universidade de aveiro deti departamento de electrónica, telecomunicações e informática

Authentication and Access Control

LEI: Segurança Informática e nas Organizações 2019-2020

89296 | Tomás Batista

88887 | Flávia Figueiredo

Índice

Introdução

Este trabalho consiste na implementação de um protocolo desenhado pelo grupo para a criação de um canal de comunicação seguro para a troca de um ficheiro de texto entre o cliente e o servidor com autenticação dos intervenientes na comunicação e o controlo de acesso aos clientes.

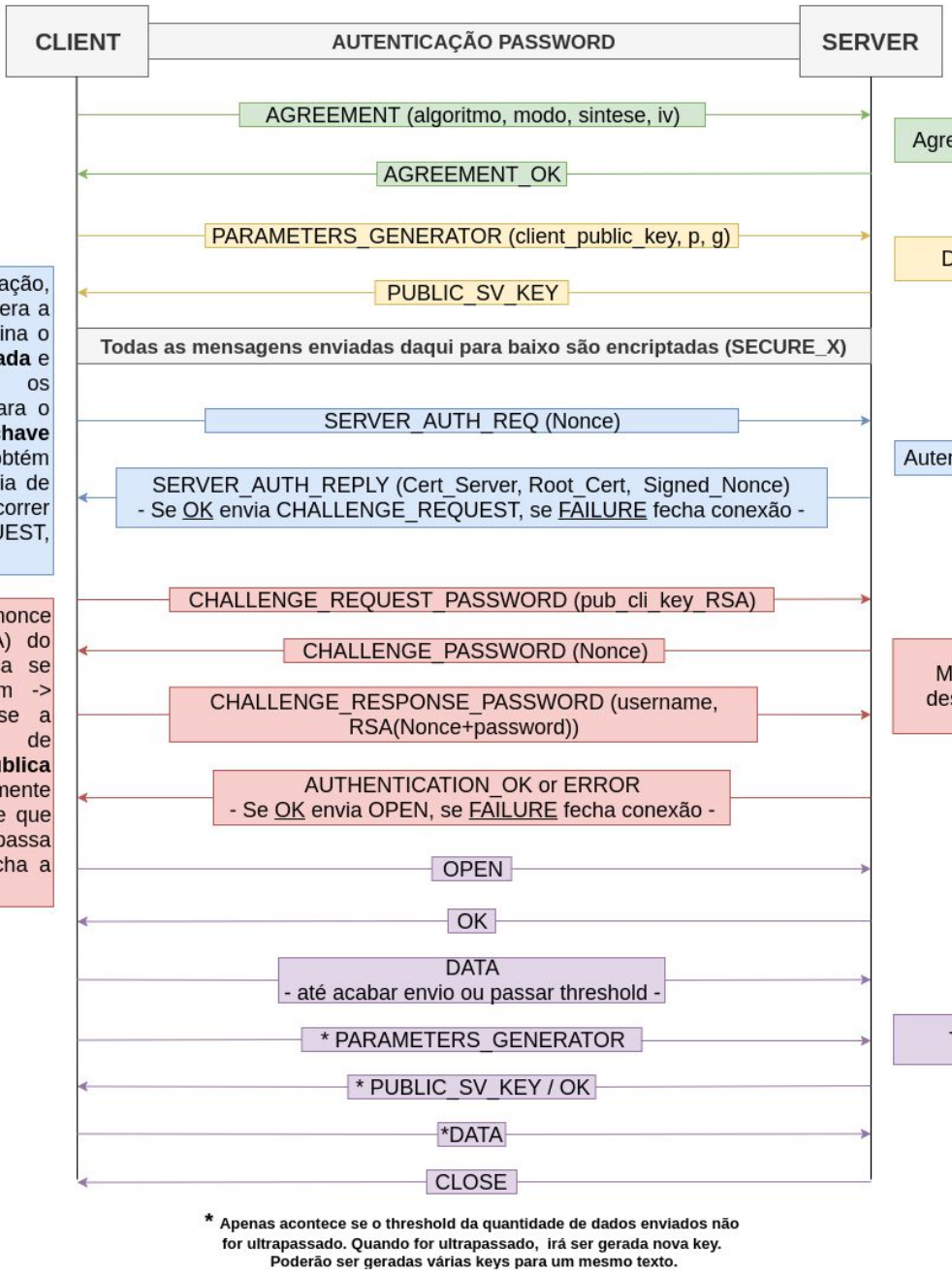
Este projeto tem como base o projeto 03 (Applied Cryptography), do qual foram mantidas a encriptação de mensagens, Diffie-Hellman, MAC para verificação de integridade e o mecanismo de rotação de chaves.

