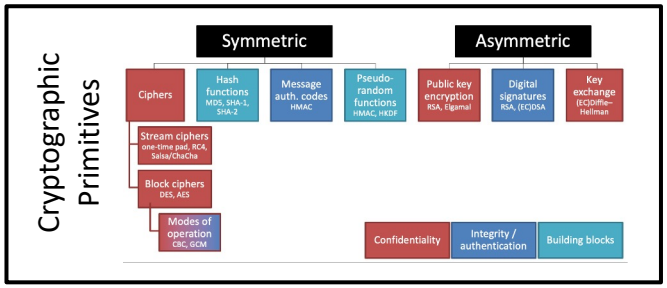
Topic 4.1 • Applications

Key management

CO 487/687

Prof. Douglas Stebila

Applications

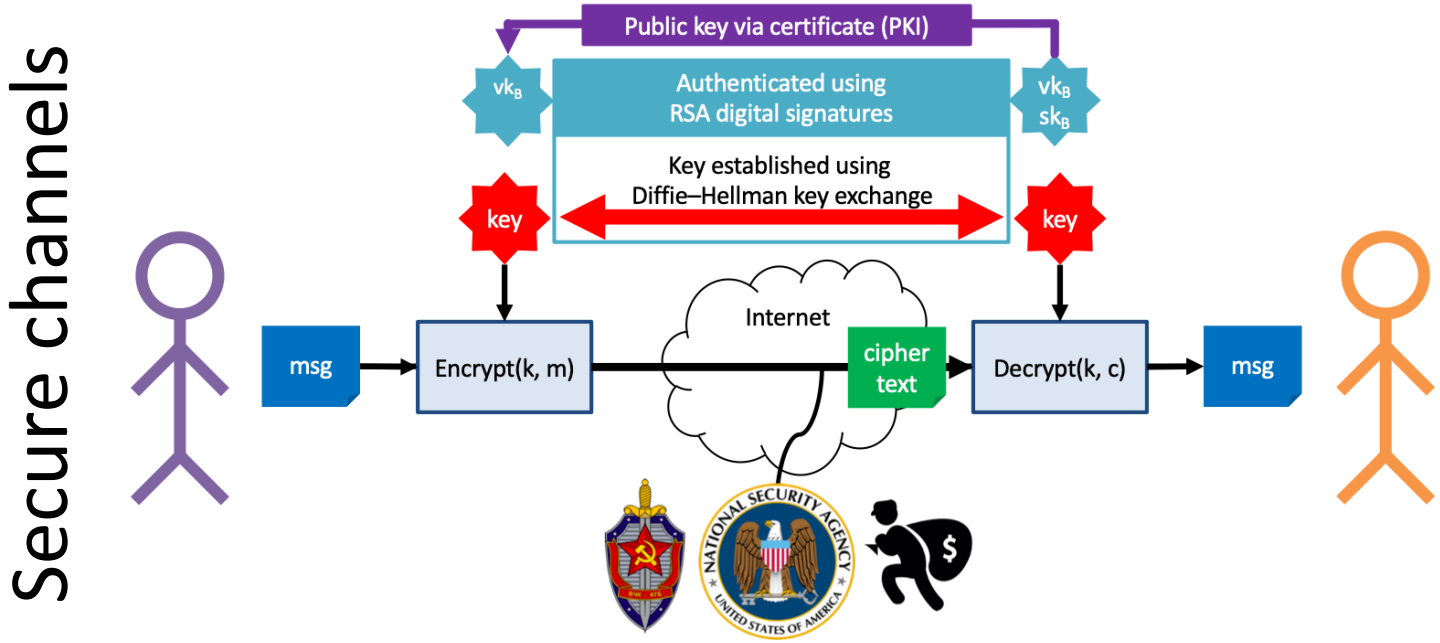


Key management
- PKI
- X.509 certificates

Secure channels
- TLS
- SSH
- Signal

Other applications
- Bitcoin
- Zero knowledge

Secure channels



The key establishment problem

- Symmetric ciphers and message authentication codes provide confidentiality and integrity against man-in-the-middle attacks
- But require a shared key between the sender and the receiver

- How to establish a shared key without a secure communication channel?

Key distribution

With asymmetric encryption

- For n parties, each party:
  - Creates their asymmetric key pair
  - Publishes their public key
  - Keeps the private key secret
- For n parties, only n key pairs must be created
- Distribute them **authentically** through out-of-band method

With symmetric encryption

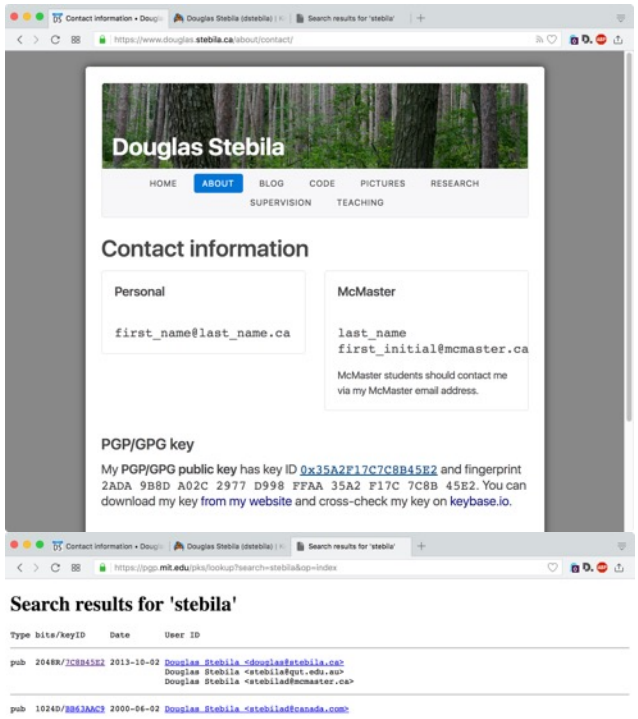
- Someone creates a secret key for each pair of communicating parties
- For n parties, n^2 secret keys must be created
- Distribute them **confidentially** through out-of-band method

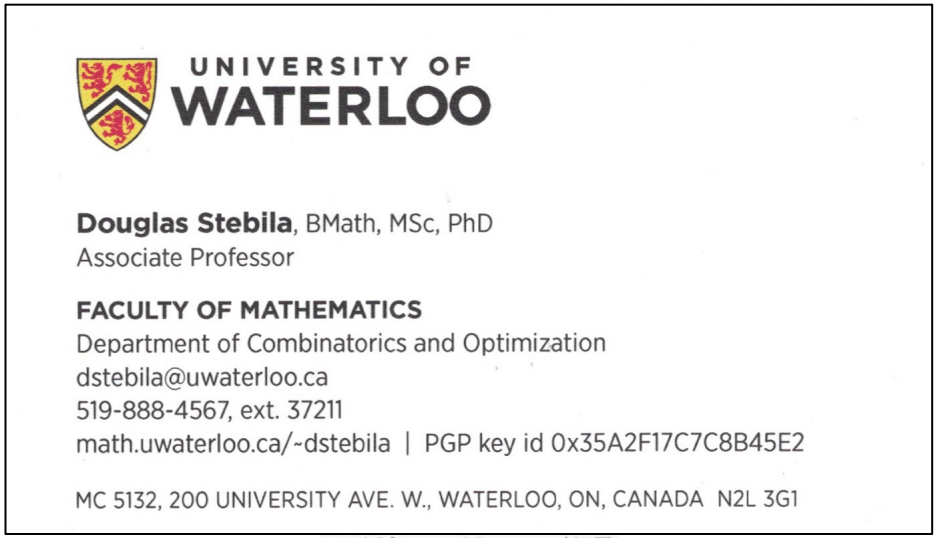
Public key distribution problem

- Man-in-the-middle who replaces public keys can then decrypt

- How can we distribute public keys authentically?
  - Especially if we don't have a basis of trust to begin with?

