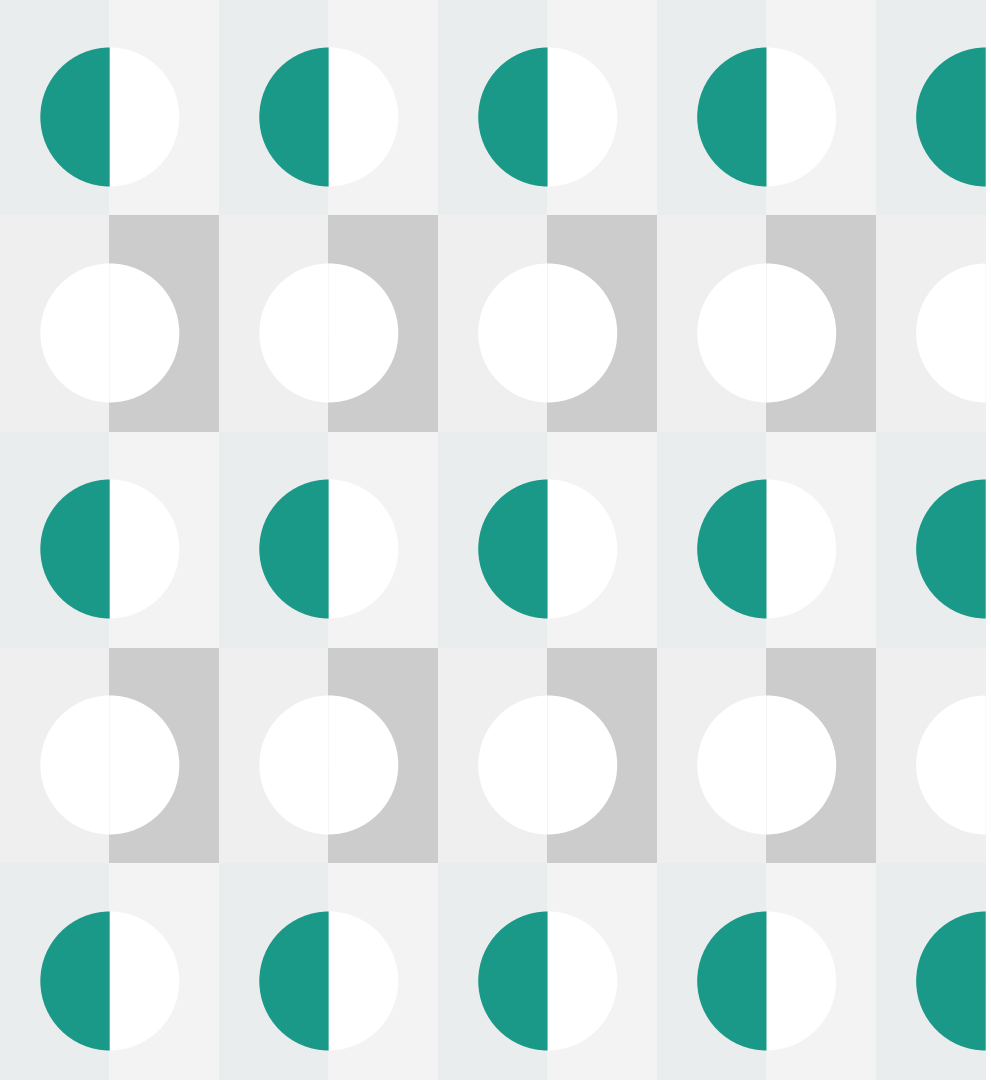
Section 5
Public Key Crypto Topics: RSA, ECC, CAs

Administrivia

- Homework 2 due next Wednesday (02-10)
  - Individual assignment
  - Hands-on cryptography
- Final Project checkpoint #1 due next next Wednesday (02-17)
  - Group members' names and UWNetIDs
  - Presentation topic

