Instituto Superior de Engenharia de Lisboa

Segurança Informática

Trabalho 2

João Bonacho Nº A49437

André Gonçalves Nº A49464

Ana Carolina Pereira Nº A49470

LEIC51D Grupo 07

Semestre de Inverno 2023/2024

14 de Dezembro de 2023

Parte 1

Exercício 1

a)

A propriedade ***perfect forward secrecy*** não é garantida usando o processo base com RSA para estabelecimento do ***master_secret*** porque se a chave privada do servidor for comprometida, as mensagens anteriores trocadas entre o cliente e o servidor podem ser decifradas. Garantido esta propriedade garante-se que se a chave for comprometida, não compromete as mensagens anteriores.

b)

Dois possíveis ataques ao *record protocol* são ataques do tipo ***POODLE*** e do tipo ***man-in-the-middle***.

Ataques do tipo ***POODLE*** baseiam-se em aproveitar uma nova tentativa de estabelecimento de ligação, quando a primeira tentativa de estabelecimento de ligação falha, com protocolos menos seguros (mais antigos) em que alguns deles já são mais facilmente atacados. Uma forma de prevenir é não descer até aos protocolos mais antigos, como por exemplo, o SSL.

Ataques do tipo ***man-in-the-middle*** permitem que um atacante seja um intermediário entre cliente e servidor, estabelecendo a ligação TLS e observando as mensagens sem o quebrar. Para o prevenir são utilizados certificados para validar o servidor (e, se possível, o cliente).

Exercício 2

A técnica ***CAPTCHA*** mitiga os ataques de dicionário, se os ataques forem à **interface de autenticação**, por dificultar aqueles que são de caráter *"brute-force"*. Apesar de não resultar em palavras-passe mais seguras, o ***CAPTCHA*** pode detetar, com um determinado nível de certeza, que aquele que está a introduzir os dados não é um humano, fazendo com que estes tipos de ataques sejam feitos mais lentamente e de forma menos eficiente.

Tendo em conta que os ataques referidos são à **informação de validação**, assume-se que a base de dados foi comprometida então a técnica ***CAPTCHA*** não mitiga este tipo de ataques.

Exercício 3

A aplicação servidor pode detetar se o conteúdo do *cookie* foi adulterado no *browser* através da marca (MAC) presente no *web token* (JWT) gerado com a identificação do utilizador. O servidor ao verificar a autenticidade/integridade do *web token* (JWT) através da marca detetará a possível adulteração.

Exercício 4

a)

A estrutura JWT é gerada e assinada pelo fornecedor de identidade com as informações do utilizador autenticado enviado para a aplicação cliente (*relying party*). Esta estrutura é enviada na resposta ao pedido ao ***token endpoint*** com o ***code*** gerado no ***authorization endpoint*** juntamente com a autenticação do *relying party* (***client_id*** e ***client_secret***).

b)

As ações da aplicação cliente para conseguir fazer os pedidos ao servidor de recursos, após o dono de recursos ter autorizado e consentido o uso de um recurso, implica a adição do ***access_token*** nos pedidos ao servidor de recursos. Este ***access_token*** é obtido através do pedido ao ***token endpoint***, como mencionado na questão 4a).

Exercício 5

Em primeiro lugar, através das relações ***user assigment*** (UA) verifica-se que o u4 é um ***(S)upervisor***.

Em segundo lugar, através das relações ***role hierarchy*** (RH) verifica-se que o ***(S)upervisor*** tem as permissões de um ***(T)ester***, ***(D)eveloper*** e ***(M)ember***.

Por último, através das relações ***permission assigment*** (PA) conclui-se que o u4 tem as permissões p1, p2 e p3.

Exercício 6

a)

1. Como as chaves e certificados no servidor são configuradas usando o formato PEM, tem que se converter ficheiros *.cer* e *.pfx* para *.pem* utilizando a ferramenta de linha de comandos *OpenSSL*.

```
C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\certificates-keys\end-entities>openssl x509 -inform der -in secure-server.cer -out secure-server.pem

C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\certificates-keys\end-entities>cd ../pfx

C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\certificates-keys\pfx>openssl pkcs12 -in secure-server.pfx -out secure-server.pem -nodes
Enter Import Password:

C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\certificates-keys\pfx>
```

Figura 1 - Conversão de ficheiros .cer e .pfx para .pem

2. Para que o endereço *www.secure-server.edu* seja resolvido para *localhost*, fez-se a configuração no ficheiro *hosts* do sistema operativo.

```
# SegInf trab2
127.0.0.1 www.secure-server.edu
```

Figura 2 – Resolução DNS para localhost

3.

