[spoofed facebook.com title page]

TRUST.

TRUST.

TRUST.

x.509v3 SSL Certificate

As Defined in the ITU-T Recommendation x.509


```
-----BEGIN CERTIFICATE-----
MIIBdTCCAS+gAwIBAgICEzcwDQYJKoZIhvcNAQEFBQAwJDENMAsGA
1UEAwwEUm9vdDETMBEGA1UECgwKUm9vdHMgSW5jLjAeFw0xNTAxMT
UwNDUwMTZaFw0xNTA3MTQwNDUwMTZaME4xCzAJBgNVBAYTAlVTMQ0
wCwYDVQQIDARPaGlvMQ8wDQYDVQQKDAZDaXR5IEIxDzANBgNVBAsM
BlVuaXQgQjEOMAwGA1UEAwwFYi5jb20wTDANBgkqhkiG9w0BAQEFA
AM7ADA4AjEArDZ7IpuvflAzhF8qbpXix59EsudjdsShdd7ebdlJR4
MuyRWVcRgUTr2+bzzh4MfPAgMBAAGjMTAvMAwGA1UdEwEB/wQCMAA
wHwYDVR0jBBgwFoAUVy6uqAhadnq2p5wOG1mrT/ZRm4wwDQYJKoZI
hvcNAQEFBQADMQBmfEdSwOSDUEYr7ia+N1u1sjS5/GBzoCxABXxau
V8PxVbZZDpIae4fh/yJCOXJ/OI=
-----END CERTIFICATE-----
```


```
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
000: 30 82 01 75 30 82 01 2F A0 03 02 01 02 02 02 13
010: 37 30 0D 06 09 2A 86 48 86 F7 0D 01 01 05 05 00
     ... 30 1E 17 0D 31 35 30 31 31 35
050: 30 34 35 30 31 36 5A 17 0D 31 35 30 37 31 34 30
060: 34 35 30 31 36 5A
     ... A3 31 30 2F 30 0C 06 03 55 1D 13 01
110: 01 FF 04 02 30 00 30 1F 06 03 55 1D 23 04 18 30
120: 16 80 14 57 2E AE A8 08 5A 76 7A B6 A7 9C 0E 1B
130: 59 AB 4F F6 51 9B 8C 30 0D 06 09 2A 86 48 86 F7
140: 0D 01 01 05 05 00 03 31 00 66 7C 47 52 C0 E4 83
150: 50 46 2B EE 26 BE 37 5B B5 B2 34 B9 FC 60 73 A0
160: 2C 40 05 7C 5A B9 5F 0F C5 56 D9 64 3A 48 69 EE
170: 1F 87 FC 89 08 E5 C9 FC E2
```

SEE DN

SEE DN

SEE PKCS#1

```
373 Bytes [certificate]
 303 Bytes [tbsCertificate]
     3 Bytes [0]
         1 Byte [Version] 3
     2 Bytes [serial number] 4919
     13 Bytes [signatureID]
         9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
         0 Bytes [null]
     36 Bytes [issuer] CN=Root, O=Roots Inc.
     30 Bytes [validity]
         13 Bytes [notBefore] 2015-01-15 04:50:16 UTC
         13 Bytes [notAfter]  2015-07-14 04:50:16 UTC
     78 Bytes [subject] C=US, ST=Ohio, O=City B, OU=Unit B, CN=b.com
     76 Bytes [subjectPublicKeyInfo] [rsaEncryption] 1.2.840.113549.1.1.1
           [modulus] 265059783540994323858542409498208100259117289099398555760
                     0655973362707827270252277499763580632001650191197639650708 7
           [exponent] 65537
     49 Bytes [extension block]
       47 Bytes [extensions]
         12 Bytes [x.509 extension]
             3 Bytes [Basic Constraints] 2.5.29.19
             1 Byte [critical] true
             2 Bytes [isCA, pathLengthConstraints]
                0 Bytes [empty] Not a CA, No Path Constraints
         31 Bytes [x.509 extension]
             3 Bytes [authorityKeyIdentifier] 2.5.29.35
           24 Bytes
             22 Bytes [keyIdentifier]
                20 Bytes [0] 572EAEA8085A767AB6A79C0E1B59AB4FF6519B8C
 13 Bytes [signatureAlgorithmID]
     9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
     0 Bytes [null]
 49 Bytes [signatureValue].f|GR...PF+.&.7[..4..`s.,@.|Z._..V.d:Hi.........
```

ASN.1 Types

30 xx	Sequence	17 xx	UTC Time
02 xx	Integer	01 01	Boolean
06 xx	OID	04 xx	Octet String
05 00	NULL	03 xx	Bit String

xx Bytes

CERTIFICATES

- Websites (TLS)
- Email (S/MIME)
- Digital Signatures
- Anywhere you need Identity & Authentication
- Anywhere you need encryption & privacy

TRADITIONAL TRUST

SECRETS & KEYS

OLD SCHOOL: KEYS

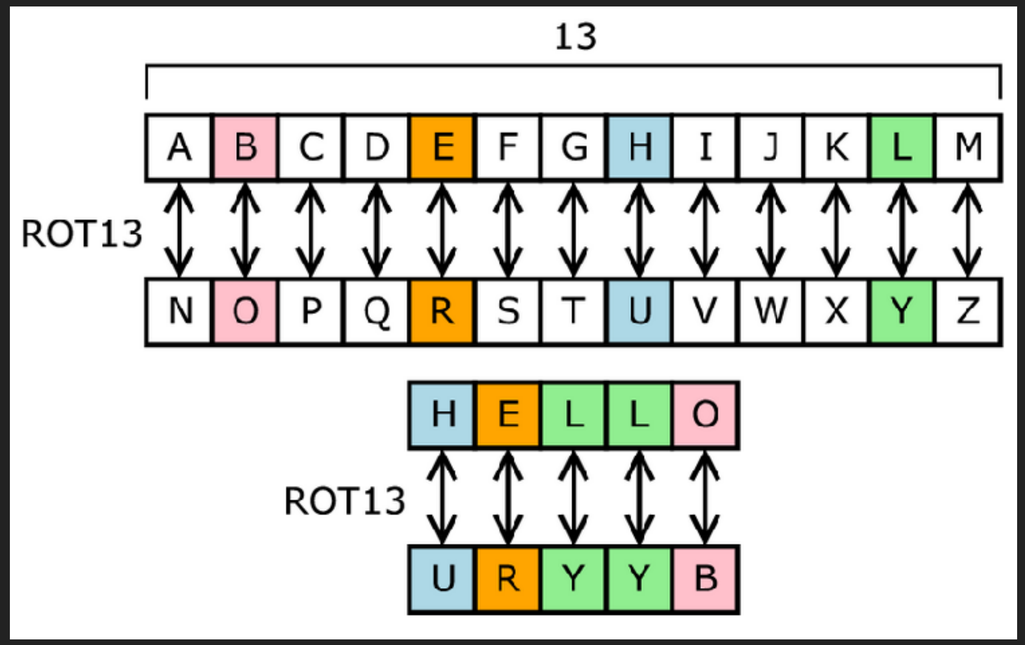
- Single physical pattern (secret) to unlock
- All parties needing access have a copy – **shared**
- Trivial to copy or take by force

OLD SCHOOL: SECRETS

- Single, static sequence or code (secret) to unlock
- All parties needing access have a copy – **shared**
- Trivial to copy

OLD SCHOOL: SUBSTITUTION CIPHERS

- Known algorithm, or technique (secret)
- No keys, but technique is **shared**



NEWER SCHOOL: ROTOR MACHINES

- Secret *not* embedded in device (ideally)
- Rotated dynamic keys
- But, ultimately, secret still **shared**

1970'S: SECRETS & KEYS

PUBLIC KEY CRYTPO

PUBLIC KEY CRYPTOGRAPHY

- Let's flip it on its head. What if everyone could manufacture locks on-the-spot that only one person could open?
- Instead of sharing the *symmetric* key, let's create *asymmetric* keys!
- This means, ideally, nobody's secret is shared. Bob and Alice each have their own **private** key that is never shared!
- A *private* key is paired with a *public* key. Public keys are just that – public and to be shared.

