

Introduction to Cryptography

Public Key Infrastructure



Authentication

- An important point from the Needham-Schroeder protocol and the attacks on it: *authentication*—knowing to whom you're actually talking—can be as important as confidentiality
- Cryptography has to provide both
- Sometimes, authentication has to be bidirectional

Bidirectional Authentication

- Recent phone experience: “Hi, I’m from your doctor’s office; could you give me your date of birth and social security number so I know I’m talking to Steve Bellovin?”
- Me: “No, you tell me; how do I know who you are?”
- Them: “I just told you!”
- Me: “And why should I believe you?”
- Them:
- Me:

Authentication versus Authorization

- Authentication is a way to prove who you are
- It is not the same as what you're allowed to do—that's *authorization*
- But most systems use authentication as a step towards authorization
- More on this later in the term

Finding a Key

Alice wants to talk to Bob, so she searches for his public key



Will she get the right answer?

Certificates

A certificate is a digitally signed binding between an identity and a public key.
Will this work?

The Google logo is displayed in its standard multi-colored font.

Bob's certificate|



Google Search

I'm Feeling Lucky

ID Requirements Are Changing

Does your ID have a star?

Beginning **October 1, 2020**, you will need a REAL ID-compliant license or another acceptable form of ID, such as a valid passport or U.S. military ID, to fly within the U.S.



Generally, a star indicates it's a REAL ID

Check with your state driver's license agency to verify that your state-issued ID is compliant.

Learn about flying with a REAL ID at tsa.gov/real-id

Questions about flying?

- The certificate has to be issued by someone you trust
- They actually have to do a good job verifying identity
- There are indications in the certificate showing the proper form

History of Certificates

- Invented in 1978 by an MIT undergrad, Loren M. Kohnfelder, in his senior thesis
- (Evidence suggests that by the early 1980s, the NSA adopted certificates for its secure phones)
- Adopted for the OSI networking standard's "directory" project
- Their format, known as X.509, is the primary certificate standard used today

What are Certificates Used For?

- Verifying website identity
- Verifying other encrypted communications sessions
- Verifying executables
- Verifying and protecting email
- More

Checking a Certificate in Firefox

Opinion

Are Republican Judges Putting Their Thumbs on the Electoral Scale?

Recent decisions certainly suggest

By Erwin Chemerinsky

Mr. Chemerinsky is dean of the law school at the University of California, Berkeley.

Sept. 25, 2020



Page Info - https://www.nytimes.com/2020/09/25/opinion/voting-courts-republicans.h...

General Media Permissions Security

Website Identity

Website: www.nytimes.com

Owner: This website does not supply ownership information.

Verified by: Sectigo Limited

Expires on: April 5, 2022

Privacy & History

Have I visited this website prior to today? Yes, 4,254 times

Is this website storing information on my computer? Yes, cookies and 48.0 KB of site data

Have I saved any passwords for this website? No

Technical Details

Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, 128 bit keys, TLS 1.2)

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.

View Certificate

Clear Cookies and Site Data

View Saved Passwords

?

All desktop browsers have some way to display sites' certificates

Columbia's Certificate: The Basics

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

Subject Name
Country US
10027
State/Province New York
Locality New York
116th Street and Broadway
Organization Columbia University
Organizational Unit Information Technology
Common Name columbia.edu

Issuer Name
Country US
State/Province MI
Locality Ann Arbor
Organization Internet2
Organizational Unit InCommon
Common Name [InCommon RSA Server CA](#)

The certificate includes the organization name and address, and the CA's name and address.

Note the top: it shows the certificate chain from the trust anchor.

Columbia: Other Essential Data

Validity	
Not Before	1/1/2020, 7:00:00 PM (Eastern Standard Time)
Not After	1/1/2022, 6:59:59 PM (Eastern Standard Time)
Subject Alt Names	
DNS Name	columbia.edu
DNS Name	*.columbia.edu
Public Key Info	
Algorithm	RSA
Key Size	2048
Exponent	65537
Modulus	C0:33:DA:C2:98:DE:44:41:A2:F5:87:D3:3E:AD:30:88:03:34:2D:36:C9:59:EB:86:64:37:B2:B3:E...
Miscellaneous	
Serial Number	00:DC:96:95:D4:2C:DF:5E:DE:EA:4B:13:5C:F7:0D:B5:5A
Signature Algorithm	SHA-256 with RSA Encryption
Version	3
Download	PEM (cert) PEM (chain)

Note the validity period, actual domain names (called “Subject Alt Names”), and the key and algorithm.

