



UNIVERSIDADE DO MINHO

DEPARTAMENTO DE INFORMÁTICA

MESTRADO EM ENGENHARIA INFORMÁTICA

CRİPTOGRAFIA E SEGURANÇA DE INFORMAÇÃO

Engenharia de Segurança

Ficha Prática 7

Grupo Nº 3

Ariana Lousada (PG47034) Luís Carneiro (PG46541)
Rui Cardoso (PG42849)

1 de maio de 2022

1 Protocolo TLS

1.1 P1.1

Escolha dois sites de Universidades não Europeias.

As duas universidades não Europeias escolhidas foram Harvard University <https://www.harvard.edu/> e University of Cape Town <https://www.uct.ac.za/>

1.1.1 Anexe os resultados do SSL Server test à sua resposta.

Resultados da *Harvard University*:

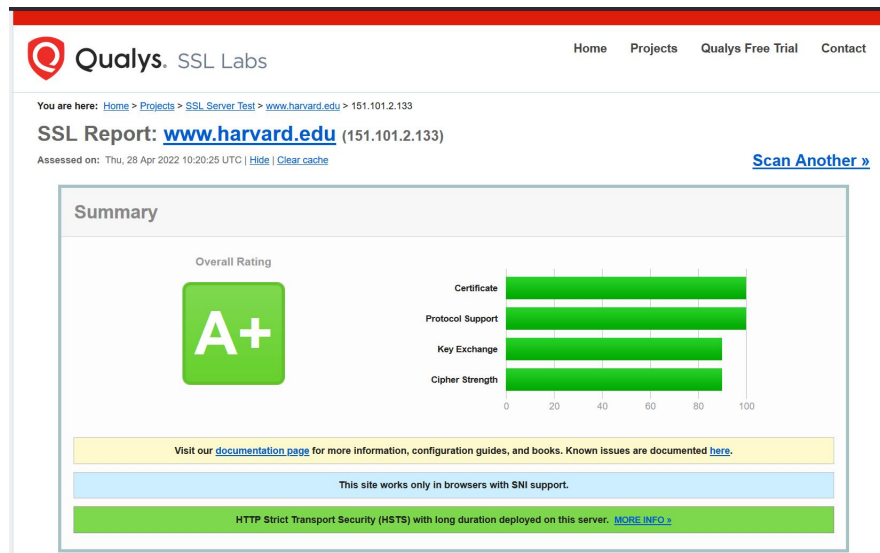


Figura 1: Harvard University SSL Server Test Score



Certificate #1: RSA 2048 bits (SHA256withRSA)	
<div>  Server Key and Certificate #1 </div>	
Subject	www.harvard.edu Fingerprint SHA256: 22ba50e9218a05ac04701e29fbc290094c76384d07c7c6912af84aa8a2d9 Pin SHA256: AwL2s24L8ZgV2DM4MnB0XLYCE0Jr8Asgab8AYU=
Common names	www.harvard.edu
Alternative names	www.harvard.edu
Serial Number	0443d402d7f39fca3e4aed1c52e6aca052fa
Valid from	Wed, 13 Apr 2022 13:44:26 UTC
Valid until	Tue, 12 Jul 2022 13:44:25 UTC (expires in 2 months and 14 days)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No
Issuer	R3 AAA: http://r3.o.kencr.org/
Signature algorithm	SHA256withRSA
Extended Validation	No
Certificate Transparency	Yes (certificate)
OCSF Must Staple	No
Revocation information	OCSF OCSF: http://r3.o.kencr.org
Revocation status	Good (not revoked)
DNS CAA	No (more info)
Trusted	Yes Mozilla Apple Android Java Windows
<div>  Additional Certificates (if supplied) </div>	
Certificates provided	3 (4007 bytes)
Chain issues	None
#2	
Subject	R3 Fingerprint SHA256: 67add1160b020ae01b895c90813c04c2aa589900790805572a3c7e737013d4d Pin SHA256: jQJ7bhh0grw01T4hSumVb+Fsd0gng621gT3PvPKG0=
Valid until	Mon, 15 Sep 2025 16:00:00 UTC (expires in 3 years and 4 months)
Key	RSA 2048 bits (e 65537)
Issuer	ISRG Root X1
Signature algorithm	SHA256withRSA
#3	
Subject	ISRG Root X1 Fingerprint SHA256: 6d99fb265eb1c5b37447693bc648f3cd9e1bffa6c4c298b0d47c7f7f1c24f Pin SHA256: CS+lgZ7cYwmwQMcRFPbsQIVLAB0XQznpa0wHf8M=
Valid until	Mon, 30 Sep 2024 18:14:03 UTC (expires in 2 years and 5 months)
Key	RSA 4096 bits (e 65537)
Issuer	DST Root CA X3
Signature algorithm	SHA256withRSA

Figura 2: Harvard University SSL Server Test Score



Configuration	
<div>  Protocols </div>	
TLS 1.3	No
TLS 1.2	Yes
TLS 1.1	No
TLS 1.0	No
SSL 3	No
SSL 2	No
(*) Experimental: Server negotiated using No-SNI	
<div>  Cipher Suites </div>	
# TLS 1.2 (suites in server-preferred order)	
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)	128
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	256
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)	256 ^(P)
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)	128 WEAK
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	256 WEAK
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	128 WEAK
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	256 WEAK
TLS_RSA_WITH_AES_128_GCM_SHA256 (0xc03c)	128 WEAK
TLS_RSA_WITH_AES_128_CBC_SHA (0xc02f)	128 WEAK
TLS_RSA_WITH_AES_256_CBC_SHA (0xc035)	256 WEAK
(P) This server prefers ChaCha20 suites with clients that don't have AES-NI (e.g., Android devices)	

Figura 3: Harvard University SSL Server Test Score

Resultados da *University of Cape Town*:

