CSE 127 Computer Security

Stefan Savage, Spring 2020, Lecture 10

Cryptography II: PKI

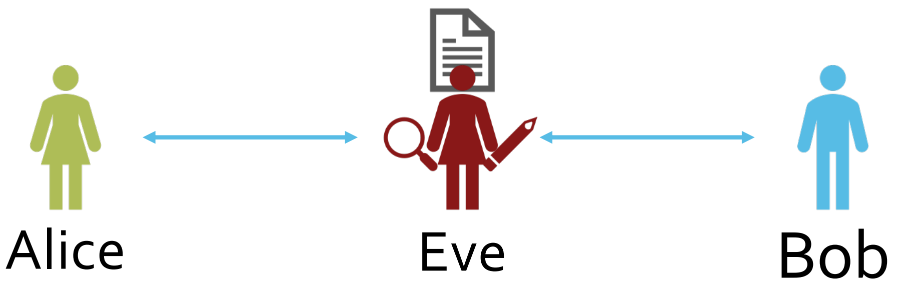
Announcements

- Midterm: We'll get it back to you next week
  - Source of delay: canvas has a grading policy for multi-answer questions that isn't what we want, so we need to offline regrade

- Reminder: PA 3 is due Thursday
- PA 4 is out on Thursday
  - Web security (which we'll start covering the same day)
  - Discussion section this week will get you started

Using Cryptography (review)

- Alice wants to send (a plaintext) *m* to Bob, via a channel that is controlled by Eve



Cryptographic Primitives (review)

- Confidentiality
  - Symmetric Encryption
    - $c = E_k(m), m = D_k(c)$
  - Asymmetric Encryption
    - $c = E_K(m), m = D_k(c)$
  - Combining Asymmetric with Symmetric
    - $k' \leftarrow r, E_K(k')||E_{k'}(m)$
  - You can rely on plaintext remaining secret.
    - Ciphertext reveals nothing about plaintext contents
  - You **cannot** rely on plaintext remaining unmodified.

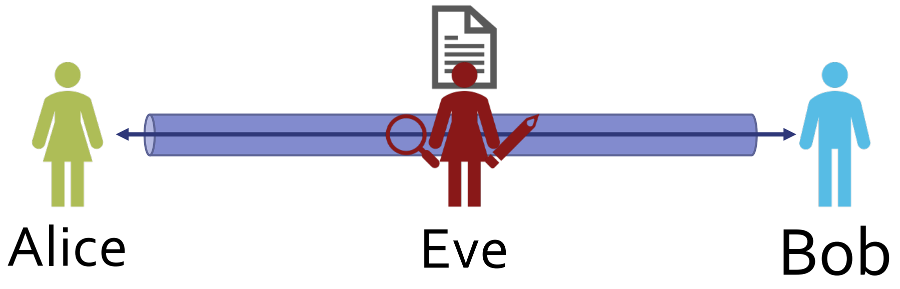
- Integrity and Authenticity
  - Symmetric MAC
    - $a = MAC_k(m)$
  - Asymmetric Signature
    - $s = S_k(H(m))$
    - $V_K(s,H(m))$: returns true or false
  - You can rely that whoever generated the tag (MAC or signature) had the secret key.
  - You **cannot** rely on tag not leaking information about the message.

Cryptographic Primitives (review)

- ***Authenticated encryption*** simultaneously provides confidentiality, integrity, and authenticity.
- A ***cryptographic hash function*** maps arbitrary length input into a fixed-size string and has the following properties:
  - ***Pre-image Resistance***
    - Impractical to find an input (pre-image) that generates specified output.
  - ***Collision Resistance***
    - Impractical to find two inputs that hash to the same output.
  - Common uses
    - Can be used to ***commit*** to a value without sharing value
    - Can be used as a fixed-size ***fingerprint*** of a longer string

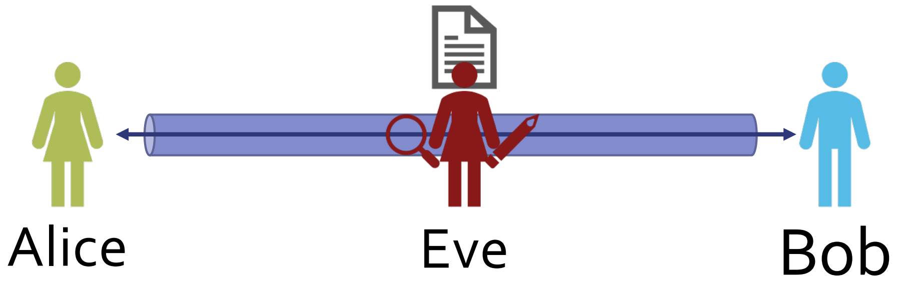
Using Cryptography

- Alice wants to send (a plaintext) *m* to Bob, via a channel that is controlled by Eve.
- Alice and Bob know each other's public keys. (assume pk crypto)
- Alice and Bob establish a secure "pipe".
- Eve cannot see plaintext contents inside the pipe, or modify them without detection.



Using Cryptography

- Alice and Bob got secrecy + integrity + authenticity and everyone lived happily ever after, right?
- Let's try to understand exactly what we achieved



Using Cryptography

- Bob knows that
  - Alice (or someone with her private key) knows the plaintext
  - Alice (or someone with her private key) signed the plaintext at some point in the past
- Alice knows that
  - Only Bob (or someone with his private key) can extract the plaintext from the encrypted channel
  - Bob (or anyone that gets the plaintext and Alice's signature) can prove that Alice signed the plaintext

Using Cryptography

- Alice does not know:
  - Whether Bob received the message
  - When Bob received the message
  - How many times Bob received the message
  - Whether Bob keeps the message secret
- Bob does not know:
  - Did Alice address this message to Bob
  - Who sent this copy of the message
  - When the message was sent
  - Who else knows the plaintext

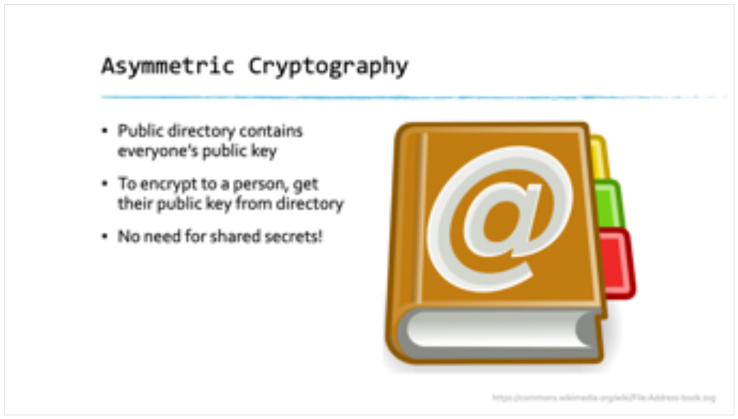
Aside: Digital Signatures

- What Does Signing Mean?
  - Signing is a mechanical operation that has no meaning in itself.
- What cryptography promises:
  - Only someone who knows the private key can create a signature that verifies using the corresponding public key
- Meaning of a digital signature is a matter of convention
  - Code signing: signer attests software is authorized to be installed
  - Email signing: signer attests she wrote message
  - Certificate signing: (coming up next!)
- Both signer and verifier need to agree on meaning

Public Key Infrastructure (PKI)

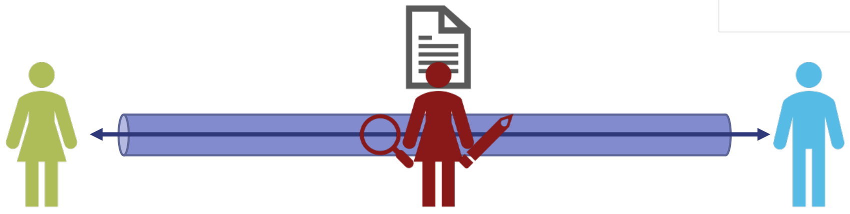
Using Cryptography

- Alice wants to send (a plaintext) *m* to Bob, via a channel that is controlled by Eve.
- Alice and Bob know each other's public keys.
- Alice and Bob establish a secure "pipe".



Asymmetric Cryptography

- Public directory contains everyone's public key
- To encrypt to a person, get their public key from directory
- No need for shared secrets!

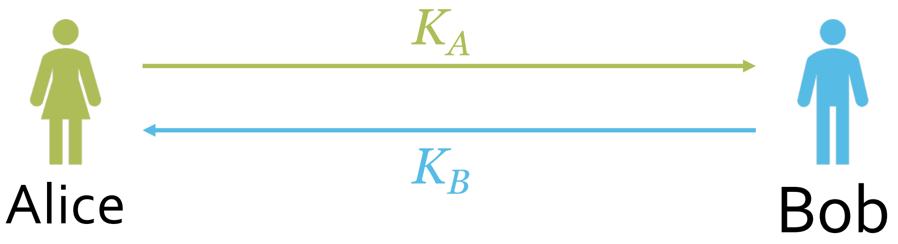


Getting Public Keys

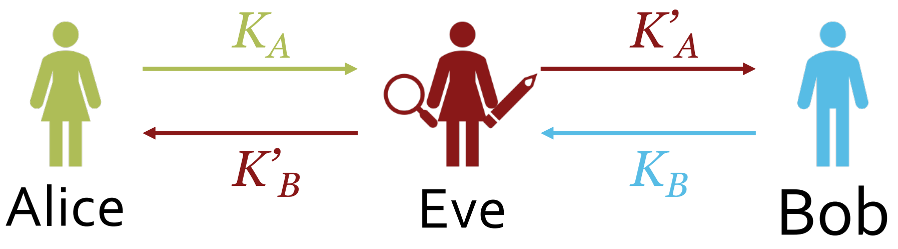
- Alice and Bob need a way to get each other’s public key.
- Alice can send an unencrypted message to Bob:
  - “Hey, send me your public key. Here’s mine.”
- Bob sends Alice his public key.
- They communicate securely ever after?

Getting Public Keys

- What they want to happen

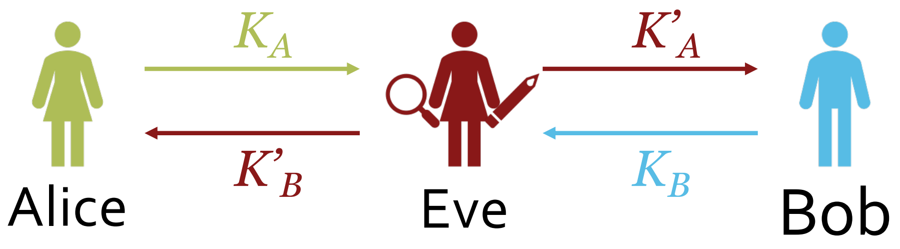


- What might happen instead



Getting Public Keys

- If Eve has man-in-the-middle capability, she can impersonate Alice to Bob and Bob to Alice.
  - Eve becomes invisible gateway between them.
  - Alice and Bob have no idea Eve is there.