- Why limit certificate lifetime?
- Limit the damage in case of private key compromise
- Limit the amount of revocation data that has to be kept
- Organizations have finite lifetimes—why leave their abilities around forever?
- And algorithms age

Columbia: Policies and Revocation

CRL Endpoints

Distribution Point <http://crl.incommon-rsa.org/InCommonRSAServerCA.crl>

Authority Info (AIA)

Location http://crt.usertrust.com/InCommonRSAServerCA_2.crt

Method CA Issuers

Location <http://ocsp.usertrust.com>

Method Online Certificate Status Protocol (OCSP)

Certificate Policies

Policy Statement Identifier (1.3.6.1.4.1)

Value 1.3.6.1.4.1.5923.1.4.3.1.1

Qualifier Practices Statement (1.3.6.1.5.5.7.2.1)

Value https://www.incommon.org/cert/repository/cps_ssl.pdf

Policy Certificate Type (2.23.140.1.2.2)

Value Organization Validation

The certificate has the CA's policies and revocation mechanisms.

How Do You Revoke a Certificate?

- Revocation is hard! Verification can be done offline; revocation requires some form of connectivity
- Publish the URL of a list of revoked certificates in a Certificate Revocation List (CRL)
 - 👉 One reason for certificate expiration dates; you don't need to keep revocation data forever
- Online status checking (OCSP)—check a server in real-time for the certificate's status
- Note: OCSP has privacy implications
- NSA secure phones apparently use a flooding algorithm to pass around lists of revoked certificates—works well because of comparatively closed communities

Why Revoke Certificates?

- Private key compromised
- Cancel authorization associated with certificate
- CA key compromised, e.g., DigiNotar
- Algorithm compromised (rare use)

Who Issues Certificates?

- Who can create this binding?
- It's a trust issue—the *other party* to the connection has to trust whoever issued your certificate
- Generally, the decision is made by our software
- Certificate issuers are called *certificate authorities*—CAs

- What makes a CA a CA?
- More precisely, how do you know who is authorized to issue a given certificate?
- And why do you trust them?

- *All* cryptographic trust has to start somewhere
- For any given application, you need to pick your trust anchors
- For the Web, that choice has been made for you: every OS or browser has a built-in CA
- The collection of hardware, software, people, policies, and procedures to issue certificates is called a *PKI*: public key infrastructure

Who Should Issue Your Certificates?

Three situations

- 1 Public-facing website
- 2 Internal corporate website
- 3 Your app talking to your server

Certificates for Public-Facing Website

- Remember that the other party has to trust your CA
- That means that everyone else's browser has to trust the CA you choose
- And that in turn means that you have to use a CA trusted by most browsers in the world
- Microsoft can add its own CAs to Edge, Apple can with Safari, Mozilla can for Firefox, and Google can for Chrome—but even they want their sites to be reachable by everyone else
- Conclusion: you have to trust the common set of CAs, even if you're really big

Certificates for Internal Websites

- In medium or larger-size enterprises, the IT group manages most internal computers
- It generally has the power to install new CAs on employee machines
- Accordingly, companies can use their own CAs internally if they want—but they have to know how to run a CA
- (This is sometimes done for firewalls—more on that later in the term)

How to Issue Certificates

- Typically, user generates a key pair, and presents the public key, identity, and proof of identity
- *Certificate Authority* (CA) signs the certificate and gives it back
- Note: certificates are self-secured; they can be verified offline

“Organizations are regularly told that they are complex, require ultra-high security, and perhaps are best outsourced to competent parties. Setting up a certificate authority (CA) requires a “ceremony”, a term with a technical meaning but nevertheless redolent of high priests in robes, acolytes with censers, and more. This may or may not be true in general; for most IPsec uses, however, little of this is accurate. (High priests and censers are definitely not needed; we are uncertain about the need for acolytes. . .)”

Can You Run Your Own CA?

- Yes, but it's hard
- It's not *inherently* hard, but the documentation is pretty poor
- Two major issues: verifying the identity of the requester, and protecting the certificate-signing key
- There are decent web pages on how to create “self-signed certificates”
- Should I give a homework assignment on creating certificates? Maybe...

Certificate Lifecycle

- 1 Generate a key pair
- 2 Request a certificate from a CA
- 3 Use the certificate
- 4 The private key is compromised
- 5 Revoke the certificate

Let's Encrypt

- The Electronic Frontier Foundation started a free CA called `letsencrypt.org`
- It verifies identity by making sure the requester has control of the web site or the DNS entries
- There are automated client tools, e.g., `certbot` for obtaining certificates

- If you control the app, you control whom it trusts
- Your own app can verify that your own CA issued the certificate your server is using
- You not only don't need a popular CA, you shouldn't use one

But What About Identity?