How to distribute public keys?





Public key distribution

- **Direct**: Get public key directly from subject
  - Example: on business card, scanning QR code

- **From a friend**
- **From a friend of a friend**
  - "Web of trust"

- **From a public directory**
  - Example: PGP key server
  - "Public key infrastructure"

Public key trust models

User-centric model

- Web of trust

- Each user maintains a **key ring** containing public keys of other users they trust
- Users are completely responsible for deciding which public keys to trust

- Examples:
  - PGP (Pretty Good Privacy)
  - GPG (GNU Privacy Guard - open source version of PGP)

Trusted authority model

- Public key infrastructure

- Trusted authorities perform checks and issue **certificates** endorsing public keys
- User trusts all certificates issued by an authority

- Examples:
  - PKI in web browsers

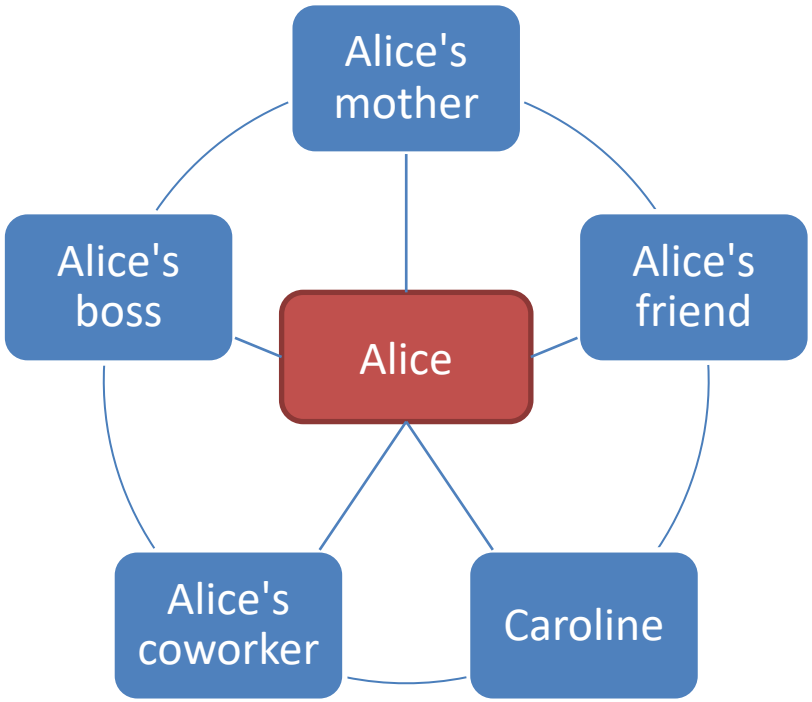
Outline

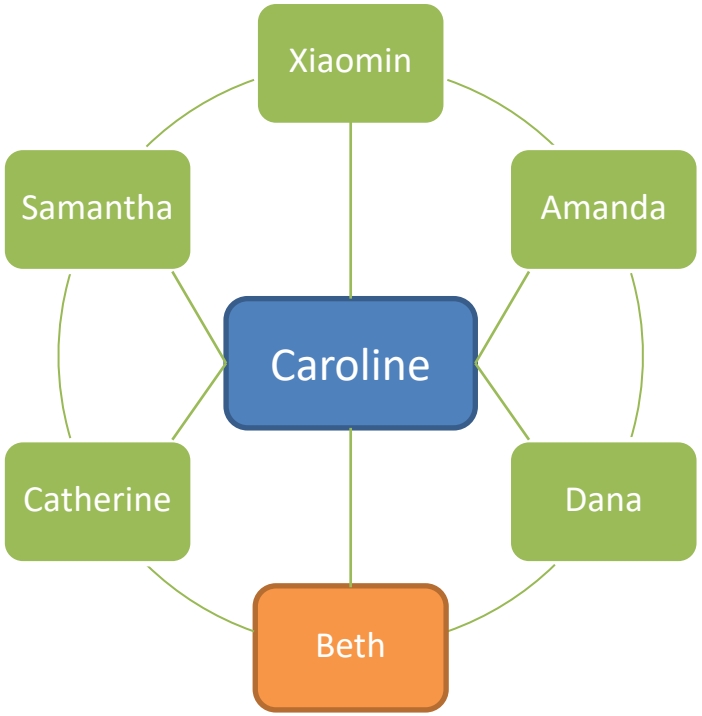
- **Web of trust**
- Public key infrastructure
  - PKI on the web

User-centric

- Each user maintains a key ring containing public keys of other users they trust
- Public keys can be distributed in a variety of ways:
  - Directly
  - Via mutual contacts
  - Key servers
- Each user decides on a case-by-case whether to trust a public key

Distributing keys via mutual contacts





Distributing keys via mutual contacts

Basic idea

- Caroline tells Alice "Here's a public key, and I verified that it belongs to Beth"

Technical implementation

- Caroline signs a statement saying "I, Caroline, assert that the following public key belongs to Beth: 5480…489"

Issuer	Caroline
Subject	Beth
Subject's public key	5480….489
Issuer's signature on all of the above	…

- Alice has a copy of Caroline's public key
- Alice verifies the signature

Key signing party - Wikipedia

en.wikipedia.org/wiki/Key_signing_party

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search Wikipedia

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF

Key signing party

From Wikipedia, the free encyclopedia

This article **does not cite any sources**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
(March 2015) (Learn how and when to remove this template message)

In public-key cryptography, a **key signing party** is an event at which people present their public keys to others in person, who, if they are confident the key actually belongs to the person who claims it, digitally sign the certificate containing that public key and the person's name, etc. Key signing parties are common within the PGP and GNU Privacy Guard community, as the PGP public key infrastructure does not depend on a central key certifying authority, but to a distributed web of trust approach. Key signing parties are a way to strengthen the web of trust. Participants at a key signing party are expected to present adequate identity documents.

Although PGP keys are generally used with personal computers for Internet-related applications, key signing parties themselves generally do not involve computers, since that would give adversaries increased opportunities for subterfuge. Rather, participants write down a string of letters and numbers, called a *public key fingerprint*, which represents their key. The fingerprint is created by a cryptographic hash function, which condenses the public key down to a string which is shorter and more manageable. Participants exchange these fingerprints as they verify each other's identification. Then, after the party, they obtain the public keys corresponding to the fingerprints they received and digitally sign them.

Key signing in front of FOSDEM 2008.

https://en.wikipedia.org/wiki/Key_signing_party

Web of trust

Advantages

- Simple
- Free
- Works well for a small number of users
- Does not require expensive infrastructure to operate