Foram implementados protocolos para a autenticação de utentes através de um mecanismo de desafio resposta, para controlo de acesso, para a autenticação de utentes através do cartão de cidadão, para a autenticação do servidor utilizando certificados X509.

O modo de autenticação a escolher (Password ou CC) é escolhido através de um parâmetro: **-mode CC/PWD**

Desenvolvimento

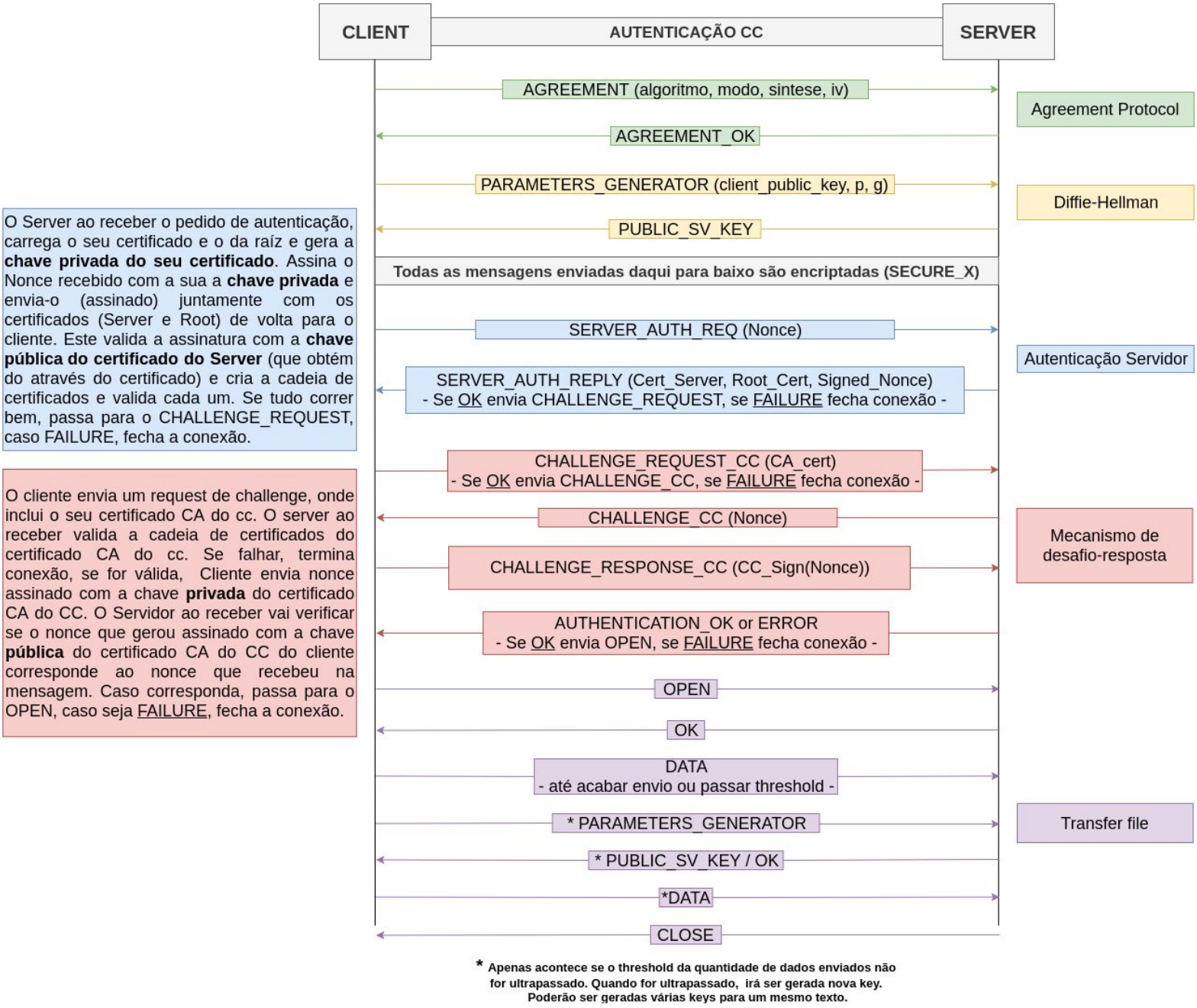
Desenho do Protocolo



O Server ao receber o pedido de autenticação, carrega o seu certificado e o da raíz e gera a **chave privada do seu certificado**. Assina o Nonce recebido com a sua a **chave privada** e envia-o (assinado) juntamente com os certificados (Server e Root) de volta para o cliente. Este valida a assinatura com a **chave pública do certificado do Server** (que obtém do através do certificado) e cria a cadeia de certificados e valida cada um. Se tudo correr bem, passa para o CHALLENGE_REQUEST, caso FAILURE, fecha a conexão.

O Cliente envia username e password+nonce assinado com a chave **privada** (RSA) do cliente. O Servidor ao receber verifica se contém o username (se não contém -> FAILURE). Caso contenha, verifica se a password(guardada na sua "base de dados")+nonce assinado com a chave **pública** (RSA) do cliente (que foi anteriormente trocada) corresponde à password+nonce que recebeu do cliente. Caso corresponda, passa para o OPEN, caso seja FAILURE, fecha a conexão.

Autenticação por apresentação senhas



O Server ao receber o pedido de autenticação, carrega o seu certificado e o da raíz e gera a **chave privada do seu certificado**. Assina o Nonce recebido com a sua a **chave privada** e envia-o (assinado) juntamente com os certificados (Server e Root) de volta para o cliente. Este valida a assinatura com a **chave pública do certificado do Server** (que obtém do através do certificado) e cria a cadeia de certificados e valida cada um. Se tudo correr bem, passa para o CHALLENGE_REQUEST, caso FAILURE, fecha a conexão.

