comando `catalog`.

3.3. COMPRAR UM E-BOOK

```
IEDCS> download AliceInWonderland
Sending 'GET' request to URL : http://localhost:8080/rest/hello/carroll-alice-in-wonderland-illustrations.epub
Response Code : 200
Downloading: carroll-alice-in-wonderland-illustrations.epub
```

Quando se compra um e-book é requisitado ao servidor que o cifre com a a File Key, esta é depois cifrada com três chaves (Device, User e Player key) . O resultado dessa cifra é escrito no header do e-book cifrado, formando o cryptoheader, e enviado para o cliente.
A device key é formada por um digest do MAC address + Serial number do HDD.

3.4. LISTAR E-BOOKS COMPRADOS

```
IEDCS> show
*******************************************************
****************** DOWNLOADED EBOOKS*******************
*******************************************************
1984
AliceInWonderland
GreatExpectations
IEDCS>
```

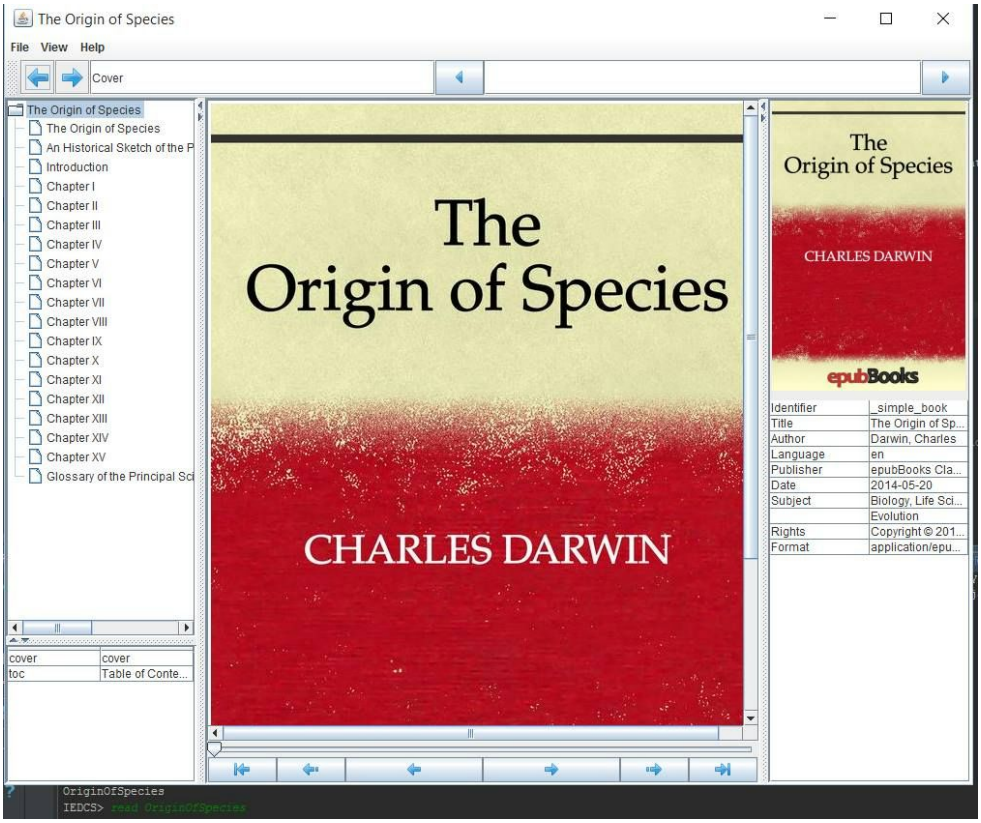
Para listar os livros já comprados, temos o comando `show`.

3.5 LISTA DE COMANDOS POSSÍVEIS

```
IEDCS> ?list
abbrev  name    params
    exit    ()
a   add (p1, p2)
r   read    (p1)
h   hello   ()
s   show    ()
c   catalog ()
d   download    (p1)
IEDCS>
```

Todos os comandos disponíveis podem ser consultados usando o comando `?list`.