3.1. Autenticação do servidor sem autenticação do cliente

Servidor:

1 – Configurou-se a chave privada (***secure-server-key.pem***) do servidor.

2 – Configurou-se o certificado (***secure-server.pem***), proveniente da concatenação do certificado do servidor (***secure-server-cer.pem***) e do certificado intermédio da cadeia de certificados (***CA1-int.pem***).

Cliente:

Adicionou-se a raiz de confiança da entidade fornecedora de certificados (***CA1.cer***) no sistema operativo (no caso de estudo, ***Google Chrome***) e no *browser* (no caso de estudo, ***FireFox***).

3.2. Autenticação do servidor com autenticação do cliente

Servidor:

Adicionou-se a raiz de confiança da entidade fornecedora de certificados (***CA2.pem***).

Cliente:

Adicionou-se a chave privada do utilizador (no caso de estudo, ***Alice_2.pfx***) e o certificado intermédio da cadeia (no caso de estudo, ***CA2-int.cer***) ao *browser* (no caso de estudo, ***FireFox***).

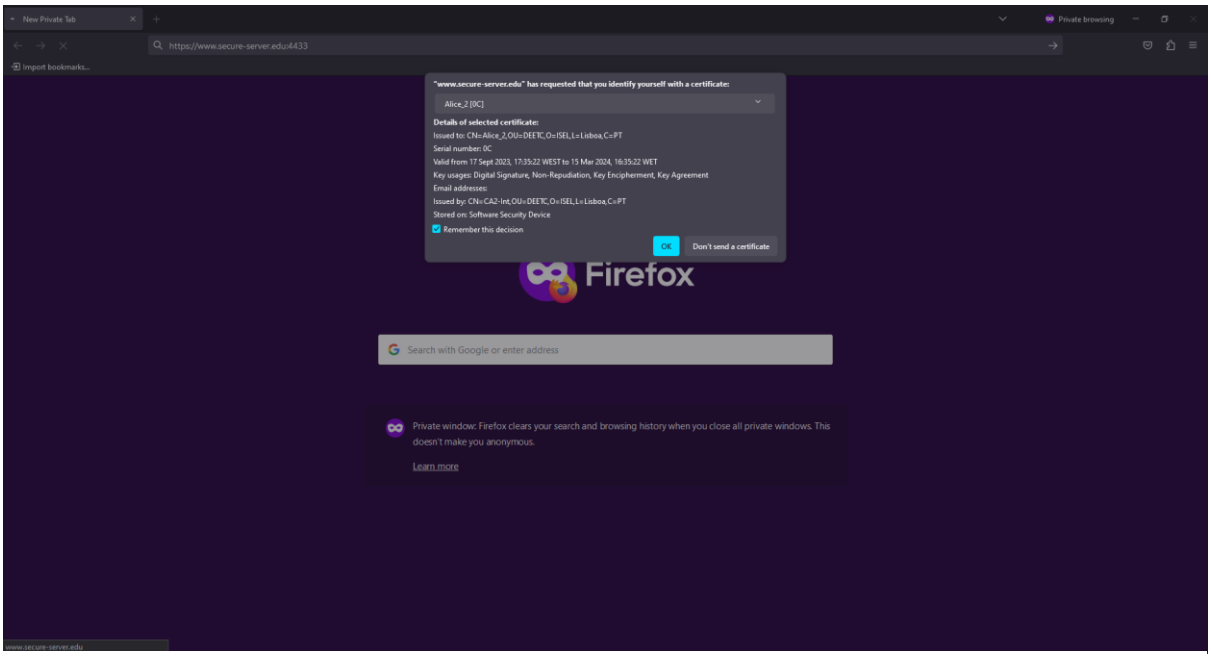


Figura 3 – Autenticação do cliente solicitada pelo servidor

NOTA: O código fonte encontra-se no ficheiro ***https-server.js,*** em anexo na diretoria *code/ex6/HTTPS-server*.

b)

Como a raiz de confiança na aplicação necessita de ser configurada usando o formato JKS, realizou-se a conversão usando a ferramenta de linha de comandos *keytool* proveniente do JDK.