- Certificates are about binding identity to a public key
- What identity should be there?
- It depends on the type of certificate

Web Site Identities

Validity

Not Before 1/2/2020, 7:00:00 PM (Eastern Daylight Time)

Not After 4/5/2022, 8:00:00 PM (Eastern Daylight Time)

Subject Alt Names

DNS Name nytimes.com

DNS Name *.api.dev.nytimes.com

DNS Name *.api.nytimes.com

DNS Name *.api.stg.nytimes.com

DNS Name *.blogs.nytimes.com

DNS Name *.blogs.stg.nytimes.com

DNS Name *.dev.nyt.com

DNS Name *.dev.nyt.net

DNS Name *.dev.nytimes.com

DNS Name *.newsdev.nyt.net

DNS Name *.newsdev.nytimes.com

DNS Name *.nyt.com

DNS Name *.nyt.net

DNS Name *.nytco.com

DNS Name *.nytimes.com

- For web sites, you need to verify the user-visible name: the domain name
- But many websites have multiple names—if nothing else, `example.com` and `www.example.com`
- Those names must all be listed in the subject Alt Names field

Email	The sender's or the recipient's email address
Executables	The vendor
iPhone Executables	Apple!

Verifying Certificates


- Verifying certificates is a complex business
- There are many things to check, and many ways to get it wrong
- If you can, use a standard library to do this checking

The Certificate Itself

- Is it syntactically correct?
- Is it expired?
- Does the resource's name match what's in the certificate?
- Who signed it?

Certificate Chains

- Very few CAs sign certificates directly—and that's the correct decision
- CAs issue a lot of certificates, which means that the signing key has to be online a lot, and hence could be hackable
- Instead, they periodically issue themselves an intermediate certificate with a limited lifespan

 Limit the effect of any compromise!

- So: you need to verify the certificate all the way up the chain

Certificate Scope

- Some intermediate certificates restrict the scope of the certificates they themselves can sign
- Example: if a company has an “official” CA certificate for internal use, that certificate can only sign other certificates within the company
- This has to be verified, too

Checking the Cryptography

- Are all of the algorithms listed acceptable to you?
- Example: you do *not* want MD5 or short RSA moduli anywhere in the certificate chain
- Does the message or file in question have the right hash?
- Does the signature of that hash validate?
- You then have to verify the signatures on every certificate up to a CA you trust

Security Analysis: Is PKI Secure?

- What are the weak points?
- Who can attack them?

Attacking the Cryptography

- Today's standards—SHA256; 2048-bit RSA or 256-bit ECDSA—are believed to be secure
- But when web encryption started, people used MD5 and 1024-bit RSA; both are currently believed to be insecure
- Some sites continued using older algorithms well after they were known to be bad
- And there have been exploits—but probably by intelligence agencies

- The Flame malware relied on a software vendor using MD5 in its certificates, well after they should have stopped (and well after Windows should have rejected such certificates)
- MD5 was known to be weak—but Flame used a previously unknown cryptanalytic mechanism to attack MD5
- Most hackers are not expert cryptanalysts. . .
- Flame also exploited a design error by Microsoft and hijacked the Windows Update mechanism

- Microsoft has excellent cryptographers; they *knew* that MD5 was broken
- But: they had to preserve backwards compatibility for long enough for all of their customers to migrate to a stronger hash function
- Sites that had upgraded their own infrastructure could still be vulnerable to Microsoft's decision

Private Key Compromise

- Sites' private keys can be stolen
- Stuxnet relied on two different stolen vendor code-signing keys
- (Were these keys in HSMs? They should be)

Identity Spoofing

- In 2001, someone impersonated Microsoft and got two fake code-signing certificates from VeriSign
- VeriSign employees did not check the requester's credentials properly
- But: their back-end audit process caught the problem
- Revocation didn't really work then, so Microsoft shipped a patch to ignore those two certificates

- Windows 10 recently had a certificate-checking bug: under some circumstances, it would accept *anything*
- This bug was found by the NSA, which thought it was so scary it told Microsoft and issued its [own advisory](#)

- Code-signing certificates are (at best) statements of where code came from, not that it's harmless
- Some hackers have purchased genuine code-signing certificates from Apple or Microsoft
- Victims' operating systems accept the malware, because it's digitally signed
- There is a confusion of identity with authorization and/or benignity

- Hackers, believed to be linked to Iran, compromised the DigiNotar and Comodo CAs
- The DigiNotar private keys were stored in an HSM, but the attacker controlled the computer that controlled the HSM, and hence was able to get it to sign bogus certificates
- The attack put DigiNotar out of business

Conclusions

- Certificates are useful but they aren't panaceas
- Buggy code can beat good crypto
- Note the analysis: at some point, *every piece* of the security chain for certificates has been compromised

Questions?



(American redstart, Morningside Park, September 21, 2020)