O cliente envia um request de challenge, onde inclui o seu certificado CA do cc. O server ao receber valida a cadeia de certificados do certificado CA do cc. Se falhar, termina conexão, se for válida, Cliente envia nonce assinado com a chave **privada** do certificado CA do CC. O Servidor ao receber vai verificar se o nonce que gerou assinado com a chave **pública** do certificado CA do CC do cliente corresponde ao nonce que recebeu na mensagem. Caso corresponda, passa para o OPEN, caso seja FAILURE, fecha a conexão.

Autenticação por apresentação do CC

Autenticação do servidor com Certificados X.509

Foram gerados 2 certificados através do XCA, um para a Root CA e outro para o servidor (foram também geradas as chaves privadas destes mesmos certificados). Como é possível observar no desenho do nosso protocolo, após o Diffie-Helman o cliente envia um pedido ao servidor para se autenticar, contendo um nonce.

Ao receber este pedido, o servidor vai carregar o **seu certificado**, o **certificado da sua raíz** e a **chave privada do seu certificado**. Após isto, o nonce recebido pelo cliente é passado por uma função hash (SHA256) e assinado com a sua **chave privada do seu certificado**, usando RSA, e vai enviar de volta para o servidor:

- o seu certificado,
- o certificado da sua raíz,
- o nonce assinado.

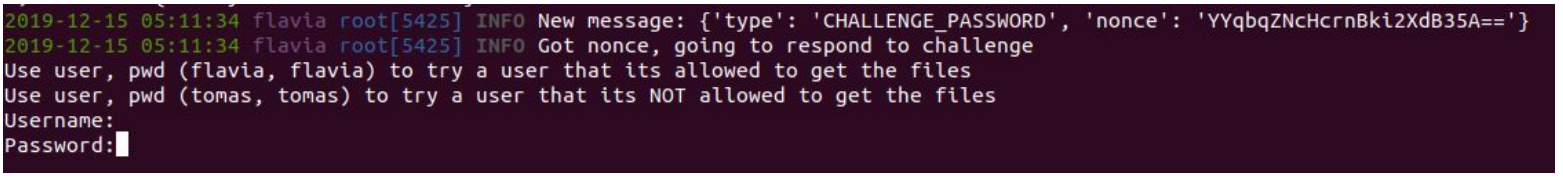
Quando receber a resposta do servidor, o cliente vai passar o nonce que tinha enviado anteriormente por uma função hash (idêntica à mencionada anteriormente) e assiná-lo com a **chave pública do certificado do servidor** que vai construir através do certificado do servidor que recebeu na mensagem e vai comparar com o nonce que recebeu que foi assinado pelo servidor com a **chave privada do certificado do servidor.** Se esta verificação falhar, a autenticação do servidor falha e é fechada a conexão.