Getting Public Keys

- Alice and Bob need a way to know that each has the **real** public key of the other.
- Ideal solution: Alice and Bob meet in person and exchange public keys
- Equivalent: Alice and Bob meet in person and exchange public key fingerprints
  - Key fingerprint: cryptographic hash of public key
  - Public key itself can be sent in the open
  - (aside: this is what Signal does)

Alice

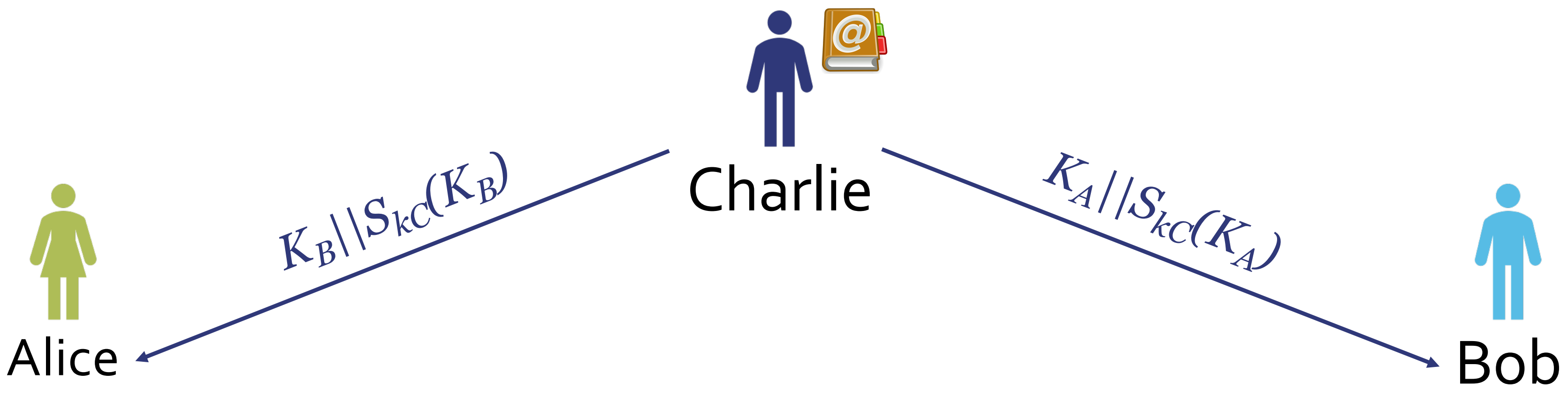
Bob

Getting Public Keys

- Problem with ideal:
  - We are back to pair-wise key establishment
  - Alice and Bob need to meet
  - Impractical to meet and verify key of everyone you talk to
- Any security problem can be solved with a ***trusted third party***

Getting Public Keys

- Using a trusted intermediary
  - Alice and Bob have already exchanged keys with Charlie
  - Charlie sends signed message with Alice's key to Bob
  - Charlie sends signed message with Bob's key to Alice
  - Alice and Bob trust Charlie to send the real public keys
  - Alice and Bob now have each other's public key



Getting Public Keys

- But every transaction is centralized through Charlie! We can do better...
- Charlie creates a ***certificate***:
  - "I, Charlie, verified that Alice's key is … "
- Charlie signs the message and gives it to Alice
  - Alice now has certificate **attesting** to her public key
- Alice sends Bob her public key and Charlie's certificate
- Bob verifies signature on certificate
- Bob trusts Charlie, accepts public key from Alice

Who is Charlie?

- Two common models:
  - PGP: Charlie is any other person you trust.
  - Almost everywhere else: Charlie is a ***Certificate Authority***.

PGP Web of Trust

- Pretty Good Privacy (PGP) is an application (and associated protocols) used for signing, encrypting, and decrypting texts, e-mails, files, directories, etc.
- PGP allows one user to attest to the accuracy of another user's public key — ***key signing***
  - PGP does not use the term "certificate", but that's because its old…
  - Public key has set of attestation signatures (certificates)
- A user can indicate how much she trusts another user's signature on a key.

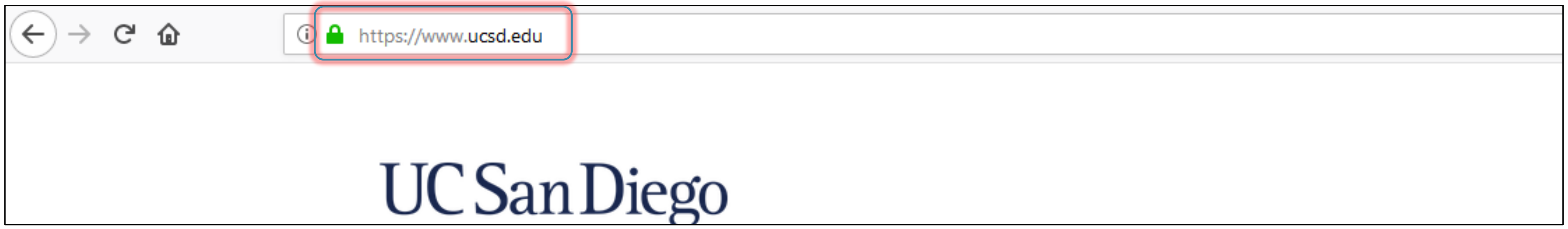
PGP Web of Trust

- Alice's signature on Bob's PGP key means Alice claims that this is really Bob's key (and ideally has verified this)
  - Email address and name associated with key are really his
- Other people who trust Alice can use her signature on Bob's key to be sure it is Bob's key
- How to decide?

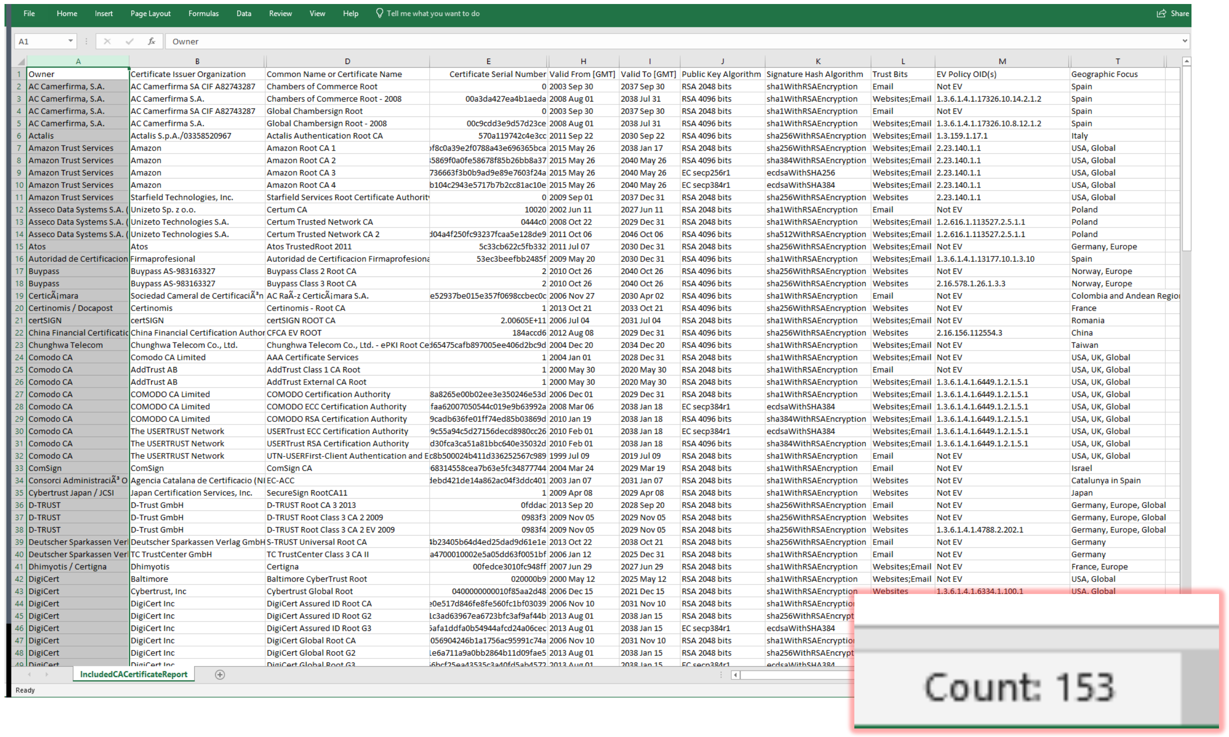
https://xkcd.com/364/

Certificate Authorities

- An alternative to PGP-like web of trust is to rely on centralized ***Certificate Authorities***: trusted signers of public keys.
- CA model used to sign certificates used on Web.
- Your browser has a set of public keys of trusted CAs.
  - Who makes this list?
  - How many CAs are on the list?
  - Who are these CAs?



