UNIVERSIDADE DE AVEIRO | Departamento de Eletrónica, Telecomunicações e Informática

Segurança Informática e nas Organizações – 1º Semestre 2019-20

Projeto 3: Autenticação

Curso:

Licenciatura em Engenharia Informática

Data:

16 de Dezembro de 2019

Docentes:

Professor João Barraca
Professor Vítor Cunha

Discentes:

João Magalhães - 79923
João Ferreira - 80041

Índice

Introdução

Este trabalho foi desenvolvido no âmbito da Unidade Curricular Segurança Informática e nas Organizações e visa explorar os conceitos relacionados com o estabelecimento de uma sessão segura entre dois interlocutores, bem como a autenticação dos intervenientes na comunicação e respectivo controlo de acesso.

No desenvolvimento deste trabalho iremos abordar temas como: autenticação, controlo de acesso, leitura de smartcards, *hash* de passwords a serem guardadas para uso futuro, entre outros.

Para a realização deste projeto utilizamos o código base fornecido pelo professor para o Projecto 2 da mesma UC.

Autenticação - Desafio Resposta

A autenticação desafio resposta é um mecanismo que é utilizado para a autenticação de utentes. Em que é apresentada uma pergunta/desafio ao utente e este deve fornecer uma resposta válida para conseguir realizar a autenticação com sucesso.

O nosso planeamento da autenticação por desafio resposta consiste nos seguintes passos:

1. Cliente irá aceder ao servidor
2. É lançado o desafio pelo servidor, neste caso, é gerado um NONCE.
3. O cliente pode resolver o desafio, usando o mecanismo de password ou o cartão de cidadão sendo o resultado enviado posteriormente
4. O servidor depois de receber o resultado valida o mesmo.

   Se for o desafio da password, esta é calculada da mesma maneira que foi calculada para o cliente.

   Se for o cartão de cidadão é usada a chave pública do cliente que é dada pelo cartão de cidadão

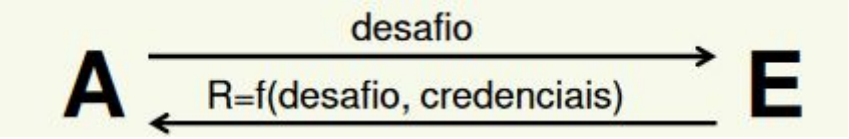


Figura 1- Exemplificação do desafio resposta

Este mecanismo tem vantagens e desvantagens:

Vantagens:

- Credenciais não são expostas:
  - Nunca circulam no canal de comunicação, de forma explícita;
  - É calculada uma hash;
- Robustas contra ataques MITM:
  - O atacante, mesmo conseguindo interceptar qual o desafio e a resposta deste, não consegue replicar a transformação;
- Compatíveis com outras aproximações:
  - Dispositivos físicos, chaves simétricas, chaves assimétricas;
- Autenticador escolhe transformação e complexidade do desafio.

Desvantagens:

- Clientes necessitam de um método para calcular respostas aos desafios;
- Autenticador pode necessitar de armazenar segredos em claro;
- Pode ser possível calcular todas as respostas possíveis;

- Obriga que o autenticador faça uma boa gestão de NONCEs, pois não podem ser reutilizados.

Um NONCE é um número arbitrário que só pode ser usado uma vez. É um número aleatório. As credenciais não são expostas, uma vez que circulam apenas as transformações das credenciais usadas e estas são sempre diferentes assim o atacante mesmo que consiga o desafio e o resultado não consegue replicar a transformação.