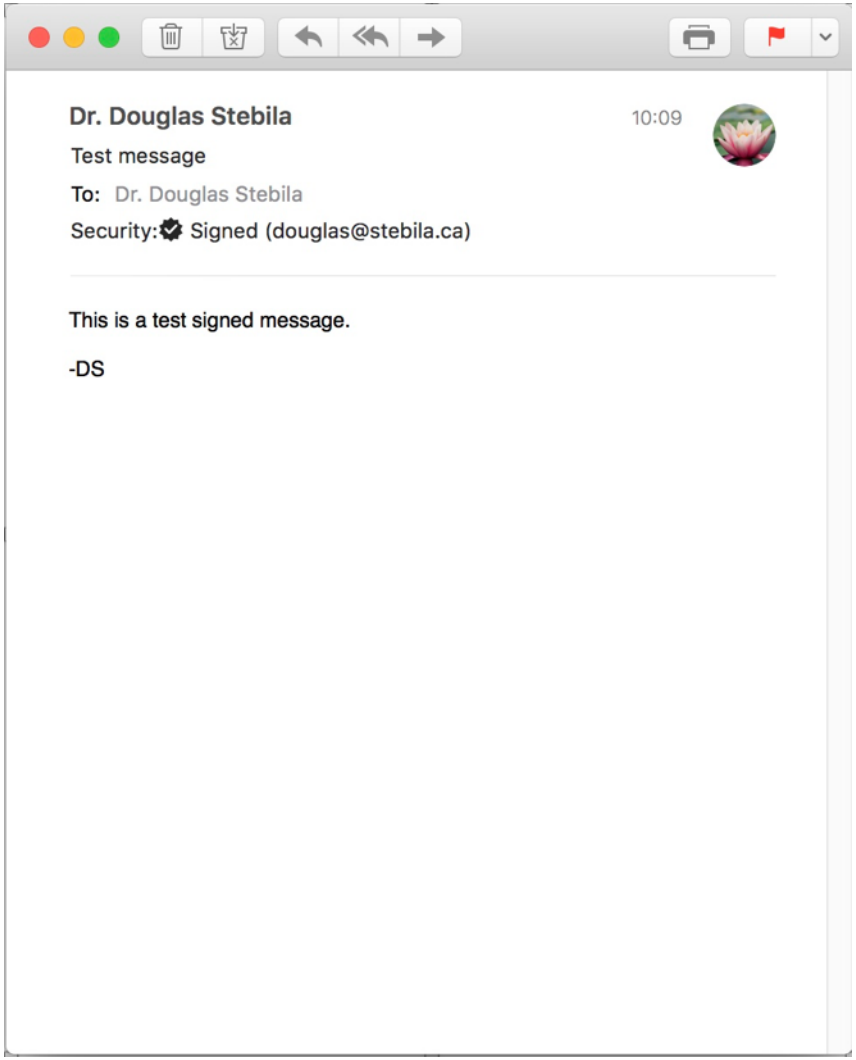
Disadvantages

- Relies on human judgement
  - Might work well with technical users aware of issues, but not general public
- Doesn't scale to large number of parties
- Not appropriate for trust-sensitive areas

PGP / GPG

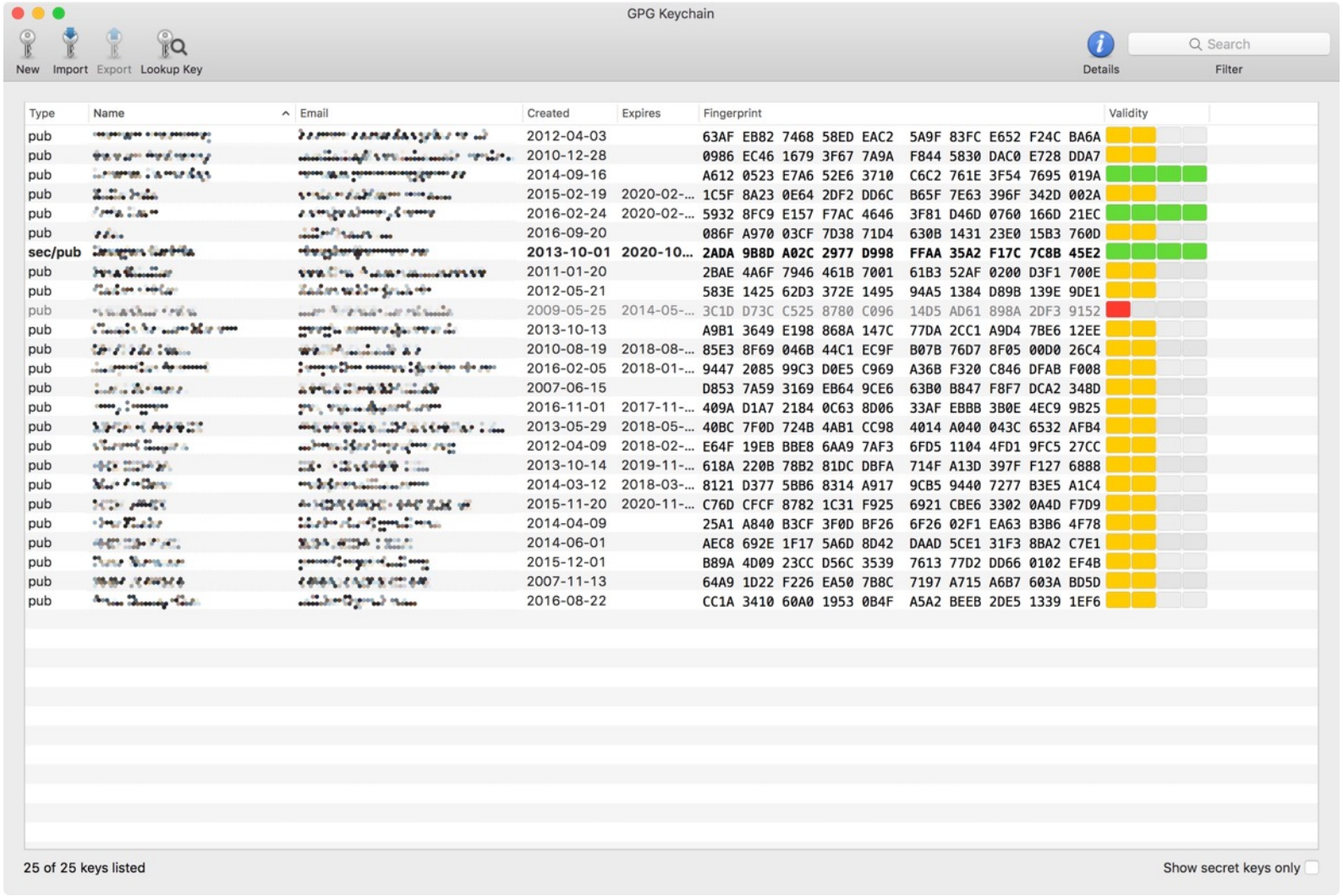
- PGP: Pretty Good Privacy
  - Invented by Phil Zimmerman in 1991
  - Software and standards for encryption and signing files and email
- OpenPGP:
  - Open standards for PGP
- GPG: GNU Privacy Guard
  - Open source implementation

PGP / GPG




```
From: "Stebila, Douglas" <dstebila@uwaterloo.ca>
To: "Stebila, Douglas" <dstebila@uwaterloo.ca>
Subject: Test message
Date: Thu, 12 Nov 2020 10:09:10 +0000
Content-Type: multipart/signed;
boundary="Apple-Mail=_421EF08F-99CC-4CC3-B07C-766A67460DCD";
protocol="application/pgp-signature"; micalg=pgp-sha512
MIME-Version: 1.0

--Apple-Mail=_421EF08F-99CC-4CC3-B07C-766A67460DCD
Content-Transfer-Encoding: 7bit
Content-Type: text/plain;
charset=us-ascii


This is a test signed message.


-DS


--Apple-Mail=_421EF08F-99CC-4CC3-B07C-766A67460DCD
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="signature.asc"
Content-Type: application/pgp-signature; name="signature.asc"
Content-Description: Message signed with OpenPGP

-----BEGIN PGP SIGNATURE-----
iQEzBAEBCgAdFiEEoUEV4SW5+wk+mWH/GYO8Cv+lJi8FAlnmDwUACgkQGYO8Cv+l
Ji8mrQgAgM+soKkr0kbORY0MMQWqN/Ykbya9pgrxNuinhI0ZNXLn2o/1S9mMESoe
s91fM3NWsfw5FotFOyAXg18WVKFL4PZ3vycmwlA1j+8JhkLms0BUgFWNmVhsBPwo
TJz2zYNBhysWrqYUX4AEkd2xGx+NH9kRLHTdWpW+YT8UCL+WBybJGCQX4psmXN/2
y26w1lMYqs1hUxbZgfEqGeXpiERE59OHC/bx92tNs2hpSS92tSEFjfMh2Lhdi81F
7ZlsjVskDFdqrOc4X35w01IDBBC6tVbeloxlCbfXIcFZp7xfqslRv0at6L7yrg+2
NRXgOodVMwquwOIibTMx9eULHu91OA==
=R/5B
-----END PGP SIGNATURE-----

--Apple-Mail=_421EF08F-99CC-4CC3-B07C-766A67460DCD--
```

GPG keychain



Outline

- Web of trust
- **Public key infrastructure**
  - PKI on the web

Certificates and certificate authorities

- Relies on trusted authorities (called **certificate authorities**) to vouch that public keys belong to certain subjects

- A **certificate** is an assertion by a trusted third party that a particular public key belongs to a particular entity.

