

Tarefa Prática (Handshake TLS + OpenSSL)

Nome: Bruno Aurélio Rôzza de Moura Campos (14104255)

PARTE 1: Handshake SSL/TLS (Secure Socket Layer/Transport Layer Security)

Questão 1

É possível verificar as possibilidades do SSL/TLS do seu browser e do seu servidor. Cole os resultados (screenshot) aqui e comente o que chamou a sua atenção em cada um dos resultados.

a. <https://www.ssllabs.com/> este site e teste o seu browser (diferentes tipos de browser podem ter resultados diferentes na sua máquina).

b. <https://www.ssllabs.com/> este site e teste um servidor que usa o SSL. Cuide para não acessar apenas um proxy de servidor real.

Obs.: forward secrecy significa que se uma chave for comprometida durante uma sessão, esse conhecimento/fato não afeta a segurança de sessões anteriores. A troca de chaves RSA (RSA key Exchange) não fornece forward secrecy pois se alguma chave privada for comprometida, todo o tráfego anterior pode ser decifrado.

Resposta

a.
User Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.80 Safari/537.36

SSL/TLS Capabilities of Your Browser

[Other User Agents »](#)

User Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.80
Safari/537.36

Protocol Support

Your user agent has good protocol support.

Your user agent supports TLS 1.2, which is recommended protocol version at the moment.
Experimental: Your user agent supports TLS 1.3.

Logjam Vulnerability

Your user agent is not vulnerable.

For more information about the Logjam attack, please go to weakdh.org.
To test manually, click [here](#). Your user agent is not vulnerable if it fails to connect to the site.

FREAK Vulnerability

Your user agent is not vulnerable.

For more information about the FREAK attack, please go to www.freakattack.com.
To test manually, click [here](#). Your user agent is not vulnerable if it fails to connect to the site.

POODLE Vulnerability

Your user agent is not vulnerable.

For more information about the POODLE attack, please read [this blog post](#).

Protocol Features



Protocols

TLS 1.3	Yes
TLS 1.2	Yes
TLS 1.1	Yes
TLS 1.0	Yes
SSL 3	No
SSL 2	No



Cipher Suites (in order of preference)

TLS_GREASE_4A (0x4a4a)	-
TLS_AES_128_GCM_SHA256 (0x1301) Forward Secrecy	128
TLS_AES_256_GCM_SHA384 (0x1302) Forward Secrecy	256
TLS_CHACHA20_POLY1305_SHA256 (0x1303) Forward Secrecy	256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b) Forward Secrecy	128
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) Forward Secrecy	128
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c) Forward Secrecy	256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) Forward Secrecy	256
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xc0a9) Forward Secrecy	256
TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xc0a8) Forward Secrecy	256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013) WEAK	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014) WEAK	256
TLS_RSA_WITH_AES_128_GCM_SHA256 (0x9c) WEAK	128

TLS_RSA_WITH_AES_256_GCM_SHA384 (0x9d) WEAK	256
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f) WEAK	128
TLS_RSA_WITH_AES_256_CBC_SHA (0x35) WEAK	256
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa) WEAK	112

(1) When a browser supports SSL 2, its SSL 2-only suites are shown only on the very first connection to this site. To see the suites, close all browser windows, then open this exact page directly. Don't refresh.



Protocol Details

Server Name Indication (SNI)	Yes
Secure Renegotiation	Yes
TLS compression	No
Session tickets	Yes
OCSP stapling	Yes
Signature algorithms	SHA256/ECDSA, RSA_PSS_SHA256, SHA256/RSA, SHA384/ECDSA, RSA_PSS_SHA384, SHA384/RSA, RSA_PSS_SHA512, SHA512/RSA, SHA1/RSA
Named Groups	tls_grease_aaaa, x25519, secp256r1, secp384r1
Next Protocol Negotiation	No
Application Layer Protocol Negotiation	Yes h2 http/1.1
SSL 2 handshake compatibility	No

Mixed Content Handling



Mixed Content Tests

Images	Passive	Yes
CSS	Active	No
Scripts	Active	No
XMLHttpRequest	Active	No
WebSockets	Active	No
Frames	Active	No

- (1) These tests might cause a mixed content warning in your browser. That's expected.
(2) If you see a failed test, try to reload the page. If the error persists, please get in touch.

Related Functionality

Upgrade Insecure Requests request header (more info)	Yes
--	-----

b.

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [www.secnet.com.br](#)

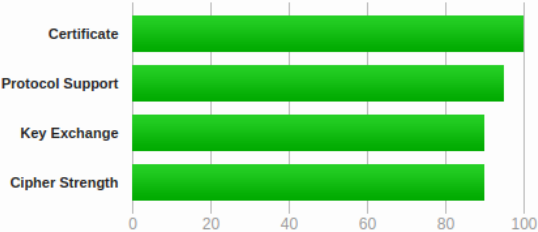
SSL Report: [www.secnet.com.br](#) (50.116.50.47)

Assessed on: Sun, 16 Jun 2019 23:06:54 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)

Summary

Overall Rating



Visit our [documentation page](#) for more information, configuration guides, and books. Known issues are documented [here](#).

Certificate #1: RSA 2048 bits (SHA256withRSA)



Server Key and Certificate #1



Subject	*.secnet.com.br Fingerprint SHA256: 6899dfc1e8ab548ebd9b2b39a9efc8734221f0ef3a20076ee2e4630523200810 Pin SHA256: AHZMZ1PW29g9A09cJYiwOQPnk44WnGP1RoAUzp+Pb00=
Common names	*.secnet.com.br
Alternative names	*.secnet.com.br secnet.com.br
Serial Number	00efbf7246d521fd30266b650140e7e824
Valid from	Fri, 09 Jun 2017 00:00:00 UTC
Valid until	Sun, 05 Jul 2020 23:59:59 UTC (expires in 1 year)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No
Issuer	COMODO RSA Domain Validation Secure Server CA AIA: http://crt.comodoca.com/COMODORSADomainValidationSecureServerCA.crt
Signature algorithm	SHA256withRSA
Extended Validation	No
Certificate Transparency	No
OCSP Must Staple	No
Revocation information	CRL, OCSP CRL: http://crl.comodoca.com/COMODORSADomainValidationSecureServerCA.crl OCSP: http://ocsp.comodoca.com
Revocation status	Good (not revoked)
DNS CAA	No (more info)
Trusted	Yes Mozilla Apple Android Java Windows



Additional Certificates (if supplied)