Autenticação - Cartão de Cidadão

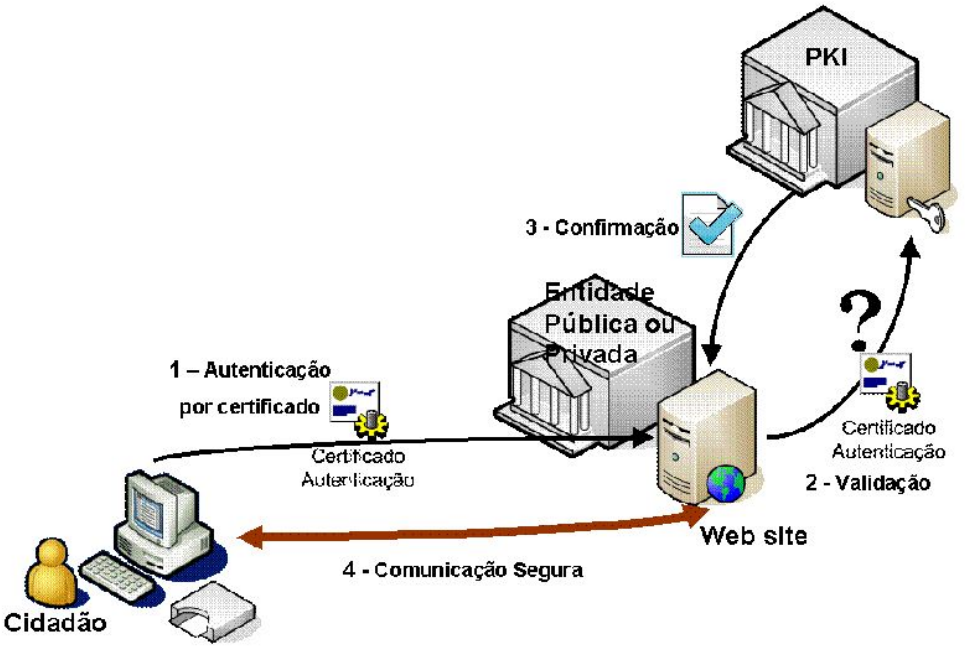
O processo de validação da identidade do cliente baseado na utilização do Cartão de Cidadão garante a veracidade dos dados do Cartão, incluindo o certificado digital produzido, do cliente em questão.

O certificado digital de autenticação produzido terá que ser alvo de uma verificação para averiguarmos se este se encontra válido.

No entanto, é a Entidade Raiz que, após verificação no seu Sistema de Informação, autentica o cliente e lhe concede os privilégios de acesso aos serviços electrónicos disponíveis.

Este tipo de autenticação tem como principal vantagem a segurança, visto que só poderá ser utilizado por um cidadão que tenha Cartão de Cidadão, que conheça o PIN de acesso ao certificado e que possua um certificado válido; por outro lado, a entidade que fornece o serviço electrónico, deverá pedir a validação do mesmo a uma entidade externa, responsável pela emissão e gestão dos certificados do Cartão de Cidadão, de modo a garantir que o certificado que o cidadão apresentou ainda se encontra válido. Só a partir desse momento o cidadão é autenticado pela Entidade, estabelecendo-se uma comunicação segura entre ambos.

Para este mecanismo de autenticação temos de obter os certificados que estão online em https://pki.cartaodecidadao.pt/, para a construção da cadeia de certificação.