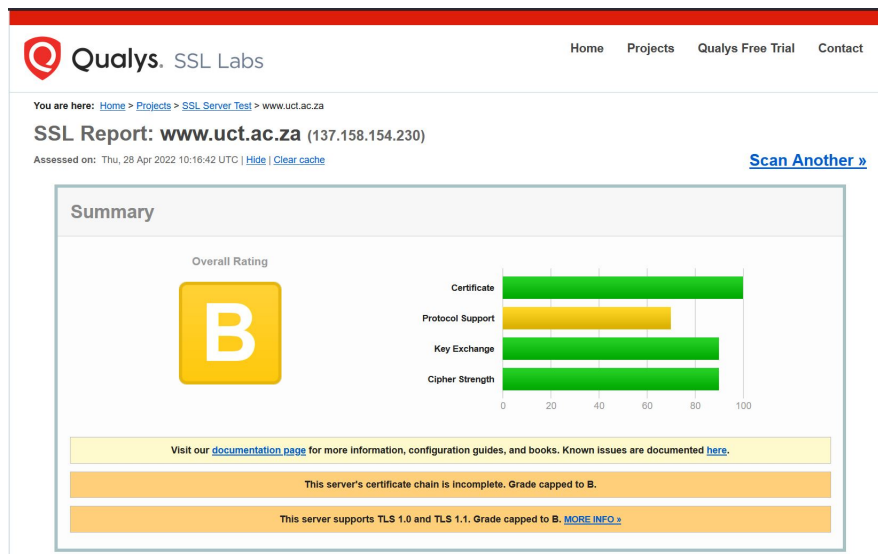


Figura 4: University of Cape Town SSL Server Test Score

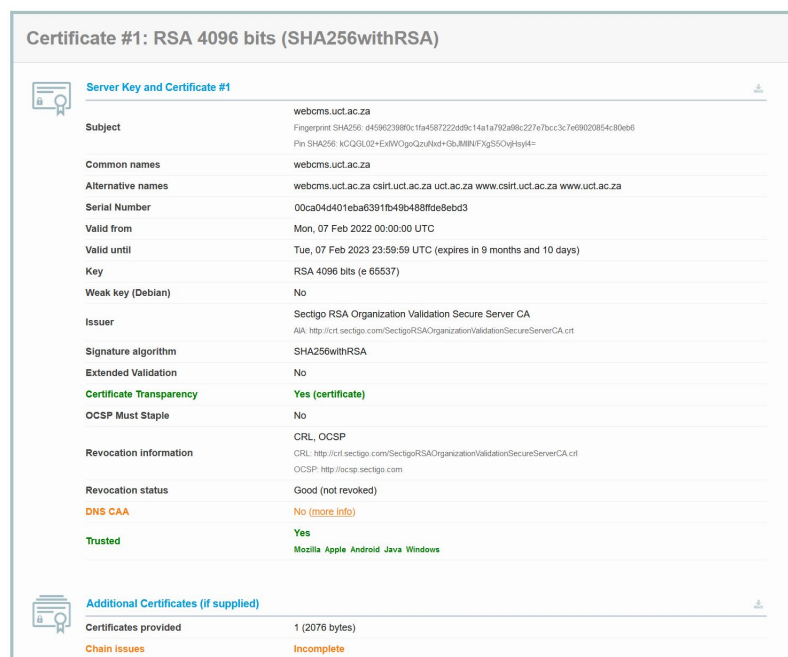


Figura 5: University of Cape Town SSL Server Test Score



Configuration		
 Protocols		
TLS 1.3		No
TLS 1.2		Yes
TLS 1.1		Yes
TLS 1.0		Yes
SSL 3		No
SSL 2		No
 Cipher Suites		
# TLS 1.2 (suites in server-preferred order)		
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH secp256r1 (eq. 3072 bits RSA) FS	256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)	ECDH secp256r1 (eq. 3072 bits RSA) FS	128
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0xc031)	DH 2048 bits FS	256
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0xc030)	DH 2048 bits FS	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK 256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK 256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0xc03b)	DH 2048 bits FS	WEAK 256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0xc039)	DH 2048 bits FS	WEAK 256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK 128
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK 128
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0xc037)	DH 2048 bits FS	WEAK 128
TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0xc033)	DH 2048 bits FS	WEAK 128
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (0xc012)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK 112
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (0xc016)	DH 2048 bits FS	WEAK 112
TLS_RSA_WITH_AES_256_GCM_SHA384 (0xc03d)		WEAK 256
TLS_RSA_WITH_AES_128_GCM_SHA256 (0xc03c)		WEAK 128
TLS_RSA_WITH_AES_256_CBC_SHA256 (0xc03d)		WEAK 256
TLS_RSA_WITH_AES_256_CBC_SHA (0xc035)		WEAK 256
TLS_RSA_WITH_AES_128_CBC_SHA256 (0xc03c)		WEAK 128
TLS_RSA_WITH_AES_128_CBC_SHA (0xc02f)		WEAK 128
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xc01a)		WEAK 112
# TLS 1.1 (suites in server-preferred order)		
# TLS 1.0 (suites in server-preferred order)		

Figura 6: University of Cape Town SSL Server Test Score

1.1.2 Analise o resultado do SSL Server test relativo ao site escolhido com pior rating. Que comentários pode fazer sobre a sua segurança. Porquê?

O teste com pior rating foi o da universidade *University of Cape Town* com o resultado de B em contraste com o da universidade *Harvard University* que teve uma classificação de A+.

O resultado é apenas B devido ao facto de a chain de certificados do servidor estar incompleta e o servidor suportar TLS 1.0 e TLS 1.1. Isto faz com que a classificação fique limitada a um máximo de B.

Também se deve ao facto do TLS 1.0 e TLS 1.1 terem sido descontinuados; uma vez que já não existem atualizações para os mesmos, o número de vulnerabilidades vai aumentando com o tempo.

Vários browsers deixaram de suportar TLS 1.0 e 1.1 durante o ano de 2020 como o Microsoft IE, Edge, Google Chrome, Mozilla Firefox e Safari.

É recomendado que os servidores deixem de utilizar TLS 1.0 e 1.1 e passem a usar apenas TLS 1.2+. As melhores práticas podem ser consultadas no ficheiro RFC-7525.

Um pior rating permite que um atacante possa conseguir fazer ataques como *man-in-the-middle*.

Também podemos ver que suporta algumas *Cipher Suites* consideradas fracas quando idealmente deveria apenas suportar as que estão a verde.

1.1.3 É natural que tenha visto a seguinte informação: "DNS CAA". O que significa, para efeitos práticos?

O DNS CAA trata-se de uma proposta para melhorar a segurança do ecossistema *PKI*, controlando quais CA's podem emitir certificados para um domínio em particular. Esta proposta foi feita em 2013 no documento RFC 6844.