Certificates provided 4 (5399 bytes)

Chain issues Contains anchor

#2


Subject	COMODO RSA Domain Validation Secure Server CA Fingerprint SHA256: 02ab57e4e67a0cb48dd2ff34830e8ac40f4476fb08ca6be3f5cd846f646840f0 Pin SHA256: kIO23nT2ehFDXCh3eHTDRESMz3asj1muO+4aldjiuY=
Valid until	Sun, 11 Feb 2029 23:59:59 UTC (expires in 9 years and 7 months)
Key	RSA 2048 bits (e 65537)
Issuer	COMODO RSA Certification Authority
Signature algorithm	SHA384withRSA


#3

Subject	COMODO RSA Certification Authority Fingerprint SHA256: 4f32d5dc00f715250abcc486511e37f501a899deb3bf7ea8adbbd3aef1c412da Pin SHA256: grX4Ta9HpZx6tSHkmCrvpApTQGo67CYDnvrLg5yRME=
Valid until	Sat, 30 May 2020 10:48:38 UTC (expires in 11 months and 13 days)
Key	RSA 4096 bits (e 65537)
Issuer	AddTrust External CA Root
Signature algorithm	SHA384withRSA


#4

Subject	AddTrust External CA Root In trust store Fingerprint SHA256: 687fa451382278ff0c8b11f8d43d576671c6eb2bceab413fb83d965d06d2ff2 Pin SHA256: ICppFqbkrlJ3EcVFAkeip0+44VaoJUymbnOaEuk7IEU=
Valid until	Sat, 30 May 2020 10:48:38 UTC (expires in 11 months and 13 days)
Key	RSA 2048 bits (e 65537)
Issuer	AddTrust External CA Root Self-signed
Signature algorithm	SHA1withRSA Weak, but no impact on root certificate




Certification Paths


Mozilla Apple Android **Java** Windows

Path #1: Trusted


1	Sent by server	*.secnet.com.br Fingerprint SHA256: 6899dfc1e8ab548ebd9b2b39a9efc8734221f0ef3a20076ee2e4630523200810 Pin SHA256: AHZMZ1PW29g9A09cJYiwOQPnk44WnGP1RoAUzp+Pb00= RSA 2048 bits (e 65537) / SHA256withRSA
2	Sent by server	COMODO RSA Domain Validation Secure Server CA Fingerprint SHA256: 02ab57e4e67a0cb48dd2ff34830e8ac40f4476fb08ca6be3f5cd846f646840f0 Pin SHA256: kQ23nT2ehFDXCfb3eHTDRESMz3asj1muO+4aldjiuY= RSA 2048 bits (e 65537) / SHA384withRSA
3	In trust store	COMODO RSA Certification Authority Self-signed Fingerprint SHA256: 52f0e1c4e58ec629291b60317f074671b85d7ea80d5b07273463534b32b40234 Pin SHA256: grX4Ta9HpZx6tSHkmCrvpApTQGo67CYDnvpRLg5yRME= RSA 4096 bits (e 65537) / SHA384withRSA

Path #2: Trusted


1	Sent by server	*.secnet.com.br Fingerprint SHA256: 6899dfc1e8ab548ebd9b2b39a9efc8734221f0ef3a20076ee2e4630523200810 Pin SHA256: AHZMZ1PW29g9A09cJYiwOQPnk44WnGP1RoAUzp+Pb00= RSA 2048 bits (e 65537) / SHA256withRSA
2	Sent by server	COMODO RSA Domain Validation Secure Server CA Fingerprint SHA256: 02ab57e4e67a0cb48dd2ff34830e8ac40f4476fb08ca6be3f5cd846f646840f0 Pin SHA256: kQ23nT2ehFDXCfb3eHTDRESMz3asj1muO+4aldjiuY= RSA 2048 bits (e 65537) / SHA384withRSA
3	Sent by server	COMODO RSA Certification Authority Fingerprint SHA256: 4f32d5dc00f715250abcc486511e37f501a899deb3bf7ea8adbbd3aef1c412da Pin SHA256: grX4Ta9HpZx6tSHkmCrvpApTQGo67CYDnvpRLg5yRME= RSA 4096 bits (e 65537) / SHA384withRSA
4	Sent by server In trust store	AddTrust External CA Root Self-signed Fingerprint SHA256: 687fa451382278ff0c8b11fd43d576671c6eb2bceab413fb83d965d06d2ff2 Pin SHA256: lCpPfqbkrlU3EcVFAkeip0+44VaoJUymbnOaEUk7IEU= RSA 2048 bits (e 65537) / SHA1withRSA Weak or insecure signature, but no impact on root certificate

Configuration



Protocols

TLS 1.3	No
TLS 1.2	Yes
TLS 1.1	Yes
TLS 1.0	Yes
SSL 3	No
SSL 2	No

For TLS 1.3 tests, we only support RFC 8446.



Cipher Suites

# TLS 1.2 (suites in server-preferred order)			<input type="checkbox"/>
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)	ECDH secp256r1 (eq. 3072 bits RSA) FS		128
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_RSA_WITH_AES_128_GCM_SHA256 (0x9c)		WEAK	128
TLS_RSA_WITH_AES_128_CBC_SHA256 (0x9c)		WEAK	128
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f)		WEAK	128
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)	ECDH secp256r1 (eq. 3072 bits RSA) FS		256
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_RSA_WITH_AES_256_GCM_SHA384 (0x9d)		WEAK	256
TLS_RSA_WITH_AES_256_CBC_SHA256 (0x3d)		WEAK	256
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)		WEAK	256
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (0xc012)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	112
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa)		WEAK	112
# TLS 1.1 (suites in server-preferred order)			<input type="checkbox"/>
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	128
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f)		WEAK	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	256
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)		WEAK	256
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (0xc012)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	112
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa)		WEAK	112
# TLS 1.0 (suites in server-preferred order)			<input type="checkbox"/>
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	128
TLS_RSA_WITH_AES_128_CBC_SHA (0x2f)		WEAK	128
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	256
TLS_RSA_WITH_AES_256_CBC_SHA (0x35)		WEAK	256
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (0xc012)	ECDH secp256r1 (eq. 3072 bits RSA) FS	WEAK	112
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa)		WEAK	112