Utilização de certificado de autorização

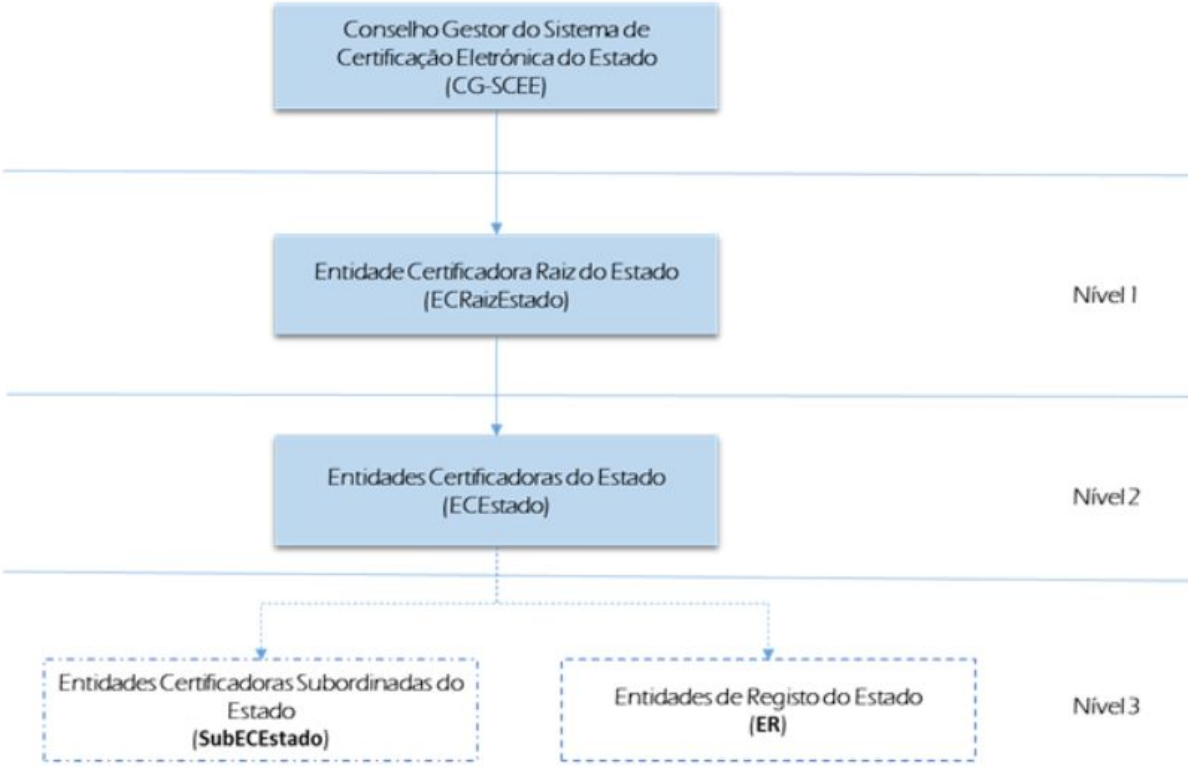


Figura 3 - Arquitetura funcional do SCEE

Para o efeito o SCEE compreende:

- Um Conselho Gestor que aprova a integração ou cessação de entidades certificadoras no SCEE, pronunciando-se igualmente sobre práticas e políticas de certificação;
- Uma Entidade Certificadora Raiz do Estado (ECRaizEstado), que constitui o primeiro nível e o topo da cadeia hierárquica de certificação;
- As entidades certificadoras do Estado (ECEstado),subordinadas diretamente à entidade raiz;
- As entidades certificadoras subordinadas do Estado (SubECEstado), subordinadas diretamente ou em múltiplos níveis de uma das ECEstado;
- As entidades de registo associadas a cada uma das entidades certificadoras.

Estados Adicionados

- Cliente
  - STATE_PICK_AUTHENTICATION → Escolher método de autenticação
  - STATE_AUTHENTICATION_SERVER → Autenticação do servidor
  - STATE_AUTHENTICATION → Autenticação do cliente

- Servidor
  - STATE_PICK_AUTHENTICATION → Receber/Definir método de autenticação
  - STATE_AUTHENTICATION_SERVER → Autenticação perante o cliente
  - STATE_AUTHENTICATION → Autenticar o cliente
  - STATE_CHALLENGE → Realiza a operação de autenticação e verifica as permissões

Implementação

Primeiro de tudo a autenticidade do servidor tem de ser validada, posteriormente o cliente terá que informar ao servidor que tipo de autenticação deseja realizar. O servidor para este efeito gera um número aleatório com 64 bits. O cliente recebe o desafio, sendo que este terá de o resolver e devolver o seu resultado. No final o resultado enviado pelo cliente é verificado para percebermos se este é válido ou não. Se deste processo resultar qualquer falha a comunicação é terminada pois é considerada insegura.

Implementação Cartão de Cidadão

Para autenticação com o cartão de cidadão, começamos por carregar (*load)* o seu certificado (cert). Posteriormente, o nonce é gerado,utilizando a sua chave privada. Finalmente, é enviada uma mensagem para o servidor com o certificado e assinatura.

Quando o servidor recebe a mensagem enviada pelo cliente, valida quer o certificado, quer a assinatura.

```
def cc(self, message: str) -> bool:
    cert_cc = base64.b64decode(message['certificate'])
    cert = x509.load_pem_x509_certificate(cert_cc, default_backend())
    self.cert = cert
    verify_cert = self.check_certificate(cert)

    signature = base64.b64decode(message['signature'])
    verify_sign = self.check_signature(signature, cert)

    if not verify_cert or not verify_sign:
        return False
    logger.info('Certificate chain and Signature verified')
    return True
```

Para a validação do certificado é necessário efectuar os seguintes passos:

1. Verificar se o certificado tem o propósito de validar assinaturas digitais através (campo digital signature da KEY_USAGE, atributo do certificado).
2. Definir dois dicionários onde guardamos os certificados (root e intermediários).
3. Definir a cadeia de certificação.
4. Percorrendo a cadeia de certificação devemos verificar o parâmetro key_cert_sign (KEY_USAGE), verificar se cada um dos certificados não se encontra na CRL, verificar se estes se encontram todos válidos na dimensão temporal e validade das respectivas assinaturas.