3.6. VISUALIZAÇÃO



Depois de comprado, o utilizador pode, através do comando `read <nome do livro>` ler o e-book no visualizador.
Neste processo, o player retira o header do e-book cifrado descarregado, decifra-o com a player key, envia-o para o servidor, no servidor é decifrado com a user key, é enviado de volta para o Player onde é decifrado com a device key. O resultado desta tripla decifra é a File Key que é usada para decifrar o e-book (já com o cryptoheader removido).

PROTECÇÃO CONTRA ATAQUES CONHECIDOS

A interacção com a BD não é dependente de input do utilizador. Isto é, nada do que o utilizador escreve na interacção com o sistema é usado directamente numa query à BD, pelo que se assegura a protecção contra SQL injection.
Além disto, devido a ser usada uma linguagem de alto nível e com gestão de memória, na maior parte do tempo, automatizada, é assegurada a protecção contra ataques do tipo Buffer Overflow.

PROBLEMAS CONHECIDOS/DEFICIÊNCIAS

A cifra simétrica é feita em modo ECB, um modo que tem algumas falhas conhecidas. Neste modo a criptanálise é facilitada pelo facto de ser possível reconhecer padrões no criptograma.
Uma possível melhoria seria usar um outro modo como CBC, alimentado por um IV aleatório.
Algum material criptográfico, como a Player Key, está hardcoded. Idealmente, existiria um distribuidor de Players que atribui a cada novo player uma Player Key distincta.
Não existe ainda uma PKI implementada apesar de o código que gera certificados e o associa a chaves públicas estar funcional.
Apesar de não conseguirmos os riscos e consequências de um ataque XSS neste sistema em particular , a existência de uma vulnerabilidade deste tipo não está fora de questão em particular num GET request usado para fazer download de um e-book, dado que neste GET entra um parâmetro num URL, que é dado pelo utilizador: trata-se do título do livro a fazer download. É inteiramente possível um utilizador usar esta entrada para introduzir código malicioso.
Todas as outras routes estão protegidas devido ao próprio Jersey que espera URLs num formato bastante específico.
Para combater esta vulnerabilidade deverá ser implementado um filtro que higieniza este parâmetro de entrada.
Deveria também ser melhor delineada a fronteira client/servidor, usar directorias diferentes para as operações de cliente e servidor e numa fase mais avançada, construir VMs, uma com o cliente , outra com o servidor configuradas de forma a estarem na mesma rede de modo a, com recurso de algo como o Wireshark, analisar a troca de pacotes entre cliente-servidor.
Ao momento da entrega, só conseguimos obter o número de série do HDD em Windows.
Numa nota não tanto ligada a Segurança em si, o catálogo dos ebooks comprados está a ser guardado em memória. Idealmente, estaria guardado num ficheiro de modo a, ao reiniciar do Player, ainda ter acesso a todos os ebooks descarregados.

RECURSOS

http://javaingrab.blogspot.pt/2014/04/aes-256bits-encryption-and-decryption.html
http://www.java2s.com/Code/Java/Security/WrapAndUnwrapKey.htm
http://www.mkyong.com/java/how-to-get-mac-address-in-java/
http://www.java2s.com/Code/Java/Security/SimpleRSApublickeyencryptionalgorithmimplementation.htm
http://jexp.ru/index.php/Java_Tutorial/Security/X509Certificate
https://github.com/pcarrier/identify/blob/master/src/main/java/identify/X509CertificateFactoryImpl.java
http://www.bouncycastle.org/wiki/display/JA1/BC+Version+2+APIs
http://www.informit.com/articles/article.aspx?p=170967&seqNum=4
https://crackstation.net/hashing-security.htm
http://stackoverflow.com/questions/5482947/how-to-get-hard-disk-serial-number-using-java