Handshake Simulation

Android 2.3.7 <small>No SNI²</small>	RSA 2048 (SHA256)	TLS 1.0	TLS_RSA_WITH_AES_128_CBC_SHA	No FS
Android 4.0.4	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
Android 4.1.1	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
Android 4.2.2	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
Android 4.3	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
Android 4.4.2	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Android 5.0.0	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Android 6.0	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Android 7.0	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Baidu Jan 2015	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
BingPreview Jan 2015	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Chrome 49 / XP SP3	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Chrome 69 / Win 7 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Chrome 70 / Win 10	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Firefox 31.3.0 ESR / Win 7	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Firefox 47 / Win 7 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Firefox 49 / XP SP3	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Firefox 62 / Win 7 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Googlebot Feb 2018	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
IE 7 / Vista	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
IE 8 / XP <small>No FS¹ No SNI²</small>	RSA 2048 (SHA256)	TLS 1.0	TLS_RSA_WITH_3DES_EDE_CBC_SHA	
IE 8-10 / Win 7 <small>R</small>	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
IE 11 / Win 7 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1 FS
IE 11 / Win 8.1 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1 FS
IE 10 / Win Phone 8.0	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
IE 11 / Win Phone 8.1 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1 FS
IE 11 / Win Phone 8.1 Update <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1 FS
IE 11 / Win 10 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Edge 15 / Win 10 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Edge 13 / Win Phone 10 <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
Java 6u45 <small>No SNI²</small>	RSA 2048 (SHA256)	TLS 1.0	TLS_RSA_WITH_AES_128_CBC_SHA	No FS
Java 7u25	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1 FS
Java 8u161	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
OpenSSL 0.9.8y	RSA 2048 (SHA256)	TLS 1.0	TLS_RSA_WITH_AES_128_CBC_SHA	No FS
OpenSSL 1.0.1j <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS
OpenSSL 1.0.2e <small>R</small>	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1 FS

Safari 5.1.9 / OS X 10.6.8	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1	FS
Safari 6 / iOS 6.0.1	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1	FS
Safari 6.0.4 / OS X 10.8.4 R	RSA 2048 (SHA256)	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	ECDH secp256r1	FS
Safari 7 / iOS 7.1 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1	FS
Safari 7 / OS X 10.9 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1	FS
Safari 8 / iOS 8.4 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1	FS
Safari 8 / OS X 10.10 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	ECDH secp256r1	FS
Safari 9 / iOS 9 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
Safari 9 / OS X 10.11 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
Safari 10 / iOS 10 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
Safari 10 / OS X 10.12 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
Apple ATS 9 / iOS 9 R	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
Yahoo Slurp Jan 2015	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS
YandexBot Jan 2015	RSA 2048 (SHA256)	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	ECDH secp256r1	FS

Not simulated clients (Protocol mismatch)




IE 6 / XP **No FS** ¹ **No SNI** ² Protocol mismatch (not simulated)

- (1) Clients that do not support Forward Secrecy (FS) are excluded when determining support for it.
 (2) No support for virtual SSL hosting (SNI). Connects to the default site if the server uses SNI.
 (3) Only first connection attempt simulated. Browsers sometimes retry with a lower protocol version.
 (R) Denotes a reference browser or client, with which we expect better effective security.
 (All) We use defaults, but some platforms do not use their best protocols and features (e.g., Java 6 & 7, older IE).
 (All) Certificate trust is not checked in handshake simulation, we only perform TLS handshake.



Protocol Details

	No, server keys and hostname not seen elsewhere with SSLv2 (1) For a better understanding of this test, please read this longer explanation (2) Key usage data kindly provided by the Censys network search engine; original DROWN website here (3) Censys data is only indicative of possible key and certificate reuse; possibly out-of-date and not complete
DROWN	
Secure Renegotiation	Supported
Secure Client-Initiated Renegotiation	No
Insecure Client-Initiated Renegotiation	No
BEAST attack	Not mitigated server-side (more info) TLS 1.0: 0xc013
POODLE (SSLv3)	No, SSL 3 not supported (more info)
POODLE (TLS)	No (more info)
Zombie POODLE	No (more info) TLS 1.2: 0xc027
GOLDENDOODLE	No (more info) TLS 1.2: 0xc027
OpenSSL 0-Length	No (more info) TLS 1.2: 0xc027
Sleeping POODLE	No (more info) TLS 1.2: 0xc027
Downgrade attack prevention	Yes, TLS_FALLBACK_SCSV supported (more info)
SSL/TLS compression	No
RC4	No
Heartbeat (extension)	Yes
Heartbleed (vulnerability)	No (more info)
Ticketbleed (vulnerability)	No (more info)
OpenSSL CCS vuln. (CVE-2014-0224)	No (more info)
OpenSSL Padding Oracle vuln. (CVE-2016-2107)	No (more info)
ROBOT (vulnerability)	No (more info)
Forward Secrecy	With modern browsers (more info)
ALPN	No
NPN	No
Session resumption (caching)	Yes
Session resumption (tickets)	Yes
OCSP stapling	Yes
Strict Transport Security (HSTS)	No
HSTS Preloading	Not in: Chrome Edge Firefox IE
Public Key Pinning (HPKP)	No (more info)
Public Key Pinning Report-Only	No
Public Key Pinning (Static)	No (more info)
Long handshake intolerance	No
TLS extension intolerance	No
TLS version intolerance	No
Incorrect SNI alerts	No
Uses common DH primes	No, DHE suites not supported
DH public server param (Ys) reuse	No, DHE suites not supported
ECDH public server param reuse	No
Supported Named Groups	secp256r1, secp521r1, secp384r1, secp256k1 (server preferred order)
SSL 2 handshake compatibility	Yes

 HTTP Requests 	
1 https://www.secnnet.com.br/ (HTTP/1.1 200 OK)	
1	Date
	Sun, 16 Jun 2019 23:04:26 GMT
	Server
	Apache
	Accept-Ranges
	bytes
	Vary
	Accept-Encoding,User-Agent
	X-Mod-Pagespeed
	1.13.35.2-0
	Cache-Control
	max-age=0, no-cache
	X-Server
	High Performance Servers - www.secnnet.com.br
	Referrer-Policy
	Pragma
	public
	Content-Length
	219585
	Connection
	close
	Content-Type
	text/html; charset=UTF-8
 Miscellaneous	
	Test date
	Sun, 16 Jun 2019 23:04:49 UTC
	Test duration
	124.894 seconds
	HTTP status code
	200
	HTTP server signature
	Apache
	Server hostname
	cloud.secnnet.host

2. Questão

Leia as recomendações da página <https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices> e faça um pequeno resumo das seções 1 e 2 dessas recomendações.