Menu de escolha de autenticação é definido da seguinte forma:

```
def pick_authentication_method(self) -> None:

    print("Authentication method: ")
    print("1 - Password Authentication")
    print("2 - Citizen card Authentication")
    while self.type_auth == None:
        option = input("Option: ")
        if option == "1":
            self.type_auth = "password"
        elif option == "2":
            self.type_auth = "cc"
        else:
            print("Invalid option\n")

    message = {'type': 'AUTHENTICATION', 'data': None}

    if self.type_auth == "password":
        username = input("Username: ")
        message['data'] = base64.b64encode((self.type_auth + username).encode()).decode()
    else:
        message['data'] = base64.b64encode(self.type_auth.encode()).decode()
    self._send(message)
```

Autenticação do Servidor (x509)

Tal como o cliente, o servidor também deve ser autenticado. Para este efeito, emitimos dois certificados x509, um pertencente ao servidor com a sua chave privada, e um que assina o certificado do servidor, sendo considerado uma Entidade de Certificação (CA). Para a autenticação do servidor é seguido o mesmo método de desafio-resposta utilizando o cartão de cidadão. O cliente envia para o servidor um nonce e o servidor assina-o com a sua chave privada, devolvendo esta assinatura em conjunto com o respectivo certificado. O cliente valida os mesmos campos que são validados quando é feita a autenticação com cartão de cidadão.

Autenticação-Passwords

Cada password antes de ser registada é transformada através do SHA256, que é um algoritmo de síntese, e um salt para impedir que passwords iguais criem a mesma síntese, assim criamos uma transformação impossível de replicar.

O servidor envia o NONCE e o salt ao utilizador. O utilizador como é o dono da password só tem de replicar o processo da password no servidor e assim realiza uma segunda operação de síntese usando o NONCE. Depois o utilizador tem que enviar o resultado ao servidor.

O servidor quando recebe o desafio já resolvido por parte do utilizador, replica o processo todo e verifica se os resultados são iguais.

Se os resultados forem iguais o utilizador é autenticado com sucesso. Para o envio de algum ficheiro do utilizador para o servidor é necessário que o cliente em questão tenha as permissões para tal, logo não basta estar autenticado.

Controlo de acesso

Definimos os clientes no ficheiro users.py, recorrendo ao uso de listas com os quatro atributos que os caracterizam: nome, password, número de cartão de cidadão e permissão. Estes atributos são separados por vírgulas.

A cada posição do array corresponde a mesma posição nos diferentes arrays. Por exemplo, tendo as listas :

nomes = [“joao”,”francisco”]
passwords = [“pw1”,”pw2”]
n_cc = [“1”,”2”]
permissoes = [True,False]

ao utilizador “joao” corresponde a password “pw1”, número de cartão de cidadão “1” e tem permissão para enviar ficheiros (True).

Após o cliente ser autenticado são verificadas as suas permissões para enviar o ficheiro (True ou False).

No caso da autenticação por desafio-resposta (username-password), percorremos o ficheiro e, no caso da permissão ser “True”, então é permitido o envio de ficheiros ao cliente.

No caso da autenticação por cartão de cidadão, obtemos o número do cartão, através do certificado associado. Percorremos os utilizadores até encontrarmos um com um número igual ao fornecido e, em caso afirmativo, verificamos se o utilizador está autorizado ou não.

Para o teste desta funcionalidade utilizamos um dos nossos números de cartão de cidadão (15130620) combinando as duas possibilidades de autenticação (True ou False). De modo a alterar as permissões estabelecidas basta apenas editar o ficheiro de texto users_db.txt.

Outputs

1. Autenticação por password, em que o cliente tem as permissões para transferir ficheiros.

client.py

```
(venv) user@vm:~/project-3-auth$ python client.py teste.txt
2019-12-16 06:30:25 vm root[10363] INFO Sending file: /home/user/project-3-auth/teste.txt to 127.0.0.1:5000 LogLevel: 20
2019-12-16 06:30:25 vm root[10363] INFO Channel open
2019-12-16 06:30:25 vm root[10363] INFO Nonce sended
2019-12-16 06:30:25 vm root[10363] INFO Authentication done.
Authentication method:
1 - Password Authentication
2 - Citizen card Authentication
Option: 1
Username: joao
2019-12-16 06:30:39 vm root[10363] INFO Type Authentication chosen.
Password:
2019-12-16 06:30:41 vm root[10363] INFO Challenge finished.
2019-12-16 06:30:41 vm root[10363] WARNING Ignoring message from server
```