- A **digital certificate** contains
  - The subject's identity
  - The subject's public key
  - Additional information (e.g., validity period)
  - The issuer's digital signature

Certificates and certificate authorities

The **certificate authority** generates a certificate by

1. Obtaining the subject's public key by some trusted mechanism.
2. Verifying that the subject really is who she says she is.
3. Signing (using the certificate authority's private key) the subject's public key and name.

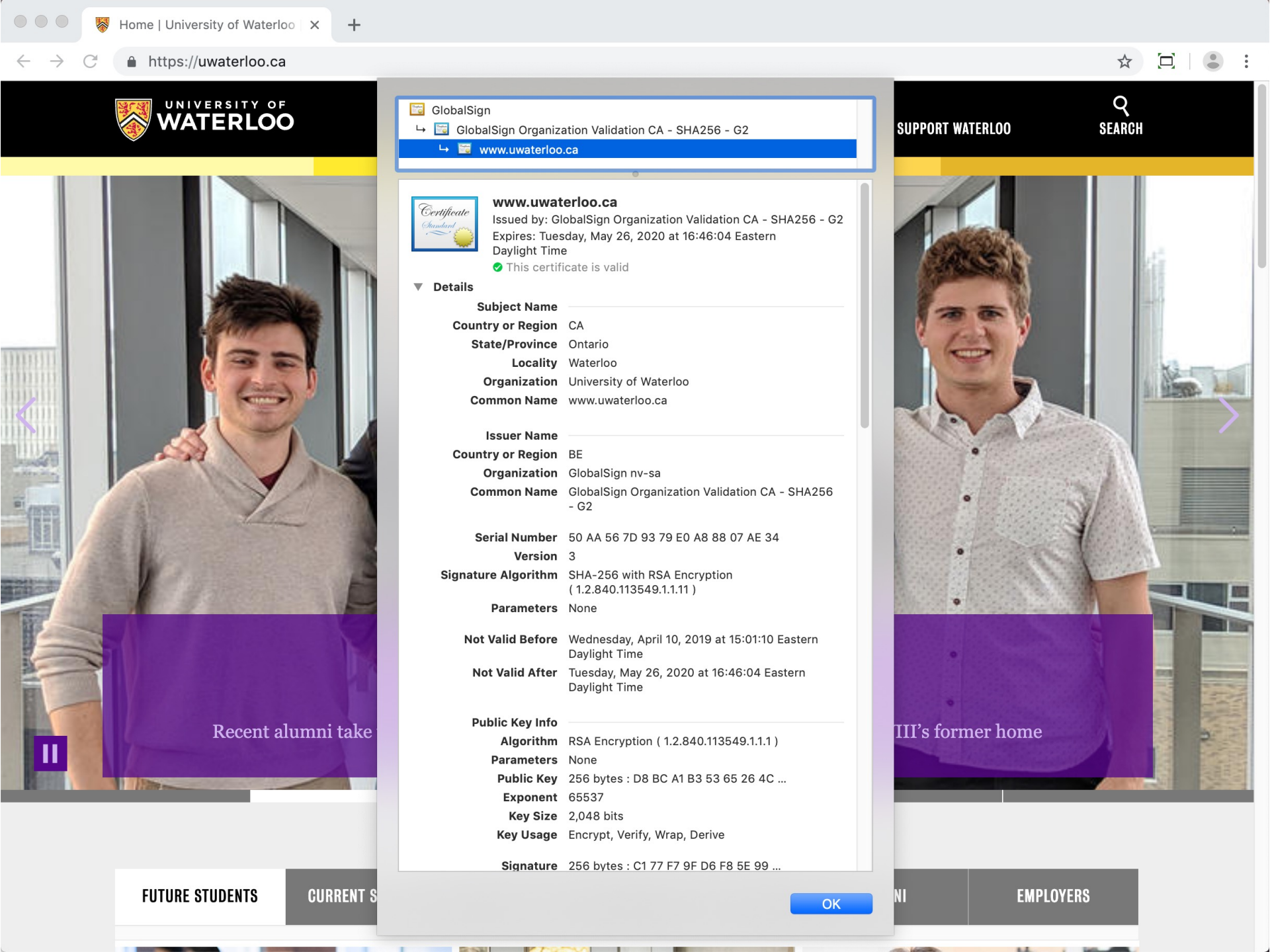
This allows two parties who have never met to establish trust between them:

- Exchange certificates.
- Do authentication using digital signatures.
- If they each trust the certificate authority that signed the other party's certificate, they can now be certain who the other party is.

X.509 certificates

- X.509 is a standard format for digital certificates
- Current version: v3
- Standardized by International Telecommunication Union (ITU-T)

- Important fields in X.509 digital certificates are:
  - Version number
  - Serial Number (set by the CA)
  - Signature Algorithm identifier (Algorithm used for dig sigs)
  - Issuer (Name of the CA)
  - Subject (Name of entity to which certificate has been issued)
  - Subject Public Key Information
  - Validity period (certificate should not be used outside this time)
  - Digital signature (of the certificate, signed by the CA)


Home | University of Waterloo
https://uwaterloo.ca
UNIVERSITY OF WATERLOO
SUPPORT WATERLOO
SEARCH
GlobalSign
GlobalSign Organization Validation CA - SHA256 - G2
www.uwaterloo.ca
www.uwaterloo.ca
Issued by: GlobalSign Organization Validation CA - SHA256 - G2
Expires: Tuesday, May 26, 2020 at 16:46:04 Eastern Daylight Time
This certificate is valid
Details
Subject Name
Country or Region CA
State/Province Ontario
Locality Waterloo
Organization University of Waterloo
Common Name www.uwaterloo.ca
Issuer Name
Country or Region BE
Organization GlobalSign nv-sa
Common Name GlobalSign Organization Validation CA - SHA256 - G2
Serial Number 50 AA 56 7D 93 79 E0 A8 88 07 AE 34
Version 3
Signature Algorithm SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters None
Not Valid Before Wednesday, April 10, 2019 at 15:01:10 Eastern Daylight Time
Not Valid After Tuesday, May 26, 2020 at 16:46:04 Eastern Daylight Time
Public Key Info
Algorithm RSA Encryption (1.2.840.113549.1.1.1)
Parameters None
Public Key 256 bytes : D8 BC A1 B3 53 65 26 4C ...
Exponent 65537
Key Size 2,048 bits
Key Usage Encrypt, Verify, Wrap, Derive
Signature 256 bytes : C1 77 F7 9F D6 F8 5E 99 ...
OK
Recent alumni take
III's former home
FUTURE STUDENTS
CURRENT S
NI
EMPLOYERS

Domain name

Certificate authority

Validity period

X.509 certificates

A standardized format for certificates.

Uses a strange (old) format called ASN.1 and a strange binary encoding.