Resposta

Chave privada e certificado

O TLS começa com a identificação criptográfica do servidor. Para isso, é usado uma chave privada forte afim de evitar ataques de falsificação de identidade. Para garantir a segurança, há algumas dicas como:

- **Use chaves particulares de 2048 bits:** Para grande parte dos sites, chaves RSA de 2048 já são o suficiente.
- **Proteger Chaves Privadas:** Restringindo o acesso, gerando chaves com entropia suficiente.
- **Garantir cobertura suficiente de Hostname:** É uma forma de garantir que todas as rotas estão acessíveis e evitar avisos de certificados inválidos.
- **Obter certificados de uma CA confiável:** Isso é, escolher uma Autoridade de Certificação (CA) que seja confiável e séria. Para escolher uma CA, deve-se levar em consideração:
 - **Postura de segurança:** uma opção é examinar seu histórico de segurança.
 - **As CAs com foco nos negócios:** cujas atividades constituem uma parte substancial de seus negócios, têm tudo a perder se algo der errado

- **Serviços oferecidos:** No mínimo, sua AC selecionada deve fornecer suporte para os métodos de revogação da Lista de Revogação de Certificados (CRL) e do Protocolo de Status de Certificados Online (OCSP), com disponibilidade e desempenho de rede sólidos.
- **Opções de gerenciamento de certificados** se for necessário operar um grande número de certificados e operar em um ambiente complexo, escolha uma autoridade de certificação que ofereça boas ferramentas para gerenciá-los.
- **Suporte** é uma tranquilidade ter um bom suporte.
- **Use Algoritmos de Assinatura de Certificado Forte:** A segurança do certificado depende (1) da força da chave privada que foi usada para assinar o certificado e (2) da força da função de hash usada na assinatura.

Configuração

É uma garantia que as credenciais sejam apresentadas corretamente aos visitantes do site. Há uma série de medidas para ser levado em conta:

- **Use protocolos seguros:** Na maioria das implantações, o certificado do servidor sozinho é insuficiente; Dois ou mais certificados são necessários para construir uma cadeia completa de confiança.
- **Use Conjuntos de Codificação Segura** Existem cinco protocolos na família SSL / TLS: SSL v2, SSL v3, TLS v1.0, TLS v1.1 e TLS v1.2:
 - O SSL v2 é inseguro e não deve ser usado.
 - O SSL v3 é inseguro quando usado com HTTP (o ataque POODLE).
 - O TLS v1.0 também é um protocolo legado que não deve ser usado.
 - O TLS v1.1 e v1.2 são ambos sem problemas de segurança conhecidos
 - O TLS v1.2 deve ser seu protocolo principal porque é a única versão que oferece criptografia autenticada moderna
- **Use Conjuntos de Codificação Segura:** Em SSL e TLS, os conjuntos de criptografia definem como a comunicação segura ocorre. Eles são compostos de diferentes blocos de construção com a idéia de alcançar a segurança através da diversidade. Se um dos blocos de construção for fraco ou inseguro, é possível mudar para outro.
- **Selecione as melhores suítes de codificação:** Ter servidores selecionando ativamente o melhor conjunto de criptografia disponível é fundamental para obter a melhor segurança.
- **Use o sigilo antecipado:** O sigilo de encaminhamento (às vezes também chamado de sigilo de encaminhamento perfeito) é um recurso de protocolo que permite conversas seguras que não dependem da chave privada do servidor.
- **Use troca de chaves forte:** Para a troca de chaves, os sites públicos normalmente podem escolher entre a troca de chaves Diffie-Hellman (DHE) efêmera clássica e sua variante de curva elíptica, ECDHE.
- **Mitigar Problemas Conhecidos:** Nada é perfeitamente seguro, e é por isso que é uma boa prática ficar de olho no que acontece na segurança. Aplique prontamente correções de fornecedores se e

quando elas estiverem disponíveis; caso contrário, confie em soluções alternativas para mitigação.

3. Questão

Explique de forma geral as quatro fases do handshake de acordo com as páginas do livro do Stallings 386, 387, 388 e 389 (o pdf do capítulo está anexado junto na tarefa).

Resposta

- 1. Estabelecer capacidades de segurança:** É a fase que inicia a comunicação. O cliente envia mensagem contendo alguns parâmetros:
 - versão do SSL
 - ID da sessão
 - Conjunto de cifras (cipherSuite) - é uma lista contendo algoritmo de troca de chave. Por exemplo, RSA, Diffie-Hellman, Diffie-Hellman anônimo Fortezza
 - Método de compactação(compression method) - é uma lista dos métodos de compactação que o cliente admite Em seguida, o cliente aguarda a resposta do servidor.
- 2. Autenticação de servidor e troca de chaves:** Nesta etapa, o servidor encaminha seus certificados, se necessário autenticar. A mensagem de certificado é exigida para qualquer troca de chaves que tenham sido acordadas, exceto se for Diffie-Hellman anônimo. A mensagem final desta fase é sempre exigida, é uma mensagem `server_done` enviada pelo servidor para indicar o final da resposta dele. Em seguida, o servidor aguardará uma resposta do cliente
- 3. Autenticação do cliente e troca de chaves:** Quando o cliente recebe uma mensagem `server_done` ele verifica se o certificado é válido e se os parâmetros `server_hello` são aceitáveis. Se tudo ok, o cliente responde seja com uma mensagem `certificate` ou `no_certificate`. Na sequência, é recebido uma mensagem `client_key_exchange` contendo algum conjunto de cifras:
 - RSA
 - Diffie-Hellman anônimo ou efêmero
 - Diffie-Hellman fixo
 - Fortezza No fim desta fase, o cliente pode enviar uma mensagem `certificate_verify` para oferecer uma certificação explícita de um certificado. Contudo, essa mensagem só é enviada após qualquer certificado que tenha capacidade de assinatura, ou seja qualquer certificado menos Diffie-Hellman fixo
- 4. Término:** Na última etapa, o cliente envia uma mensagem `change_cipher_spec`. Cabe notar que essa mensagem não é considerada parte do protocolo de estabelecimento de sessão mas sim enviada usando o protocolo de mudança de especificação de cifra. Além da mensagem anterior, o cliente encaminha a mensagem `finished_message` sob os novos algoritmos, chaves e segredos. Em resposta, o servidor envia sua mensagem `change_cipher_spec`, transfere o CipherSpec pendente para o atual e envia sua `finished_message`. A partir daqui, o cliente e servidor podem trocar dados na camada de aplicação.