servidor.py

```
2019-12-16 06:30:23 vm a10-tcpsever[10360] INFO Starting worker [10360]
2019-12-16 06:30:25 vm root[10360] INFO

Connection from ('127.0.0.1', 41488)
2019-12-16 06:30:25 vm root[10360] INFO File open
2019-12-16 06:30:25 vm root[10360] INFO Certificate sended
2019-12-16 06:30:25 vm root[10360] INFO Client validate server
2019-12-16 06:30:39 vm root[10360] INFO Nonce and salt sended
2019-12-16 06:30:41 vm root[10360] INFO Successfully authenticated
2019-12-16 06:30:41 vm root[10360] INFO Authentication finished and permissions verified
```

2.. Autenticação por password, em que o cliente nao tem as permissões para transferir ficheiros.

client.py

```
(venv) user@vm:~/project-3-auth$ python client.py teste.txt
2019-12-16 06:33:27 vm root[10385] INFO Sending file: /home/user/project-3-auth/teste.txt to 127.0.0.1:5000 LogLevel: 20
2019-12-16 06:33:27 vm root[10385] INFO Channel open
2019-12-16 06:33:27 vm root[10385] INFO Nonce sended
2019-12-16 06:33:27 vm root[10385] INFO Authentication done.
Authentication method:
1 - Password Authentication
2 - Citizen card Authentication
Option: 1
Username: guilherme
2019-12-16 06:33:35 vm root[10385] INFO Type Authentication chosen.
Password:
2019-12-16 06:33:37 vm root[10385] INFO Challenge finished.
2019-12-16 06:33:37 vm root[10385] WARNING Got error from server: Permission denied
(venv) user@vm:~/project-3-auth$
```

server.py

```
(venv) user@vm:~/project-3-auth$ python server.py
2019-12-16 06:33:24 vm root[10381] INFO Port: 5000 LogLevel: 20 Storage: /home/user/project-3-auth/files
[2019-12-16 06:33:24 +0000] [10381] [INFO] Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
2019-12-16 06:33:24 vm aio-tcpsever[10381] INFO Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
[2019-12-16 06:33:24 +0000] [10381] [INFO] Starting worker [10381]
2019-12-16 06:33:24 vm aio-tcpsever[10381] INFO Starting worker [10381]
2019-12-16 06:33:27 vm root[10381] INFO

Connection from ('127.0.0.1', 41502)
2019-12-16 06:33:27 vm root[10381] INFO File open
2019-12-16 06:33:27 vm root[10381] INFO Certificate sended
2019-12-16 06:33:27 vm root[10381] INFO Client validate server
2019-12-16 06:33:35 vm root[10381] INFO Nonce and salt sended
2019-12-16 06:33:37 vm root[10381] ERROR Permission denied
NoneType: None
2019-12-16 06:33:37 vm root[10381] INFO Authentication finished and permissions verified
2019-12-16 06:33:37 vm root[10381] INFO Closing transport
```

3.Autenticação por Cartão de Cidadão em que o cliente tem permissões para transferir ficheiros.

client.py

```
(venv) user@vm:~/project-3-auth$ python client.py teste.txt
2019-12-16 06:35:19 vm root[10397] INFO Sending file: /home/user/project-3-auth/teste.txt to 127.0.0.1:5000 LogLevel: 20
2019-12-16 06:35:19 vm root[10397] INFO Channel open
2019-12-16 06:35:19 vm root[10397] INFO Nonce sended
2019-12-16 06:35:19 vm root[10397] INFO Authentication done.
Authentication method:
1 - Password Authentication
2 - Citizen card Authentication
Option: 2
2019-12-16 06:35:22 vm root[10397] INFO Type Authentication chosen.
QApplication: invalid style override passed, ignoring it.
2019-12-16 06:35:28 vm root[10397] INFO Challenge finished.
2019-12-16 06:35:28 vm root[10397] WARNING Ignoring message from server
```