Caso a autenticação do nonce seja válida, este prossegue para a análise da chain do **certificado do servidor.** Se este, ao analisar a chain, encontrar a root com certificado autoassinado, valida a chain e prossegue para o mecanismo resposta.

Autenticação do servidor

Autenticação do cliente com Senhas

O server tem em memória uma "base de dados" com as credenciais dos usernames.



Username e Password

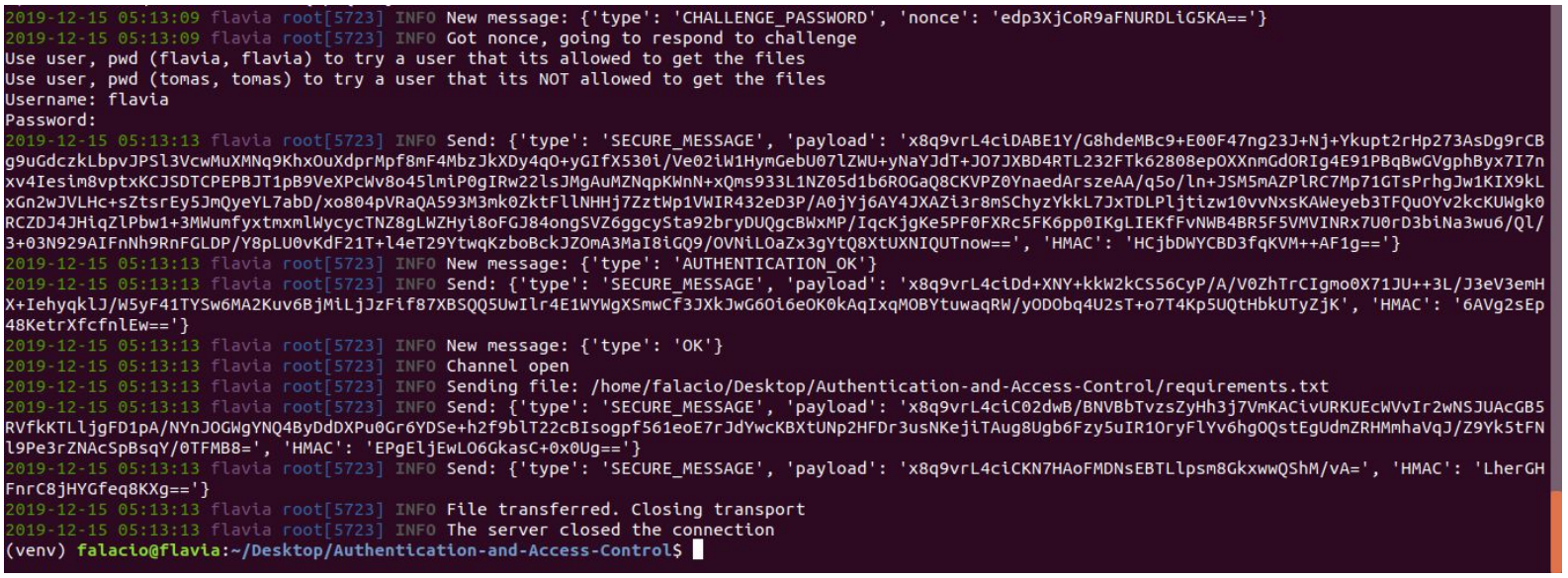
Ao ser selecionada a opção de argumento de autenticação por senhas, este vai ser o modo de autenticação a ser usado.

O desafio mecanismo resposta para a autenticação com senhas inicia-se com o envio de uma mensagem do cliente com um pedido de challenge onde adiciona a **chave pública** usada no RSA. Após isto, o server guarda a chave que foi enviada e envia para o cliente um nonce, ao receber este nonce, são introduzidos o username e password (tal como se pode ver na figura acima), depois a password é concatenada ao nonce e assinadas com a **chave privada.** Para o servidor são enviados o username e nonce+password assinados.