Certificate Authorities



	Owner	Certificate Issuer Organization	Common Name or Certificate Name	Certificate Serial Number	Valid From [GMT]	Valid To [GMT]	Public Key Algorithm	Signature Hash Algorithm	Trust Bits	EV Policy OID(s)	Geographic Focus
2	AC Camerfirma, S.A.	AC Camerfirma SA CIF A82743287	Chambers of Commerce Root	0	2003 Sep 30	2037 Sep 30	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	Spain
3	AC Camerfirma, S.A.	AC Camerfirma S.A.	Chambers of Commerce Root - 2008	00a3da427ea4b1aeda	2008 Aug 01	2038 Jul 31	RSA 4096 bits	sha1WithRSAEncryption	Websites;Email	1.3.6.1.4.1.17326.10.14.2.1.2	Spain
4	AC Camerfirma, S.A.	AC Camerfirma SA CIF A82743287	Global Chambersign Root	0	2003 Sep 30	2037 Sep 30	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	Spain
5	AC Camerfirma, S.A.	AC Camerfirma S.A.	Global Chambersign Root - 2008	00c9cdd3e9d57d23ce	2008 Aug 01	2038 Jul 31	RSA 4096 bits	sha1WithRSAEncryption	Websites;Email	1.3.6.1.4.1.17326.10.8.12.1.2	Spain
6	Actalis	Actalis S.p.A./03358520967	Actalis Authentication Root CA	570a119742c4e3cc	2011 Sep 22	2030 Sep 22	RSA 4096 bits	sha256WithRSAEncryption	Websites;Email	1.3.159.1.17.1	Italy
7	Amazon Trust Services	Amazon	Amazon Root CA 1	f8c0a39e2f0788a43e696365bca	2015 May 26	2038 Jan 17	RSA 2048 bits	sha256WithRSAEncryption	Websites;Email	2.23.140.1.1	USA, Global
8	Amazon Trust Services	Amazon	Amazon Root CA 2	5869f0a0fe58678f85b26bb8a37	2015 May 26	2040 May 26	RSA 4096 bits	sha384WithRSAEncryption	Websites;Email	2.23.140.1.1	USA, Global
9	Amazon Trust Services	Amazon	Amazon Root CA 3	736663f3b0b9ad9e89e7603f24a	2015 May 26	2040 May 26	EC secp256r1	ecdsaWithSHA256	Websites;Email	2.23.140.1.1	USA, Global
10	Amazon Trust Services	Amazon	Amazon Root CA 4	b104c2943e5717b7b2cc81ac10e	2015 May 26	2040 May 26	EC secp384r1	ecdsaWithSHA384	Websites;Email	2.23.140.1.1	USA, Global
11	Amazon Trust Services	Starfield Technologies, Inc.	Starfield Services Root Certificate Authority	0	2009 Sep 01	2037 Dec 31	RSA 2048 bits	sha256WithRSAEncryption	Websites	2.23.140.1.1	USA, Global
12	Asseco Data Systems S.A. (	Unizeto Sp. z o.o.	Certum CA	10020	2002 Jun 11	2027 Jun 11	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	Poland
13	Asseco Data Systems S.A. (	Unizeto Technologies S.A.	Certum Trusted Network CA	0444c0	2008 Oct 22	2029 Dec 31	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	1.2.616.1.113527.2.5.1.1	Poland
14	Asseco Data Systems S.A. (	Unizeto Technologies S.A.	Certum Trusted Network CA 2	d04a4f250fc93237fcaa5e128de9	2011 Oct 06	2046 Oct 06	RSA 4096 bits	sha512WithRSAEncryption	Websites;Email	1.2.616.1.113527.2.5.1.1	Poland
15	Atos	Atos	Atos TrustedRoot 2011	5c33cb622c5fb332	2011 Jul 07	2030 Dec 31	RSA 2048 bits	sha256WithRSAEncryption	Websites;Email	Not EV	Germany, Europe
16	Autoridad de Certificacion	Firmaprofesional	Autoridad de Certificacion Firmaprofesiona	53ec3beefbb2485f	2009 May 20	2030 Dec 31	RSA 4096 bits	sha1WithRSAEncryption	Websites;Email	1.3.6.1.4.1.13177.10.1.3.10	Spain
17	Buypass	Buypass AS-983163327	Buypass Class 2 Root CA	2	2010 Oct 26	2040 Oct 26	RSA 4096 bits	sha256WithRSAEncryption	Websites	Not EV	Norway, Europe
18	Buypass	Buypass AS-983163327	Buypass Class 3 Root CA	2	2010 Oct 26	2040 Oct 26	RSA 4096 bits	sha256WithRSAEncryption	Websites	2.16.578.1.26.1.3.3	Norway, Europe
19	CerticÃ¡mara	Sociedad Cameral de CertificaciÃ³n	AC RaÃz CerticÃ¡mara S.A.	e52937be015e357f0698ccbec0c	2006 Nov 27	2030 Apr 02	RSA 4096 bits	sha1WithRSAEncryption	Email	Not EV	Colombia and Andean Regio
20	Certinomis / Docapost	Certinomis	Certinomis - Root CA	1	2013 Oct 21	2033 Oct 21	RSA 4096 bits	sha256WithRSAEncryption	Websites	Not EV	France
21	certSIGN	certSIGN	certSIGN ROOT CA	2.00605E+11	2006 Jul 04	2031 Jul 04	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	Not EV	Romania
22	China Financial Certificatio	China Financial Certification Author	CFCA EV ROOT	184accd6	2012 Aug 08	2029 Dec 31	RSA 4096 bits	sha256WithRSAEncryption	Websites	2.16.156.112554.3	China
23	Chunghwa Telecom	Chunghwa Telecom Co., Ltd.	Chunghwa Telecom Co., Ltd. - ePKI Root Ce	65475cafb897005ee406d2bc9d	2004 Dec 20	2034 Dec 20	RSA 4096 bits	sha1WithRSAEncryption	Websites;Email	Not EV	Taiwan
24	Comodo CA	Comodo CA Limited	AAA Certificate Services	1	2004 Jan 01	2028 Dec 31	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	Not EV	USA, UK, Global
25	Comodo CA	AddTrust AB	AddTrust Class 1 CA Root	1	2000 May 30	2020 May 30	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	USA, UK, Global
26	Comodo CA	AddTrust AB	AddTrust External CA Root	1	2000 May 30	2020 May 30	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
27	Comodo CA	COMODO CA Limited	COMODO Certification Authority	8a8265e00b02ee3e350246e53d	2006 Dec 01	2029 Dec 31	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
28	Comodo CA	COMODO CA Limited	COMODO ECC Certification Authority	faa62007050544c019e9b63992a	2008 Mar 06	2038 Jan 18	EC secp384r1	ecdsaWithSHA384	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
29	Comodo CA	COMODO CA Limited	COMODO RSA Certification Authority	9cadb636fe01ff74ed85b03869d	2010 Jan 19	2038 Jan 18	RSA 4096 bits	sha384WithRSAEncryption	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
30	Comodo CA	The USERTRUST Network	USERTrust ECC Certification Authority	9c55a94c5d27156decd8980cc26	2010 Feb 01	2038 Jan 18	EC secp384r1	ecdsaWithSHA384	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
31	Comodo CA	The USERTRUST Network	USERTrust RSA Certification Authority	d30fca3ca51a81bbc640e35032d	2010 Feb 01	2038 Jan 18	RSA 4096 bits	sha384WithRSAEncryption	Websites;Email	1.3.6.1.4.1.6449.1.2.1.5.1	USA, UK, Global
32	Comodo CA	The USERTRUST Network	UTN-USERFirst-Client Authentication and E	c8b500024b411d336252567c989	1999 Jul 09	2019 Jul 09	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	USA, UK, Global
33	ComSign	ComSign	ComSign CA	68314558cea7b63e5fc34877744	2004 Mar 24	2029 Mar 19	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	Israel
34	Consorci AdministraciÃ³ O	Agencia Catalana de Certificacio (NI	EC-ACC	debd421de14a862ac04f3ddc401	2003 Jan 07	2031 Jan 07	RSA 2048 bits	sha1WithRSAEncryption	Websites	Not EV	Catalunya in Spain
35	Cybertrust Japan / JCSI	Japan Certification Services, Inc.	SecureSign RootCA11	1	2009 Apr 08	2029 Apr 08	RSA 2048 bits	sha1WithRSAEncryption	Websites	Not EV	Japan
36	D-TRUST	D-Trust GmbH	D-TRUST Root CA 3 2013	0fddac	2013 Sep 20	2028 Sep 20	RSA 2048 bits	sha256WithRSAEncryption	Email	Not EV	Germany, Europe, Global
37	D-TRUST	D-Trust GmbH	D-TRUST Root Class 3 CA 2 2009	0983f3	2009 Nov 05	2029 Nov 05	RSA 2048 bits	sha256WithRSAEncryption	Websites	Not EV	Germany, Europe, Global
38	D-TRUST	D-Trust GmbH	D-TRUST Root Class 3 CA 2 EV 2009	0983f4	2009 Nov 05	2029 Nov 05	RSA 2048 bits	sha256WithRSAEncryption	Websites	1.3.6.1.4.1.4788.2.202.1	Germany, Europe, Global
39	Deutscher Sparkassen Verl	Deutscher Sparkassen Verlag GmbH	S-TRUST Universal Root CA	4b23405b64d4ed25dad9d61e1e	2013 Oct 22	2038 Oct 21	RSA 2048 bits	sha256WithRSAEncryption	Email	Not EV	Germany
40	Deutscher Sparkassen Verl	TC TrustCenter GmbH	TC TrustCenter Class 3 CA II	a4700010002e5a05dd63f0051bf	2006 Jan 12	2025 Dec 31	RSA 2048 bits	sha1WithRSAEncryption	Email	Not EV	Germany
41	Dhimyotis / Certigna	Dhimyotis	Certigna	00fedce3010fc948ff	2007 Jun 29	2027 Jun 29	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	Not EV	France, Europe
42	DigiCert	Baltimore	Baltimore CyberTrust Root	020000b9	2000 May 12	2025 May 12	RSA 2048 bits	sha1WithRSAEncryption	Websites;Email	Not EV	USA, Global
43	DigiCert	Cybertrust, Inc	Cybertrust Global Root	0400000000010f85aa2d48	2006 Dec 15	2021 Dec 15	RSA 2048 bits	sha1WithRSAEncryption	Websites	1.3.6.1.4.1.6334.1.100.1	USA, Global
44	DigiCert	DigiCert Inc	DigiCert Assured ID Root CA	e0e517d846fe8fe560fc1bf03039	2006 Nov 10	2031 Nov 10	RSA 2048 bits	sha1WithRSAEncryptio			
45	DigiCert	DigiCert Inc	DigiCert Assured ID Root G2	1c3ad63967ea6723bfc3af9af44b	2013 Aug 01	2038 Jan 15	RSA 2048 bits	sha256WithRSAEncrypt			
46	DigiCert	DigiCert Inc	DigiCert Assured ID Root G3	5afa1ddfa0b54944afcd24a06cec	2013 Aug 01	2038 Jan 15	EC secp384r1	ecdsaWithSHA384			
47	DigiCert	DigiCert Inc	DigiCert Global Root CA	056904246b1a1756ac95991c74a	2006 Nov 10	2031 Nov 10	RSA 2048 bits	sha1WithRSAEncryptio			
48	DigiCert	DigiCert Inc	DigiCert Global Root G2	1e6a711a9a0bb2864b11d09fae5	2013 Aug 01	2038 Jan 15	RSA 2048 bits	sha256WithRSAEncrypt			
49	DigiCert	DigiCert Inc	DigiCert Global Root G3	6bcf25ea43535c3a40fd5ab4572	2013 Aug 01	2038 Jan 15	EC secp384r1	ecdsaWithSHA384			

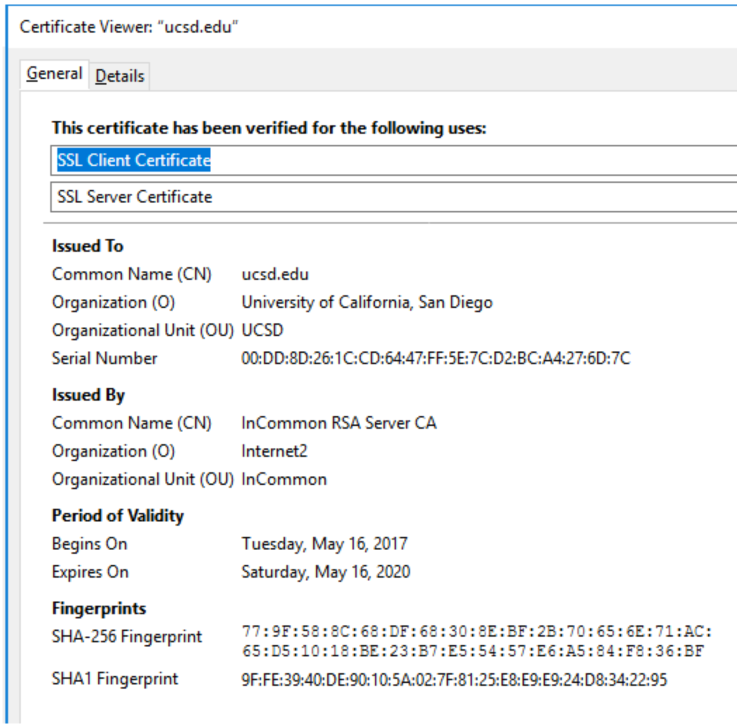
Count: 153

Certificate Authorities

- Mozilla
  - ~150 root certificates
  - https://wiki.mozilla.org/CA/Included_Certificates
- iOS
  - ~150 root certificates
  - https://support.apple.com/en-us/HT208125
- Microsoft
  - ~300 active root certificates
  - http://aka.ms/RootCert

Certificate Authorities

- Certificate semantics:
  - Subject (name, domain)
  - Issuing CA
  - Validity period
  - Limitations on use



Certificate Viewer: "ucsd.edu"

This certificate has been verified for the following uses:

SSL Client Certificate

SSL Server Certificate

Issued To

Common Name (CN)	ucsd.edu
Organization (O)	University of California, San Diego
Organizational Unit (OU)	UCSD
Serial Number	00:DD:8D:26:1C:CD:64:47:FF:5E:7C:D2:BC:A4:27:6D:7C

Issued By

Common Name (CN)	InCommon RSA Server CA
Organization (O)	Internet2
Organizational Unit (OU)	InCommon

Period of Validity

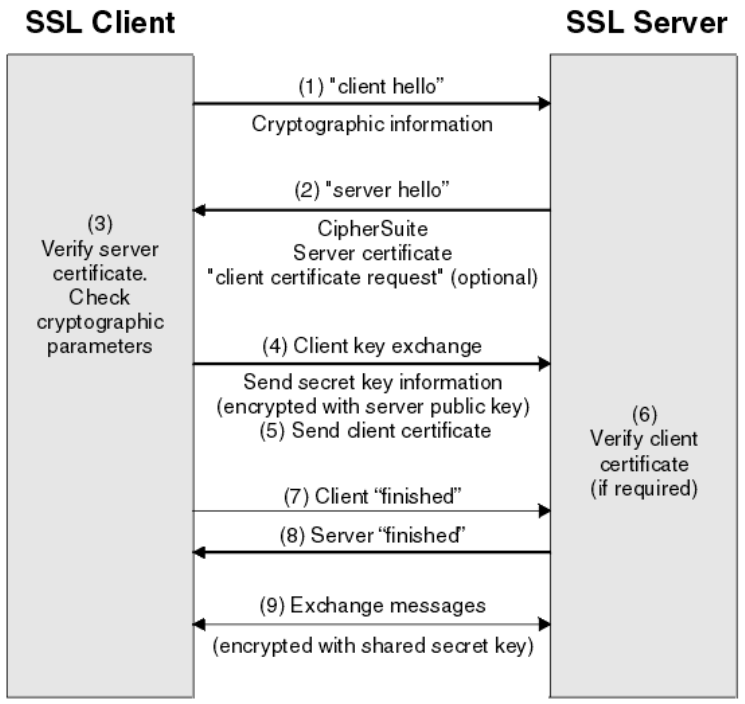
Begins On	Tuesday, May 16, 2017
Expires On	Saturday, May 16, 2020

Fingerprints

SHA-256 Fingerprint	77:9F:58:8C:68:DF:68:30:8E:BF:2B:70:65:6E:71:AC: 65:D5:10:18:BE:23:B7:E5:54:57:E6:A5:84:F8:36:BF
SHA1 Fingerprint	9F:FE:39:40:DE:90:10:5A:02:7F:81:25:E8:E9:E9:24:D8:34:22:95

Using Certificates: Transport Layer Security (TLS)