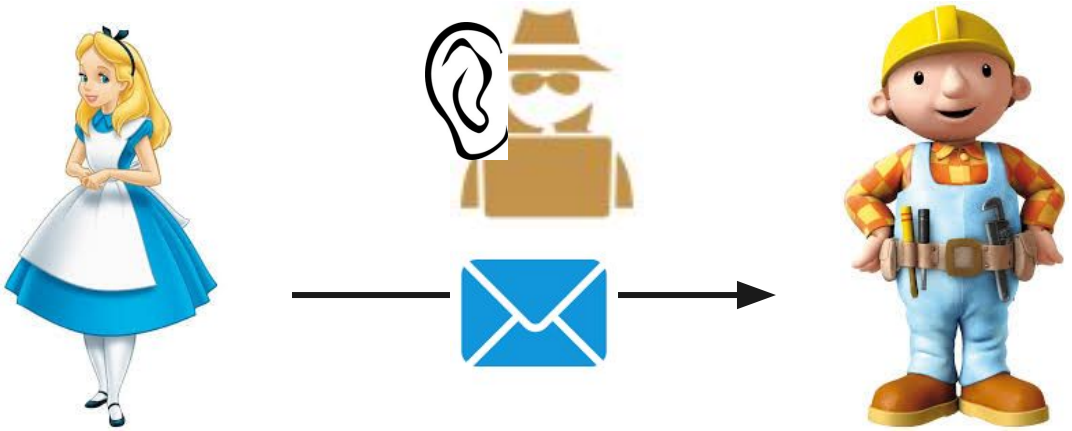


RSA: Review, Practice, and Future

Public Key Cryptography Review

Alice wants to send Bob an encrypted message

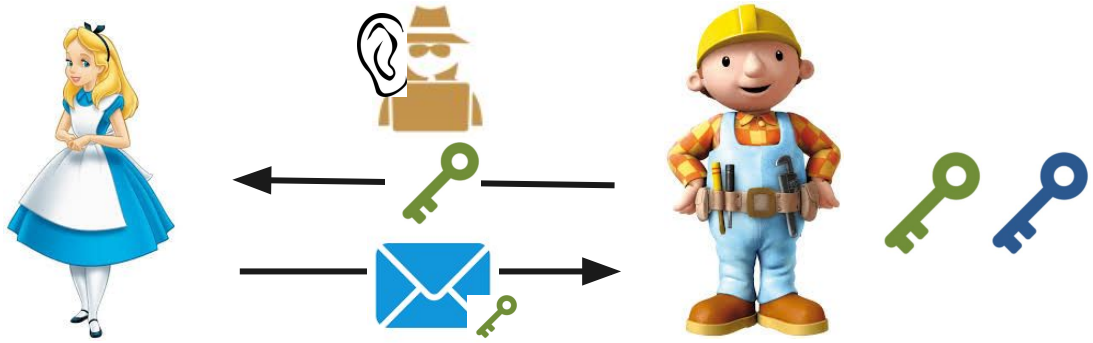
- Goal: Confidentiality
- Problem: Eve can intercept key



Public Key Cryptography Review

Solution: public key cryptography (aka **asymmetric** cryptography)

- Public-private keypair
- Alice encrypts using Bob's public key
- Bob decrypts using Bob's private key



RSA Cryptosystem Review

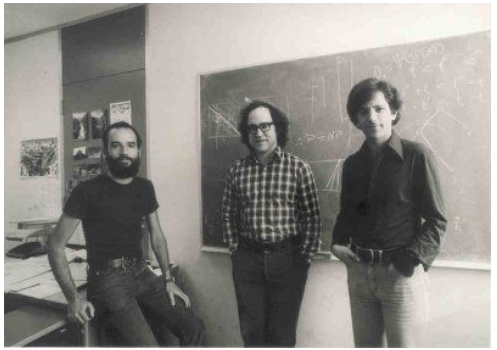
Key generation:

- Generate large primes p, q
- Compute $N=pq$ and $\varphi(N)=(p-1)(q-1)$
- Choose e coprime to $\varphi(N)$
  - Typically $e=3$ or $e=2^{16}+1=65537$
- Find (unique) d such that $ed \equiv 1 \pmod{\varphi(N)}$

Public key = (e, N); Private key = (d, N)

Encryption of m: $c = m^e \bmod N$

Decryption of c: $c^d \bmod N = (m^e \bmod N)^d \bmod N = m^1 \bmod N = m$



Adi **S**hamir, Ron **R**ivest, Len **A**dleman
[Photo from Dan Wright]

RSA Practice

Public key: N = 33, e = 7

Step 1: Find $\varphi(N)$

Step 2: Find the decryption key, d

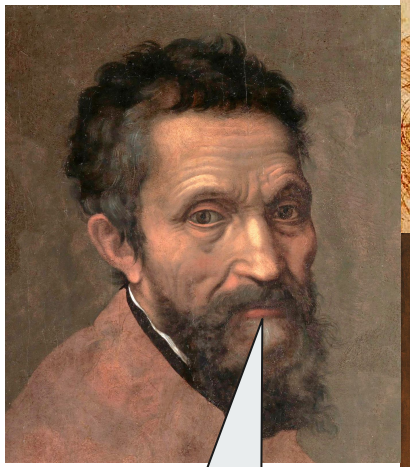
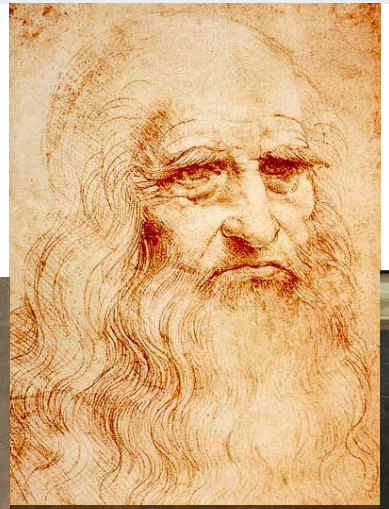
- $ed \equiv 1 \pmod{\varphi(N)}$

Step 3: Decrypt the cryptogram

- $c^d \bmod N = m$
- ‘A’ = 1, ‘B’ = 2, ...

Cryptograms:

```
12 14 27 20 1 6 16 27
6 1 25 2 1 14 12
7 15 9 2 14 12 1 20 28 14 12 27
16 27 20 1 26 14 12 12 27
```


Cowabunga!



TEENAGE MUTANT NINJA
TURTLES

RSA Strength

"RSA problem": decrypt only using the public key

- Factoring N is hard
- No known efficient algorithm
- Trapdoor function: easy to go forward, hard to go back

RSA Factoring Challenge (1991-2007)

- Cash prizes for factoring large N values (up to $200,000 (!))
- Only the smallest 23 of 54 factored so far...