Public key

Revocation information

CA's signature on everything

www.uwaterloo.ca

Issued by: GlobalSign Organization Validation CA - SHA256 - G2

Expires: Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time

This certificate is valid

Details

Subject Name	
Country	CA
State/Province	Ontario
Locality	Waterloo
Organization	University of Waterloo
Common Name	www.uwaterloo.ca
Issuer Name	
Country	BE
Organization	GlobalSign nv-sa
Common Name	GlobalSign Organization Validation CA - SHA256 - G2
Serial Number	2E 4B 75 8D 35 75 9F A0 27 1F F1 BC
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	none
Not Valid Before	Thursday, December 1, 2016 at 16:26:05 Eastern Standard Time
Not Valid After	Monday, March 26, 2018 at 16:46:04 Eastern Daylight Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	none
Public Key	256 bytes : D8 BC A1 B3 53 65 26 4C 39 D5 92 56 84 66 37 CC 1F 57 FD 5B 87 A7 38 36 D5 05 83 3E 6C 96 02 12 3E 6A C3 CF F3 BC 3C 6D BF FE BB BB 08 02 8C 97 AF D9 86 2A 6B F6 EE D7 0C DA E8 2F DA B1 14 E8 B5 EA 04 FF 12 3D BA ED 42 FA CE A7 93 AC 15 29 66 63 2E 39 7F F2 69 D7 82 01 CB B8 92 81 75 B3 F9 4A 87 32 05 67 E0 42 78 55 1F 03 17 A8 8F 6E 85 56 1B 1C AF 8E 35 A5 14 91 E9 25 61 AC 05 6F 9A FC 58 F8 7F 64 BE C7 D4 6A EB 2A BC 47 D6 30 35 51 1D BD 57 09 49 19 9E BC 43 09 F1 58 0C 88 E5 D1 9C CB 00 AA A8 66 E8 4B C9 CE AA 63 63 5A A9 AF 3D 63 90 E8 7A 2F 95 1B CC EC 2E 48 16 4A 0E B8 1F 69 45 82 3C F1 09 53 2C B6 69 8C 70 4C 99 89 6F 4E CA 0C 8D F5 1E 3A 5F 07 46 7D 63 ED 3D 38 B7 0E 88 ED 4F FD 00 C2 76 35 F7 99 5B 39 CE 26 CC C4 19 CA 47 DA 6D 80 61 7E 01 8E 96 DD
Exponent	65537
Key Size	2048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 5B 01 1C 81 17 01 07 2F ...

Extension	Key Usage (2.5.29.15)
Critical	YES
Usage	Digital Signature, Key Encipherment
Extension	Basic Constraints (2.5.29.19)
Critical	NO
Certificate Authority	NO
Extension	Extended Key Usage (2.5.29.37)
Critical	NO
Purpose #1	Server Authentication (1.3.6.1.5.5.7.3.1)
Purpose #2	Client Authentication (1.3.6.1.5.5.7.3.2)
Extension	Subject Key Identifier (2.5.29.14)
Critical	NO
Key ID	DB B9 21 BC CD 3E AF 70 C9 E9 3D 9B FF 42 B0 C8 88 8F 78 C3
Extension	Authority Key Identifier (2.5.29.35)
Critical	NO
Key ID	96 DE 61 F1 BD 1C 16 29 53 1C C0 CC 7D 3B 83 00 40 E6 1A 7C
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	www.uwaterloo.ca
DNS Name	uwaterloo.ca
Extension	Certificate Policies (2.5.29.32)
Critical	NO
Policy ID #1	(1.3.6.1.4.1.4146.1.20)
Qualifier ID #1	Certification Practice Statement (1.3.6.1.5.5.7.2.1)
CPS URI	https://www.globalsign.com/repository/
Policy ID #2	(2.23.140.1.2.2)
Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://crl.globalsign.com/gs/gsorganizationvalsha2g2.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://secure.globalsign.com/cacert/gsorganizationvalsha2g2r1.crt
Method #2	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://ocsp2.globalsign.com/gsorganizationvalsha2g2

Fingerprints	
SHA-256	C7 DC B4 CD 45 9E D5 1A AA 03 86 73 31 4B F8 A9 53 A6 9C F1 B9 C4 35 A3 AD C6 4F 87 97 93 AD D6
SHA-1	53 9E 06 03 64 F9 24 F6 ED 9B A1 0E F9 46 81 1A E0 88 F8 A5

Certificate revocation

- Once a certificate's been issued, what happens if the user's private key has been compromised?

- We would like to be able to **revoke** the certificate, or indicate that it should no longer be trusted.

Certificate revocation mechanisms

Certificate Revocation Lists (CRLs)

- Each CA can publish a file containing a list of certificates that have been revoked.
- Have to download whole list.
- CRL address often included in certificate.

Online Certificate Status Protocol (OCSP)

- An online service run by a CA for checking in real-time if a certificate has been revoked.
- Don’t have to download whole list.
- Not widely implemented.
- Compromises user privacy
  - As of 2024, starting to be deprecated on the web

Public key infrastructure

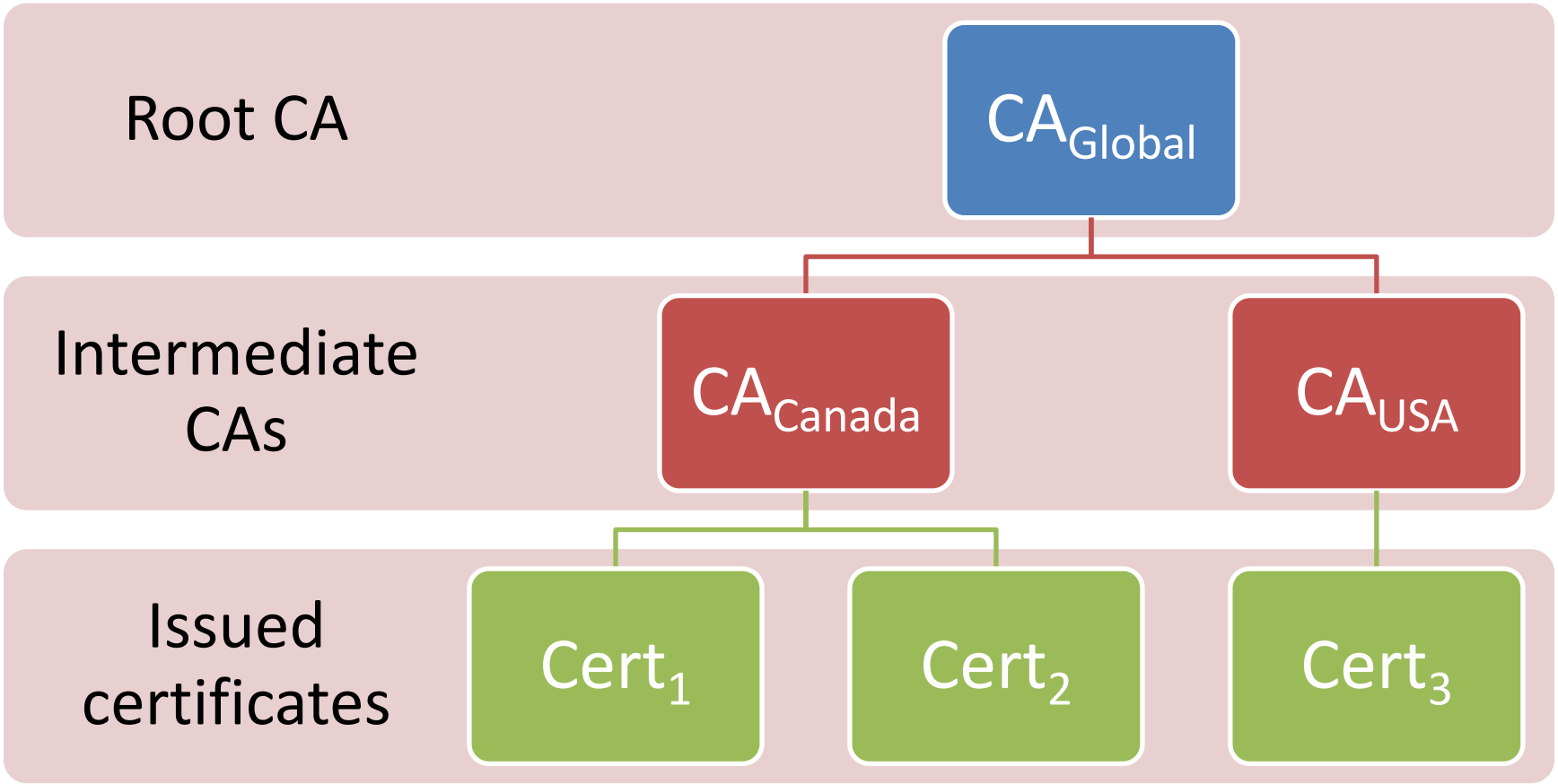
A **public key infrastructure (PKI)** is

- a set of systems (hardware, software, policies, procedures)
- for managing (creating, distributing, storing, revoking)
- digital certificates.