SHARED KEYS

Shared Key

Alice

Bob

SHARED KEYS

Shared Key

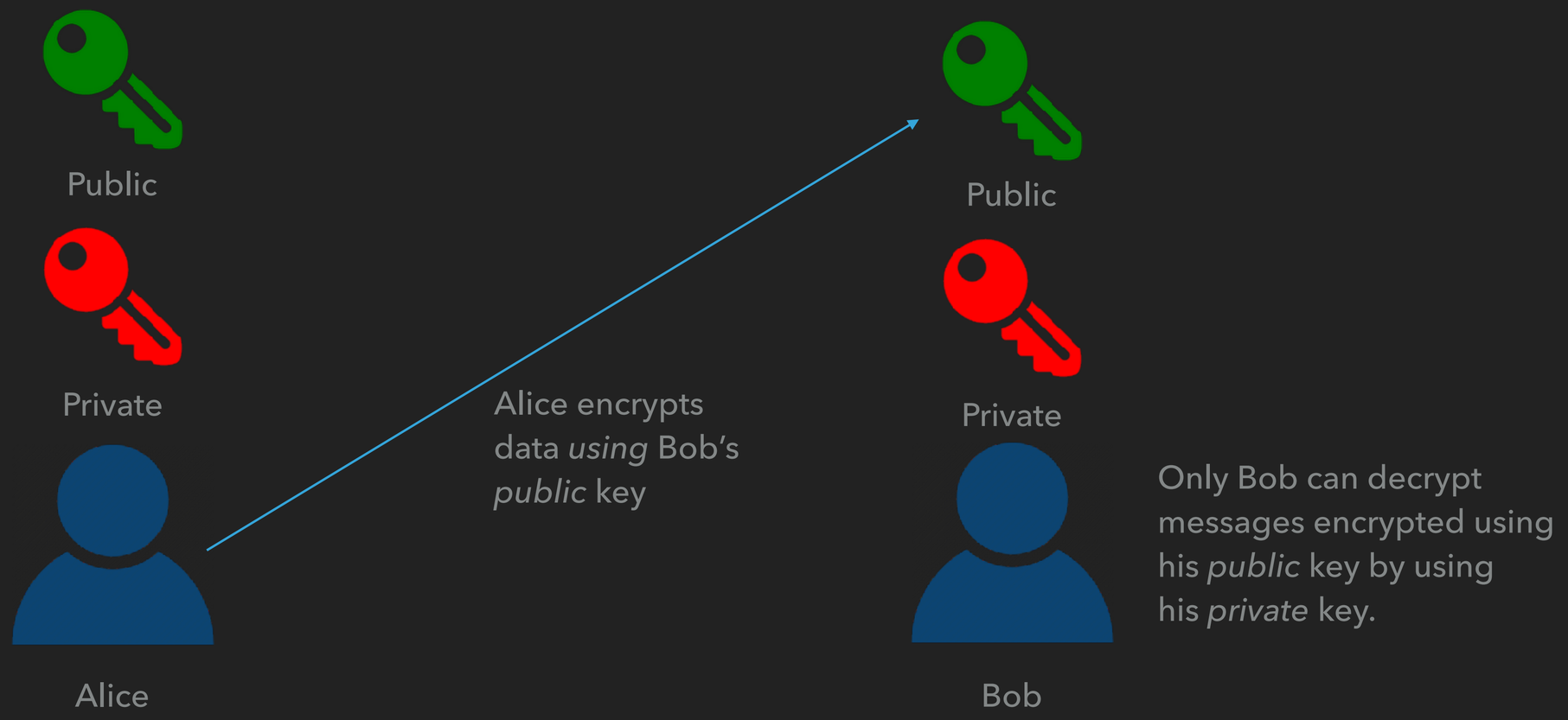
Shared Key

Alice

Bob

PUBLIC KEY CRYPTOGRAPHY

PUBLIC KEY CRYPTOGRAPHY

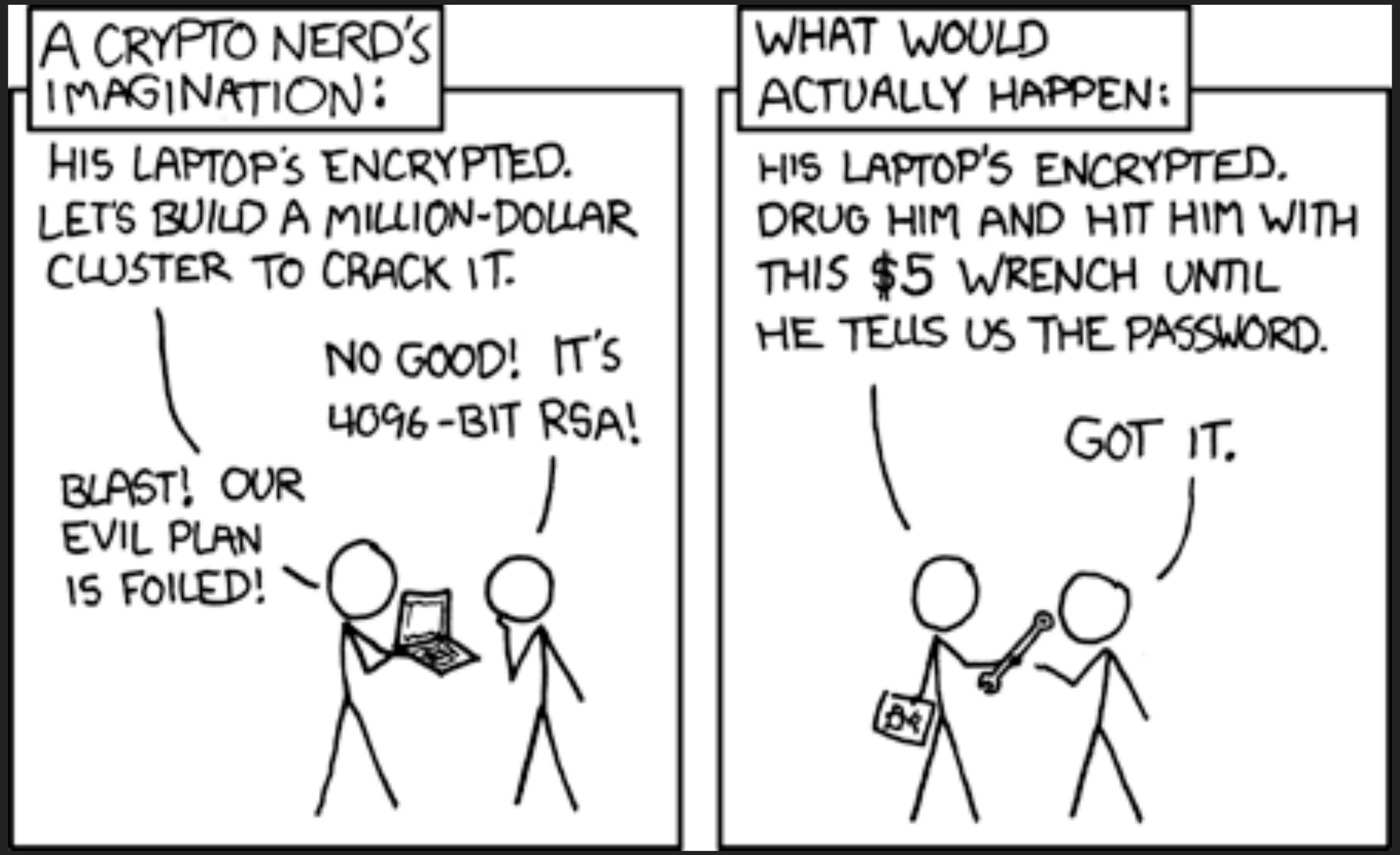


LIKE A MAIL LOCK BOX

- This asymmetric secret (key) relationship
- Anybody can drop mail in; only owner can get get it out

THE MATH

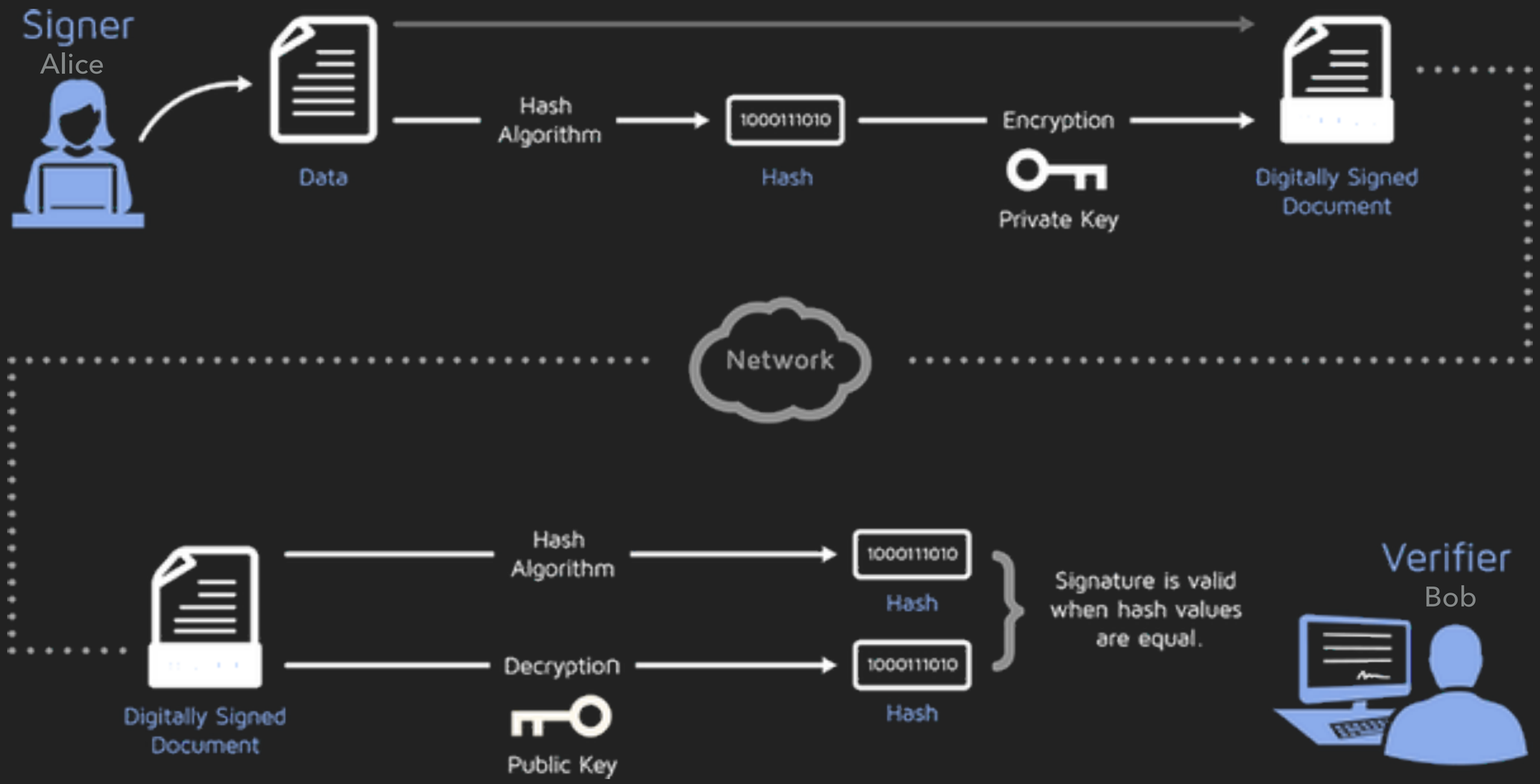
- 2 large primes and an exponent (where it's at)
  - RSA key size relates to size of primes. E.g. 2048 bit = 617 digits
  - The modulus for the primes is included in both pub and priv keys
- Public and private keys are mathematically derived from these values
  - Using fancy math symbols like λ and ϕ and ≡
- Learning more: Kid-RSA
- Fun fact: RSA was invented twice *at the same time*


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

20TH CENTURY JOHN HANCOCK — DIGITAL SIGNATURES

- With *a public key* you can verify a signature (data) came from a given *private key*
- Allows us to guarantee the authenticity (source) of data as well as the integrity (non-modification) and non-repudiation
- Different than mere *hashing* – data actually entangled with the public key

DIGITAL SIGNATURE FLOW



X.509

DIGITAL CERTIFICATES

GETTING CODIFIED

- Have mathematical and conceptual way of understanding asymmetric cryptography (public key crypto)
- Need a standard, specified way of representing that information for exchange and communication
  - Needs to be very well documented & widely available
- Sounds like a job for … standards bodies!

  ITU-T to the rescue!

DEFINED

- Defined in a few different places.
- The X. prefix comes from the International Telecommunication Union (ITU-T) series of standards related to networking and communications
  - Same folks you brought you V.32bis (14.4kbps modem speeds!)

- Also defined in the most recent RFC 5280
- PKCS-set of specifications – now most are RFCs

CERTIFICATES BEGAN WITH… LDAP?

- X.500 from 1988 envisioned ubiquitous, centralized directory services
- This would be a centralized place to *get access to and find* digital certificates which contain public keys
- X.509 was going to be the authentication method to access/modify an X.500 directory
- History had other plans, and our (browser-based) trust is instead based around delegated trust in CAs

CERTIFICATES: JUST WHAT ARE THEY?

- **ASN.1** encoded data structures
- Contain a standard set of fields
  - Critically, the *public key* and *signature*
  - Subject & issuer DNs, expiry dates, etc.
  - Plus a number of optional extensions
- Often wrapped in other formats like PEM, P12, etc.