- This is what makes https:// work
- "*When secured by TLS, connections between a client (e.g., a web browser) and a server (e.g., wikipedia.org) have one or more of the following properties:*
  - *The connection is private (or secure) because* ***symmetric cryptography*** *is used* ***to encrypt*** *the data transmitted…*
  - *The identity of the communicating parties can be* ***authenticated using public-key cryptography****…*
  - *The connection ensures integrity because each message transmitted includes a message integrity check using a* ***message authentication code*** *to prevent undetected loss or alteration of the data during transmission.*"
- Details of protocol are complex, but the basic idea isn't:
  - Browser gets and verifies server's certificate, and extracts PK
  - Use PK to encrypt random symmetric session key
  - Use session key to encrypt session and to key HMAC for integrity


SSL Client
SSL Server
(1) "client hello"
Cryptographic information
(2) "server hello"
CipherSuite
Server certificate
"client certificate request" (optional)
(3) Verify server certificate. Check cryptographic parameters
(4) Client key exchange
Send secret key information (encrypted with server public key)
(5) Send client certificate
(6) Verify client certificate (if required)
(7) Client "finished"
(8) Server "finished"
(9) Exchange messages
(encrypted with shared secret key)

https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_7.1.0/com.ibm.mq.doc/sy10660_.htm

A quick step back…

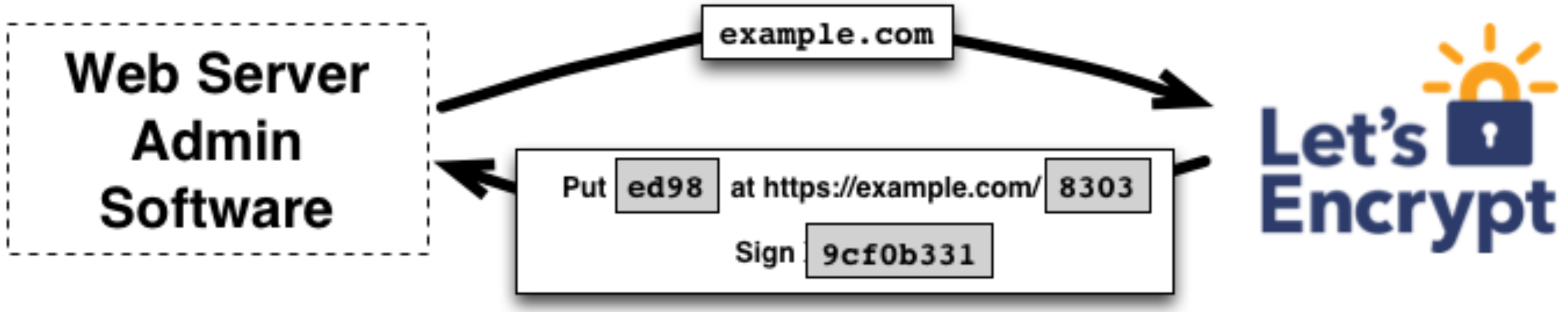
- Using TLS (i.e., https) what security is being claimed?
  - Authenticity
    - That the server is who they say they are (e.g., www.amazon.com)
    - That the client is who they say they are?
  - Confidentiality
    - That no one but the client and server can read the messages sent between them
  - Integrity
    - That no one can alter messages between client and server without being detected

- What are we depending on?
  - Crypto works
  - The attestations of the CAs are reliable and their services are secure

Certificate Authorities

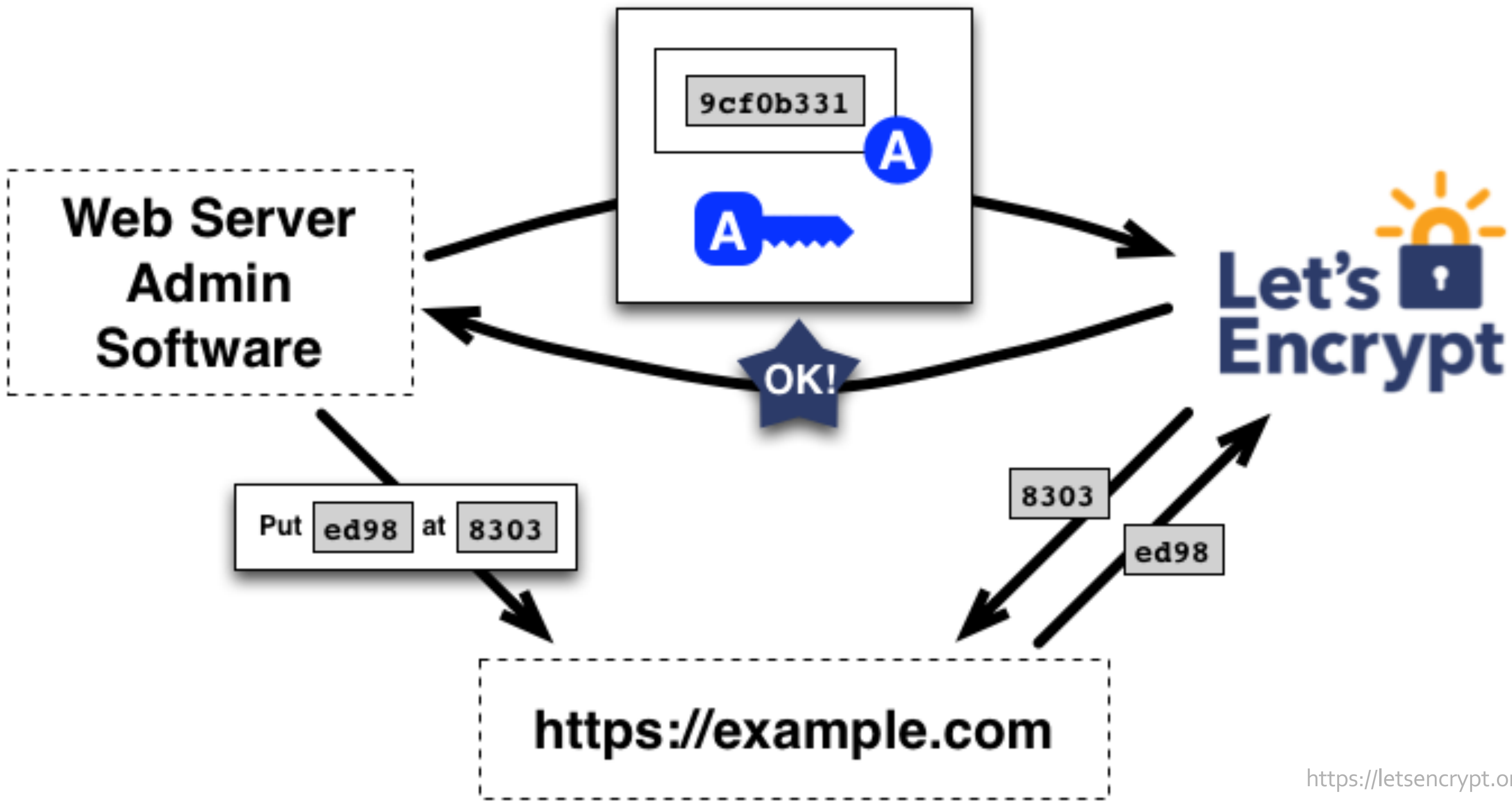
- Which CA can issue a certificate for mycompany.com?
- For fbi.gov?

Let’s Encrypt


Web Server Admin Software
example.com
Put ed98 at https://example.com/ 8303
Sign 9cf0b331
Let’s Encrypt

https://letsencrypt.org/how-it-works/

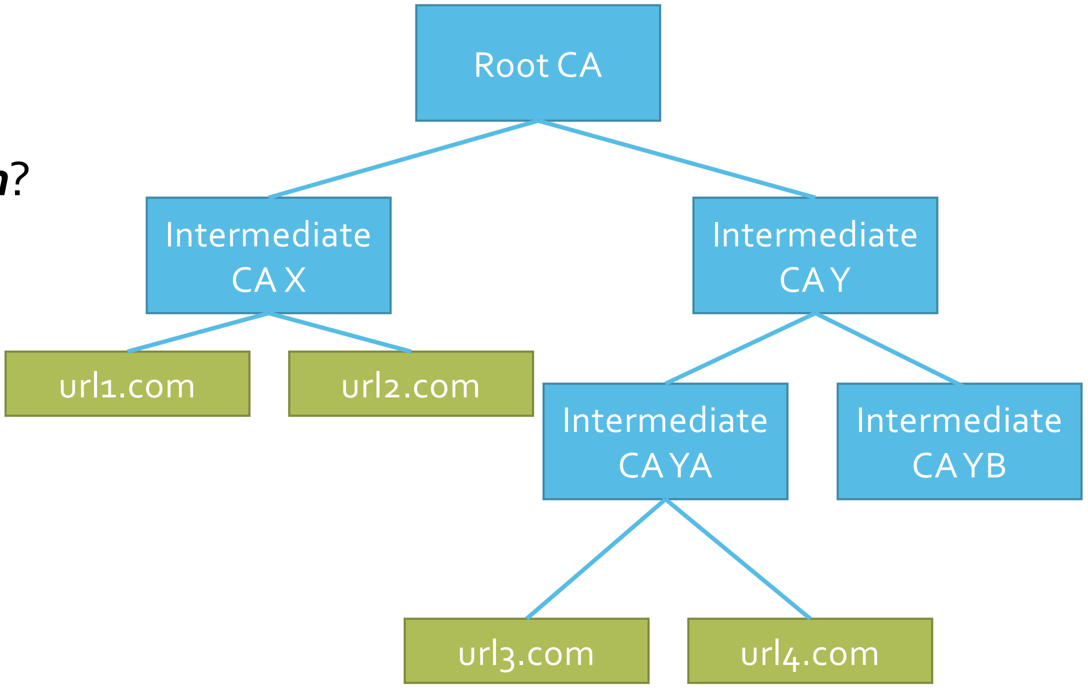
Let’s Encrypt



https://letsencrypt.org/how-it-works/

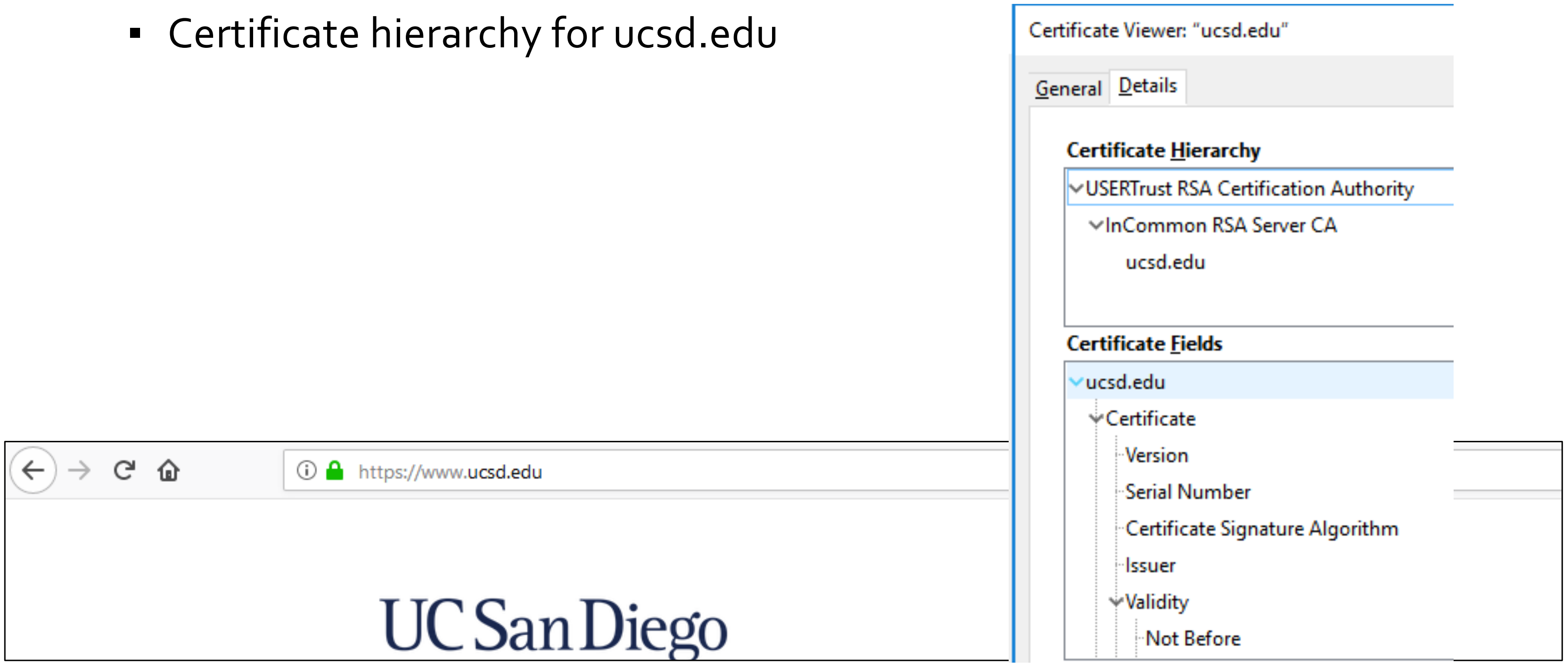
Certificate Authorities

- What if we take a ***Trusted Third Party*** and combine it with ***Another Layer of Indirection***?
- ***Certificate Hierarchy***
- ***Root CA*** signs keys for ***Intermediate CAs***, which in turn sign keys for users (or other intermediate CAs)