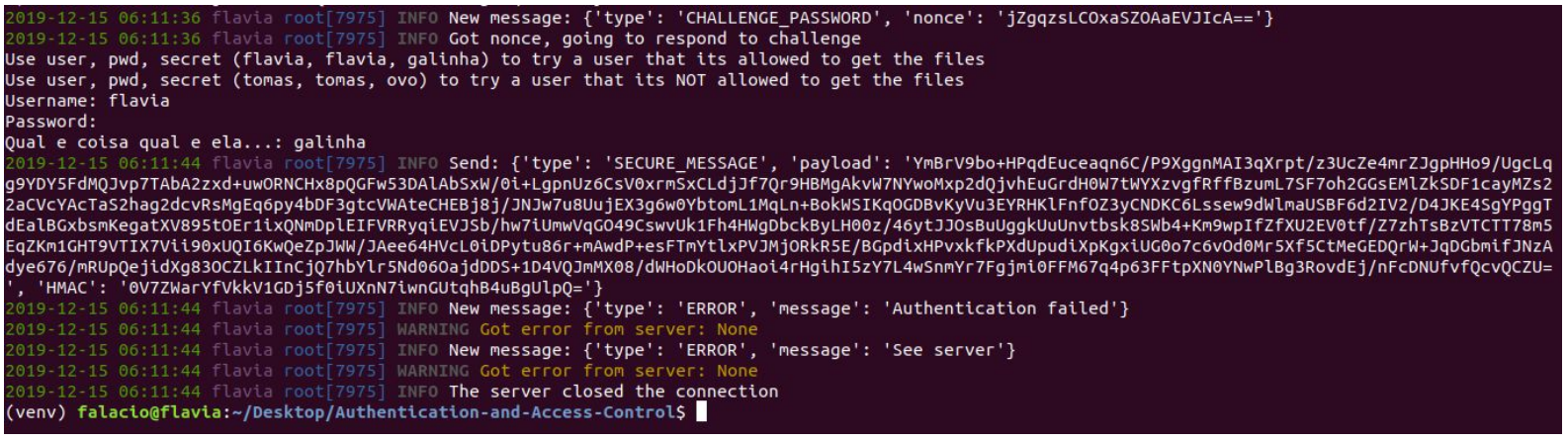
Ao receber estes, em primeiro lugar, verifica se contém o username, se não contém cancela a conexão e avisa, visto que não há necessidade de tentar validar a password.

Se o username existe, ele vai pegar no nonce que enviou, vai buscar a password do username em questão, vai concatená-las e vai assinar com a chave pública que tinha sido enviada antes pelo cliente. Após assinar, compara a password assinada com a chave privada que recebeu do cliente com a que ele assinou

com a chave pública. Caso sejam iguais, a autenticação está concluída e é prosseguida a troca de mensagens, caso não, é desligada a conexão.



Autenticação efetuada com sucesso



Autenticação sem sucesso (Password errada)

Autenticação do cliente com Cartão de Cidadão

Criação e Validação da chain de certificados:

São passados os certificados pela função e caso o issuer do certificado esteja presente em self.user_roots (certificados intermédios) ou self.roots (certificados do sistema) é adicionado o certificado ao array **chains** que é calculado novamente até que o issuer do cero que diferetificado que está a ser passado seja igual ao subject do certificado.

Na validação da chain, são validados os seguintes campos: **purpose**, **common_name**, **signature**, **crl e datas**.

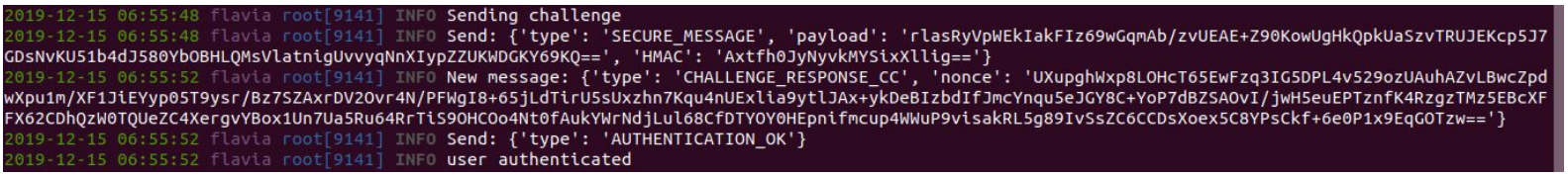
- **purpose:** é o campo no certificado que vai indicar para que é que o certificado vai ser usado. SERVER_AUTH é usado para indicar certificados que podem ser usados para TLS Web Server Authentication. Para o cliente existem duas situações, o primeiro certificado da cadeia que é o CC, tem um parâmetro no key usage que é o digital_signature (que é true quando a chave pública é utilizada para verificar assinaturas digitais), para a restante cadeia é key_cert_sign (true quando a chave pública é utilizada para verificar assinaturas)
- **common_name**: common name refere-se ao nome do issuer. Para validar o certificado, o common name do certificado que vem a seguir na cadeia tem de ser igual ao common name deste..