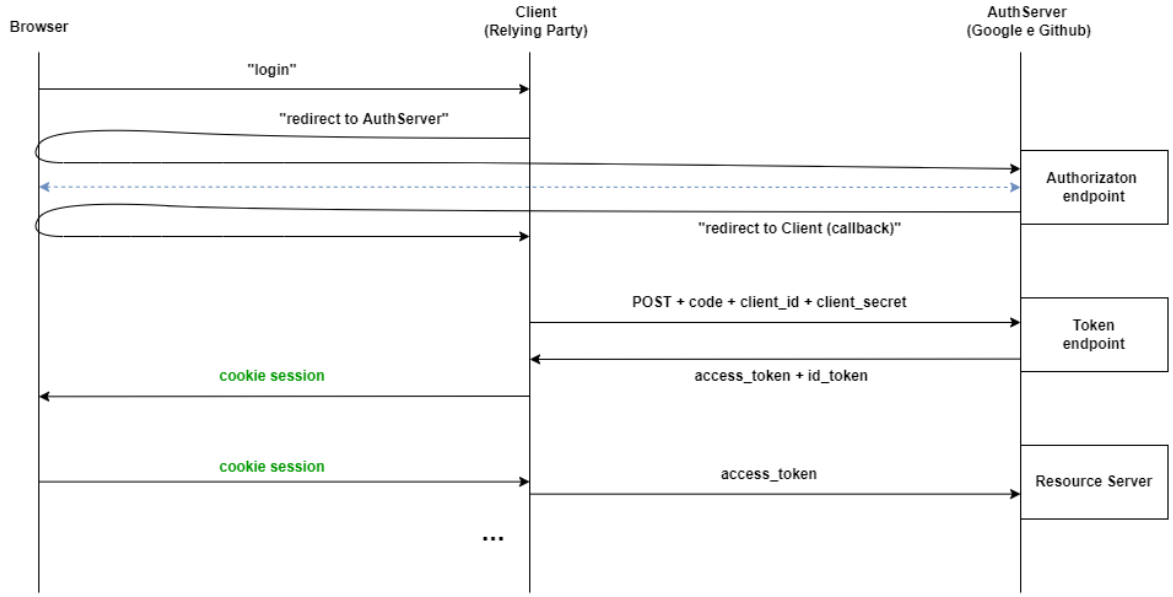
```
C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\App\src>keytool -importcert -keystore CA1.jks -file CA1.cer -alias default -storepass changeit
Owner: CN=CA1, OU=DEETC, O=ISEL, L=Lisboa, C=PT
Issuer: CN=CA1, OU=DEETC, O=ISEL, L=Lisboa, C=PT
Serial number: 5a
Valid from: Sun Sep 17 17:35:20 WEST 2023 until: Fri Mar 15 16:35:20 WET 2024
Certificate fingerprints:
        SHA1: C7:FB:AD:FE:BF:52:92:33:09:2C:FE:3F:5F:D9:1A:1D:92:C6:2E:C4
        SHA256: A4:78:99:60:DB:47:BA:97:2A:EC:77:A8:FD:DE:BF:A6:B3:A8:50:33:83:27:91:F7:D6:A7:89:2C:4A:0A:C1:1C
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 73 03 DF B8 EA 47 83 87   D4 1D 54 9B 8A 9B 3B 46  s....G....T...;F
0010: B2 9E A1 62                                        ...b
]
[CN=CA1, OU=DEETC, O=ISEL, L=Lisboa, C=PT]
SerialNumber: [    5a]
]

#2: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen: no limit
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 73 03 DF B8 EA 47 83 87   D4 1D 54 9B 8A 9B 3B 46  s....G....T...;F
0010: B2 9E A1 62                                        ...b
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore

C:\Users\guilh\Documents\ISEL\5s\SegInf\code\SegInf-2324-G07\trab2\ex6\App\src>
```

Figura 4 - Conversão do certificado para .jks

NOTA: O código fonte encontra-se no ficheiro ***App.java,*** em anexo na diretoria *code/ex6/App*.

Exercício 7

Fluxo do tipo *authorization code*



Política de controlo de acesso

Papéis: { (F)ree, (P)remium, (A)dmin }

Recurso: Google Tasks

- U = { emailA, emailB, emailC }
- RH = { F ⪯ P, P ⪯ A }
- UA = { (emailA, F), (emailB, P), (emailC, A) }
- PA = { (F, read), (P, write) }

NOTA: O código fonte encontra-se no ficheiro ***relying_party.js*** e as políticas de acesso nos ficheiros ***app_model.conf*** e ***app_policy.csv,*** em anexo na diretoria *code/ex7*.