O facto de qualquer CA poder emitir um certificado para qualquer domínio é considerado o ponto mais fraco do ecossistema *PKI*. Como existem centenas de CA's e nenhum controlo sobre as suas ações em particular, o número de riscos pode potencialmente aumentar.

Estes riscos são mitigados com a utilização de um CAA, uma vez que este cria um mecanismo de DNS que permite aos donos do domínio escolher apenas os CA's que pretendem autorizar para a geração de certificados dos respetivos *hostnames*.

Antes de um CA gerar um certificado, este deve verificar o DNS. Caso o CA não se encontre no DNS, este deve recusar o pedido para gerar o certificado.

Em nenhuma das universidades testadas se verificou algum tipo de suporte do DNS CAA. Contudo, existem empresas que já possuem serviços a utilizar esta tecnologia, como por exemplo a Google(nomeadamente no seu site *google.com*).

2 Protocolo SSH

2.1 P2.1

Escolha dois servidores ssh de Universidades não Europeias.

Os seguintes IP's foram encontrados utilizando o website <https://www.shodan.io/>:

- Harvard University com o IP: 140.247.175.39
- University of Cape Town com o IP: 137.158.234.87

Os dois servidores escolhidos são das mesmas universidades escolhidas na pergunta 1.1.

2.1.1 Anexe os resultados do ssh-audit à sua resposta.

```
(base) [obsession@ARES ~]$ ssh-audit 140.247.175.39
# general
(gen) banner: SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.13
(gen) software: OpenSSH 6.6.1p1
(gen) compatibility: OpenSSH 6.5-6.6, Dropbear SSH 2013.62+ (some functionality from 0.52)
(gen) compression: enabled (zlib@openssh.com)
# security
(cve) CVE-2018-15473 -- (CVSSv2: 5.3) enumerate usernames due to timing discrepancies
(cve) CVE-2016-3115 -- (CVSSv2: 5.5) bypass command restrictions via crafted X11 forwarding data
(cve) CVE-2016-1907 -- (CVSSv2: 5.0) cause DoS via crafted network traffic (out of bounds read)
(cve) CVE-2015-8325 -- (CVSSv2: 7.2) privilege escalation via triggering crafted environment
(cve) CVE-2015-6564 -- (CVSSv2: 6.9) privilege escalation via leveraging sshd uid
(cve) CVE-2015-6563 -- (CVSSv2: 1.9) conduct impersonation attack
# key exchange algorithms
(kex) curve25519-sha256@libssh.org -- [info] available since OpenSSH 6.5, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp256 -- [fail] using weak elliptic curves
(kex) ecdh-sha2-nistp384 -- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp521 -- [fail] using weak elliptic curves
(kex) diffie-hellman-group-exchange-sha256 (1024-bit) -- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) diffie-hellman-group-exchange-sha1 (1024-bit) -- [fail] using small 1024-bit modulus
(kex) diffie-hellman-group14-sha1 -- [warn] using weak hashing algorithm
(kex) diffie-hellman-group1-sha1 -- [info] available since OpenSSH 2.3.0
(kex) diffie-hellman-group1-sha1 -- [warn] using weak hashing algorithm
(kex) diffie-hellman-group1-sha1 -- [info] available since OpenSSH 3.9, Dropbear SSH 0.53
(kex) diffie-hellman-group1-sha1 -- [fail] using small 1024-bit modulus
(kex) diffie-hellman-group1-sha1 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
(kex) diffie-hellman-group1-sha1 -- [fail] disabled (in client) since OpenSSH 7.0, logjam attack
(kex) diffie-hellman-group1-sha1 -- [warn] using weak hashing algorithm
(kex) diffie-hellman-group1-sha1 -- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.28
# host-key algorithms
(key) ssh-rsa (2048-bit) -- [fail] using weak hashing algorithm
(key) ssh-rsa (2048-bit) -- [info] available since OpenSSH 2.5.0, Dropbear SSH 0.28
(key) ssh-rsa (2048-bit) -- [info] a future deprecation notice has been issued in OpenSSH 8.2: https://www.openssh.com/txt/release-8.2
(key) ssh-dss -- [fail] using small 1024-bit modulus
(key) ssh-dss -- [fail] removed (in server) and disabled (in client) since OpenSSH 7.0, weak algorithm
(key) ssh-dss -- [warn] using weak random number generator could reveal the key
(key) ecdsa-sha2-nistp256 -- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28
(key) ecdsa-sha2-nistp256 -- [fail] using weak elliptic curves
(key) ecdsa-sha2-nistp256 -- [warn] using weak random number generator could reveal the key
(key) ssh-ed25519 -- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(key) ssh-ed25519 -- [info] available since OpenSSH 6.5
```

Figura 7: Harvard University SSH Audit

```

# encryption algorithms (ciphers)
(enc) aes128-ctr -- [info] available since OpenSSH 3.7, Dropbear SSH 0.52
(enc) aes192-ctr -- [info] available since OpenSSH 3.7
(enc) aes256-ctr -- [info] available since OpenSSH 3.7, Dropbear SSH 0.52
(enc) arcfour256 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher
(enc) arcfour128 -- [info] available since OpenSSH 4.2
-- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher
-- [info] available since OpenSSH 4.2
(enc) aes128-gcm@openssh.com -- [info] available since OpenSSH 6.2
(enc) aes256-gcm@openssh.com -- [info] available since OpenSSH 6.2
(enc) chacha20-poly1305@openssh.com -- [info] available since OpenSSH 6.5
-- [info] default cipher since OpenSSH 6.9.
(enc) aes128-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.28
(enc) 3des-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.4, unsafe algorithm
-- [warn] using weak cipher
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
-- [info] available since OpenSSH 1.2.2, Dropbear SSH 0.28
(enc) blowfish-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [fail] disabled since Dropbear SSH 0.53
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
(enc) cast128-cbc -- [info] available since OpenSSH 1.2.2, Dropbear SSH 0.28
-- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
(enc) aes192-cbc -- [info] available since OpenSSH 2.1.0
-- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0
(enc) aes256-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.47
(enc) arcfour -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher
-- [info] available since OpenSSH 2.1.0
(enc) rijndael-cbc@lysator.liu.se -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0