- **signature:** é um campo de cada certificado que é criado com a chave privada do seu issuer (o que o signature faz é pegar na chave do issuer e validar)
- **crl:** lista de certificados revogados e para cada certificado da cadeia é verificado se não está nessa lista
- **datas:** é verificado se o certificado está expirado baseando-se no seus métodos **not_valid_before** e **not_valid_after**. Se o tempo atual estiver entre os valores retornados por estes métodos, o certificado é válido.

A diferença entre a validação da chain dos certificados do cartão de cidadão e dos certificados do servidor (autenticação do servidor através de X.509), é a validação do purpose, que utiliza métodos diferentes.

O desafio mecanismo resposta para a autenticação com cartão de cidadão inicia-se com o envio de uma mensagem do cliente com um pedido de challenge onde adiciona o **certificado CA do CC**. Após isto, o server guarda o certificado que foi enviado e vai validar a cadeia de certificados do certificado enviado. Se falhar a validação, encerra a conexão, se conseguir validar, envia para o cliente um nonce; ao receber este nonce, o cliente assina o nonce com a **chave privada do cartão de cidadão** e envia-o para o servidor.

O servidor, ao receber o nonce assinado, vai assinar o nonce que enviou anteriormente com a **chave pública do cartão de cidadão** (que extrai do certificado que lhe foi enviado anteriormente) e compara com o que recebeu. Se a comparação for válida a troca de mensagens prossegue, caso contrário, a conexão é terminada.



User autenticado com CC

Mecanismo Controlo de Acesso

Na criação dos users exemplo criámos 1 de cada tipo, user com permissões para descarregar o file e outro sem permissões.

O controlo de acesso foi feito com "flags", ao verificar a autenticação do username + password verifica também se este tem permissões, caso tenha prossegue para a transferência do ficheiro, caso não tenha, cancela a transferência e termina a sessão.

```
2019-12-15 05:13:59 flavia root[5800] INFO Got nonce, going to respond to challenge
Use user, pwd (flavia, flavia) to try a user that its allowed to get the files
Use user, pwd (tomas, tomas) to try a user that its NOT allowed to get the files
Username: tomas
Password:
2019-12-15 05:14:02 flavia root[5800] INFO Send: {'type': 'SECURE_MESSAGE', 'payload': 'eRAhKF0DIRYfX9gEPIt3EIF0jCL2usJ276TxSCHqV4cPE1+v8Yput1KK6c7YnJ
dh2rHVBJgTCydfwYKNZYXyIj7YIDLVigYLFGIbIA91OcY//fg2qVDxWliXraTyeQLQbMOwOT0/YtZpQ2nkWGFqKzTqArB/d9eOMdwk3S2rm6OasuyogG2BFiL9b+HIK7C92cwR4rOW1UdQhlEN1/9N
rYEPcWTxGrfA9d6yEsq4DS06Ai4wmqkoSlo/lJ5Xbx5Ogqsnnl8blMG3cV0dRlLfyKFDoHASa6SBICA3lz4rMo6TDiNthuMxgCpp78TJ7pP9wKI8tSy/mMwCw/WGBcTLPDaPaoJrcbLZFO2tXbyKDb
uhC1WsMOSxdKYwgINt9hYRg7pgtOFr7N61KWfpRt7ZyRjLEhezIaGLBKcTozxKRv5K7vbrm4AVKJTD7X2pbghUGEDrJVwXXGt6ZaiVrb9Zph17VAV+p5UzvunZP1giwUGcJxIRhMFZGBL72PrqZuK7
Bxa+8+uJrhzK6UtWfGx27DjBTg5XDYubohUt7t8IJ5YvvVa2wpzHP65/hR+ccJ9Tfo6qEajudlMRShlQIQnVxjJhTdj94IilQQaFaDkttS4SOl5bEKq/dHWIXVg3KaF6XZ+T3dsUjMqCyuu0BEOgPc
YXvBaWR10ibYbD9lloLOk//gt4IgONjgVd187hKbWdKwk1qRqEV04ovqnYV62Awzf5vWj5TqbiudqpO0ltY84SGBIj7M3EKThjd2sE0RNr', 'HMAC': 'dl7ROfBrLnys0Hg5R6dQ4A==''}
2019-12-15 05:14:02 flavia root[5800] INFO New message: {'type': 'ERROR', 'message': 'User not authorized'}
2019-12-15 05:14:02 flavia root[5800] WARNING Got error from server: None
2019-12-15 05:14:02 flavia root[5800] INFO New message: {'type': 'ERROR', 'message': 'See server'}
2019-12-15 05:14:02 flavia root[5800] WARNING Got error from server: None
2019-12-15 05:14:02 flavia root[5800] INFO The server closed the connection
(venv) falacio@flavia:~/Desktop/Authentication-and-Access-Control$
```