Certificate Authorities

- Certificate hierarchy for ucsd.edu


https://www.ucsd.edu

UC San Diego

Certificate Viewer: "ucsd.edu"

Certificate Hierarchy

- USERTrust RSA Certification Authority
  - InCommon RSA Server CA
    - ucsd.edu

Certificate Fields

- ucsd.edu
  - Certificate
    - Version
    - Serial Number
    - Certificate Signature Algorithm
    - Issuer
    - Validity
      - Not Before

Aside: other uses of certificates

- Certificates also used in code signing
  - https://source.android.com/security/apksigning/
  - https://docs.microsoft.com/en-us/windows-hardware/drivers/install/driver-signing
  - https://developer.apple.com/support/code-signing/
- Who is the CA?
- What is the meaning of the signature?
  - Alice released this app?
  - Alice authorizes this app to run?
  - Alice authorizes this app to access privileged resources?

Certificate Revocation

- What happens if someone steals your private key?
  - They can impersonate you and read messages encrypted to you
- Certificate expiration helps with this but not enough
- CA and PGP PKIs support revocation
  - Owner says: "I, Alice, revoke my public key … do not use it."
  - Signs revocation with her private key
  - Others can verify Alice's signature, stop using key

Certificate Revocation

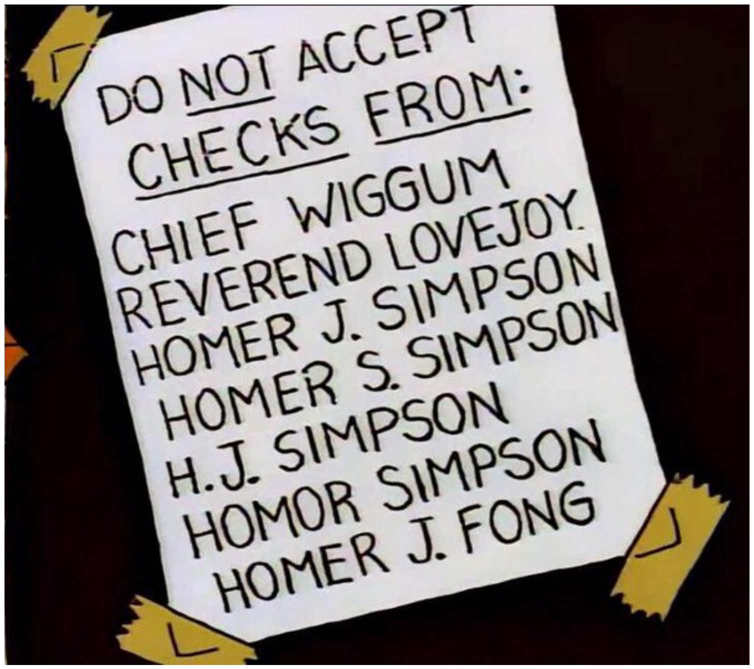
- How does Bob know if Alice’s key has been revoked?
- Bob asks Alice: “Has your key been revoked?”
- Alice sends signed message: “No.”
- Does not work: if Alice’s key has been compromised, then Eve could have forged the message “No.”
- Availability of revocation list critical

Certificate Revocation

- In PGP model, only Alice can revoke her own key
  - If Alice loses her private key, she can’t revoke
    - Do not lose private key
  - Option: generate revocation with key, store in secure place
- In CA model, Alice asks CA to revoke certificate
  - Alice does not need private key to do this, can authenticate herself through other means

Certificate Revocation

- Two Mechanisms: CRL and OCSP
- ***Certificate Revocation List (CRL)***:
  - Certificate says where to get CRL
  - Clients periodically download updated CRLs
  - What if CRL server is down?

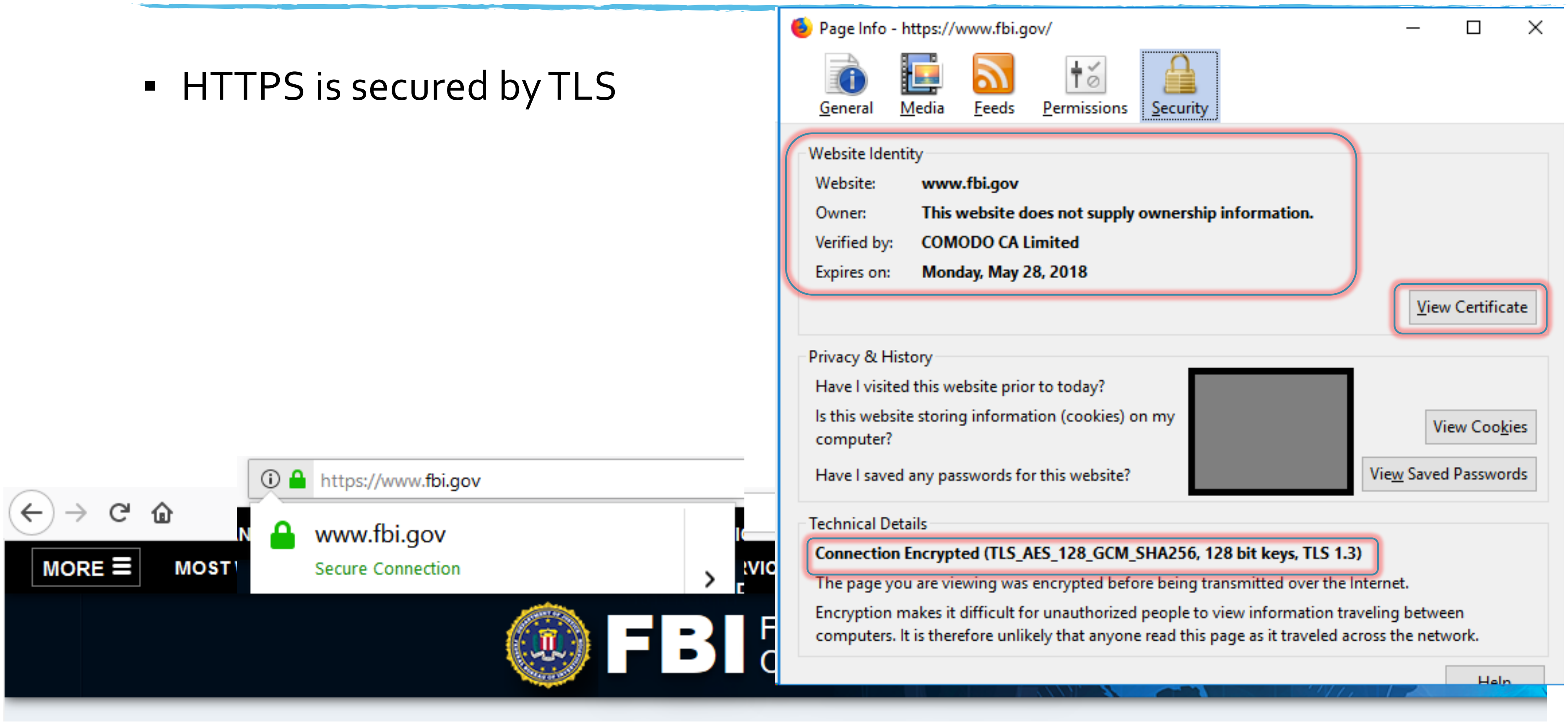


Certificate Revocation

- Two Mechanisms: CRL and OCSP
- ***Online Certificate Status Protocol (OCSP)***:
  - Query CA about status of cert before trusting it
  - “You said I can trust this key, but are you still sure?”
- OCSP Stapling
  - Server includes recent OCSP status (signed by CA)
- Aside: Certificate Pinning
  - Remember which certificate was used for a particular domain and raise an alert if a different one is used later
- Visit https://revoked-isrgrootx1.letsencrypt.org/ with your browser

Some additional complexity: CDNs

- HTTPS is secured by TLS



Page Info - https://www.fbi.gov/

Website Identity

Website: **www.fbi.gov**

Owner: **This website does not supply ownership information.**

Verified by: **COMODO CA Limited**

Expires on: **Monday, May 28, 2018**

Privacy & History

Have I visited this website prior to today?

Is this website storing information (cookies) on my computer?

Have I saved any passwords for this website?

[redacted]

Technical Details

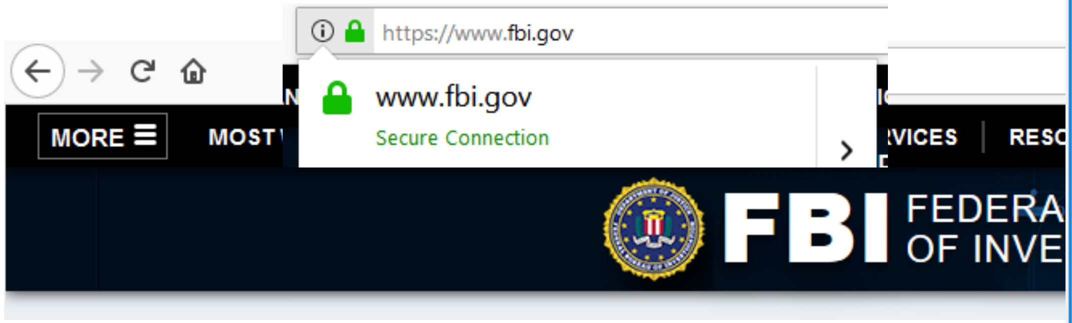
Connection Encrypted (TLS_AES_128_GCM_SHA256, 128 bit keys, TLS 1.3)

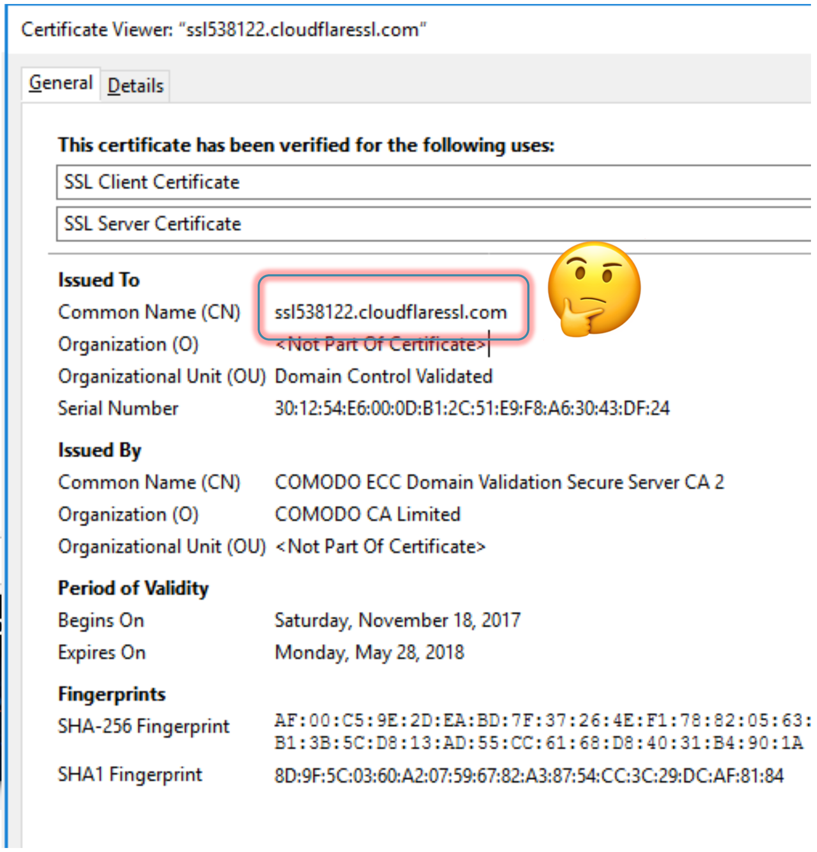
The page you are viewing was encrypted before being transmitted over the Internet.

Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.

Some additional complexity: CDNs

- HTTPS is secured by TLS





Certificate Viewer: "ssl538122.cloudflaressl.com"

General Details

This certificate has been verified for the following uses:

SSL Client Certificate

SSL Server Certificate

Issued To

Common Name (CN)	ssl538122.cloudflaressl.com
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	Domain Control Validated
Serial Number	30:12:54:E6:00:0D:B1:2C:51:E9:F8:A6:30:43:DF:24

Issued By

Common Name (CN)	COMODO ECC Domain Validation Secure Server CA 2
Organization (O)	COMODO CA Limited
Organizational Unit (OU)	<Not Part Of Certificate>

Period of Validity

Begins On	Saturday, November 18, 2017
Expires On	Monday, May 28, 2018

Fingerprints

SHA-256 Fingerprint	AF:00:C5:9E:2D:EA:BD:7F:37:26:4E:F1:78:82:05:63: B1:3B:5C:D8:13:AD:55:CC:61:68:D8:40:31:B4:90:1A
SHA1 Fingerprint	8D:9F:5C:03:60:A2:07:59:67:82:A3:87:54:CC:3C:29:DC:AF:81:84

Content Delivery Networks (CDNs)

- CDN: geographically distributed network of proxy servers
  - Cache static content closer to the requester
  - Improve latency
  - Decrease network congestion
  - Improve reliability and availability
    - DDOS protection
  - Cloudflare, Akamai, CloudFront, etc
- Mess up our nice security abstractions
  - Now Alice deliberately wants her CDN to impersonate her to Bob!

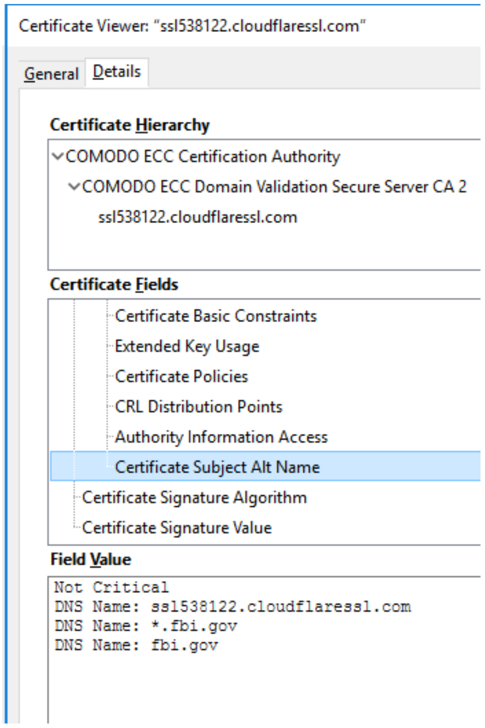
Content Delivery Networks (CDNs)

- Bob wants to connect to www.fbi.gov
- Bob’s browser attempts to get the corresponding IP address via DNS
- Because FBI is using Cloudflare CDN, DNS resolves to a Cloudflare edge server
- But Bob’s browser thinks it’s talking to fbi.gov
- Cloudflare needs to convince Bob’s browser that it’s really FBI

Content Delivery Networks (CDNs)

- Deputized via "Subject Alternate Name" field
  - "Yeah, I'm cloudflaressl.com, but I'm authorized to communicate on behalf fbi.gov"

- Who decides whether a CDN can get a given Subject Alternate Name in its cert?



Certificate Viewer: "ssl538122.cloudflaressl.com"

Certificate Hierarchy

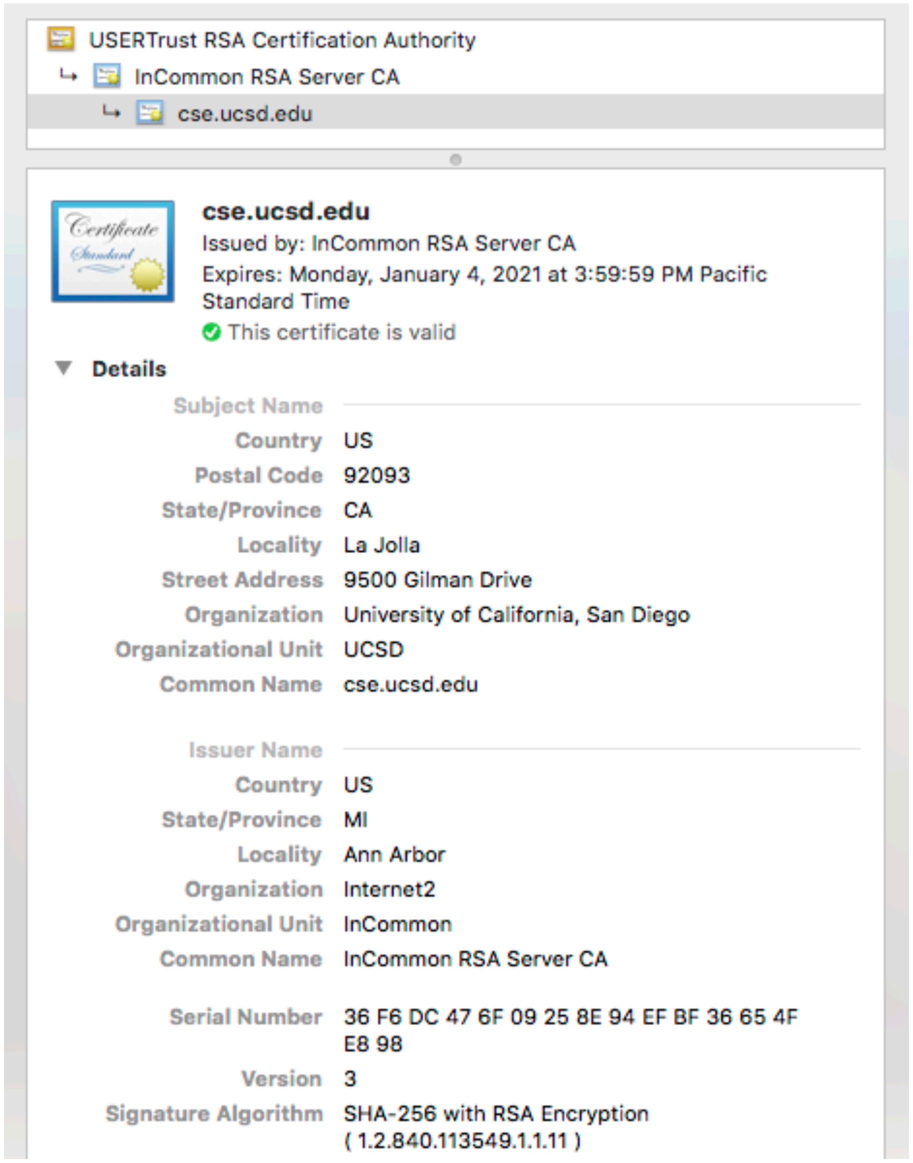
- COMODO ECC Certification Authority
  - COMODO ECC Domain Validation Secure Server CA 2
    - ssl538122.cloudflaressl.com

Certificate Fields

- Certificate Basic Constraints
- Extended Key Usage
- Certificate Policies
- CRL Distribution Points
- Authority Information Access
- Certificate Subject Alt Name
- Certificate Signature Algorithm
- Certificate Signature Value

Field Value

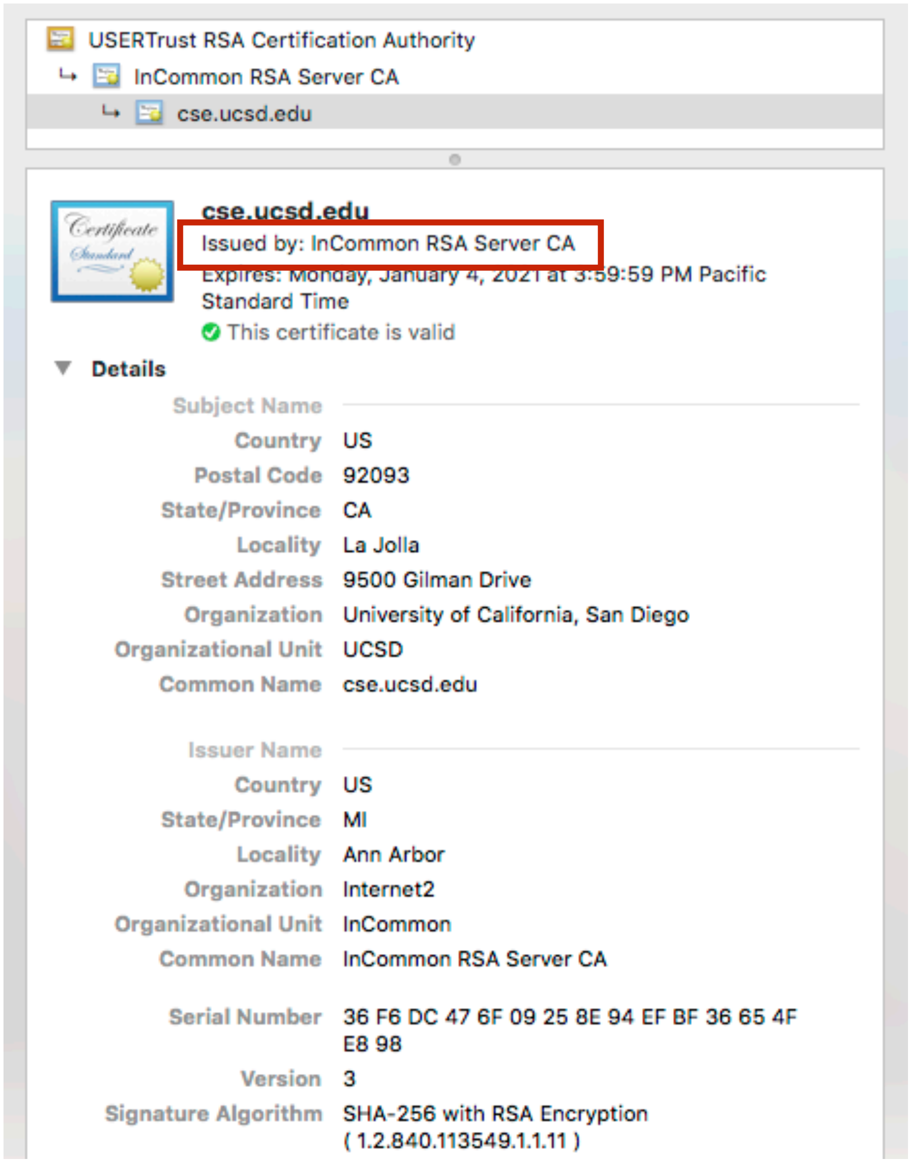
```
Not Critical
DNS Name: ssl538122.cloudflaressl.com
DNS Name: *.fbi.gov
DNS Name: fbi.gov
```


USERTrust RSA Certification Authority
↳ InCommon RSA Server CA
↳ cse.ucsd.edu

cse.ucsd.edu
Issued by: InCommon RSA Server CA
Expires: Monday, January 4, 2021 at 3:59:59 PM Pacific Standard Time
This certificate is valid

Details

Subject Name	
Country	US
Postal Code	92093
State/Province	CA
Locality	La Jolla
Street Address	9500 Gilman Drive
Organization	University of California, San Diego
Organizational Unit	UCSD
Common Name	cse.ucsd.edu
Issuer Name	
Country	US
State/Province	MI
Locality	Ann Arbor
Organization	Internet2
Organizational Unit	InCommon
Common Name	InCommon RSA Server CA
Serial Number	36 F6 DC 47 6F 09 25 8E 94 EF BF 36 65 4F E8 98
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)