Shor's Algorithm

- Quantum computer algorithm to factor integers
- Largest number factored so far: 21 😎

RSA-2048:

```
25195908475657893494027183240
04839857142928212620403202777
71378360436620207075955562640
18525880784406918290641249515
08218929855914917618450280848
91200728449926873928072877767
35971418347270261896375014971
82469116507761337985909570009
73304597488084284017974291006
42458691817195118746121515172
65463228221686998754918242243
36372590851418654620435767984
23387184774447920739934236584
82382428119816381501067481045
16603773060562016196762561338
44143603833904414952634432190
11465754445417842402092461651
57233507787077498171257724679
62926386356373289912154831438
16789988504044536402352738195
13786365643912120103971228221
20720357
```

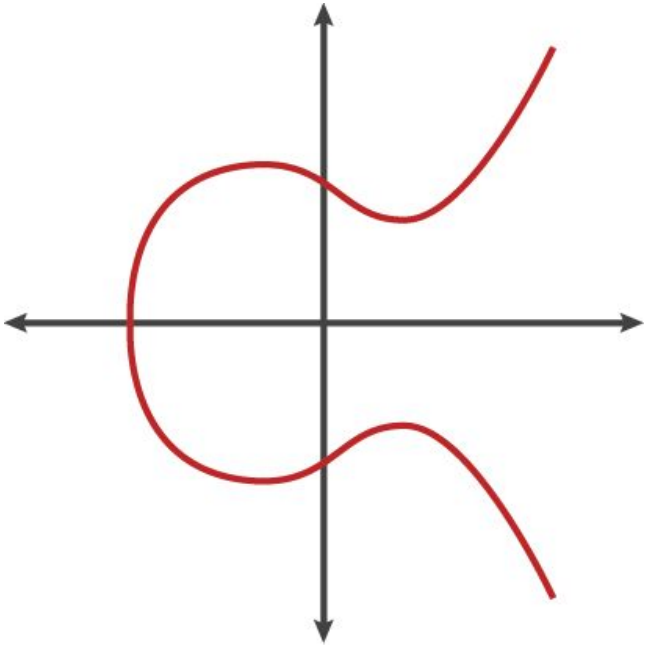
RSA Today

- Still usable in practice
  - SSH keys, TLS, etc.
  - But not preferred...
- Need big keys for RSA
  - At least 2048 bits
- Bigger keys ⇒ slower computation
- Can we do better?

Elliptic-Curve Cryptography (ECC)

$y^2 = x^3 + ax + b$

- First suggested independently by Neal Koblitz (UW Math faculty!) and Victor S. Miller in 1985
- Widespread adoption started in the last 2 decades



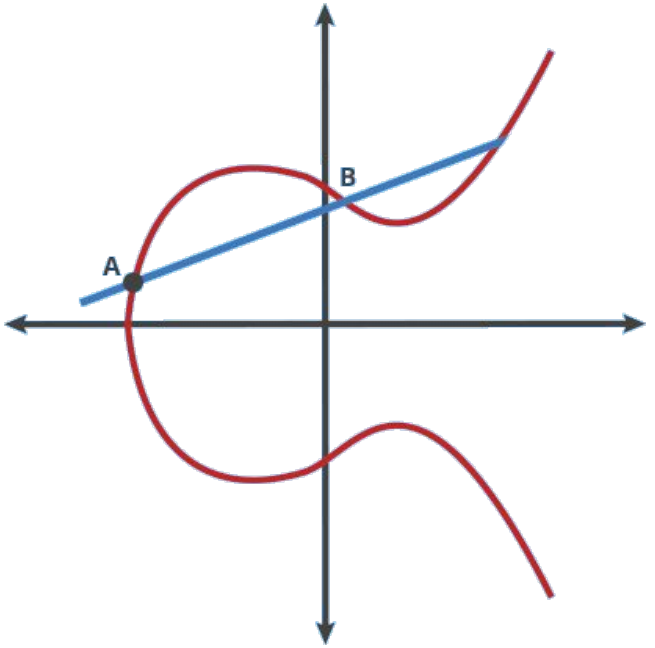
[visuals from Cloudflare]

Elliptic-Curve Cryptography (ECC)

Special operation: ◦ (“dot”)

- A ◦ B = C, A ◦ C = D, …
- nA = A ◦ … ◦ A (n times)
- x(yA) = y(xA) = xyA
- Given point P, hard to find n s.t. nA = P
- Pattern behaves “randomly”

Private key: n (integer)
Public key: P (point on curve, P = nG)
Public knowledge: G (generator point) and curve parameters



[visuals from Cloudflare]

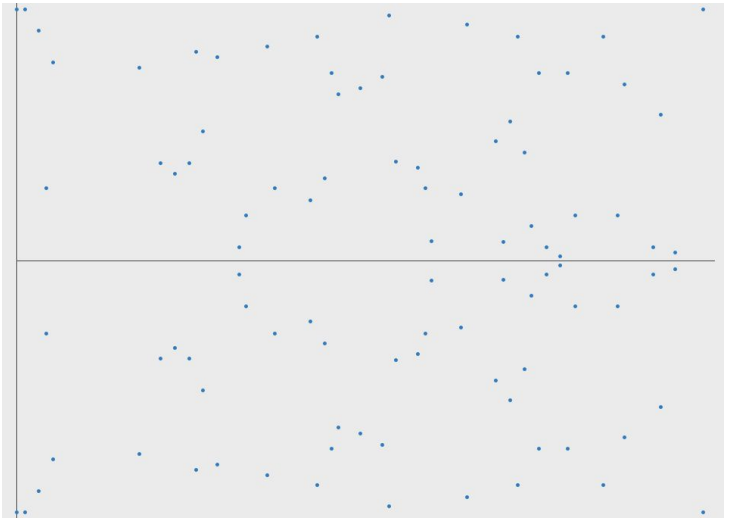
ECC In Practice

Wrap the graph about x and y axes

- Achieves the same effect as modulo, in RSA
- Want prime numbers as the bounds
- Elliptic Curve Discrete Logarithm Problem™

“Safe” Curves?