```

Figura 8: Harvard University SSH Audit

```

# message authentication code algorithms
(mac) hmac-md5-etm@openssh.com -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                                  ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                                  ^- [warn] using weak hashing algorithm
                                  ^- [info] available since OpenSSH 6.2
(mac) hmac-sha1-etm@openssh.com -- [warn] using weak hashing algorithm
                                  ^- [info] available since OpenSSH 6.2
(mac) umac-64-etm@openssh.com -- [warn] using small 64-bit tag size
                                  ^- [info] available since OpenSSH 6.2
(mac) umac-128-etm@openssh.com -- [info] available since OpenSSH 6.2
(mac) hmac-sha2-256-etm@openssh.com -- [info] available since OpenSSH 6.2
(mac) hmac-sha2-512-etm@openssh.com -- [info] available since OpenSSH 6.2
(mac) hmac-ripemd160-etm@openssh.com -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                                  ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                                  ^- [info] available since OpenSSH 6.2
(mac) hmac-sha1-96-etm@openssh.com -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                                  ^- [warn] using weak hashing algorithm
                                  ^- [info] available since OpenSSH 6.2
(mac) hmac-md5-96-etm@openssh.com -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                                  ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                                  ^- [warn] using weak hashing algorithm
                                  ^- [info] available since OpenSSH 6.2
(mac) hmac-md5 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                  ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                  ^- [warn] using encrypt-and-MAC mode
                  ^- [warn] using weak hashing algorithm
                  ^- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28
(mac) hmac-sha1 -- [warn] using encrypt-and-MAC mode
                  ^- [warn] using weak hashing algorithm
                  ^- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28
(mac) umac-64@openssh.com -- [warn] using encrypt-and-MAC mode
                          ^- [warn] using small 64-bit tag size
                          ^- [info] available since OpenSSH 4.7
(mac) umac-128@openssh.com -- [warn] using encrypt-and-MAC mode
                          ^- [info] available since OpenSSH 6.2
(mac) hmac-sha2-256 -- [warn] using encrypt-and-MAC mode
                      ^- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
(mac) hmac-sha2-512 -- [warn] using encrypt-and-MAC mode
                      ^- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
(mac) hmac-ripemd160 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                      ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                      ^- [warn] using encrypt-and-MAC mode
                      ^- [info] available since OpenSSH 2.5.0
(mac) hmac-ripemd160@openssh.com -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                                  ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                                  ^- [warn] using encrypt-and-MAC mode
                                  ^- [info] available since OpenSSH 2.1.0
(mac) hmac-sha1-96 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
                     ^- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
                     ^- [warn] using encrypt-and-MAC mode
                     ^- [warn] using weak hashing algorithm
                     ^- [info] available since OpenSSH 2.5.0, Dropbear SSH 0.47
(mac) hmac-md5-96 -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm

```

Figura 9: Harvard University SSH Audit


```

[warn] using weak hashing algorithm
[info] available since OpenSSH 2.5.0, Dropbear SSH 0.47
[fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
[warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
[warn] using encrypt-and-MAC mode
[warn] using weak hashing algorithm
[info] available since OpenSSH 2.5.0

# fingerprints
(fin) ssh-ed25519: SHA256:uv+w2qQYA0+kan7VvIm6LIJcAqmAu3pyu+6ya9A/TUY
(fin) ssh-rsa: SHA256:zcztDItSN9B3D/KCUqvcrHNEkiac88P2B2DeL/LjhFQ

# algorithm recommendations (for OpenSSH 6.6.1)
(rec) -diffie-hellman-group-exchange-sha256 -- kex algorithm to change (increase modulus size to 2048 bits or larger)
(rec) -3des-cbc -- enc algorithm to remove
(rec) -aes128-cbc -- enc algorithm to remove
(rec) -aes192-cbc -- enc algorithm to remove
(rec) -aes256-cbc -- enc algorithm to remove
(rec) -arcfour -- enc algorithm to remove
(rec) -arcfour128 -- enc algorithm to remove
(rec) -arcfour256 -- enc algorithm to remove
(rec) -blowfish-cbc -- enc algorithm to remove
(rec) -cast128-cbc -- enc algorithm to remove
(rec) -diffie-hellman-group-exchange-sha1 -- kex algorithm to remove
(rec) -diffie-hellman-group1-sha1 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp256 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp384 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp521 -- kex algorithm to remove
(rec) -ecdsa-sha2-nistp256 -- key algorithm to remove
(rec) -hmac-md5 -- mac algorithm to remove
(rec) -hmac-md5-96 -- mac algorithm to remove
(rec) -hmac-md5-96-etm@openssh.com -- mac algorithm to remove
(rec) -hmac-md5-etm@openssh.com -- mac algorithm to remove
(rec) -hmac-ripemd160 -- mac algorithm to remove
(rec) -hmac-ripemd160-etm@openssh.com -- mac algorithm to remove
(rec) -hmac-ripemd160@openssh.com -- mac algorithm to remove
(rec) -hmac-sha1-96 -- mac algorithm to remove
(rec) -hmac-sha1-96-etm@openssh.com -- mac algorithm to remove
(rec) -rijndael-cbc@lysator.liu.se -- enc algorithm to remove
(rec) -ssh-dss -- key algorithm to remove
(rec) -ssh-rsa -- key algorithm to remove
(rec) -diffie-hellman-group14-sha1 -- kex algorithm to remove
(rec) -hmac-sha1 -- mac algorithm to remove
(rec) -hmac-sha1-etm@openssh.com -- mac algorithm to remove
(rec) -hmac-sha2-256 -- mac algorithm to remove
(rec) -hmac-sha2-512 -- mac algorithm to remove
(rec) -umac-128@openssh.com -- mac algorithm to remove
(rec) -umac-64-etm@openssh.com -- mac algorithm to remove
(rec) -umac-64@openssh.com -- mac algorithm to remove

# additional info
(info) For hardening guides on common OSes, please see: <https://www.ssh-audit.com/hardening\_guides.html>