PEM

- PEM (for Privacy Enhanced Mail): a defunct standard for encoding cryptography for email before S/MIME and PGP
- Base64 encoded data
  - ```
    $ echo hello | openssl base64 -e
    aGVsbG8K
    $ echo aGVsbG8K | openssl base64 -d
    hello
    ```
- ```
  -----BEGIN CERTIFICATE-----
  MIIGsjCCBZqgAwIBAgIQCzw7YBoY9Z7itrsFYF7ywDANBgkqhkiG9w0BAQsFADBw
  MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMRkwFwYDVQQLExB3
  …
  -----END CERTIFICATE-----
  ```

ABSTRACT SYNTAX NOTATION ONE (ASN.1)

- A generalized data structure specification with a few standard encoding types
  - Supports composite types – effectively hierarchical
- Think of it sorta like binary XML
  - Or maybe Thrift/protobufs
- Used for LDAP, VoIP/H.323, Kerberos, SNMP, 3G/4G (LTE)
- Raw ASN.1 X509v3 certificates can be called DER format

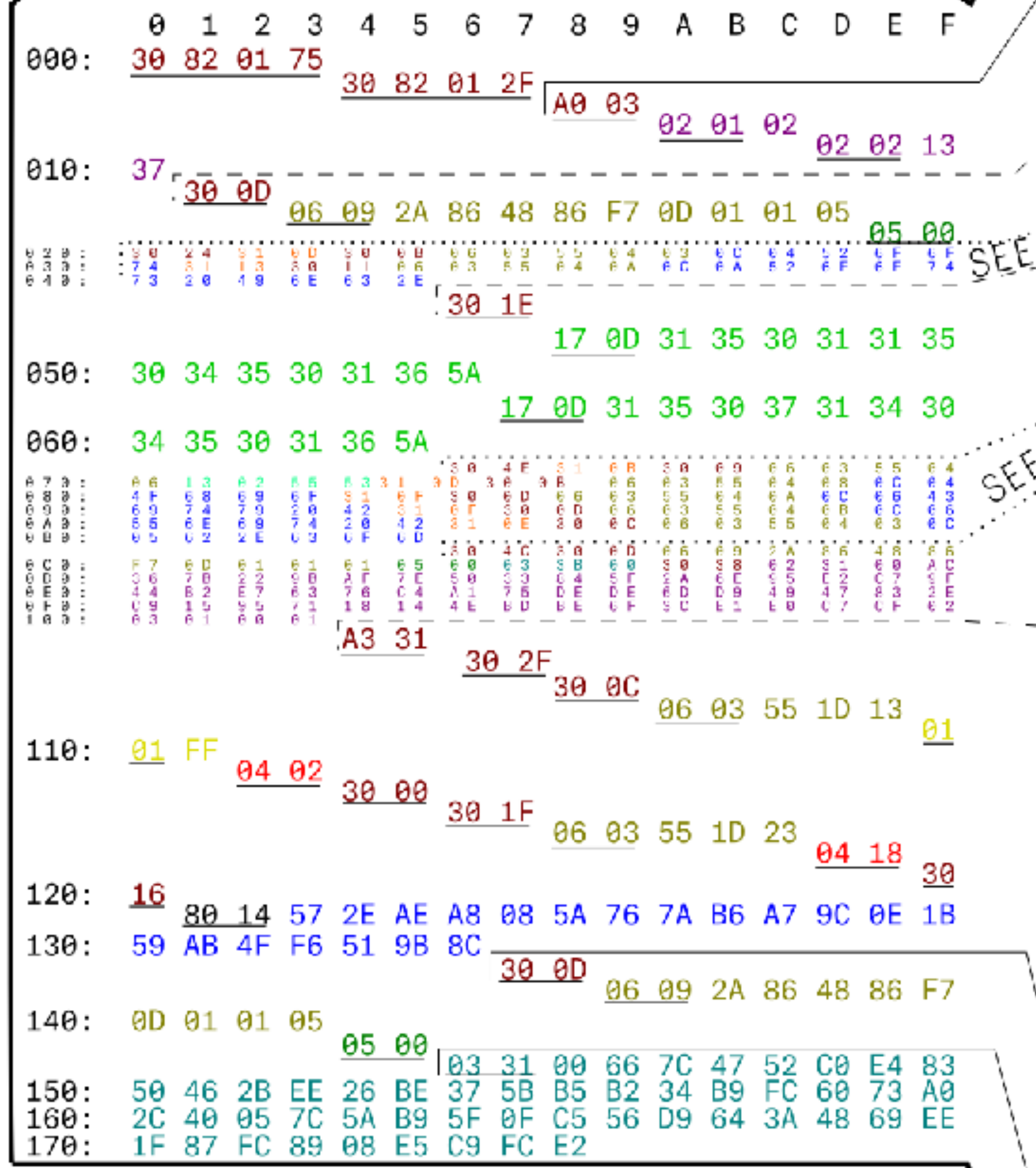
ABSTRACT SYNTAX NOTATION ONE (ASN.1)

- Object Identifiers – OIDs
  - Standardized hierarchical numbering schema for field types and data structures and identifiers
  - Can represent types, fields, companies, columns, etc.
- Used a lot in ASN.1. See: SNMP
- `1.3.6.1.4.1.63` is Apple Corporation
- `1.2.840.113549.1.1.11` is for sha256WithRSAEncryption

x.509v3 SSL Certificate

As Defined in the ITU-T Recommendation x.509

-----BEGIN CERTIFICATE-----
MIIBdTCCAS+gAwIBAgICEzcwDQYJKoZIhvcNAQEFBQAwJDENMAsGA
1UEAwwEUm9vdDETMBEGA1UECgwKUm9vdHMgSW5jLjAeFw0xNTAxMT
UwNDUwMTZaFw0xNTA3MTQwNDUwMTZaME4xCzAJBgNVBAYTAlVTMQ0
wCwYDVQQIDARPaGlvMQ8wDQYDVQQKDAZDaXR5IEIxDzANBgNVBAsM
BlVuaXQgQjEOMAwGA1UEAwwFYi5jb20wTDANBgkqhkiG9w0BAQEFA
AM7ADA4AjEArDZ7IpuvflAzhF8qbpXix59EsudjdsShdd7ebdlJR4
MuyRWVcRgUTr2+bzzh4MfPAgMBAAGjMTAvMAwGA1UdEwEB/wQCMAA
wHwYDVR0jBBgwFoAUVy6uqAhadnq2p5wOG1mrT/ZRm4wwDQYJKoZI
hvcNAQEFBQADMQBmfEdSwOSDUEYr7ia+N1u1sjS5/GBzoCxABXxau
V8PxVbZZDpIae4fh/yJCOXJ/OI=
-----END CERTIFICATE-----



373 Bytes [certificate]
- 303 Bytes [tbsCertificate]
  - 3 Bytes [0]
    - 1 Byte [Version] 3
  - 2 Bytes [serial number] 4919
  - 13 Bytes [signatureID]
    - 9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
    - 0 Bytes [null]
  - 36 Bytes [issuer] CN=Root, O=Roots Inc.
  - 30 Bytes [validity]
    - 13 Bytes [notBefore] 2015-01-15 04:50:16 UTC
    - 13 Bytes [notAfter] 2015-07-14 04:50:16 UTC
  - 78 Bytes [subject] C=US, ST=Ohio, O=City B, OU=Unit B, CN=b.com
  - 76 Bytes [subjectPublicKeyInfo] [rsaEncryption] 1.2.840.113549.1.1.1
    - SEE PKCS#1
    - [modulus] 2650597835409943238585424094982081002591172890993985557600655973362707827270252277499763580632001650191197639650708 7
    - [exponent] 65537
  - 49 Bytes [extension block]
    - 47 Bytes [extensions]
      - 12 Bytes [x.509 extension]
        - 3 Bytes [Basic Constraints] 2.5.29.19
        - 1 Byte [critical] true
        - 2 Bytes [isCA, pathLengthConstraints]
          - 0 Bytes [empty] Not a CA, No Path Constraints
      - 31 Bytes [x.509 extension]
        - 3 Bytes [authorityKeyIdentifier] 2.5.29.35