- NIST recommendations are “fast”, but suspicious
- djb et al. show their work for recommendations
- More: https://safecurves.cr.yp.to/



[visuals from Cloudflare]

ECC vs RSA

Pros:

- Same strength using smaller keys
- Smaller keys ⇒ faster computation
- ECDLP harder(?) than DLP

Cons:

- Hard to understand
- Hard to implement correctly
- Suspicious implementations (NSA 🤔)

Security Strength	Symmetric Key Algorithms	FFC (DSA, DH, MQV)	IFC* (RSA)	ECC* (ECDSA, EdDSA, DH, MQV)
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

[table from NIST (SP 800-57 PART 1 REV. 5)]

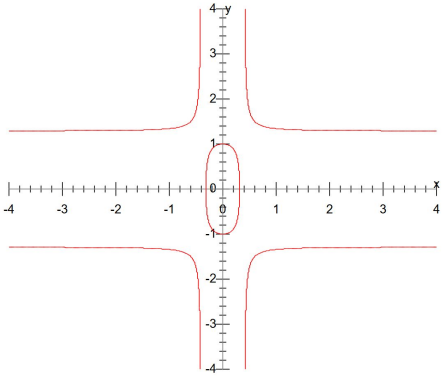
Ultimately: ECC can achieve the same security with smaller keys and faster operations.

ECC In The Wild

ECC can be substituted for $(\mathbb{Z}_p)^\times$ in DL-based protocols:

- Elliptic Curve **Diffie-Hellman**
- Elliptic Curve Integrated **Encryption** Scheme
- Elliptic Curve Digital **Signature** Algorithm
- Edwards-curve* Digital **Signature** Algorithm

Most digital certificates use ECDSA (e.g. P-256)
or EdDSA (e.g. ed25519)



*Twisted Edwards curve
[Wikipedia]

Certificates in Practice
&
Certificate Authority
(CA)

What are certificates

- A security certificate is a small data file used to establish the identity, authenticity and reliability of a website.

  Think of it as a passport!

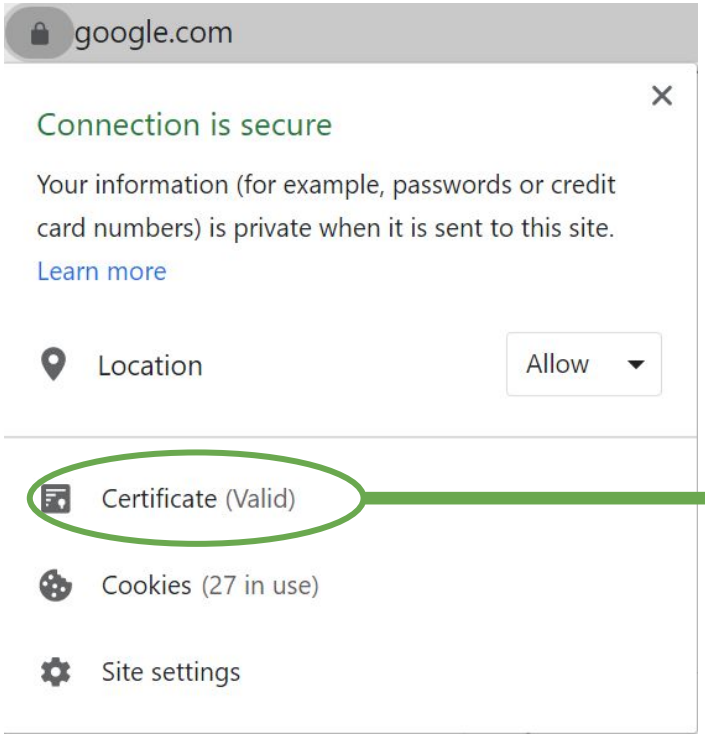
- TLS/SSL: Encryption and authentication for connections