4. Questão - handshake

Observando o handshake dos protocolos TLS v1.2 e TLS v1.3 através da leitura do material dos seguintes sites e da observação das figuras, cite pelo menos 3 diferenças do handshake entre as duas versões.

1. Melhora de desempenho
2. Remoção de SHA1, MD5
3. Adição de Assinatura de handshake completa

```
openssl s_client -connect www.ufsc.br:443 -tls1_2
openssl s_client -connect youtube.com:443 -tls1_3
```

```
CONNECTED(00000003)
depth=2 OU = GlobalSign Root CA - R3, O = GlobalSign, CN = GlobalSign
verify return:1
depth=1 C = BE, O = GlobalSign nv-sa, CN = GlobalSign RSA OV SSL CA 2018
verify return:1
depth=0 C = BR, ST = SC, L = Florianopolis, O = UNIVERSIDADE FEDERAL DE
SANTA CATARINA, CN = *.ufsc.br
verify return:1
---
Certificate chain
 0 s:C = BR, ST = SC, L = Florianopolis, O = UNIVERSIDADE FEDERAL DE SANTA
CATARINA, CN = *.ufsc.br
  i:C = BE, O = GlobalSign nv-sa, CN = GlobalSign RSA OV SSL CA 2018
 1 s:C = BE, O = GlobalSign nv-sa, CN = GlobalSign RSA OV SSL CA 2018
  i:OU = GlobalSign Root CA - R3, O = GlobalSign, CN = GlobalSign
 2 s:OU = GlobalSign Root CA - R3, O = GlobalSign, CN = GlobalSign
  i:OU = GlobalSign Root CA - R3, O = GlobalSign, CN = GlobalSign
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIGmTCCBYGgAwIBAgIMZgvf9yb37PeNbiMrMA0GCSqGSIb3DQEBCwUAMFAxCzAJ
BgNVBAYTAKJFMRkwFwYDVQQKEXBHbG9iYWxTaWduIG52LXNhMSYwJAYDVQQDEx1H
bG9iYWxTaWduIFJ1TQSBPVlBTU0wgQ0EgMjAxODAEwMDA3MTAyMjIxMDJhFw0y
MjA3MTEyMjIxMDJhMhcxZCZAJBgNVBAYTAKJSMQswCQYDVQQIEwJ1TQZEWMBQGA1UE
BxMNRmxvcmlhbm9wb2xpczEvMC0GA1UEChMmVU5JVkVSU0lEQURFIEZFREVSQUwg
REUgU0FOVEEGQ0FUQVJ1TTEkExEjAQBgNVBAMMCSoudWZzYy5icjCCASIwDQYJKoZI
hvcNAQEBBQADggEPADCCAQoCggEBAPCZzjPXRpDab2nMZXDmxXgPMrT7UyZ0n1kt
ewJ32DfYNhcDjcjgpm5FZENCMC/SI0Zg+ol8Bt5Ce6QpE+XPSHdGa/L6PChkdviK
gmzQYHYDIz0jKnFDWNCuniyttaLnaE+R+Vei1aoZBZxVFila6hC84xTbfy5FLzRZ
zKGjH6ZMTyWCRdu+Dxu84kKJ2Htftp7nKFHjKpa5tQFjCrZdLwPMpCcLKUxz+0+Ww0
eXB/old+8u/hxIzy1/IicMmW/swF01qGXqRdlehKIndlRh8geVn/MXlQWRafp9vW
8k0RajkfUzBacSQz3Y33YBSrfq8upNXRro1SgLz9rdXm7hIYs68CAwEAAa0CA0ow
```

```

ggNGMA4GA1UdDwEB/wQEAwIFoDCBjgYIKwYBBQUHAQEegYEWfzBEBggrBgEFBQcw
AoY4aHR0cDovL3NlY3VyZS5nbG9iYWxzawduLmNvbS9jYWNlcnQvZ3Nyc2FvbnNz
bGNhMjAxOC5jcncQWwYIKwYBBQUHMAGGK2h0dHA6Ly9vY3NwLmdsb2JhbHNpZ24u
Y29tL2dzcncNhb3Zzc2xjYTIwMTgwVgYDVR0gBE8wTTBBBgkrBgEEAaAyARQWNDAY
BggrBgEFBQcCARYmaHR0cHM6Ly93d3cuZ2xvYmFsc2lnbi5jb20vcnVvb3NpdG9y
eS8wCAYGZ4EMAQICMAKGA1UdEwQCAAwPwYDVR0fBDgwNjA0oDKgMIYuaHR0cDov
L2Nybc5nbG9iYWxzawduLmNvbS9nc3JzYW92c3NsY2EyMDE4LmNybdAdBgNVHREE
FjAUGgkqLnVmc2MuYnKCB3Vmc2MuYnIwHQYDVR0lBBYwFAYIKwYBBQUHAwEGCCsG
AQUFBwMCMB8GA1UdIwQYMBaAFPjvf/LNeGeo3m+PJI2I8YcDARPrMB0GA1UdDgQW
BBQ3u1t6DrFB+gG2GuF0sKUTEiYyjdCCAX8GCisGAQQB1nkCBAIEggFvBIIBawFp
AHcAb1N2rDHwMRnYmQCkURX/dxUcEdkCwQApBo2yCJo32RMAAAFz0tFbLgAABAMA
SDBGAIeA2SarecMZHJxPFT1tw9kCqLqL08KJQv1XhpuVkG6BaocCIQDaJL0nXuS0
du/68MkiuhtqldjIDQwCdqb2CE0f3Kc2VAB2ACl5vvCe0Tkh8FZzn20ld+W+V32c
YAr4+U1dJlwlXceEAAABczrRXlsAAAQDAECwRQIgU+YTrwmH+g07xYeQBjS0cUuH
36u00zt26cfi3wkDAfgCIQCuWd33/RW/081zC4epKfDe8ecF6+L8B5k/e5ujbZ7t
SgB2AFGjsPX9AXmcVm24N3iPDKR6zBsny/eeiEKaDf7UiwXlAAABczrRXeMAAAQD
AECwRQIhAMLC0vHWI90ZuA+iRjkhdmE7Vw11kD0yjhCrIBbrTqLJAiBw/q8ebNiv
AxjCPdAXlqV4s3Uowoh7fJsPoJM/iaFKYDANBgkqhkiG9w0BAQsFAAOCACEABv+H
4wgIYvB5ML05izXZ47P0ipwGSjR/xHpQ1aYAQhjlZMPkX84h8tsms/3cT5BxD9mK
gm9D8fCCUSZ3ya4Cztfxkg3qZBhEVSUB75DrH2lmgd2BsLLdtaf3ZR5N/OQbtHei
mw1hg3tkmLwvKfkmPhmUilVX4U0kBPXoVMukB9jeksXsDaLf0iGllLaEnDG20bizQ
emtLAX1yTvvXMrSMwKkKeRiR17q0hz6LoGEi5mM5IS9VqB/R7Vr1i5HmpplLgc4t
bKIAU9cZ/czuUHSij75iNmWHxgUfWnw7kE+zIUL7VKGr1Z+a2bV0e1A2QKb5guKV
mAFqJ5do4hLMU5k26g==