```

Figura 10: Harvard University SSH Audit

```
(base) [obsession@ARES ~]$ ssh-audit 137.158.234.87
# general
(gen) banner: SSH-2.0-OpenSSH_7.4
(gen) software: OpenSSH 7.4
(gen) compatibility: OpenSSH 7.4+ (some functionality from 6.6); Dropbear SSH 2018.76+ (some functionality from 0.52)
(gen) compression: enabled (zlib@openssh.com)

# security
(cve) CVE-2018-15473 -- (CVSSv2: 5.3) enumerate usernames due to timing discrepancies

# key exchange algorithms
(kex) curve25519-sha256 -- [info] available since OpenSSH 7.4, Dropbear SSH 2018.76
(kex) curve25519-sha256@libssh.org -- [info] available since OpenSSH 6.5, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp256 -- [fail] using weak elliptic curves
(kex) ecdh-sha2-nistp384 -- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(kex) ecdh-sha2-nistp521 -- [fail] using weak elliptic curves
(kex) diffie-hellman-group-exchange-sha256 (1024-bit) -- [fail] using small 1024-bit modulus
-- [info] available since OpenSSH 4.4
(kex) diffie-hellman-group16-sha512 -- [info] available since OpenSSH 7.3, Dropbear SSH 2016.73
(kex) diffie-hellman-group18-sha512 -- [info] available since OpenSSH 7.3
(kex) diffie-hellman-group-exchange-sha1 (1024-bit) -- [fail] using small 1024-bit modulus
-- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 2.3.0
(kex) diffie-hellman-group14-sha256 -- [info] available since OpenSSH 7.3, Dropbear SSH 2016.73
(kex) diffie-hellman-group14-sha1 -- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 3.9, Dropbear SSH 0.53
(kex) diffie-hellman-group1-sha1 -- [fail] using small 1024-bit modulus
-- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [fail] disabled (in client) since OpenSSH 7.0, logjam attack
-- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.28

# host-key algorithms
(key) ssh-rsa (2048-bit) -- [fail] using weak hashing algorithm
-- [info] available since OpenSSH 2.5.0, Dropbear SSH 0.28
-- [info] a future deprecation notice has been issued in OpenSSH 8.2: https://www.openssh.com/txt/release-8.2
(key) rsa-sha2-512 (2048-bit) -- [info] available since OpenSSH 7.2
(key) rsa-sha2-256 (2048-bit) -- [info] available since OpenSSH 7.2
(key) ecdsa-sha2-nistp256 -- [fail] using weak elliptic curves
-- [warn] using weak random number generator could reveal the key
-- [info] available since OpenSSH 5.7, Dropbear SSH 2013.62
(key) ssh-ed25519 -- [info] available since OpenSSH 6.5
```

Figure 11: University of Cape Town

```
# encryption algorithms (ciphers)
(enc) chacha20-poly1305@openssh.com -- [info] available since OpenSSH 6.5
-- [info] default cipher since OpenSSH 6.9.
(enc) aes128-ctr -- [info] available since OpenSSH 3.7, Dropbear SSH 0.52
(enc) aes192-ctr -- [info] available since OpenSSH 3.7
(enc) aes256-ctr -- [info] available since OpenSSH 3.7, Dropbear SSH 0.52
(enc) aes128-gcm@openssh.com -- [info] available since OpenSSH 6.2
(enc) aes256-gcm@openssh.com -- [info] available since OpenSSH 6.2
(enc) aes128-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.28
(enc) aes192-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0
(enc) aes256-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] using weak cipher mode
-- [info] available since OpenSSH 2.3.0, Dropbear SSH 0.47
(enc) blowfish-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [fail] disabled since Dropbear SSH 0.53
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
-- [info] available since OpenSSH 1.2.2, Dropbear SSH 0.28
(enc) cast128-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.2, legacy algorithm
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
-- [info] available since OpenSSH 2.1.0
(enc) 3des-cbc -- [fail] removed (in server) since OpenSSH 6.7, unsafe algorithm
-- [warn] disabled (in client) since OpenSSH 7.4, unsafe algorithm
-- [warn] using weak cipher
-- [warn] using weak cipher mode
-- [warn] using small 64-bit block size
-- [info] available since OpenSSH 1.2.2, Dropbear SSH 0.28
```

Figure 12: University of Cape Town

```
# message authentication code algorithms
(mac) umac-64-etm@openssh.com -- [warn] using small 64-bit tag size
-- [info] available since OpenSSH 6.2
(mac) umac-128-etm@openssh.com -- [info] available since OpenSSH 6.2
-- [info] available since OpenSSH 6.2
(mac) hmac-sha2-256-etm@openssh.com -- [info] available since OpenSSH 6.2
-- [info] available since OpenSSH 6.2
(mac) hmac-sha1-etm@openssh.com -- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 6.2
-- [warn] using encrypt-and-MAC mode
-- [warn] using small 64-bit tag size
-- [info] available since OpenSSH 4.7
(mac) umac-64@openssh.com -- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
(mac) umac-128@openssh.com -- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28
(mac) hmac-sha2-256 -- [info] available since OpenSSH 6.2
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
(mac) hmac-sha1 -- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28
(mac) hmac-sha1 -- [info] available since OpenSSH 5.9, Dropbear SSH 2013.56
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using encrypt-and-MAC mode
-- [warn] using weak hashing algorithm
-- [info] available since OpenSSH 2.1.0, Dropbear SSH 0.28

# fingerprints
(fin) ssh-ed25519: SHA256:8A9FwG9agJT0YGTDLJP/Ah8EfKZlKLxDILVbUnBNcDI
(fin) ssh-rsa: SHA256:J6/bMheyYwK7aAy2SDSRVF34LDLHIjgz0cRz58aNczk

# algorithm recommendations (for OpenSSH 7.4)
(rec) !diffie-hellman-group-exchange-sha256 -- kex algorithm to change (increase modulus size to 2048 bits or larger)
(rec) -3des-cbc -- enc algorithm to remove
(rec) -aes128-cbc -- enc algorithm to remove
(rec) -aes192-cbc -- enc algorithm to remove
(rec) -aes256-cbc -- enc algorithm to remove
(rec) -blowfish-cbc -- enc algorithm to remove
(rec) -cast128-cbc -- enc algorithm to remove
(rec) -diffie-hellman-group-exchange-sha1 -- kex algorithm to remove
(rec) -diffie-hellman-group1-sha1 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp256 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp384 -- kex algorithm to remove
(rec) -ecdh-sha2-nistp521 -- kex algorithm to remove
(rec) -ecdsa-sha2-nistp256 -- key algorithm to remove
(rec) -ssh-rsa -- key algorithm to remove
(rec) -diffie-hellman-group14-sha1 -- kex algorithm to remove
(rec) -hmac-sha1 -- mac algorithm to remove
(rec) -hmac-sha1-etm@openssh.com -- mac algorithm to remove
(rec) -hmac-sha2-256 -- mac algorithm to remove
(rec) -hmac-sha2-512 -- mac algorithm to remove
(rec) -umac-128@openssh.com -- mac algorithm to remove
(rec) -umac-64-etm@openssh.com -- mac algorithm to remove
(rec) -umac-64@openssh.com -- mac algorithm to remove

# additional info
(info) For hardening guides on common OSes, please see: <https://www.ssh-audit.com/hardening_guides.html>
```