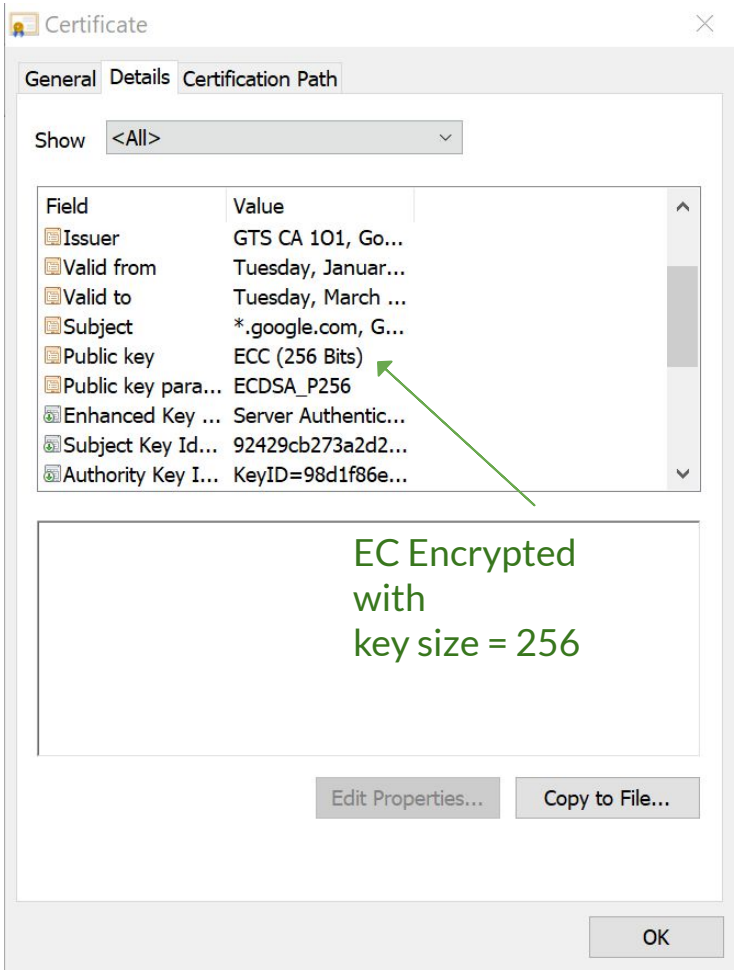
  Note that certificates are not dependent on protocols.

Information on a certificate

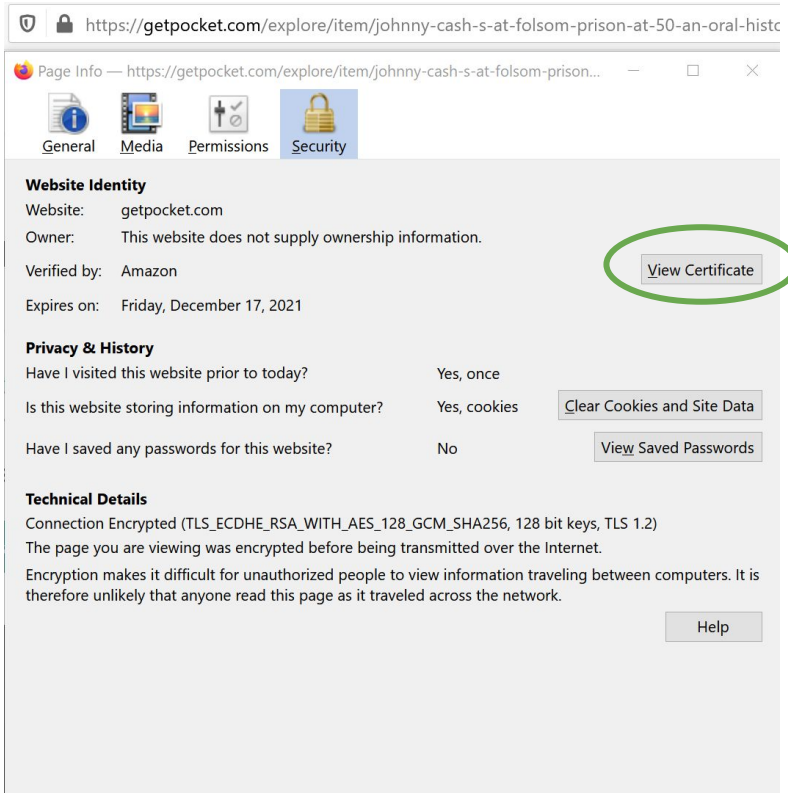
- An X.509 certificate a standard format for public key certificates.
  - Different versions, most common: X.509 v3
  - Not all certificates require public trust
- Includes:
  - public key
  - digital signature
  - Issuing CA
  - Additional information about the certificate