Includes:

- one or more certificate authorities
- subjects
- users
- relying parties
- possibly a timestamp server
- possibly a directory server storing certificates (e.g., LDAP server, Active Directory server)

Hierarchical CAs



Using certificates for confidentiality

- Suppose Alice wants to send a message confidentially to Bob
  1. Alice needs Bob's public key
     1. Alice obtains $Cert_{Bob}$, signed by CA_1
     2. Alice checks that the identity in $Cert_{Bob}$ is the Bob she wants
     3. Alice verifies CA_1's signature on $Cert_{Bob}$ using CA_1's public key
     4. Alice extracts pk_{Bob} from $Cert_{Bob}$
  2. Alice uses pk_{Bob} to encrypt message M for Bob

- Does this provide confidentiality – can only Bob read the message?
  - If Alice trusts the CA that issued $Cert_{Bob}$ to
    - Check the identity of subjects before issuing certificates
    - Not issue fraudulent certificates
  - And Alice is certain of the CA's public key
  - Then Alice can be sure that only Bob will be able to decrypt the message

Using certificates for authentication/integrity

- Suppose Alice wants to check if a message really came from Bob
  1. Alice needs Bob's public key
     1. Alice obtains $Cert_{Bob}$, signed by CA_1
     2. Alice checks that the identity in $Cert_{Bob}$ is the Bob she wants
     3. Alice verifies CA_1's signature on $Cert_{Bob}$ using CA_1's public key
     4. Alice extracts pk_{Bob} from $Cert_{Bob}$
  2. Alice uses pk_{Bob} to verify the signature on a given message supposedly from Bob

- Does this provide integrity– can only Bob send messages?
  - If Alice trusts the CA that issued $Cert_{Bob}$ to
    - Check the identity of subjects before issuing certificates
    - Not issue fraudulent certificates
  - And Alice is certain of the CA's public key
  - Then Alice can be sure that only Bob will be able to sign messages that verify

Trustworthiness of CAs

- We assume that CAs
  - Check the identity of subjects before issuing certificates
  - Don't issue fraudulent certificates
  - Protect their own signing key

Applications of PKI

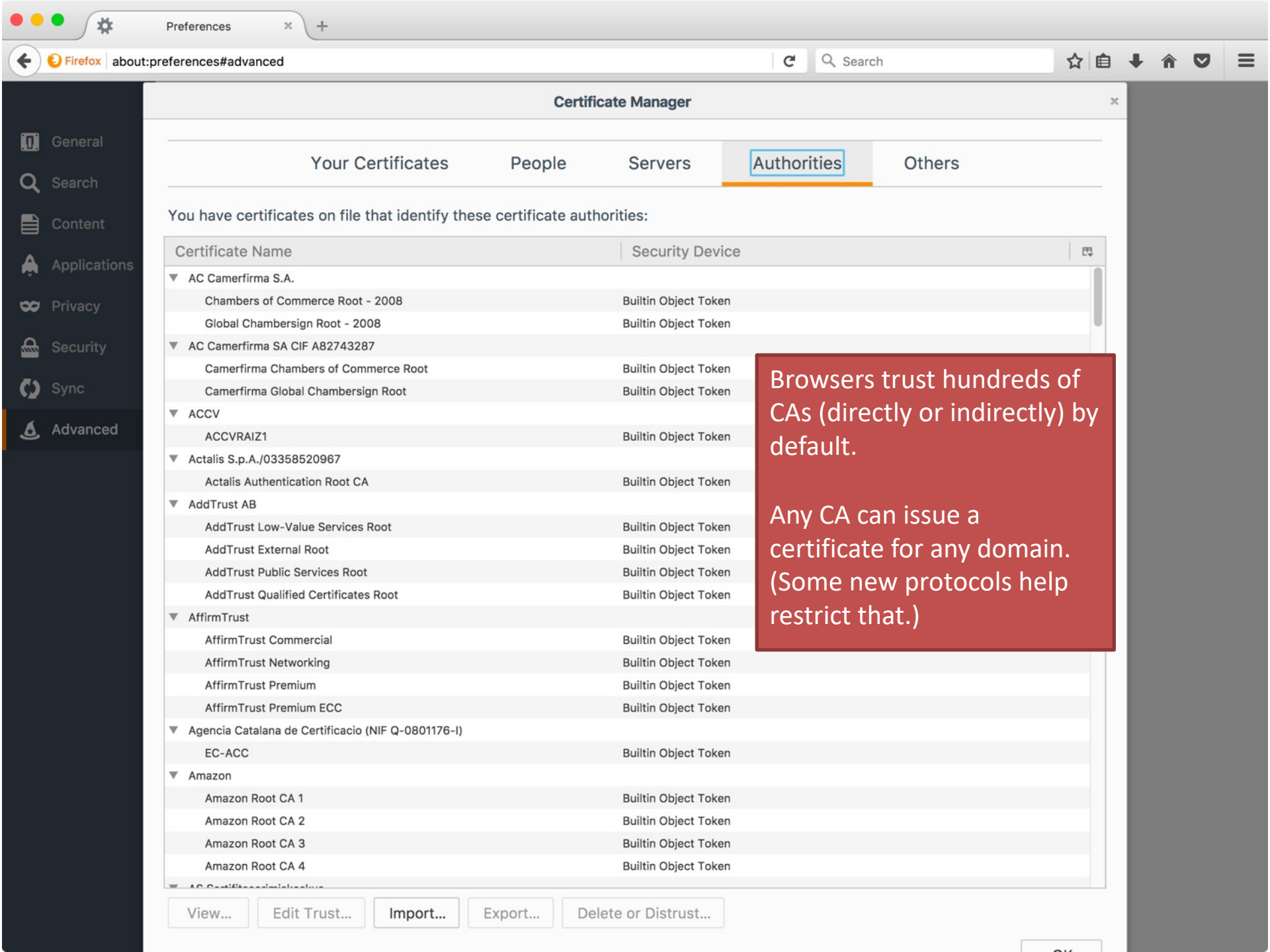
- Web site authentication (TLS)
- Email authentication (S/MIME, PGP)
- Domain names (DNSSEC)
- Digital identities
  - e.g., national identity cards (Belgium, Spain, Germany)
- Business-to-business e-commerce
  - e.g., digitally signing transactions, XML signatures

Secure email

- X.509 certificates can also be used to send secure email:
  - digitally signed
  - encrypted
- **S/MIME** (Secure/Multipurpose Internet Mail Extensions):
  - Supported in most desktop mail programs.
  - Relies on a public key infrastructure.
- **PGP** (Pretty Good Privacy):
  - Available as an add-on to most desktop mail programs.
  - Uses public keys, but doesn't require CAs: users manually distribute their keys in a "web of trust"
- Not widely used:
  - Users must know how set up public keys and obtain S/MIME X.509 certificate or distribute PGP public keys.
  - Little to no support in webmail.