Figura 13: University of Cape Town

2.1.2 Indique o software e versão utilizada pelos servidores ssh.

O servidor SSH da University of Harvard utiliza OpenSSH 6.6.1p1, compatibility OpenSSH 6.5-6.6, Dropbear SSH 2013.62+, enquanto que o servidor SSH da University of Cape Town utiliza OpenSSH 7.4, compatibility OpenSSH 7.3+, Dropbear ssh 2016.73.

2.1.3 Qual dessas versões de software tem mais vulnerabilidades?

Segundo o SSH-Audit a versão 6.6.1p1 da Harvard University, este possui mais vulnerabilidades (6) em comparação com a versão 7.4 da University of Cape Town, que apenas possui uma vulnerabilidade.

2.1.4 E qual tem a vulnerabilidade mais grave (de acordo com o CVSS score identificado no CVE details)?

As vulnerabilidades encontradas pertencentes a University of Harvard foram: CVE-2018-15473, CVE-2016-3115, CVE-2016-1907, CVE-2015-8325, CVE-2015-6564, CVE-2015-6563. Enquanto isto, a University of Cape Town apenas tem a vulnerabilidade CVE-2018-15473.

Desta lista de vulnerabilidades, a mais grave de acordo com o CVSS Details corresponde à CVE-2015-8325 com um score de 7.2.

Vulnerability Details : CVE-2015-8325	
<p>The <code>do_setup_env</code> function in <code>session.c</code> in <code>sshd</code> in <code>OpenSSH</code> through 7.2p2, when the <code>UseLogin</code> feature is enabled and <code>PAM</code> is configured to read <code>.pam_environment</code> files in user home directories, allows local users to gain privileges by triggering a crafted environment for the <code>/bin/login</code> program, as demonstrated by an <code>LD_PRELOAD</code> environment variable.</p> <p>Publish Date : 2016-05-01 Last Update Date : 2018-06-30</p>	
Collapse All Expand All Select Select&Copy Scroll To Comments External Links Search Twitter Search YouTube Search Google	
- CVSS Scores & Vulnerability Types	
CVSS Score	7.2
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Gain privileges
CWE ID	264

Figura 14: CVSS Details score

2.1.5 Para efeitos práticos, a vulnerabilidade indicada no ponto anterior é grave? Porquê?

A função `do_setup_env` no ficheiro `session.c` no `sshd` possui esta vulnerabilidade até a versão 7.2p2. Caso o sistema possua `UseLogin` activo e o *privileged access management* esteja configurada para ler ficheiros `.pam_environment` na directoria dos utilizadores, permite que os utilizadores locais ganhem privilégios e permissões que não deveriam.

Como a complexidade para efetuar o ataque é baixa e os resultados são graves, esta vulnerabilidade é grave, uma vez que compromete a integridade e confidenciabilidade do sistema.

3 TOR (The Onion Router)

3.1 P3.1

Para aceder a alguns sites nos EUA tem que estar localizado nos EUA.

3.1.1 Efetuando o comando *sudo anonsurf start* consegue garantir que está localizado nos EUA?

Após efetuar o comando

```
sudo anonsurf start
```

foi executado o comando

```
sudo anonsurf myip
```

para descobrir qual o endereço IP atribuído.

Após uma pesquisa deste endereço IP no website [1], confirma-se que este se situa nos Países Baixos. Portanto, a execução deste comando não garante que a localização do endereço IP seja nos EUA.

```
luis@luis-VivoBook:~/kali-anonsurf$ sudo anonsurf myip
My ip is:
83.137.158.10
-----
```

Figura 15: Endereço IP após iniciar o módulo *anonsurf*

3.1.2 Porquê? Utilize características do protocolo TOR para justificar.

A rede TOR é constituída por *Onion Proxy's* (OP), que são executados em cada utilizador e *Onion Routers* (OR), que são utilizados como intermediários entre o utilizador e o *website* que quer aceder.

Para garantir anonimidade, os OP's vão mudando de 10 em 10 minutos - se o nodo de saída a partir do qual se extrai a localização do utilizador muda, a localização do mesmo também muda, pelo que não é possível garantir que o utilizador está localizado nos EUA.

3.2 P3.2

No seguimento da experiência 3.2, acesse a <https://www.bbcweb3hytmzhn5d532owbu6oqadra5z3ar726vq5kgwwn6aucdccrad.onion/>, <http://ciadotgov4sjwlzihbbgxnqg3xiyrg7so2r2o3lt5wz5ypk4sxyjstad.onion> ou <https://www.facebookkwkhpilnemxj7asaniu7vnjbjltxjqhye3mhbshg7kx5tfyd.onion/>.

3.2.1 Clique no lado esquerdo da barra de URL (no símbolo do onion) e verifique qual é o circuito para esse site.

O circuito para este site pode ser observado na figura seguinte.