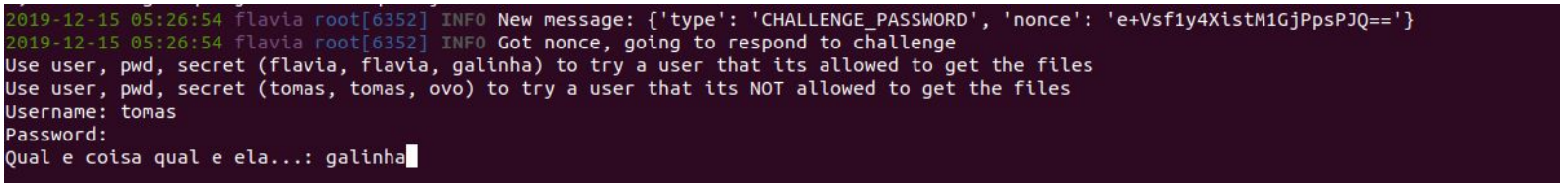
Utilizador NÃO autorizado a transferir ficheiro

```
2019-12-15 05:13:09 flavia root[5723] INFO New message: {'type': 'CHALLENGE_PASSWORD', 'nonce': 'edp3XjCoR9aFNURDLiG5KA==''}
2019-12-15 05:13:09 flavia root[5723] INFO Got nonce, going to respond to challenge
Use user, pwd (flavia, flavia) to try a user that its allowed to get the files
Use user, pwd (tomas, tomas) to try a user that its NOT allowed to get the files
Username: flavia
Password:
2019-12-15 05:13:13 flavia root[5723] INFO Send: {'type': 'SECURE_MESSAGE', 'payload': 'x8q9vrL4ciDABE1Y/G8hdeMBc9+E00F47ng23J+Nj+Ykupt2rHp273AsDg9rCB
g9uGdczkLbpvJPSl3VcwMuXMNq9KhxOuXdprMpf8mF4MbzJkXDy4qO+yGIfX530i/Ve02iW1HymGebU07lZWU+yNaYJdT+JO7JXBD4RTL232FTk62808epOXXnmGdORIg4E91PBqBwGVgphByx7I7n
xv4Iesim8vptxKCJSDTCPEPBJT1pB9VeXPcWv8o45lmiP0gIRw22lsJMgAuMZNqpKWnN+xQms933L1NZ05d1b6ROGaQ8CKVPZ0YnaedArszeAA/q5o/ln+JSM5mAZPlRC7Mp71GTsPrhgJw1KIX9kL
xGn2wJVLHc+sZtsrEy5JmQyeYL7abD/xo804pVRaQA593M3mk0ZktFllNHHj7ZztWp1VWIR432eD3P/A0jYj6AY4JXAZi3r8mSChyzYkkL7JxTDLPljtizw10vvNxsKAWeyeb3TFQuOYv2kcKUWgk0
RCZDJ4JHiqZlPbw1+3MWumfyxtmxmlWycycTNZ8gLWZHyi8oFGJ84ongSVZ6ggcySta92bryDUQgcBWxMP/IqcKjgKe5PF0FXRc5FK6pp0IKgLIEKfFvNWB4BR5F5VMVINRx7U0rD3biNa3wu6/Ql/
3+03N929AIFnNh9RnFGLDP/Y8pLU0vKdF21T+l4eT29YtwqKzboBckJZOmA3MaI8iGQ9/OVNiLOaZx3gYtQ8XtUXNIQUTnow==', 'HMAC': 'HCjbDWYCBD3fqKVM++AF1g==''}
2019-12-15 05:13:13 flavia root[5723] INFO New message: {'type': 'AUTHENTICATION_OK'}
2019-12-15 05:13:13 flavia root[5723] INFO Send: {'type': 'SECURE_MESSAGE', 'payload': 'x8q9vrL4ciDd+XNY+kkW2kCS56CyP/A/V0ZhTrCIgmo0X71JU++3L/J3eV3emH
X+IehyqklJ/W5yF41TYSw6MA2Kuv6BjMiLjJzFif87XBSQQ5UwIlr4E1WYWgXSmwCf3JXkJwG6Oi6eOK0kAqIxqMOBYtuwaqRW/yODObq4U2sT+o7T4Kp5UQtHbkUTyZjK', 'HMAC': '6AVg2sEp
48KetrXfcfnlEw==''}
2019-12-15 05:13:13 flavia root[5723] INFO New message: {'type': 'OK'}
2019-12-15 05:13:13 flavia root[5723] INFO Channel open
2019-12-15 05:13:13 flavia root[5723] INFO Sending file: /home/falacio/Desktop/Authentication-and-Access-Control/requirements.txt
2019-12-15 05:13:13 flavia root[5723] INFO Send: {'type': 'SECURE_MESSAGE', 'payload': 'x8q9vrL4ciC02dwB/BNVBbTvzsZyHh3j7VmKACivURKUEcWVvIr2wNSJUAcGB5
RVfkKTLljgFD1pA/NYnJOGWgYNQ4ByDdDXPu0Gr6YDSe+h2f9blT22cBIsogpf561eoE7rJdYwcKBXtUNp2HFDr3usNKejiTAug8Ugb6Fzy5uIR1OryFlYv6hgOQstEgUdmZRHMmhaVqJ/Z9Yk5tFN
l9Pe3rZNAcSpBsqY/0TFMB8=', 'HMAC': 'EPgEljEwLO6GkasC+0x0Ug==''}
2019-12-15 05:13:13 flavia root[5723] INFO Send: {'type': 'SECURE_MESSAGE', 'payload': 'x8q9vrL4ciCKN7HAoFMDNsEBTLlpsm8GkxwwQShM/vA=', 'HMAC': 'LherGH
FnrC8jHYGfeq8KXg==''}
2019-12-15 05:13:13 flavia root[5723] INFO File transferred. Closing transport
2019-12-15 05:13:13 flavia root[5723] INFO The server closed the connection
(venv) falacio@flavia:~/Desktop/Authentication-and-Access-Control$
```