Outline

- Web of trust
- Public key infrastructure
  - **PKI on the web**


Preferences
Firefox about:preferences#advanced
Search
General
Search
Content
Applications
Privacy
Security
Sync
Advanced
Certificate Manager
Your Certificates
People
Servers
Authorities
Others
You have certificates on file that identify these certificate authorities:
Certificate Name
Security Device
AC Camerfirma S.A.
Chambers of Commerce Root - 2008
Builtin Object Token
Global Chambersign Root - 2008
Builtin Object Token
AC Camerfirma SA CIF A82743287
Camerfirma Chambers of Commerce Root
Builtin Object Token
Camerfirma Global Chambersign Root
Builtin Object Token
ACCV
ACCVRAIZ1
Builtin Object Token
Actalis S.p.A./03358520967
Actalis Authentication Root CA
Builtin Object Token
AddTrust AB
AddTrust Low-Value Services Root
Builtin Object Token
AddTrust External Root
Builtin Object Token
AddTrust Public Services Root
Builtin Object Token
AddTrust Qualified Certificates Root
Builtin Object Token
AffirmTrust
AffirmTrust Commercial
Builtin Object Token
AffirmTrust Networking
Builtin Object Token
AffirmTrust Premium
Builtin Object Token
AffirmTrust Premium ECC
Builtin Object Token
Agencia Catalana de Certificacio (NIF Q-0801176-I)
EC-ACC
Builtin Object Token
Amazon
Amazon Root CA 1
Builtin Object Token
Amazon Root CA 2
Builtin Object Token
Amazon Root CA 3
Builtin Object Token
Amazon Root CA 4
Builtin Object Token
View...
Edit Trust...
Import...
Export...
Delete or Distrust...
Browsers trust hundreds of CAs (directly or indirectly) by default.
Any CA can issue a certificate for any domain. (Some new protocols help restrict that.)

Certificate types

Domain validation

- Identity confirmed by validating control over DNS record
  - Let's Encrypt $0
  - Comodo $77
  - Thawte $149

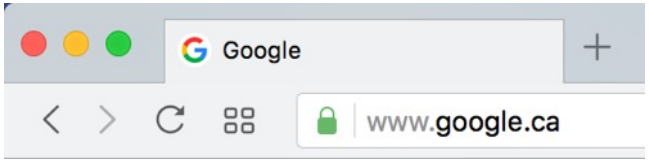
Organization validation

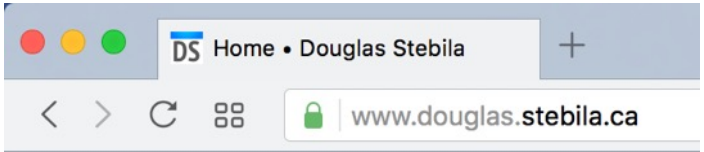
- Identity confirmed by some checks of legal status of organization
  - Symantec $995
  - Thawte $199

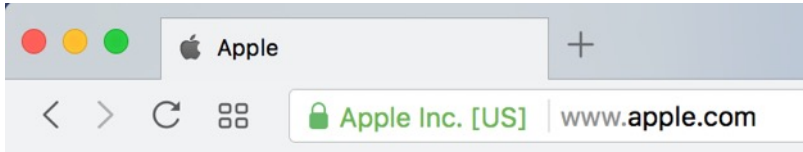
Extended validation

- More rigourous check of organization's existence
  - Symantec $995
  - Thawte $299

Eye-tracking studies show that users do not notice these additional security indicators







Certificate authority breaches and errors

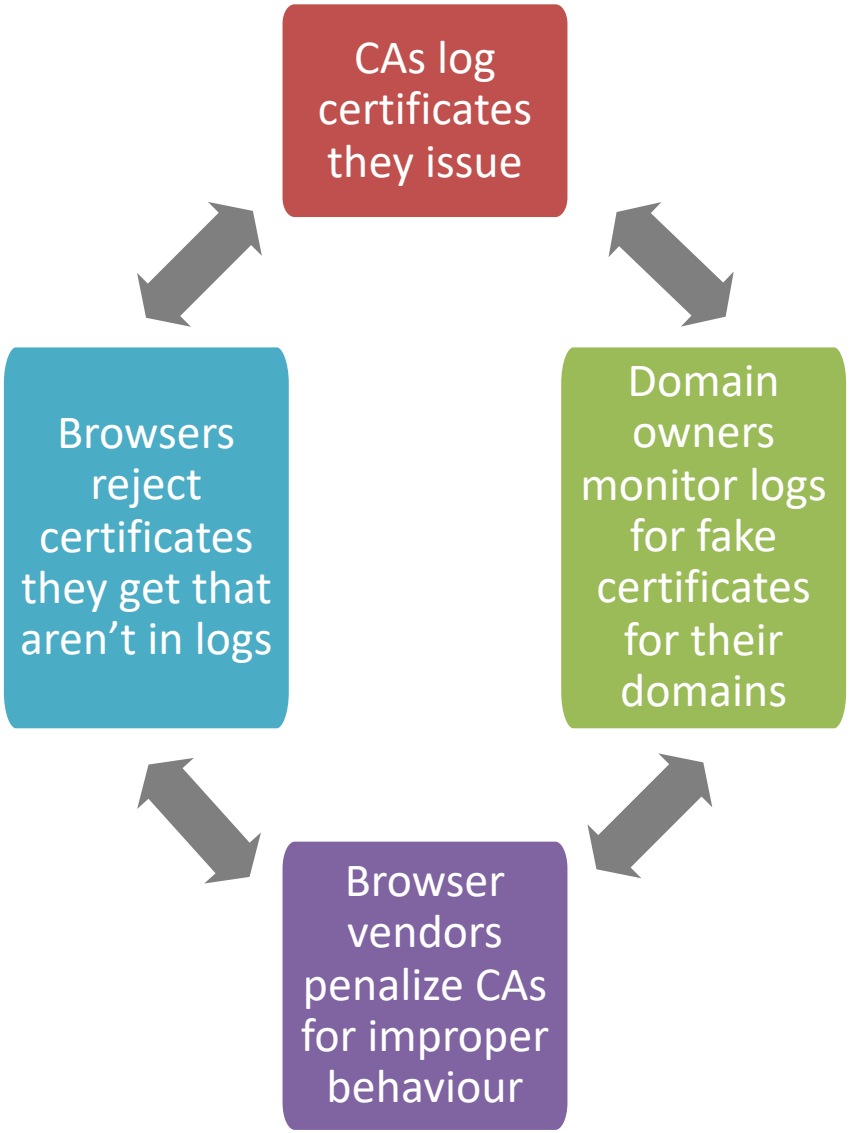
- Verisign in 2001
  - Issued 2 certificates for "Microsoft" to someone not from Microsoft
- DigiNotar in Jul. 2011
  - Security breach, malicious certificates for many domains issued
    - Including "wildcard" certificate for *.google.com
  - Observed in use by multiple customers of Iranian ISPs
  - Also affected a Dutch national PKI
  - Went out of business
- TURKTRUST in Aug. 2011
- Digicert Malaysia in Nov. 2011
- KPN/Getronics in Nov. 2011
- Trustwave in Feb. 2012
- Symantec in 2015
  - https://security.googleblog.com/2015/10/sustaining-digital- certificate-security.html
- Entrust in 2024
  - https://security.googleblog.com/2024/06/sustaining-digital-certificate-security.html
- ...