        - 24 Bytes
          - 22 Bytes [keyIdentifier]
            - 20 Bytes [0] 572EAEA8085A767AB6A79C0E1B59AB4FF6519B8C
- 13 Bytes [signatureAlgorithmID]
  - 9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
  - 0 Bytes [null]
- 49 Bytes [signatureValue].f|GR...PF+.&.7[..4..`s.,@.|Z._..V.d:Hi.........

ASN.1 Types

xx Bytes

30 xx	Sequence	17 xx	UTC Time
02 xx	Integer	01 01	Boolean
06 xx	OID	04 xx	Octet String
05 00	NULL	03 xx	Bit String

 www.cem.me


x.509v3 SSL Certificate
As Defined in the
ITU-T Recommendation x.509
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
373 Bytes [certificate]
303 Bytes [tbsCertificate]
3 Bytes [0]
1 Byte [Version] 3
2 Bytes [serial number] 4919
13 Bytes [signatureID]
9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
0 Bytes [null]
36 Bytes [issuer] CN=Root, O=Roots Inc.
30 Bytes [validity]
13 Bytes [notBefore] 2015-01-15 04:50:16 UTC
13 Bytes [notAfter] 2015-07-14 04:50:16 UTC
78 Bytes [subject] C=US, ST=Ohio, O=City B, OU=Unit B, CN=b.com
76 Bytes [subjectPublicKeyInfo] [rsaEncryption] 1.2.840.113549.1
[modulus]
[exponent] 65537
SEE DN
SEE DN
SEE PKCS#1
49 Bytes [extension block]
47 Bytes [extensions]
12 Bytes [x.509 extension]
3 Bytes [Basic Constraints] 2.5.29.19
1 Byte [critical] true
2 Bytes [isCA, pathLengthConstraints]
0 Bytes [empty] Not a CA, No Path Constraints
31 Bytes [x.509 extension]
3 Bytes [authorityKeyIdentifier] 2.5.29.35
24 Bytes
22 Bytes [keyIdentifier]
20 Bytes [0] 572EAEA8085A767AB6A79C0E1B59AB4FF651
13 Bytes [signatureAlgorithmID]
9 Bytes [sha1WithRSAEncryption] 1.2.840.113549.1.1.5
0 Bytes [null]
49 Bytes [signatureValue]
ASN.1 Types
xx Bytes
30 xx Sequence
02 xx Integer
06 xx OID
05 00 NULL
17 xx UTC Time
01 01 Boolean
04 xx Octet String
03 xx Bit String
Copyright © Carl Mehner 2015
www.cem.me
*.facebook.com.cer
Open with Keychain Access
*.facebook.com
Subject Name
Country US
State/Province California
Locality Menlo Park
Organization Facebook, Inc.
Common Name *.facebook.com
Issuer Name
Country US
Organization DigiCert Inc
Organizational Unit www.digicert.com
Common Name DigiCert SHA2 High Assurance Server CA
Serial Number 0B 3C 3B 60 1A 18 F5 9E E2 B6 BB 05 60 5E F2 C0
Version 3
Signature Algorithm SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Not Valid Before Thursday, December 14, 2017 at 4:00:00 PM Pacific Standard Time
Not Valid After Friday, March 22, 2019 at 5:00:00 AM Pacific Daylight Time
Public Key Info
Algorithm Elliptic Curve Public Key (1.2.840.10045.2.1)
Parameters Elliptic Curve secp256r1 (1.2.840.10045.3.1.7)
Public Key 65 bytes : 04 88 03 CE C8 8E AA 8C ...
Key Size 256 bits
Key Usage Encrypt, Verify, Derive
Signature 256 bytes : 6B B4 BB 16 43 F8 84 57 ...
Extension Key Usage (2.5.29.15)
Critical YES
Usage Digital Signature
Extension Basic Constraints (2.5.29.19)
Critical YES
Certificate Authority NO
Extension Extended Key Usage (2.5.29.37)

Distinguished Name (DN)

x.500 as defined in RFC 2247