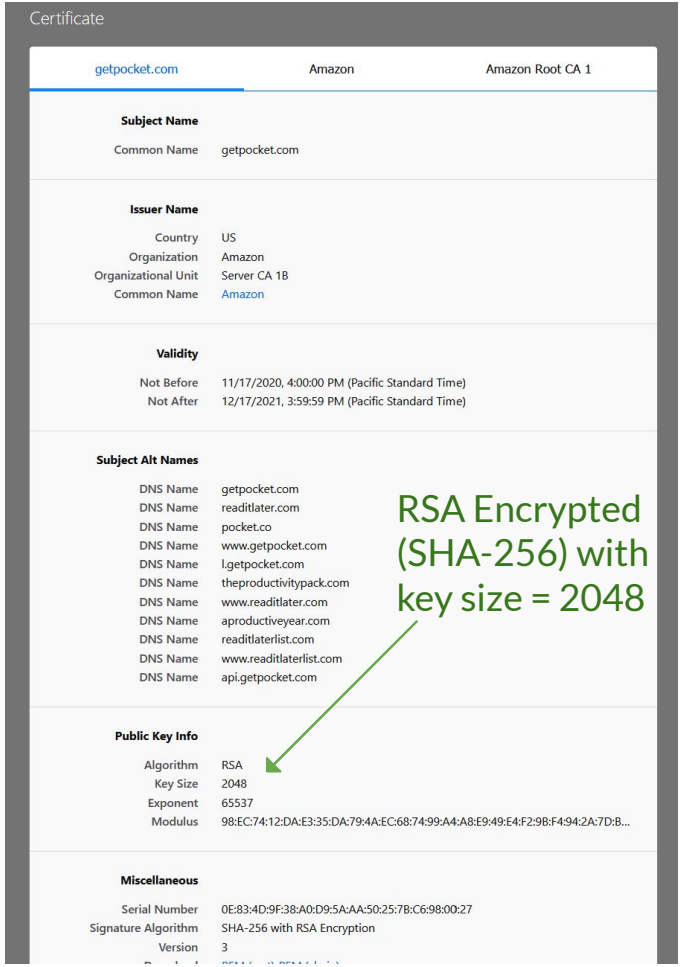
Example: Chrome





Example: Firefox





RSA Encrypted (SHA-256) with key size = 2048

The Handshake

Client says hello

The Handshake

- Server hello
- Client certificate request

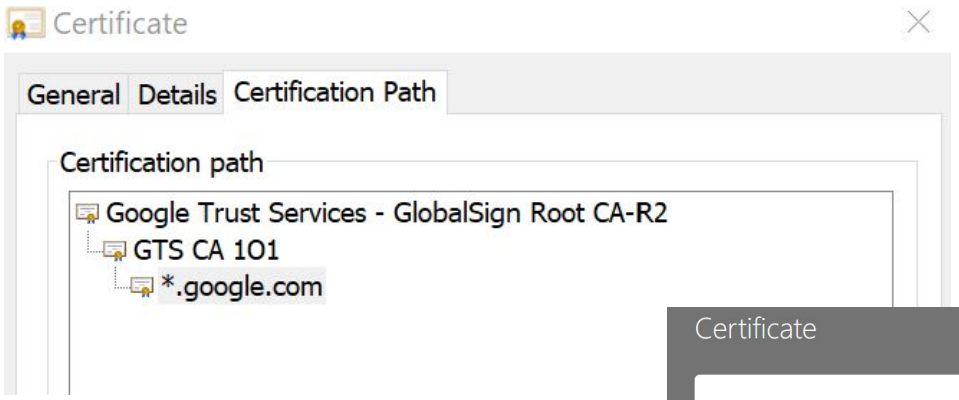
The Handshake

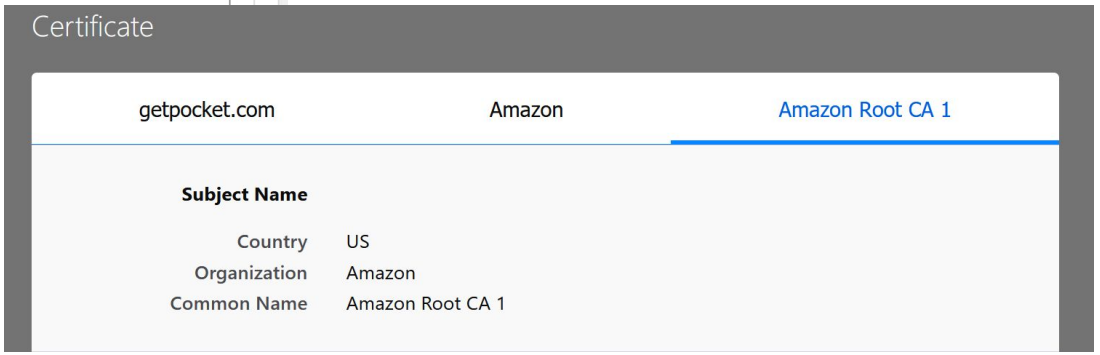
- Client certificate
- Client sends key info (encrypted with server's public key)
- Certificate verify (with digital signature)
- Finished message (encrypted with symmetric key)

The Handshake

Finished message (encrypted with symmetric key)