CA/Browser Forum

- Voluntary consortium of CAs and browser vendors
- Issue guidelines for CA management and procedures
  - Effectively requirements for CAs to have their certificates installed in browsers

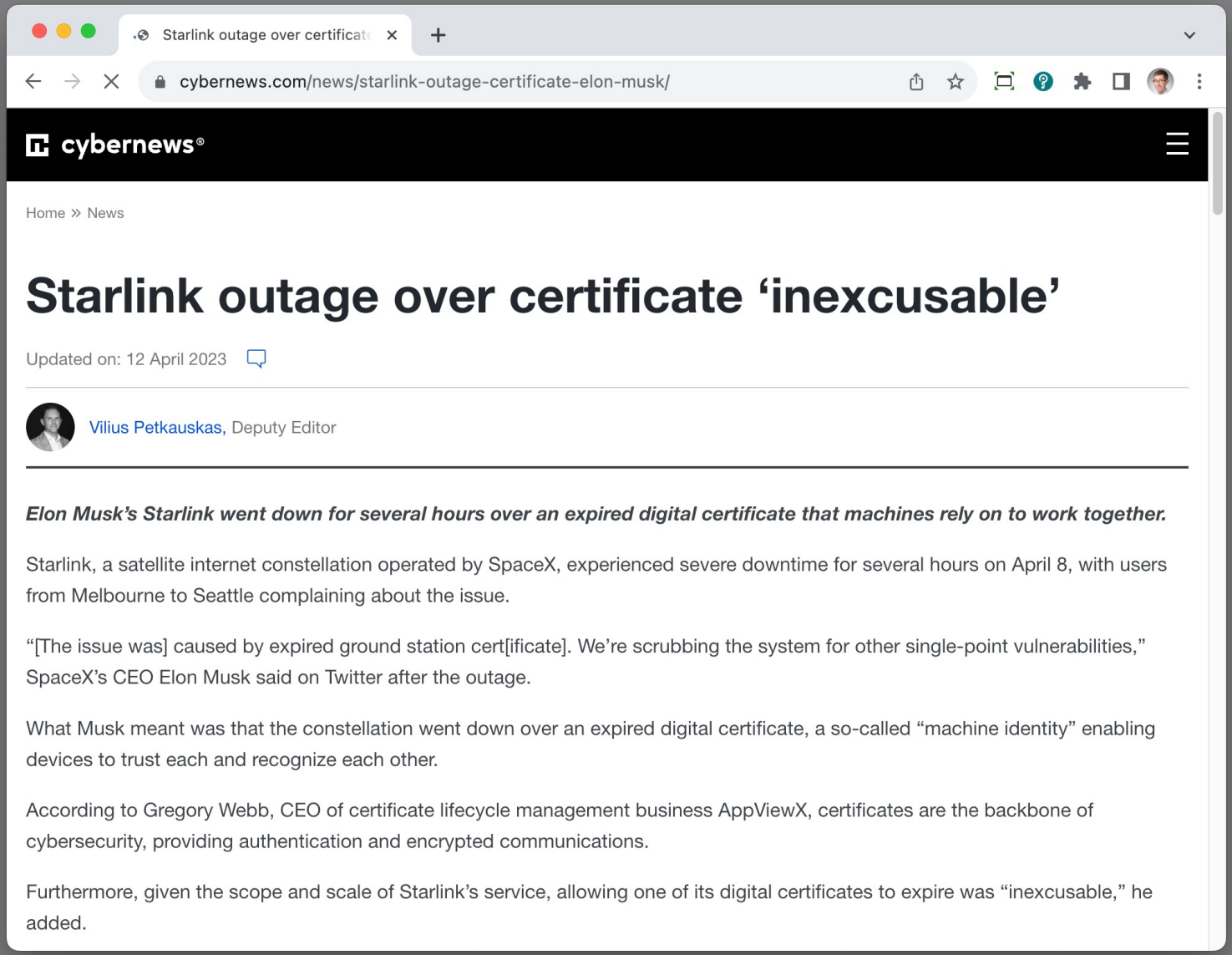
Certificate Transparency

- An auditing mechanism to monitor the behaviour of CAs
- Doesn't prevent malicious behaviour by CAs, but increases chance of detection and penalization

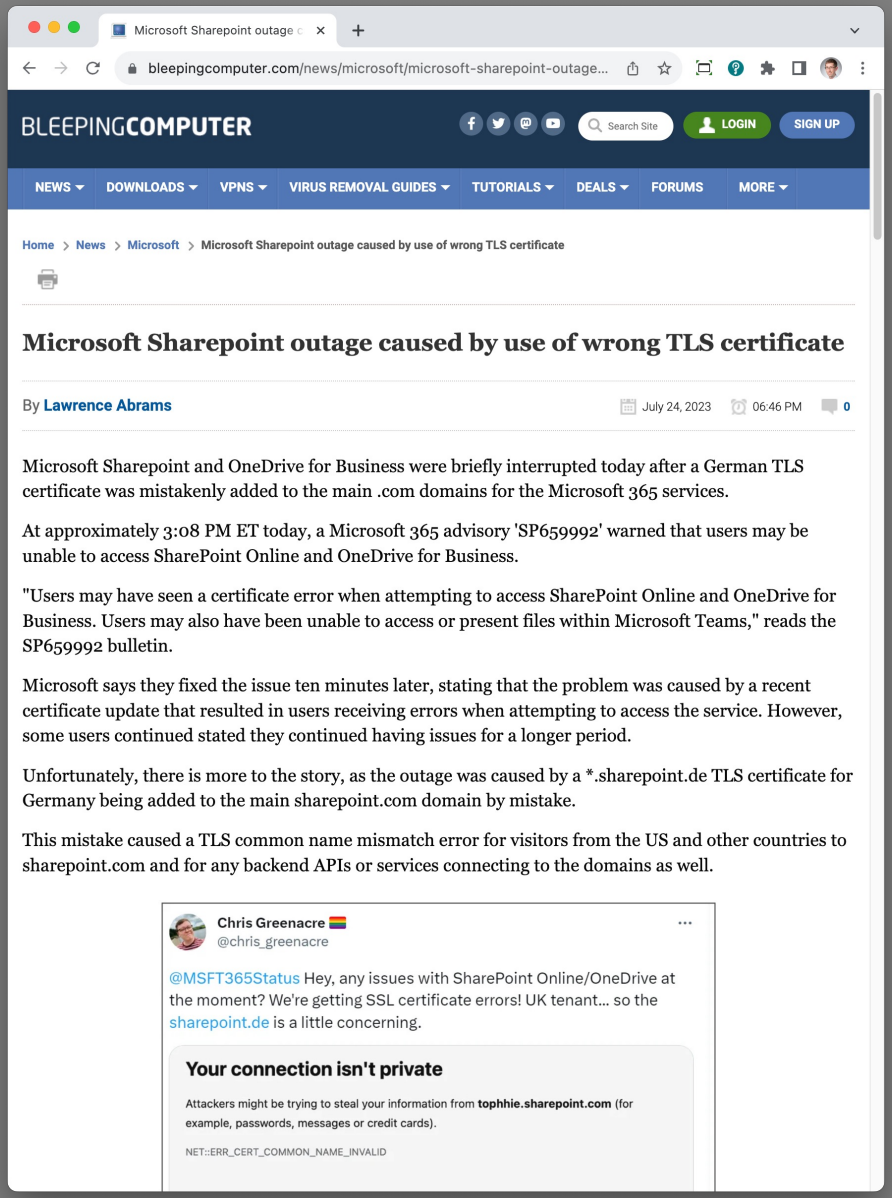


Let's Encrypt

- Provides free X.509 certificates for web sites
- Uses automated issuance process
  - "ACME protocol"
  - Command-line tool can automatically configure Apache web server
- Relies on domain validation
- 411 million active certificates as of November 2024



https://cybernews.com/news/starlink-outage-certificate-elon-musk/



https://www.bleepingcomputer.com/news/microsoft/microsoft-sharepoint-outage-caused-by-use-of-wrong-tls-certificate/

4+8+7 things to remember from CO 487

CO 487/687 • Fall 2024

Public key infrastructure

Things to remember

A PKI allows us to distribute public keys by relying on a trusted **certificate authority** to issue credentials linking a public key to a particular party.

A **certificate** (usually X.509 format) is an assertion, signed by the CA, that a particular public key belongs to a particular party.

CA public keys distributed in advance (e.g., built in to browser/OS).