```

-----END CERTIFICATE-----

subject=C = BR, ST = SC, L = Florianopolis, O = UNIVERSIDADE FEDERAL DE SANTA CATARINA, CN = *.ufsc.br

issuer=C = BE, O = GlobalSign nv-sa, CN = GlobalSign RSA OV SSL CA 2018

No client certificate CA names sent

Peer signing digest: SHA256

Peer signature type: RSA

Server Temp Key: ECDH, P-256, 256 bits

SSL handshake has read 4362 bytes and written 340 bytes

Verification: OK

New, TLSv1.2, Cipher is ECDHE-RSA-AES256-GCM-SHA384

Server public key is 2048 bit

Secure Renegotiation IS supported

Compression: NONE

Expansion: NONE

No ALPN negotiated

SSL-Session:

Protocol : TLSv1.2

Cipher : ECDHE-RSA-AES256-GCM-SHA384

Session-ID:

D86C4D885B3CEB13BC42E56CC5439E2D438EDD5D79770C6EB2EAD7FD891EE1F2

Session-ID-ctx:

Master-Key:

616FA86D2DAF2CAAB93C721622CEB024533D30AF4ACC37A51236DBC245301AC1693C68F6B5
BB7FE8AD839BD11EA3865

```

PSK identity: None
PSK identity hint: None
SRP username: None
TLS session ticket lifetime hint: 300 (seconds)
TLS session ticket:
0000 - 28 a3 b4 86 46 f7 cc a6-54 9d 3a 80 cb 5a 1d 3b
(...F...T:...Z.;
0010 - 45 74 71 56 3a 87 5b 5c-2c 48 6e 37 fd d1 e7 4f   EtqV:.
[\,Hn7...0
0020 - 02 47 0f 4c 7f 46 ef fb-b6 8d 46 3d 38 e4 00 47
.G.L.F....F=8..G
0030 - 3d af d4 da f2 81 16 d4-8a 34 d3 b5 b1 9f a7 81
=.....4.....
0040 - dc 54 1d c9 a8 10 69 fb-d3 15 6c f4 56 14 4c 92
.T....i...l.V.L.
0050 - 24 09 9c d0 d8 5e 6b 79-20 60 d0 0b e6 7f d6 07   $....^ky
`.....
0060 - 8c 11 06 be 90 3a 8d 50-35 b9 0c bf cc 1a ac 71
.....:P5.....q
0070 - 67 17 80 55 79 bd 77 80-0d 09 b4 9a 32 4f e7 f0
g..Uy.w.....20..
0080 - 35 be f7 19 c5 84 bd b6-78 1c 86 9a c7 a2 0b 26
5.....x.....&
0090 - 04 25 8c 26 9b 34 29 d3-5d 9e d5 c2 e0 5f 0c d8
.%.&.4).]...._..
00a0 - ce e8 52 6d d5 ed 7f 20-cc b1 f1 62 df 22 0d 5c   ..Rm...
...b.".\
00b0 - 5b 50 8f b1 c9 ed 3a 25-a9 6b 31 ca bc bd e8 bc
[P....:%.k1.....

Start Time: 1604879949
Timeout    : 7200 (sec)
Verify return code: 0 (ok)
Extended master secret: no
---
closed

```

2. Parte: `openssl s_client -connect youtube.com:443 -tls1_3`

```

└─ openssl s_client -connect youtube.com:443 -tls1_3

CONNECTED(00000003)
depth=2 OU = GlobalSign Root CA - R2, O = GlobalSign, CN = GlobalSign
verify return:1
depth=1 C = US, O = Google Trust Services, CN = GTS CA 101
verify return:1
depth=0 C = US, ST = California, L = Mountain View, O = Google LLC, CN =
*.google.com
verify return:1
---
Certificate chain