Utilizador autorizado a transferir ficheiro

Mecanismos de Segurança Adicionais

Como mecanismo de segurança adicional acrescentamos uma pergunta de verificação do utilizador. Cada utilizador tem a sua resposta. Do estilo de perguntas de segurança como "qual o nome do seu primo mais velho", etc.



Questão segurança ao utilizador

Caso a resposta à questão secreta não esteja correta a sessão é terminada. Se estiver correta, a autenticação é efetuada (caso tenha permissões).

Conclusão

A realização do nosso protocolo foi, a nossa ver, realizada com sucesso, uma vez que a comunicação entre o servidor e o cliente está estruturada de maneira bastante segura e robusta e o controlo de acessos dos clientes está a funcionar na plenitude. Foram implementados todos os tópicos que nos foram propostos: Autenticação do Servidor, Autenticação do Cliente através de CC e Password, Mecanismos de desafio resposta e Mecanismos de defesa adicionais.

O tópico onde tivemos mais dificuldades foi a autenticação através de CC.

Todas as validações estão a ser corretas e exatas e sem quaisquer problemas.

Em suma, apesar dos problemas enfrentados durante o desenvolvimento, concluímos que o balanço foi positivo, resultando assim num protocolo robusto e funcional para vários tipos de autenticação.

Bibliografia

- Consulta de vários temas da documentação do Cryptography IO