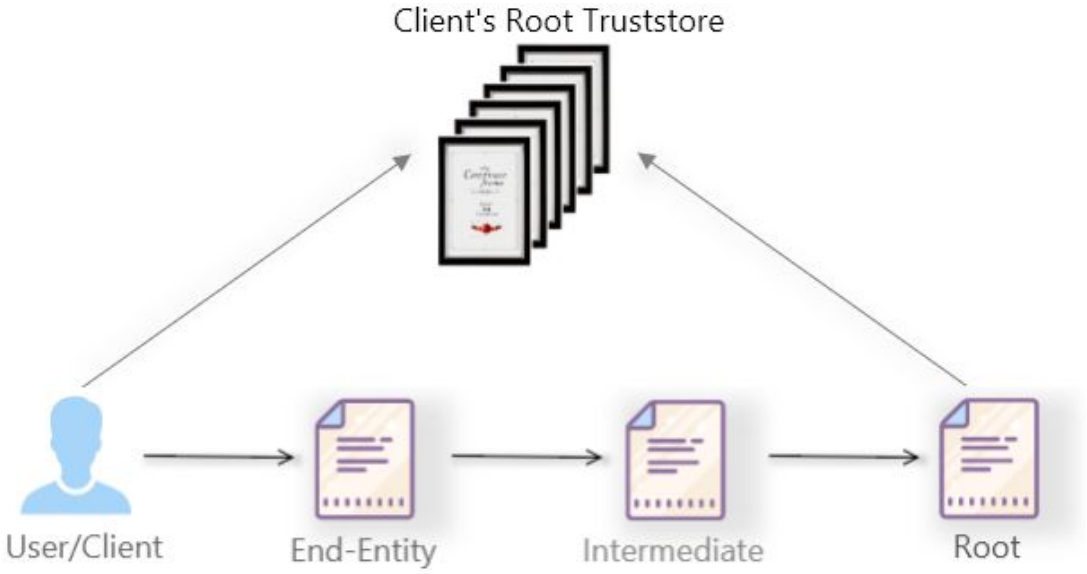
Chain of Trust



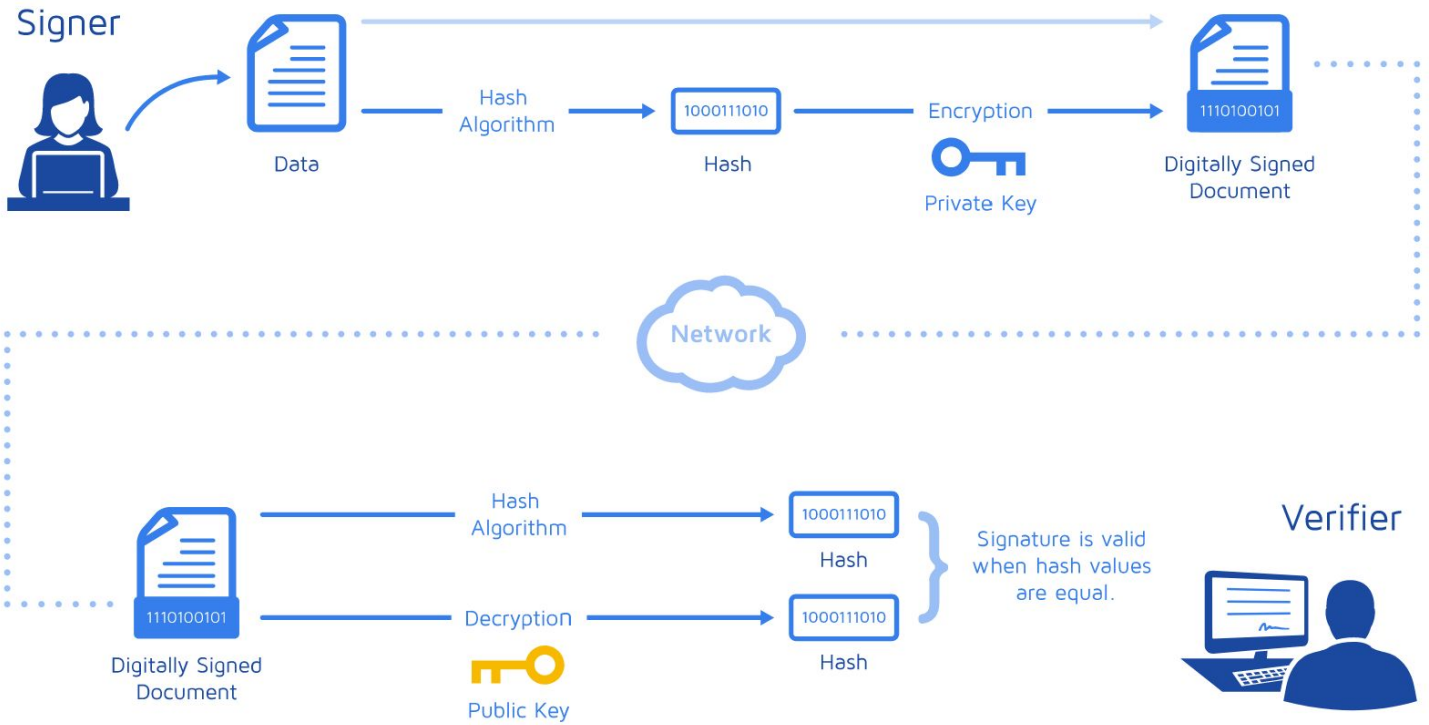


Certificate Authority (CA)

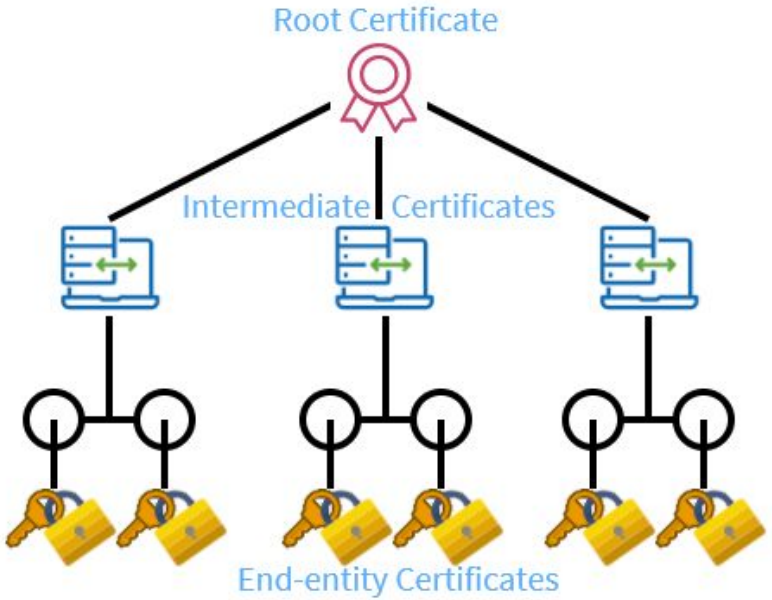
A company or organization that acts to validate the identities of entities and bind them to cryptographic keys through the issuance of digital certificates.