```



```
0 s:C = US, ST = California, L = Mountain View, O = Google LLC, CN =
*.google.com
  i:C = US, O = Google Trust Services, CN = GTS CA 101
1 s:C = US, O = Google Trust Services, CN = GTS CA 101
  i:OU = GlobalSign Root CA - R2, O = GlobalSign, CN = GlobalSign
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIJcDCCCFigAwIBAgIRAJEUCUm/106mAgAAAAB/FD0wDQYJKoZIhvcNAQELBQAw
QjELMAKGA1UEBhMCVVMxHjAcBgNVBAOTFudvb2dsZSBUcnVzdCBTZXJ2aWNlczET
MBEGA1UEAxMKR1RTIENBIDFPMTAeFw0yMDEwMjAxODAzMjhaFw0yMTAxMTIxODAz
MjhaMGYxCzAJBgNVBAYTAlVTMRMwEQYDVQKIIEwPDYXpZm9ybmlhMRYwFAYDVQQH
Ew1Nb3VudGFpbWV3MRMwEQYDVQKKEwPHb29nbGUgTEExDMRUwEwYDVQQDDAwq
Lmdvb2dsZS5jb20wWTATBgqhkJOPQIBBgqhkJOPQMBBwNCAAQ+BGdFp7krslU
hY3Xcy7QwWmBe6ldsDUSufEVS79SiNE50r1UoK91Mm9pUxfR0zBt9T5jwM2U0lBO
ONFF3nMdo4IHBjCCBwIwDgYDVR0PAAQH/BAQDAgeAMBGA1UdJQMMAoGCCsGAQUF
BwMBMAwGA1UdEwEB/wQCMAAwHQYDVRO0BBYEFMC/keyyTMG1w7rwKubzCEm6+5DI
MB8GA1UdIwQYMBAAAFJjR+G4Q68+b7GCFGJAbo0t9Cf0rMGgGCCsGAQUFBwEBBFw
WjArBggrBgEFBQcwAYYfaHR0cDovL29jc3AucGtpLmdvb2cvZ3RzMW8xY29yZTAr
BggrBgEFBQcwAoYfaHR0cDovL3BraS5nb29nL2dzcjIvR1RTMU8xLmNydDCCBMIG
A1UdEQSCBLkwgGS1ggwqLmdvb2dsZS5jb22CDSouYW5kcm9pZC5jb22CFiouYXBw
ZW5naW5lLmdvb2dsZS5jb22CCSouYmRuLmRldoISKi5jbG91ZC5nb29nbGUuY29t
ghgqLmNyb3dkc291cmNlLmdvb2dsZS5jb22CGCouZGF0YWNvbXB1dGUuZ29vZ2xl
LmNvbYIGKi5nLmNvgg4qLmdjcC5ndnQyLmNvbYIRKi5nY3BjZG4uZ3Z0MS5jb22C
CiouZ2dwaHQyY26CDiouZ2tly25hcHBzLmNughYqLmdvb2dsZS1hbmFseXRpY3Mu
Y29tggsqLmdvb2dsZS5jYYILKi5nb29nbGUuY2yCDiouZ29vZ2xlLmNvLmLugg4q
Lmdvb2dsZS5jby5qcII0Ki5nb29nbGUuY28udWuCDyouZ29vZ2xlLmNvbS5hcoIP
Ki5nb29nbGUuY29tLmF1gg8qLmdvb2dsZS5jb20uYnKCyouZ29vZ2xlLmNvbS5j
b4IPKi5nb29nbGUuY29tLm14gg8qLmdvb2dsZS5jb20udHKCDyouZ29vZ2xlLmNv
bS52boILKi5nb29nbGUuZGWCyouZ29vZ2xlLmVzggqLmdvb2dsZS5mcoILKi5n
b29nbGUuHwCCyouZ29vZ2xlLmL0ggqLmdvb2dsZS5ubIILKi5nb29nbGUuGyC
CyouZ29vZ2xlLmB0ghIqLmdvb2dsZWFKYXBpcy5jb22CDyouZ29vZ2xlYXBpcy5j
boIRKi5nb29nbGVjbmFwcHMuy26CFCouZ29vZ2xlY29tbWVyY2UuY29tghEqLmdv
b2dsZXZpZGVvLmNvbYIMKi5nc3Rh dGllmNugg0qLmdzdGF0aWwMuY29tghIqLmdz
dGF0aWwNjbmFwcHMuy26CCiouZ3Z0MS5jb22CCiouZ3Z0Mi5jb22CFCoubWV0cmllj
LmdzdGF0aWwMuY29tggsqLmNvY2hpbj5jb22CECoudXJsLmdvb2dsZS5jb22CEyou
d2Vhci5na2VjbmFwcHMuy26CFiouew91dHVizS1ub2Nvb2tpZS5jb22CDSouew91
dHVizS5jb22CFiouew91dHVizWVkdWNhdGlvbi5jb22CESouew91dHVizWtpZHMuy
29tggsqLnl0LmJlggsqLnl0aw1nLmNvbYIaYw5kcm9pZC5jbGllbnRzLmdvb2ds
ZS5jb22CC2FuZHZJvawQuY29tghtkZXZlbG9wZXIuYw5kcm9pZC5nb29nbGUuY26C
HGRldmVsb3BlcnMuYw5kcm9pZC5nb29nbGUuY26CBGcuY2+CCGdncGh0LmNuggxn
a2VjbmFwcHMuy26CBmdvby5nbIIUZZ29vZ2xlLWFuYw5dGllcy5jb22CCmdvb2ds
ZS5jb22CD2dvb2dsZWnuYXBwcy5jboISZ29vZ2xlY29tbWVyY2UuY29tghhzb3Vy
Y2UuYw5kcm9pZC5nb29nbGUuY26CCnVyY2hpbj5jb22CCnd3dy5nb28uZ2yCCHlv
dXR1LmJlggt5b3V0dWJlLmNvbYIUew91dHVizWVkdWNhdGlvbi5jb22CD3lvdXR1
YmVrawRzLmNvbYIFeXQuYmUwIQYDVR0gBBowGDAIBgZngQwBAGIwDAYKKwYBBAHW
eQIFAzAzBgNVHR8ELDAqMCigJqAkhiJodHRwOi8vY3JsLnBraS5nb29nL0dUuzFP
MWNvcmUuY3JsMIIBAwwYKKwYBBAHwEQUIEAgSB9ASB8QDVAHUA9lyUL9F3MCIUVBGI
MJRWjuNNEkxkv98MLyALzE7xZOMAAAF1R2UwdQAABAMARjBEAiAvemiwbXx8bLnF
EEfS6yRmhNqvjuWrSZ2QAkGI27T0uwIguD0Y4rjIp30kd2fJtPC8ED9b0tzQPnqo
FGldkqVENT8AdgDuwJXujXJkD5Ljw7kbxxKjaWoJe0tqGhQ45keyy+3F+QAAAXVH
ZRZLAAAEAwBHMEUCID8ITStQXlyiH+djv8B0RQsfC8yGQ7jkLMMeky0kvAriAiEA
mr2pzu/QTUfyvWBe9ELte0mWnk9nbHdBf6dW2sB08gAwDQYJKoZIhvcNAQELBQAD
ggEBAG2u6W8QxVea1QnAjEtL9prL+H8XDppBdIxmvjTLJSQcxlyDadYnUW9CX5gl
```

```

9DdXFJ0yqkQwXB5ZFGLNRVLopQGIHbg8lizTic+M5wGqH6F/SuliMJ+gnnYePili
MFy8MV1Bh95rq+VFIY4XNQxVsUwdVEL0s55G6YvqbiZhCnowySqDRxzEtJYKMdnb
3XD7AIjZyt6IUwTD25Fvctq23hKNsYblBooNR/xKvSUB1U2As8WLLR43Nwn7v0dc
aeAYNXPLZlknvExzYds5Tv5TWulzVpxwXyzthXHN46sxRZghM0WurfKd7a0f8w0o
g22zGb9/UgJGjXmnkBnyzdkYfWI=
-----END CERTIFICATE-----
subject=C = US, ST = California, L = Mountain View, O = Google LLC, CN =
*.google.com

issuer=C = US, O = Google Trust Services, CN = GTS CA 101

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 3833 bytes and written 315 bytes
Verification: OK
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Server public key is 256 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)
---
read:errno=0

```

OpenSSL - GERAR PAR DE CHAVES RSA e entender seus componentes

7.

a. Gerar sua chave privada usando o comando:

```
openssl genrsa -aes256-out seunome.privada.pem 2048
```

b) Explique o que é o parâmetro -aes256 do comando.