```
-----BEGIN CERTIFICATE REQUEST-----
MIH0MIGvAgEAMFoxCzAJBgNVBAYTAlVTMQ0wCwYDVQQIDARP
aGlvMQowCAYDVQQHDAFCMQ8wDQYDVQQKDAZDaXR5IEIxDzAN
BgNVBAsMBlVuaXQgQjEOMAwGA1UEAwwFYi5jb20wTDANBgkq
hkiG9w0BAQEFAAM7ADA4AjEArDZ7IpuvflAzhF8qbpXix59E
sudjdsShdd7ebdlJR4MuyRWVcRgUTr2+bzzh4MfPAgMBAAGg
ADANBgkqhkiG9w0BAQUFAAMxADV0IWl2A7g7GvY4dZFXPlU2
RB3wruhFgv8hUY+CiiBIBaxKp2TCtImdN98QVjTzDw==
-----END CERTIFICATE REQUEST-----
```

```
    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00: 30 5A
      31 0B
         30 09
            06 03 55 04 06
                        13 02 55 53
                                    31
10: 0D
      30 0B
         06 03 55 04 08
                     0C 04 4F 68 69 6F
                                    31 0A
20: 30 08
         06 03 55 04 07
                     0C 01 42
                              31 0F
                                 30 0D
                                    06 03
30: 55 04 0A
         0C 06 43 69 74 79 20 42
                              31 0F
                                 30 0D
                                    06
40: 03 55 04 0B
         0C 06 55 6E 69 74 20 42
                                 31 0E
                                    30 0C
50: 06 03 55 04 03
                  0C 05 62 2E 63 6F 6D
```

90 Bytes [distinguishedName]

- 11 Bytes [name]
  - 9 Bytes [attributeTypeAndValue]
    - 3 Bytes [Country Name] 2.5.4.6
    - 2 Bytes US
- 15 Bytes [name]
  - 11 Bytes [attributeTypeAndValue]
    - 3 Bytes [State or Province] 2.5.4.8
    - 4 Bytes Ohio
- 10 Bytes [name]
  - 8 Bytes [attributeTypeAndValue]
    - 3 Bytes [Locality Name] 2.5.4.7
    - 1 Bytes B
- 15 Bytes [name]
  - 13 Bytes [attributeTypeAndValue]
    - 3 Bytes [Organization Name] 2.5.4.10
    - 6 Bytes City B
- 15 Bytes [name]
  - 13 Bytes [attributeTypeAndValue]
    - 3 Bytes [Organizational Unit] 2.5.4.11
    - 6 Bytes Unit B
- 14 Bytes [name]
  - 12 Bytes [attributeTypeAndValue]
    - 3 Bytes [Common Name] 2.5.4.3
    - 5 Bytes b.com

ASN.1 Types (xx Bytes)

06 xx	OID	30 xx	Sequence
0C xx	UTF-8 String	31 xx	Set
13 xx	Printable String		

www.cem.me

FORMATS AND WRAPPERS AND BEARS, OH MY

	Certificate	Private Key	PK (Encrypt)
DER	√	√	
PEM	√	√	
Armored PEM			√
PKCS#7	√		
P12	√	√	√
keystore (java)	√		√
Keychain (macOS)	√		√

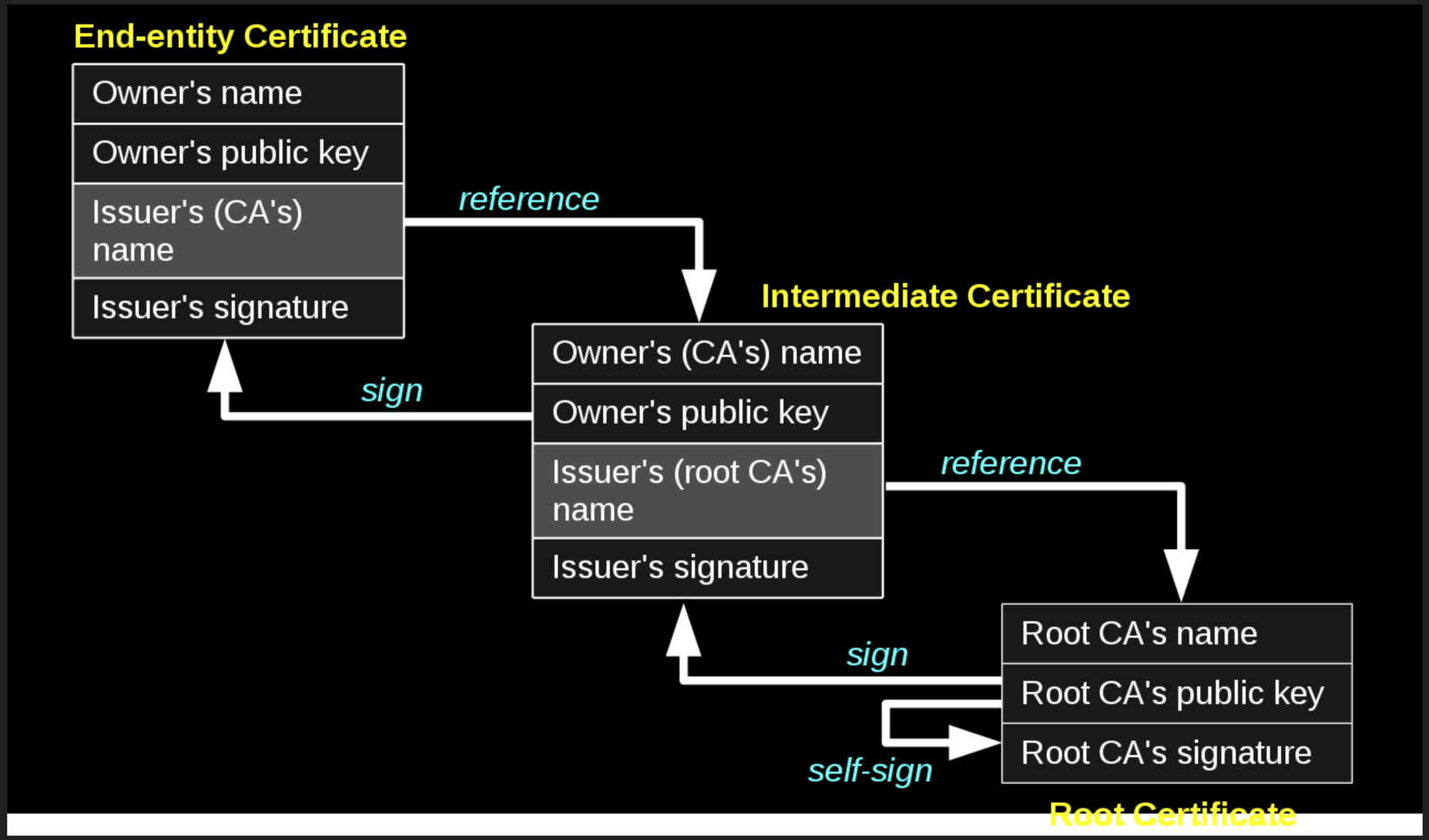
PUBLIC KEY INFRASTRUCTURE

HOW WE TRUST

A TALE OF TWO TECHNIQUES

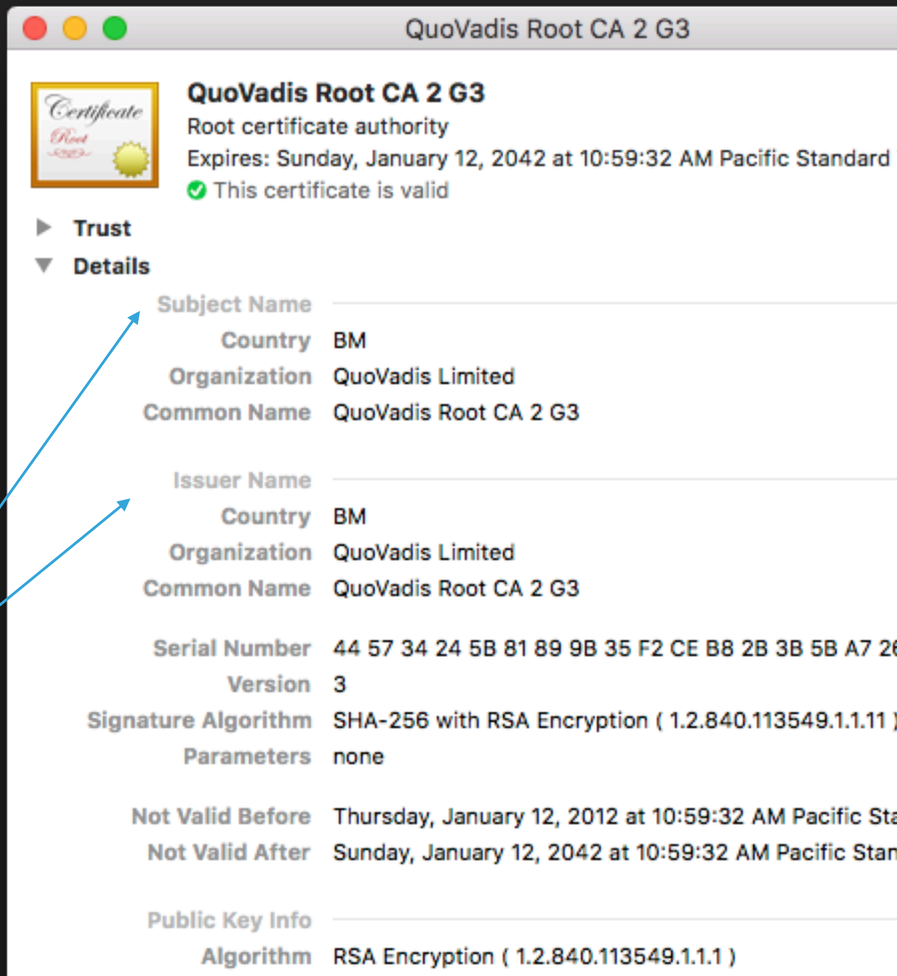