server.py

```
(venv) user@vm:~/project-3-auth$ python server.py
2019-12-16 06:35:13 vm root[10392] INFO Port: 5000 LogLevel: 20 Storage: /home/user/project-3-auth/files
[2019-12-16 06:35:13 +0000] [10392] [INFO] Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
2019-12-16 06:35:13 vm aio-tcpsever[10392] INFO Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
[2019-12-16 06:35:13 +0000] [10392] [INFO] Starting worker [10392]
2019-12-16 06:35:13 vm aio-tcpsever[10392] INFO Starting worker [10392]
2019-12-16 06:35:19 vm root[10392] INFO

Connection from ('127.0.0.1', 41506)
2019-12-16 06:35:19 vm root[10392] INFO File open
2019-12-16 06:35:19 vm root[10392] INFO Certificate sended
2019-12-16 06:35:19 vm root[10392] INFO Client validate server
2019-12-16 06:35:22 vm root[10392] INFO Nonce and salt sended
2019-12-16 06:35:28 vm root[10392] INFO Certificate chain and Signature verified
2019-12-16 06:35:28 vm root[10392] INFO Successfully authenticated
2019-12-16 06:35:28 vm root[10392] INFO Authentication finished and permissions verified
```

4.Autenticação por Cartão de Cidadão em que o cliente não tem permissões para transferir ficheiros.

client.py

```
(venv) user@vm:~/project-3-auth$ python client.py teste.txt
2019-12-16 06:38:17 vm root[10456] INFO Sending file: /home/user/project-3-auth/teste.txt to 127.0.0.1:5000 LogLevel: 20
2019-12-16 06:38:17 vm root[10456] INFO Channel open
2019-12-16 06:38:17 vm root[10456] INFO Nonce sended
2019-12-16 06:38:17 vm root[10456] INFO Authentication done.
Authentication method:
1 - Password Authentication
2 - Citizen card Authentication
Option: 2
2019-12-16 06:38:20 vm root[10456] INFO Type Authentication chosen.
QApplication: invalid style override passed, ignoring it.
2019-12-16 06:38:25 vm root[10456] INFO Challenge finished.
2019-12-16 06:38:25 vm root[10456] WARNING Got error from server: Permission denied
(venv) user@vm:~/project-3-auth$
```

server.py

```
(venv) user@vm:~/project-3-auth$ python server.py
2019-12-16 06:38:12 vm root[10452] INFO Port: 5000 LogLevel: 20 Storage: /home/user/project-3-auth/files
[2019-12-16 06:38:12 +0000] [10452] [INFO] Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
2019-12-16 06:38:12 vm aio-tcpsever[10452] INFO Single tcp server starting @0.0.0.0:5000, Ctrl+C to exit
[2019-12-16 06:38:12 +0000] [10452] [INFO] Starting worker [10452]
2019-12-16 06:38:12 vm aio-tcpsever[10452] INFO Starting worker [10452]
2019-12-16 06:38:17 vm root[10452] INFO

Connection from ('127.0.0.1', 41516)
2019-12-16 06:38:17 vm root[10452] INFO File open
2019-12-16 06:38:17 vm root[10452] INFO Certificate sended
2019-12-16 06:38:17 vm root[10452] INFO Client validate server
2019-12-16 06:38:20 vm root[10452] INFO Nonce and salt sended
2019-12-16 06:38:25 vm root[10452] INFO Certificate chain and Signature verified
2019-12-16 06:38:25 vm root[10452] ERROR Permission denied
NoneType: None
2019-12-16 06:38:25 vm root[10452] INFO Authentication finished and permissions verified
2019-12-16 06:38:25 vm root[10452] INFO Closing transport
```

Execução final

Para executar o trabalho é necessário duas janelas/dois separadores (um para o cliente e outro para o servidor), encontrando-se as duas a executar o *virtual environment* fornecido.

Antes de executar o login (por cartão de cidadão ou por palavra-passe) é necessário executar o script python users.py para gerar os utilizadores. Inicialmente, fornecemos um utilizador com username “joao” e password “pass”.

Os utilizadores podem ser alterados nesse mesmo script, sendo que deve também ser alterado o número de cartão cidadão correspondente, uma vez este irá ser verificado posteriormente na autenticação por cartão de cidadão. Por exemplo, o cartão que utilizamos para testar é o cartão com o número 151306206, dando-lhe as permissões necessárias.

Assim, para a execução deste projecto devem ser executados os seguintes comandos:

1. source venv/bin/python (em ambas as janelas)
2. python server.py (numa janela)
3. python client.py <ficheiro> (na segunda janela)