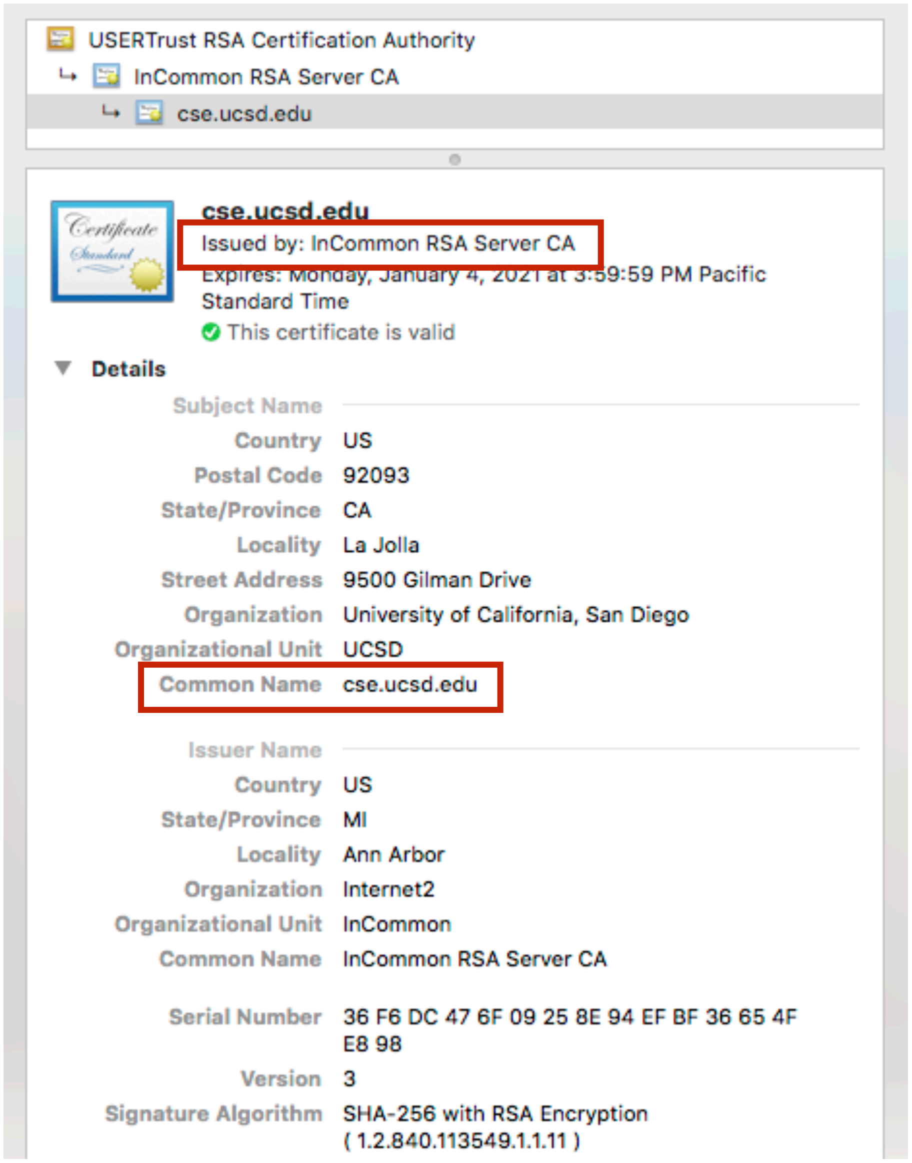
USERTrust RSA Certification Authority
↳ InCommon RSA Server CA
↳ cse.ucsd.edu

cse.ucsd.edu
Issued by: InCommon RSA Server CA
Expires: Monday, January 4, 2021 at 3:59:59 PM Pacific Standard Time
✓ This certificate is valid

▼ **Details**

Subject Name	
Country	US
Postal Code	92093
State/Province	CA
Locality	La Jolla
Street Address	9500 Gilman Drive
Organization	University of California, San Diego
Organizational Unit	UCSD
Common Name	cse.ucsd.edu
Issuer Name	
Country	US
State/Province	MI
Locality	Ann Arbor
Organization	Internet2
Organizational Unit	InCommon
Common Name	InCommon RSA Server CA
Serial Number	36 F6 DC 47 6F 09 25 8E 94 EF BF 36 65 4F E8 98
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)

Who are we trusting?



USERTrust RSA Certification Authority
↳ InCommon RSA Server CA
↳ cse.ucsd.edu

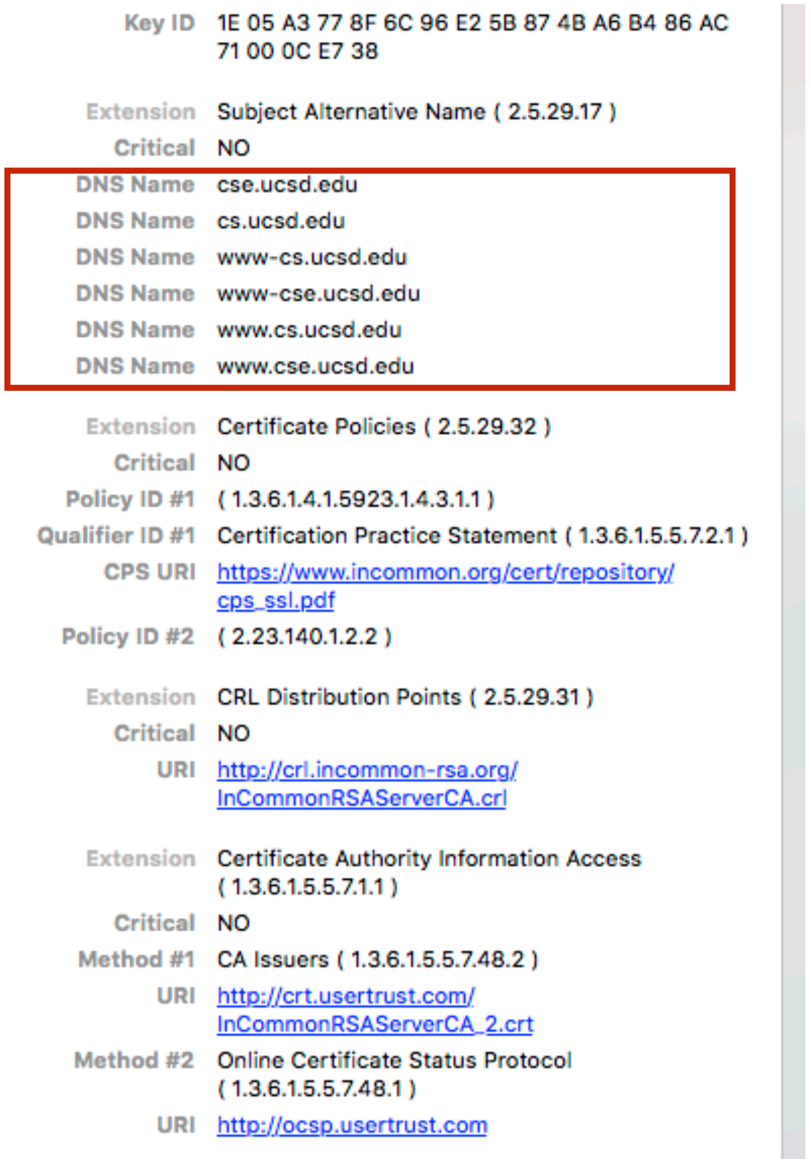
cse.ucsd.edu
Issued by: InCommon RSA Server CA
Expires: Monday, January 4, 2021 at 3:59:59 PM Pacific Standard Time
This certificate is valid

Details

Subject Name	
Country	US
Postal Code	92093
State/Province	CA
Locality	La Jolla
Street Address	9500 Gilman Drive
Organization	University of California, San Diego
Organizational Unit	UCSD
Common Name	cse.ucsd.edu
Issuer Name	
Country	US
State/Province	MI
Locality	Ann Arbor
Organization	Internet2
Organizational Unit	InCommon
Common Name	InCommon RSA Server CA
Serial Number	36 F6 DC 47 6F 09 25 8E 94 EF BF 36 65 4F E8 98
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)

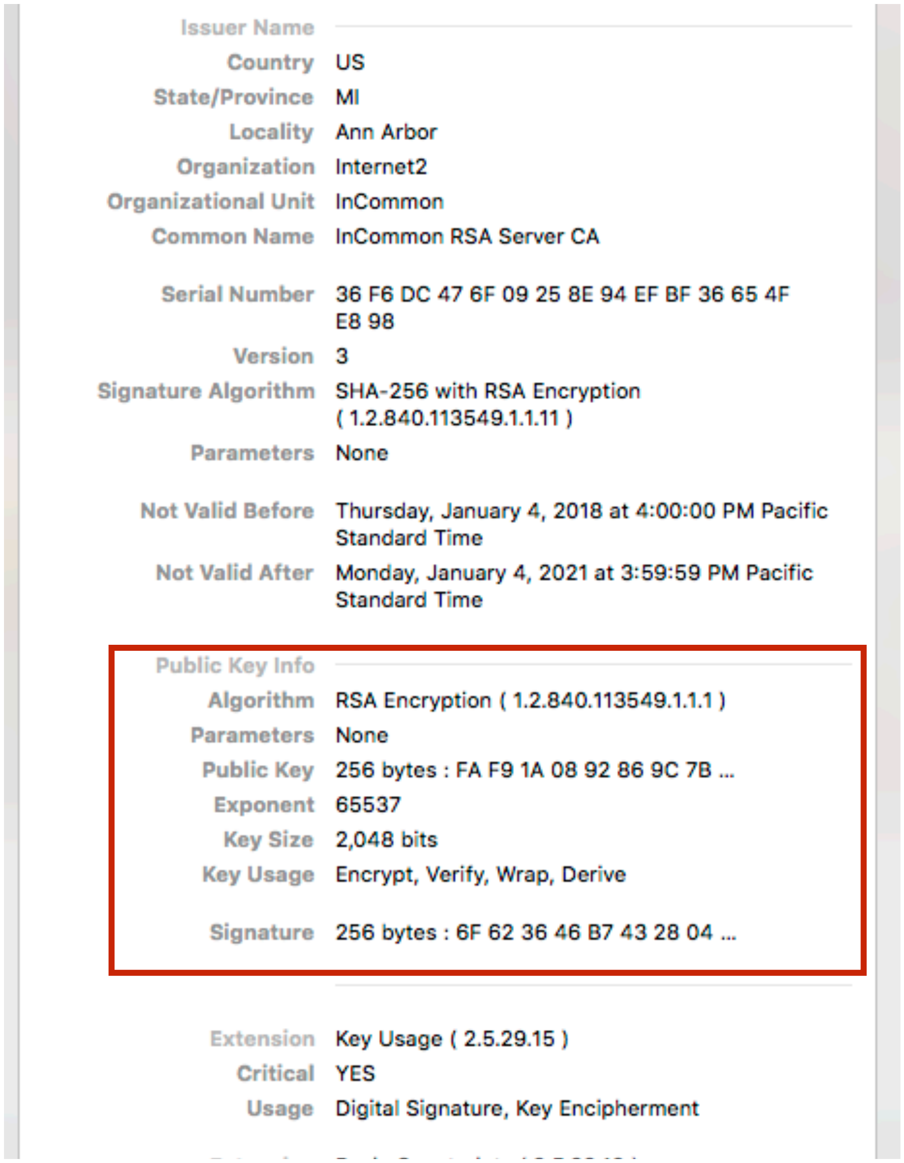
Who are we trusting?

Who is this cert for?