- **Chain of Trust (Authority)**
  - Set of Certificate Authorities (CAs) that are considered *trust anchors*
  - Can verify a certificate originated from the ‘Root’ trusted authorities via the chain
  - Typically burden of trust is with software developers
- **Web of Trust**
  - Decentralized & peer-based
  - Trust your friends, they’ll trust theirs, and eventually this distributed trust database forms
  - Places burden of trust on the end-user
  - OpenPGP – check it out!

CHAIN OF TRUST

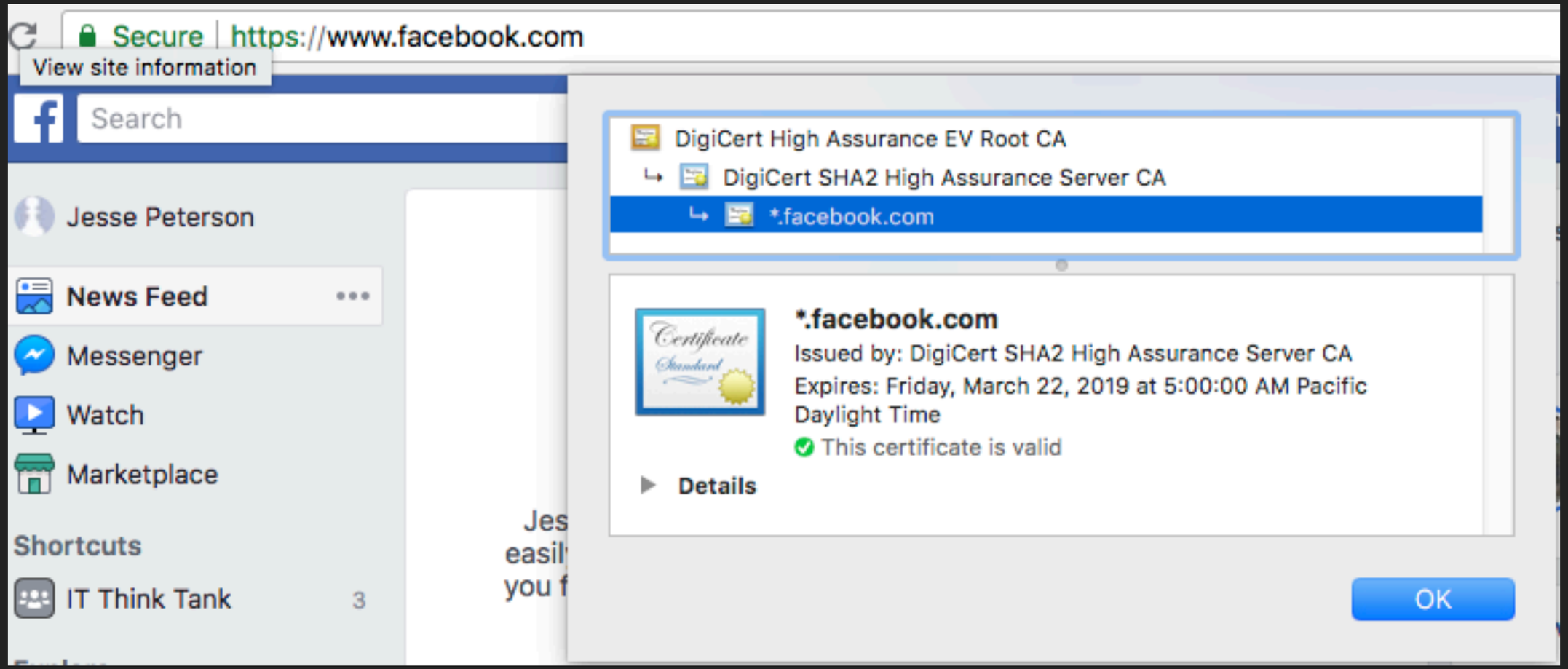


SELF-SIGNED CERTIFICATES

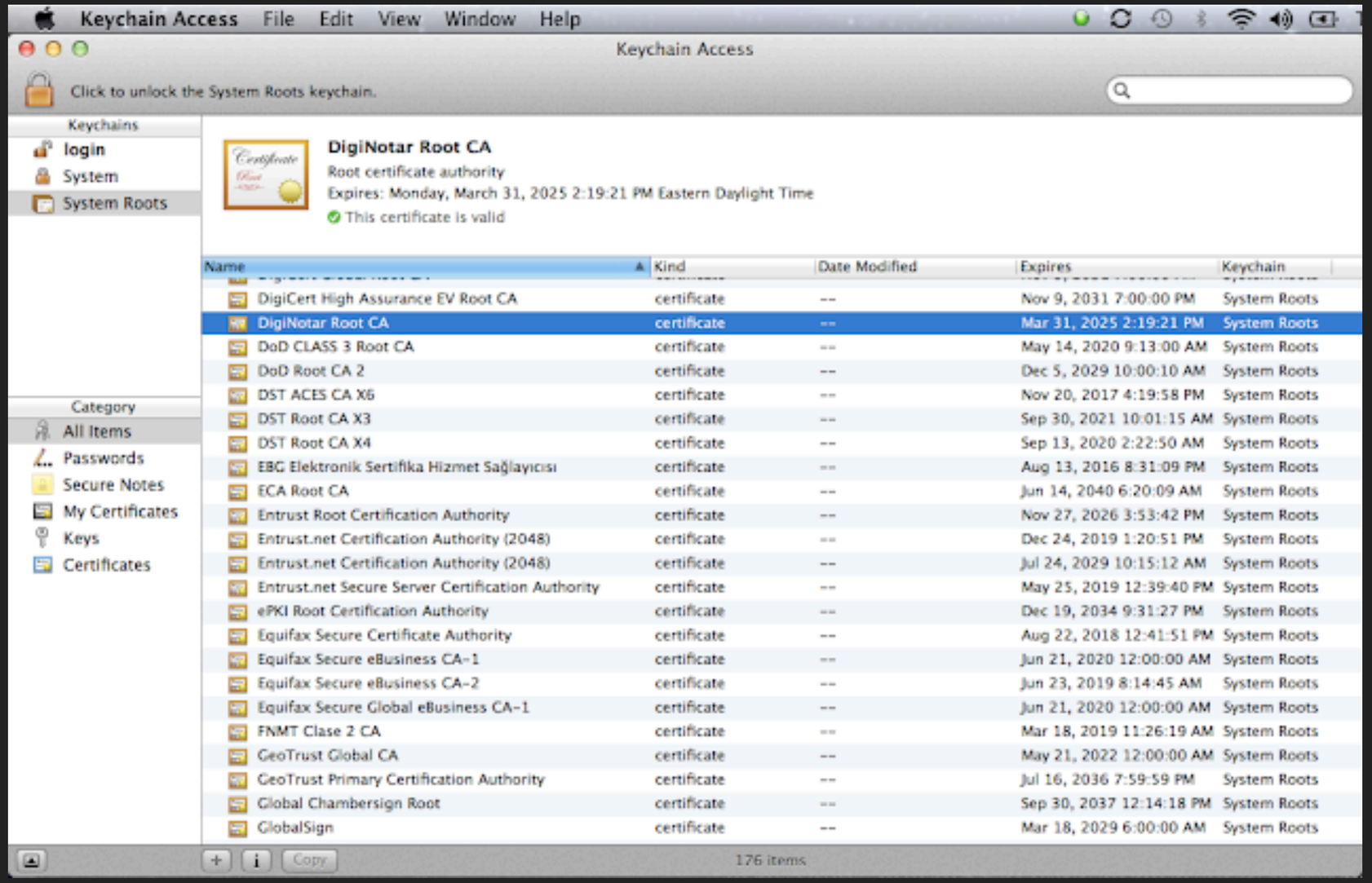
- Have same *subject & issuer* information
  - I.e. does not delegate to higher cert.
- Signed by private key corresponding to *embedded* public key
- Other attributes like CA:TRUE



CHAIN OF TRUST — FROM A BROWSER



MACOS SYSTEM ROOTS



COMPROMISED

2011

threatpost Cloud Security / Malware / Vulnerabilities / Privacy

California Attorney General Puts Mobile App Developers on Notice

South Carolina Data Breach Cast

Final Report on DigiNotar Hack Shows Total Compromise of CA Servers

COMPUTERWORLD FROM IDG

Home > Cyber Crime

NEWS

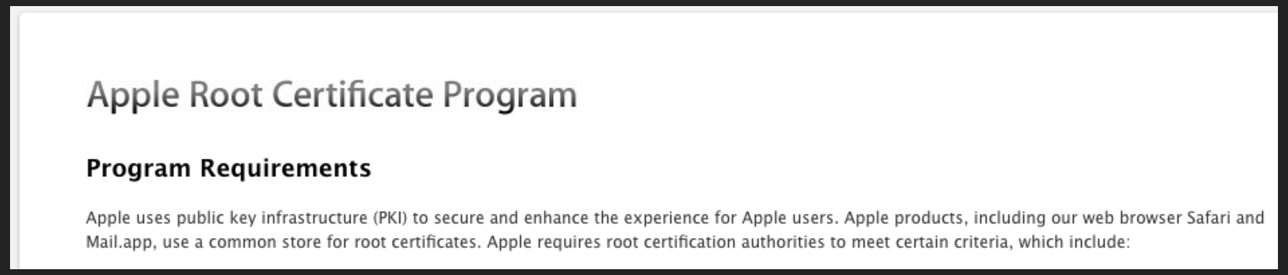
Hackers spied on 300,000 Iranians using fake Google certificate

Investigation reveals month-long, massive Gmail snooping campaign

LEAVE YOUR CALLING CARD IN ASN.1 ATTRIBUTES

- CN=*.RamzShekaneBozorg.com,SN=PK000229200006593, OU=Sare Toro Ham Mishkanam,L=Tehran,O=Hameye Ramzaro Mishkanam,C=IR
- In the Farsi language:
  "RamzShekaneBozorg" is "great cracker"
  "Hameyeh Ramzaro Mishkanam" translates to "I will crack all encryption"
  "Sare Toro Ham Mishkanam" translates to "i hate/break your head"

ROOT CERTIFICATE PROGRAMS



Apple Root Certificate Program

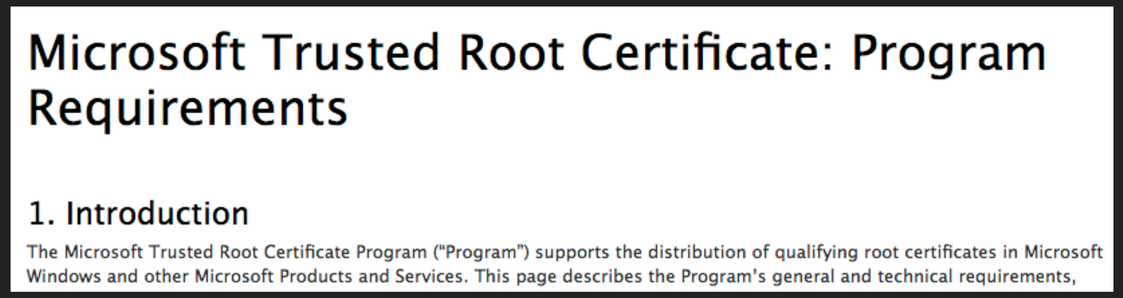
Program Requirements

Apple uses public key infrastructure (PKI) to secure and enhance the experience for Apple users. Apple products, including our web browser Safari and Mail.app, use a common store for root certificates. Apple requires root certification authorities to meet certain criteria, which include:

Mozilla Root Store Policy

Version 2.6

Effective July 1, 2018



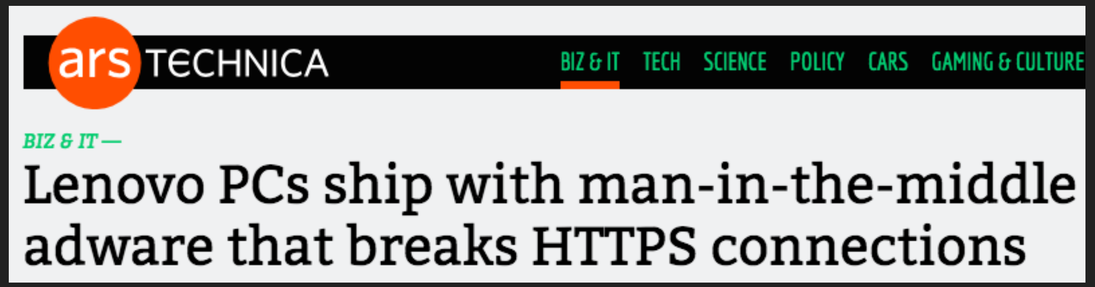
Microsoft Trusted Root Certificate: Program Requirements

1. Introduction

The Microsoft Trusted Root Certificate Program ("Program") supports the distribution of qualifying root certificates in Microsoft Windows and other Microsoft Products and Services. This page describes the Program's general and technical requirements,