Digital Signatures & Root Certificates



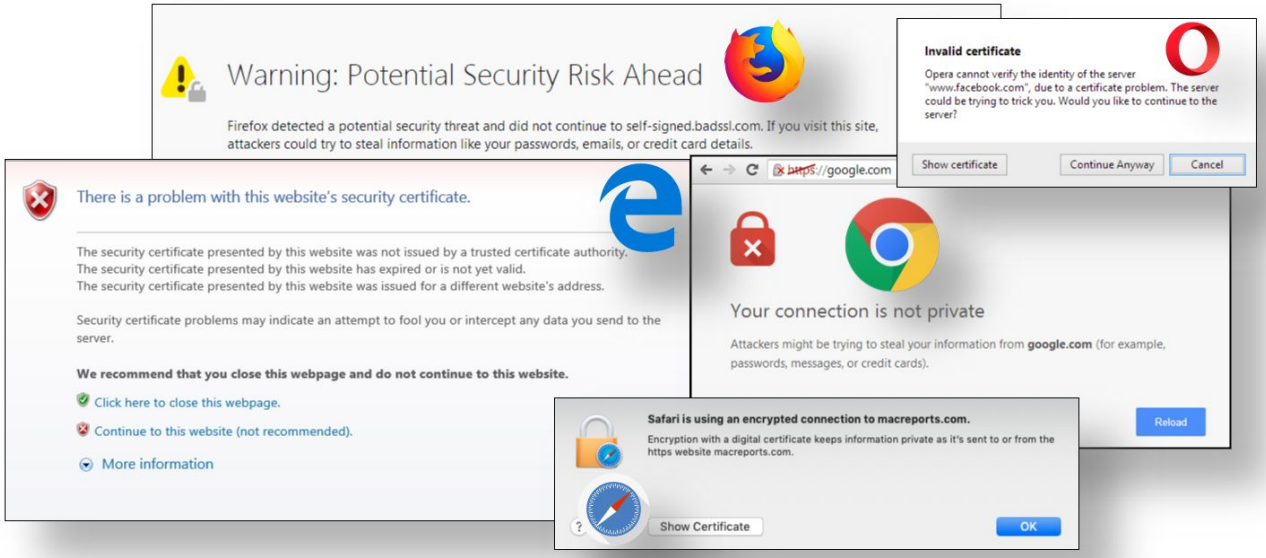
Certification Path



- The hierarchy:
  Website certificate - Intermediate CA certificate - Root CA certificate

- Multiple certification paths could exist - could lead to errors

Certificate Errors

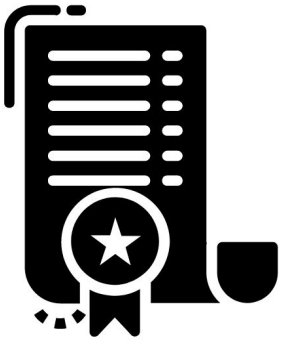


The Heartbleed Bug

- In March 2014, Google discovered a programming mistake in the popular OpenSSL library's implementation of the TLS Heartbeat Extension.
- Allows attackers to read sensitive memory from vulnerable servers, potentially including cryptographic keys, login credentials, and other private data.
- Recovery:
  Patching, revocation of the keys, reissuing keys and replacing certificates.
- Lesson:
  Support for critical projects;
  Develop a method for scalable revocation that can gracefully accommodate mass revocation events;
  Vulnerability disclosure;
  Notification and patching;

Certificate Rotation

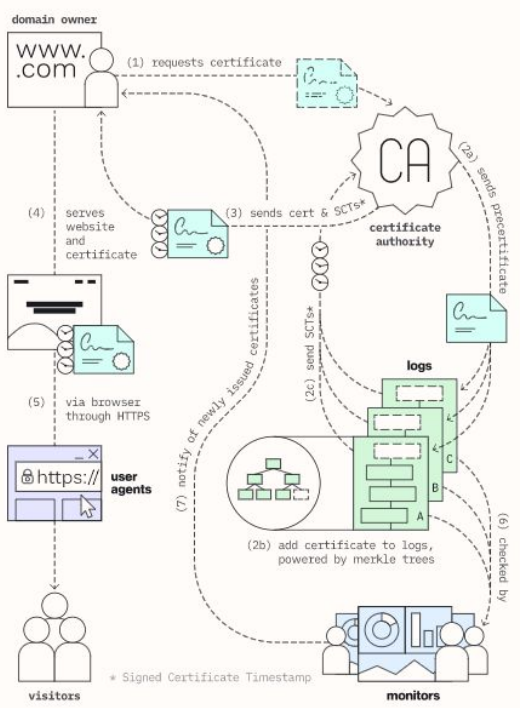
- The replacement of existing certificates with new ones
  Happens when:
  1. Any certificate expires.
  2. A new CA authority is substituted for the old; thus requiring a replacement root certificate for the cluster.
  3. New or modified constraints need to be imposed on one or more certificates.
  4. A security breach has occurred, such that existing certificate-chains can no longer be trusted.



- Example:
  Internal certificate rotation within a company: use of thumbprints vs subject name

Certificate Transparency

- Used for monitoring and auditing digital certificates
- Steps:
  - Website owner requests a certificate from the CA
  - CA issues a precertificate
  - CA sends precertificates to logs
  - Precertificates are added to the logs
  - Logs returns signed certificate timestamps (SCTs) to the CA
  - CAs send the certificate to the domain owner
  - Browsers and user agents help keep the web secure
  - Logs are cryptographically monitored



Thanks for coming to section!