Resposta

É a cifra utilizada.

c) Explique o que significa a seguinte linha do arquivo seunome.privada.pem

Resposta

A linha que começa com DEK-Info contém dois valores separados por vírgula: o nome do algoritmo de criptografia usado por EVP_get_cipherbyname() e um vetor de inicialização usado pela cifra codificada como um conjunto de dígitos hexadecimais.

8. Gere a chave pública a partir da chave privada com os comandos abaixo (guardar a chave pública no arquivo seunome.publica.pem). Explique a saída obtida em cada um dos comandos. Guarde o arquivo gerado e envie o arquivo da sua chave pública junto nas respostas da tarefa.

```
openssl rsa -in brunocampos.privada.pem -pubout -out  
brunocampos.publica.pem
```

Saída

```
└─ cat brunocampos.publica.pem  
-----BEGIN PUBLIC KEY-----  
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAmE+L08Hk80Ckj6ZBAnW  
hhyATNqAd3GjcIN7Yt/oB1WPgPdabGMZmuIkFx0SG7Wb6+esAfIyGYNNN72F5fNr  
+c132DZpcBCmknw+FK0jztdBIRSB46hrKpvvorrX2MGDm3gwaSrJMXjiWApwZtKt  
t0e1kSN5isoJHayCT4ICsNKUgnAc8kENVLR7B+Yd9D+JUXboh9gS+/1P00Qd0fNI  
piA0z32f8fnTe06S0CMRgisyl/9l+T7VwbFWzjFWqLouQ4boky5HhGoWzzijDP1I  
WxGTGz3DEXJV1lhmvYfcUeShFGeEmG5aZ1eg06G6PsThRA/K6SZeNjLk0uuYe2PV  
CQIDAQAB  
-----END PUBLIC KEY-----
```

O comando retorna um arquivo contendo uma chave pública, gerada a partir de uma chave privada.

9. Digite o seguinte comando e depois abra o arquivo seunome.publica.componentes. Explique os componentes que constam nesse arquivo(<https://tools.ietf.org/html/rfc3447#appendix-A>). Comando:

```
openssl rsa -in brunocampos.privada.pem -out  
brunocampos.publica.componentes -text -noout
```

Resposta

Este arquivo traz a sintaxe de chave privada RSA

Uma chave privada RSA deve ser representada com o tipo ASN.1
RSAPrivateKey:

```
RSAPrivateKey ::= SEQUENCE {  
    versão versão,  
    módulo INTEGER, - n  
    publicExponent INTEGER, - e
```

```

privateExponent INTEGER, - d
prime1 INTEGER, - p
prime2 INTEGER, - q
expoente1 INTEIRO, - mod d (p-1)
expoente2 INTEIRO, - mod d (q-1)
coeficiente INTEGER, - (inverso de q) mod p
otherPrimeInfos OtherPrimeInfos OPTIONAL
}

```

Os campos do tipo RSAPrivateKey têm os seguintes significados:

- versão é o número da versão, para compatibilidade com o futuro revisões deste documento. Deve ser 0 para esta versão do documento, a menos que multi-prime seja usado, caso em que deve ser 1.

```

Versão ::= INTEIRO {dois primos (0), multi (1)}
          (RESTRINGIDA POR
           {- a versão deve ser múltipla se houver otherPrimeInfos -})

```

- módulo é o módulo RSA n.
- publicExponent é o expoente público RSA e.
- privateExponent é o expoente privado RSA d.
- prime1 é o fator principal p de n.
- prime2 é o fator principal q de n.
- expoente1 é $d \bmod (p - 1)$.
- expoente2 é $d \bmod (q - 1)$.
- coeficiente é o coeficiente CRT $q^{-1} \bmod p$.
- otherPrimeInfos contém as informações para os primos adicionais r_3, \dots, r_u , em ordem. Deve ser omitido se a versão for 0 e deve conter pelo menos uma instância de OtherPrimeInfo se a versão é 1.

ASSINATURA DIGITAL

10.

a. Você deve assinar o arquivo fornecido na tarefa (msgPlana.txt). Para isso, crie o hash do arquivo msgPlana.txt e com a sua chave privada, assine o hash do arquivo

```

openssl dgst -sha256 -sign brunocampos.privada.pem -out assinatura
msgPlana.txt

```

b) Responda: qual o conteúdo do arquivo assinatura? Essa assinatura garante quais características de segurança: integridade, autenticidade, confidencialidade?

Resposta

```
[campos][avell][±][master U:7 ?:3 X][~/.../trabalho_SSL/2020] 22:06:01
■ cat assinatura
g_z aat S=VDoHqU~5[9_3/0ekS; 9(r/b/4h3Ogi~iDbCB50\Z]*,,?F=ky
J++t<[C+cnKu;If$#e<%|e; WxG
+++++3Fte|+++++"eTe,Ie
F++++1f+++17F+++U:7 ?:3 XF / (trabalho_SSL/2020]
```

- É um conteúdo ilegível
- Garante integridade

11. Verifique se o hash assinado está ok, isto é, compare o hash assinado com o hash do arquivo original usando o comando abaixo. Envie sua chave pública para que, durante a correção, possa ser feita a verificação da sua assinatura:

```
openssl dgst -sha256 -verify brunocampos publica.pem -signature assinatura
msgPlana.txt
```

Resposta

```
[campos][avell][±][master U:7 ?:3 X][~/.../trabalho_SSL/2020]
■ openssl dgst -sha256 -verify brunocampos publica.pem -signature assinatura msgPlana.txt
Verified OK
```

Tudo OK.

GERAR UM CERTIFICADO AUTO-ASSINADO("auto" porque é assinado com SUA própria chave privada)

14.

```
openssl req -new -key brunocampos.privada.pem -out certificado.csr
```

Resposta

Resposta: arquivo **certificado.csr**

15.

```
openssl req -new -key brunocampos.privada.pem -out certificado.csr
```

Resposta

```
[campos][avell][±][master U:7 ?:5 X][~/.../trabalho_SSL/2020]
■ openssl x509 -req -days 90 -sha512 -in certificado.csr -signkey brunocampos.privada.pem -out certificado.crt
Signature ok
subject=C = br, ST = sc, L = florianopolis, O = brunocampos
Getting Private key
Enter pass phrase for brunocampos.privada.pem:
```

Tudo OK.