SOFTWARE COMPANIES VS. VENDORS/RESELLERS

- Operating Systems and some software has Root CA Policies
- But often your hardware/software comes from a *vendor or reseller* who can modify said software before it reaches you…



REVOKING TRUST

- Certificate Revocation Lists (CRLs)
  - Simply a list of bad certs that were revoked
- Online Certificate Status Protocol (OCSP)
  - Perform in-the-moment verification of certs
  - OCSP “stapling”: have the server do it (regularly) and attach to the web request & Must-Staple
- OOB browser blocking: CRLset, OneCRL

GETTING BETTER AT TRUST

- Certificate Transparency (CT)
  - CAs public *all* issued certificates
  - Get notified when CAs issue certs for domains that are yours
- Certificate Authority Authorization (CAA)
  - Specify in DNS which CAs are allowed to issue certs
- Drastically lowing issuance windows: i.e. days

TRUST RECAP

- Computer/device is pre-loaded with a list of trusted Certificate Authorities
- Computer/device implicitly trusts all certificates signed from those CAs
  - Future revocation technologies not withstanding

CREATION

LET'S GET CERTIFIED

THINGS NEEDED

- Certificate Attributes you want set
  - Company name, street address, DNS address, etc.
- Public key
- Private key (for signing)

TYPICAL STEPS — REQUESTER

1. Generate Private Key (which also generates the public key)
2. Generate CSR (PKCS#10) – Certificate Signing Request using the public key and any attributes you desire
3. Use (or submit) this CSR to the CA. Or yourself, if you're self-signing.
4. (If you're the

TYPICAL STEPS — CA

1. Receive CSR from user
2. Take user's $$$
3. Take certificate attributes, public key, etc. in CSR and fabricate a new certificate using them (often discarding or modifying your CSR attributes and keeping others)
4. Sign this certificate using CA private key
5. Deliver certificate to back to user

DEMO:
SELF-SIGNED WEB CERTIFICATE

AUTOMATED ISSUANCE OPTIONS

- SCEP
  - Request certificates over the network, automatically
- DCE/RPC CertServerRequest in AD environments
  - Go see Tim Perfitt's talk to learn more!
- Let's Encrypt! 
  - Have your server *maintain it's own* certificates!

THANK YOU!

Feedback link: https://bit.ly/psumac2018-290

Q&A?

Jesse Peterson
CPE @ Facebook
Slack: @jessepeterson
Twitter: @jessecpeterson