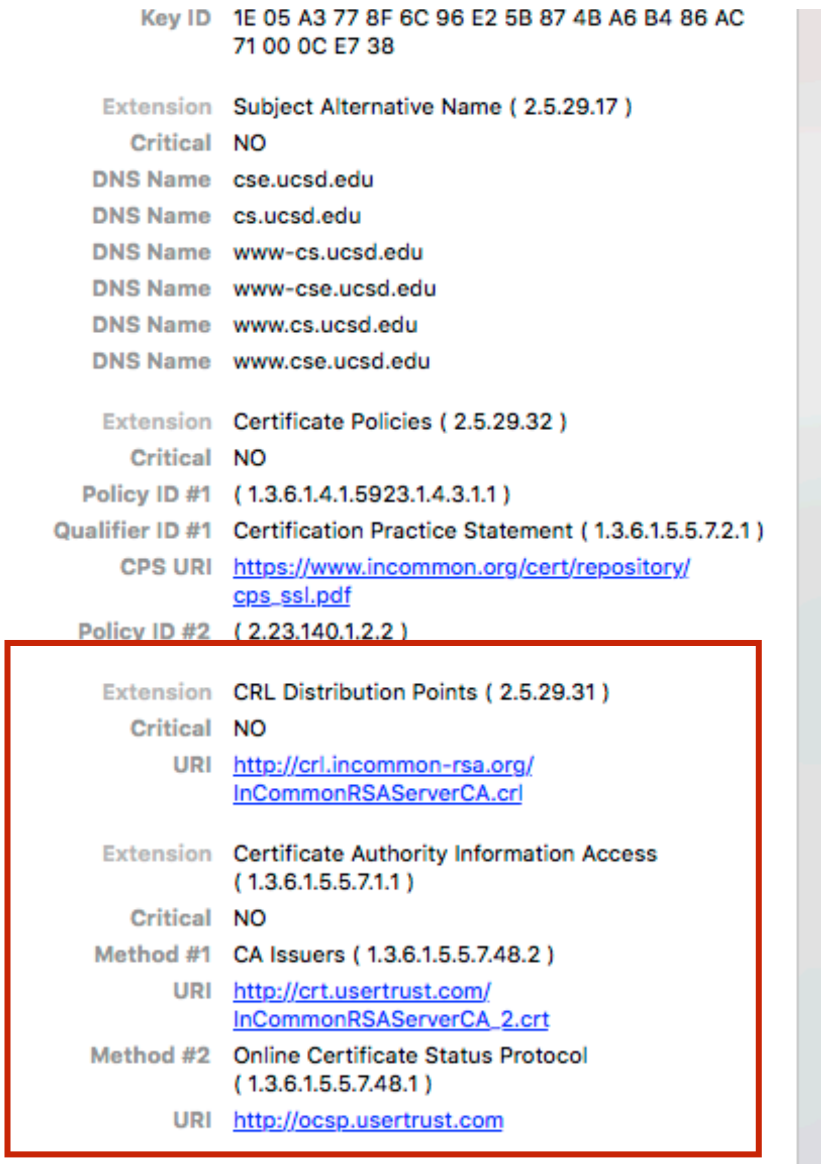
Key ID	1E 05 A3 77 8F 6C 96 E2 5B 87 4B A6 B4 86 AC 71 00 0C E7 38
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	cse.ucsd.edu
DNS Name	cs.ucsd.edu
DNS Name	www-cs.ucsd.edu
DNS Name	www-cse.ucsd.edu
DNS Name	www.cs.ucsd.edu
DNS Name	www.cse.ucsd.edu
Extension	Certificate Policies (2.5.29.32)
Critical	NO
Policy ID #1	(1.3.6.1.4.1.5923.1.4.3.1.1)
Qualifier ID #1	Certification Practice Statement (1.3.6.1.5.5.7.2.1)
CPS URI	https://www.incommon.org/cert/repository/cps_ssl.pdf
Policy ID #2	(2.23.140.1.2.2)
Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://crl.incommon-rsa.org/InCommonRSAServerCA.crl
Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO
Method #1	CA Issuers (1.3.6.1.5.5.7.48.2)
URI	http://crt.usertrust.com/InCommonRSAServerCA_2.crt
Method #2	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://ocsp.usertrust.com

Who is this cert for?



Issuer Name	
Country	US
State/Province	MI
Locality	Ann Arbor
Organization	Internet2
Organizational Unit	InCommon
Common Name	InCommon RSA Server CA
Serial Number	36 F6 DC 47 6F 09 25 8E 94 EF BF 36 65 4F E8 98
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Thursday, January 4, 2018 at 4:00:00 PM Pacific Standard Time
Not Valid After	Monday, January 4, 2021 at 3:59:59 PM Pacific Standard Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : FA F9 1A 08 92 86 9C 7B ...
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 6F 62 36 46 B7 43 28 04 ...
Extension	Key Usage (2.5.29.15)
Critical	YES
Usage	Digital Signature, Key Encipherment

CSE's pub key info

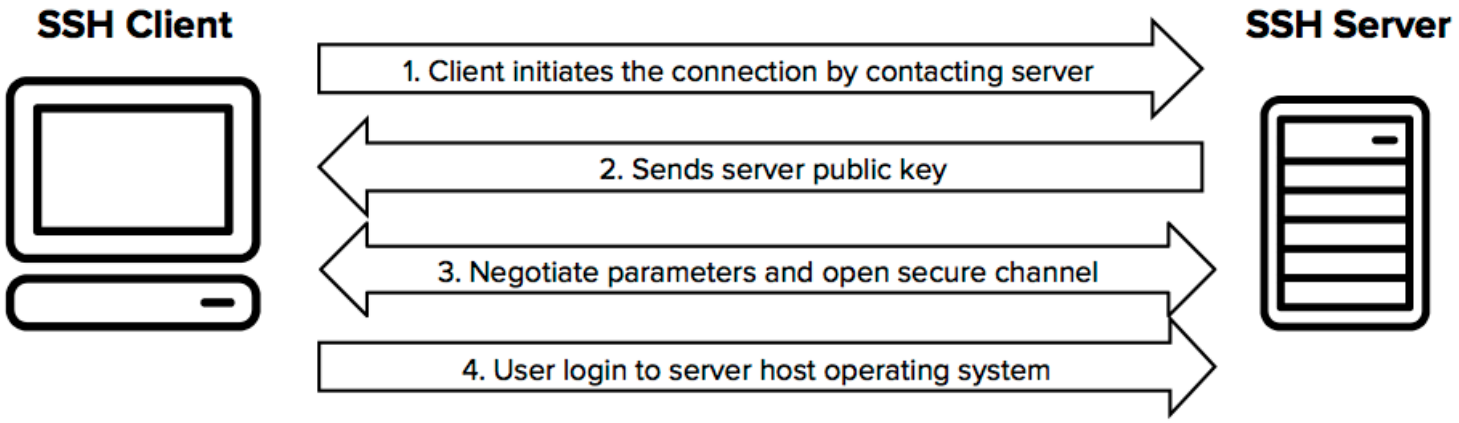


Key ID	1E 05 A3 77 8F 6C 96 E2 5B 87 4B A6 B4 86 AC 71 00 0C E7 38
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	cse.ucsd.edu
DNS Name	cs.ucsd.edu
DNS Name	www-cs.ucsd.edu
DNS Name	www-cse.ucsd.edu
DNS Name	www.cs.ucsd.edu
DNS Name	www.cse.ucsd.edu
Extension	Certificate Policies (2.5.29.32)
Critical	NO
Policy ID #1	(1.3.6.1.4.1.5923.1.4.3.1.1)
Qualifier ID #1	Certification Practice Statement (1.3.6.1.5.5.7.2.1)
CPS URI	https://www.incommon.org/cert/repository/cps_ssl.pdf
Policy ID #2	(2.23.140.1.2.2)
Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://crl.incommon-rsa.org/InCommonRSAServerCA.crl
Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO
Method #1	CA Issuers (1.3.6.1.5.5.7.48.2)
URI	http://crt.usertrust.com/InCommonRSAServerCA_2.crt
Method #2	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://ocsp.usertrust.com

Where we should check for revocation information

Another approach: Secure Shell (SSH)

- *"Secure Shell (SSH) provides a secure channel over an unsecured network, connecting an SSH client application with an SSH server. Common applications include remote command-line login and remote command execution, but any network service can be secured with SSH."*

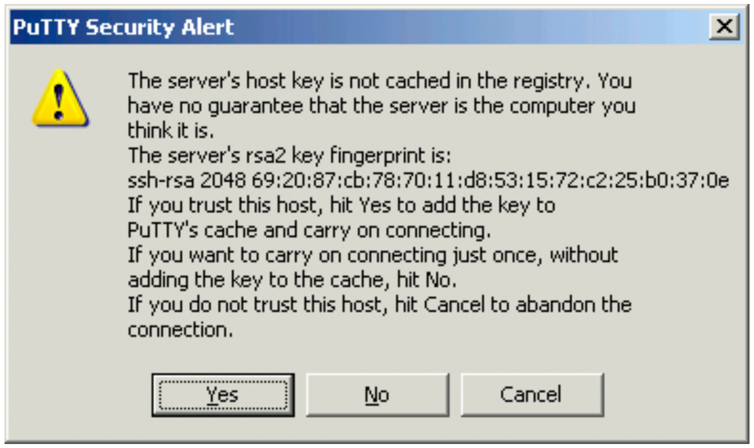

SSH Client
SSH Server
1. Client initiates the connection by contacting server
2. Sends server public key
3. Negotiate parameters and open secure channel
4. User login to server host operating system

-

https://www.ssh.com/ssh/

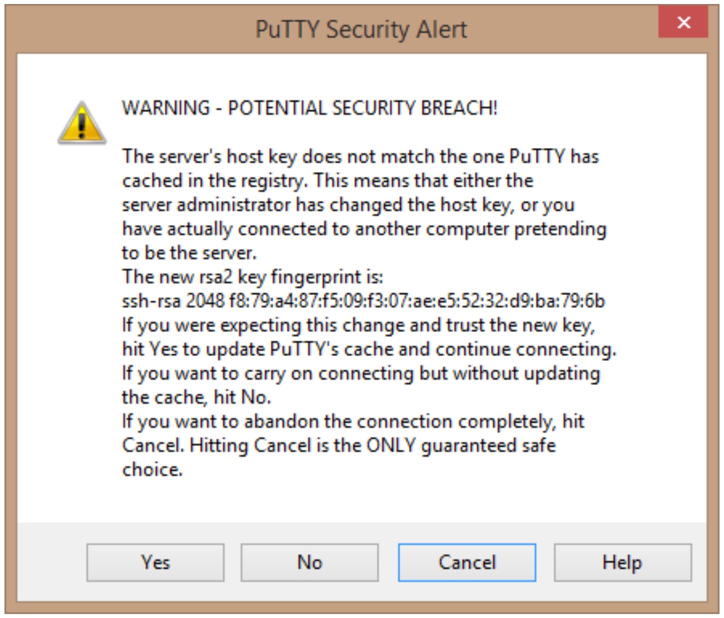
Another approach: Secure Shell (SSH)

- No trusted authorities
  - Trust on First Use
- Certificate pinning



PuTTY Security Alert

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 2048 69:20:87:cb:78:70:11:d8:53:15:72:c2:25:b0:37:0e
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.



PuTTY Security Alert

WARNING - POTENTIAL SECURITY BREACH!

The server's host key does not match the one PuTTY has cached in the registry. This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new rsa2 key fingerprint is:
ssh-rsa 2048 f8:79:a4:87:f5:09:f3:07:ae:e5:52:32:d9:ba:79:6b
If you were expecting this change and trust the new key, hit Yes to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating the cache, hit No.
If you want to abandon the connection completely, hit Cancel. Hitting Cancel is the ONLY guaranteed safe choice.

https://linode.com/docs/databases/oracle/securely-administer-oracle-xe-with-an-ssh-tunnel/
https://software.intel.com/en-us/node/734703

Summary

- Public key crypto is a powerful tool
  - Underlies https, ssh, virtually all software updates, etc…
  - But doesn’t solve the key distribution problem

- Certificate authorities (CA) occupy key (and trusted) role
  - Third-party attestation of identity or access
  - Have become hacking targets
    - 2011: Comodo & Diginotar issued fraudulent certs for Hotmail, Gmail, Skype, Yahoo Mail, Firefox…
    - 2013: TurkTrust issued cert for gmail
    - 2014: Indian Nic issued certs for Google and Yahoo!
    - 2016: WoSign issueed cert for GitHub
- Ongoing effort to police CAs

Next time

- Starting Web Security