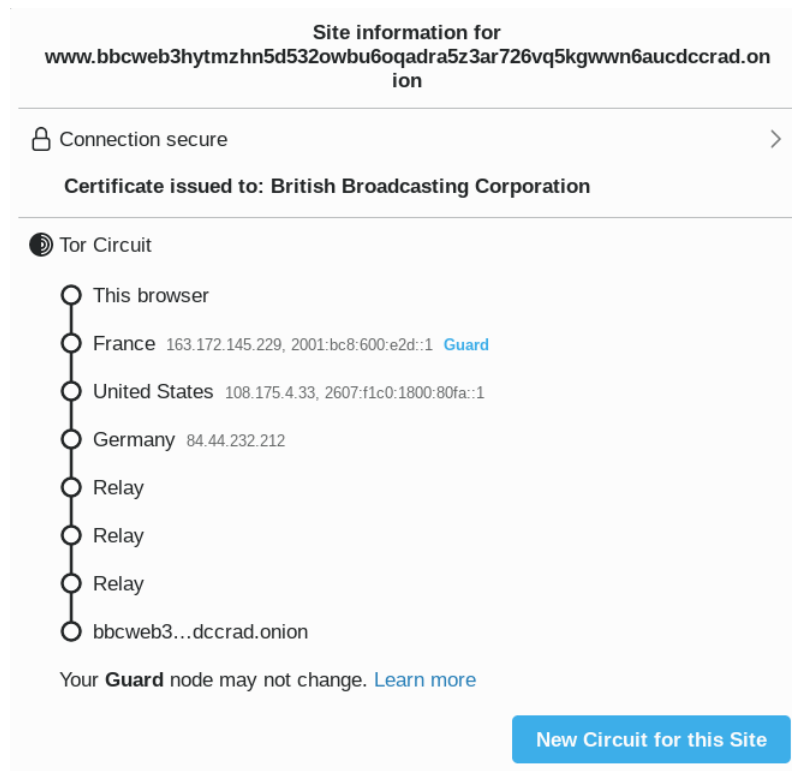


Figura 16: Circuito estabelecido pela rede TOR

3.2.2 Porque existem 6 "saltos" até ao site Onion, sendo que 3 deles são "relay"? Utilize características do protocolo TOR para justificar.

Os primeiros três endereços são os pontos de introdução que o browser do utilizador pode utilizar para se ligar à rede TOR. Os outros 3 "relays" correspondem a endereços anónimos que reencaminham o tráfego até chegar ao destino final, de modo a anonimizar o utilizador.

3.2.3 Qual é o Rendez-vous Point?

O Rendez-vous Point é o nodo escolhido de entrada na rede. Neste caso é o que se situa em França, com o endereço IPv4 163.172.145.229 e IPv6 2001:bc8:600:e2d::1.

4 Blockchain

4.1 P4.1

Na experiência 4.1, altere o método que cria o Genesis Block, de modo a que o timestamp seja a data do dia de hoje e o dado incluído nesse Bloco seja "Bloco inicial da koreCoin".

Foi modificada a seguinte função para alterar o dado do bloco para "Bloco inicial da koreCoin" e a sua timestamp para a data atual:

```
createGenesisBlock(){
  //return new Block(0, "02/01/2018", "Genesis Block", "0");
  let currentDate = new Date();
  let cDay = currentDate.getDate()
  let cMonth = currentDate.getMonth() + 1
  let cYear = currentDate.getFullYear()
  const dia = cDay+"/"+cMonth+"/"+cYear
  //console.log(dia);

  return new Block(0, dia, "Bloco inicial da koreCoin", "0"); //Mudar para data atual
}
```

Figura 17: Função alterada

4.2 P4.2

Na experiência 4.1, adicione alguns blocos simulando várias transações em cada um deles.

Foram adicionadas as seguintes linhas de código para simular alguns blocos com transações.

```
70 //Adicionar alguns blocos a blockchain simulando transações
71 koreCoin.addBlock(new Block (1, "01/01/2018", {amount: -50}));
72 koreCoin.addBlock(new Block (2, "01/01/2018", {amount: 10}));
73 koreCoin.addBlock(new Block (4, "03/01/2018", {amount: 20}));
74 koreCoin.addBlock(new Block (5, "05/01/2018", {amount: 40}));
75 koreCoin.addBlock(new Block (6, "02/02/2018", {amount: 40}));
76 koreCoin.addBlock(new Block (7, "03/01/2019", {amount: -20}));
```

Figura 18: Função alterada

O ficheiro completo com as alterações pode ser encontrado no github do grupo de trabalho na pasta Ficha-7.

4.3 P4.3

Na experiência 4.3, altere a dificuldade de minerar para 2 e veja qual o tempo que demora, utilizando o comando time do Linux (ou similar no seu sistema operativo), por exemplo `time node main.experiencia2.1.js`. Repita o exemplo para dificuldade de minerar 3, 4 e 5.

Apresente os tempos e conclua sobre os mesmos.

A dificuldade de minar pode ser alterada mudando a seguinte parte do código:

```
class Blockchain{
  constructor(){
    this.chain = [this.createGenesisBlock()];
    this.difficulty = 5;
  }
}
```

Figura 19: Alterar dificuldade de mineração

Com dificuldade 1 obtém-se o seguinte tempo de execução:

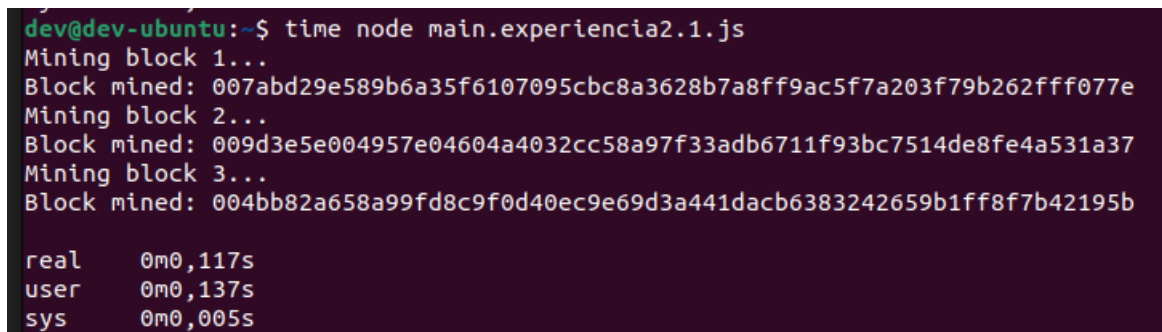
A terminal window titled 'dev@dev-ubuntu: ~' showing the execution of 'time node main.experiencia2.1.js'. The output shows three blocks being mined with their respective hashes. The timing results are: real 0m0,105s, user 0m0,099s, and sys 0m0,013s.

```
dev@dev-ubuntu:~$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 05da09d28f074ba39fe1b6ac16c23e4ffb219a1a0162b05363894d2a998c4cbe
Mining block 2...
Block mined: 07b1c4b6b78f488b209c411502182b1e05cf66716795291d696d6666526b1b4a
Mining block 3...
Block mined: 09799a37e22ee4a3f0daf197ae1d641a02e85277d77448fe09d907d4025e9a48

real    0m0,105s
user    0m0,099s
sys     0m0,013s
dev@dev-ubuntu:~$
```

Figura 20: Dificuldade 1

Com dificuldade 2 obtém-se o seguinte tempo de execução:

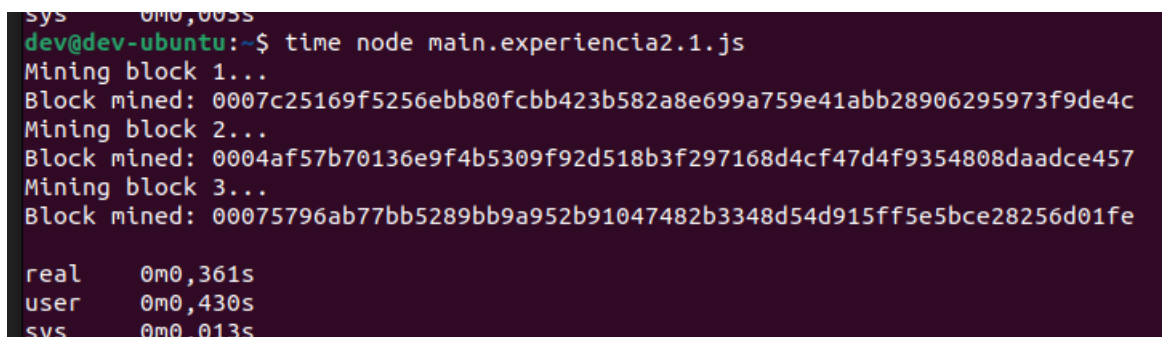
A terminal window titled 'dev@dev-ubuntu: ~' showing the execution of 'time node main.experiencia2.1.js'. The output shows three blocks being mined with their respective hashes. The timing results are: real 0m0,117s, user 0m0,137s, and sys 0m0,005s.

```
dev@dev-ubuntu:~$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 007abd29e589b6a35f6107095cbc8a3628b7a8ff9ac5f7a203f79b262fff077e
Mining block 2...
Block mined: 009d3e5e004957e04604a4032cc58a97f33adb6711f93bc7514de8fe4a531a37
Mining block 3...
Block mined: 004bb82a658a99fd8c9f0d40ec9e69d3a441dacb6383242659b1ff8f7b42195b

real    0m0,117s
user    0m0,137s
sys     0m0,005s
```

Figura 21: Dificuldade 2

Com dificuldade 3 obtém-se o seguinte tempo de execução:

A terminal window titled 'dev@dev-ubuntu: ~' showing the execution of 'time node main.experiencia2.1.js'. The output shows three blocks being mined with their respective hashes. The timing results are: real 0m0,361s, user 0m0,430s, and sys 0m0,013s.

```
dev@dev-ubuntu:~$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 0007c25169f5256ebb80fcbb423b582a8e699a759e41abb28906295973f9de4c
Mining block 2...
Block mined: 0004af57b70136e9f4b5309f92d518b3f297168d4cf47d4f9354808daadce457
Mining block 3...
Block mined: 00075796ab77bb5289bb9a952b91047482b3348d54d915ff5e5bce28256d01fe

real    0m0,361s
user    0m0,430s
sys     0m0,013s
```

Figura 22: Dificuldade 3

Com dificuldade 4 obtém-se o seguinte tempo de execução:

```
dev@dev-ubuntu:~$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 0000023168f87d968813b22c4dc92f60c127ff5084af8487d913d497ea7a7900
Mining block 2...
Block mined: 00008e0c291aaf728e015855328b14e651231cce209e6413503fd299e0df6c5e
Mining block 3...
Block mined: 0000fb4a126ef4c1c3c93bf2ed25e8db4c7da2ec89a46aad4f7bf092afd8b6b4

real    0m1,558s
user    0m1,626s
sys     0m0,075s
```

Figura 23: Dificuldade 4

Com dificuldade 5 obtém-se o seguinte tempo de execução:

```
dev@dev-ubuntu:~$ time node main.experiencia2.1.js
Mining block 1...
Block mined: 0000023168f87d968813b22c4dc92f60c127ff5084af8487d913d497ea7a7900
Mining block 2...
Block mined: 000000b950a180294edddf4340a2d5834119a41bf89e4b2027f341f0fc02365e
Mining block 3...
Block mined: 0000088dc9c3115ee6ab7e95d2e8836f932d75d303ab451a825241acde589a58

real    0m20,374s
user    0m20,640s
sys     0m0,469s
```

Figura 24: Dificuldade 5

O gráfico seguinte expõe a variação do tempo em relação à dificuldade.



Figura 25: Gráfico da variação do nível de dificuldade ao longo do tempo de execução

Podemos ver que com o aumento de dificuldade o tempo necessário para minar cada bloco aumenta exponencialmente tal como é esperado, pois a probabilidade de um minerador descobrir a substring correta diminui.

4.4 P4.4

4.4.1 Na experiência 4.4, qual é o algoritmo de 'proof of work' ?

O algoritmo usado é semelhante ao da *bitcoin*. O objetivo é o utilizador que esteja a minar, tente por tentativa e erro descobrir uma substring que gere uma hash, que comece com X números 0. A quantidade de números 0 depende do nível de dificuldade. Como por exemplo, com um nível de dificuldade **2** a hash tem que começar com dois 0, enquanto que se for de dificuldade **5** a hash tem de começar com cinco 0's.

Exemplos de tentativas: *Try 02345xhash; Try 00345xhash; Try 00375xhash.*

As hash são geradas utilizando o algoritmo SHA256.

4.4.2 Parece-lhe um algoritmo adequado para minerar? Porquê?

Sim, pois ao existir um algoritmo '*proof of work*', este previne que a *blockchain* seja bombardeada de *blocks* todos os segundos. Para além disso, é possível alterar o nível de dificuldade, o que é importante com o avanço da *performance* dos computadores. Existem outros métodos em vez de '*proof of work*' como *Proof-of-Stake* e *Proof-of-Space*, sendo estes mais eficientes em termos de energia gasta.

5 Referências

- <https://whatismyipaddress.com/>
- <https://community.torproject.org/onion-services/overview/>