Should You Trust the Padlock? Web Security and the "HTTPS Value Chain"

Keeping Current

20 November 2013

Ken Calvert

Outline

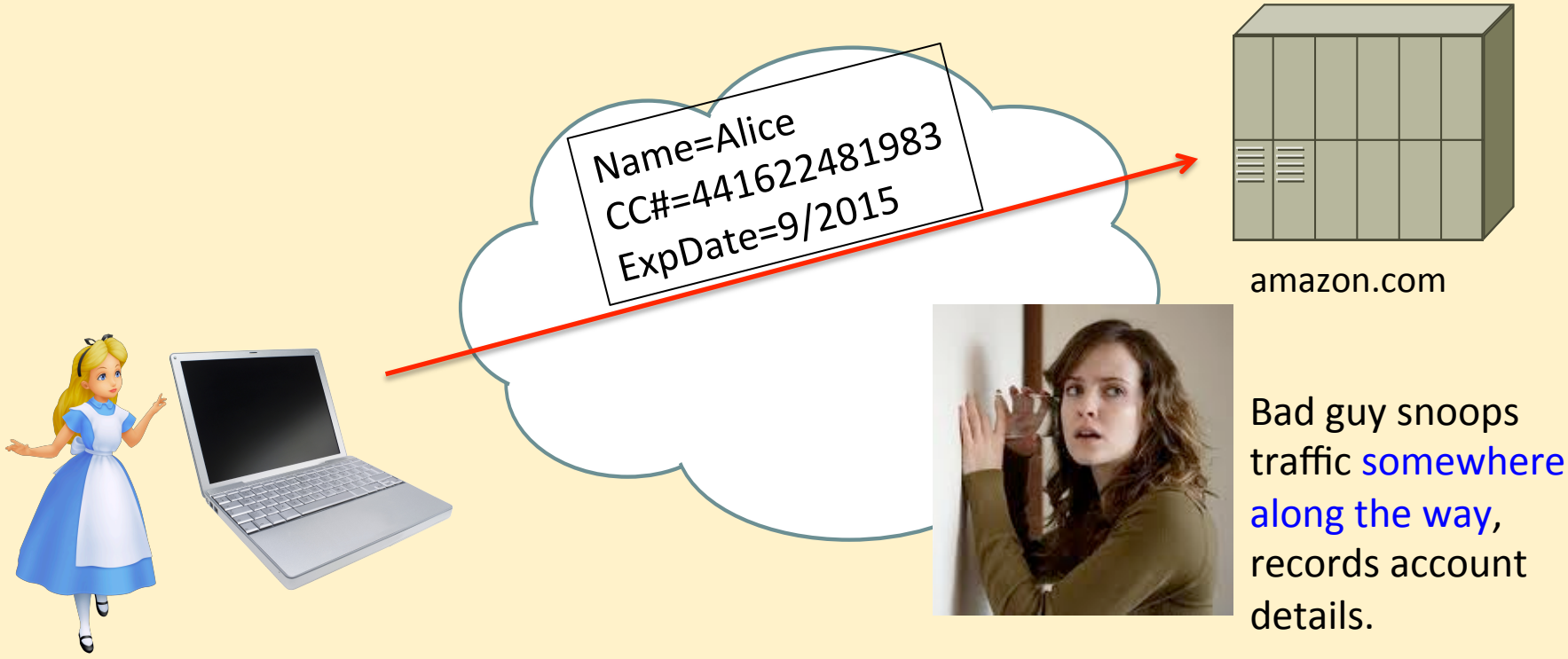
1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

Outline

1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

Threats:
What are we afraid of?

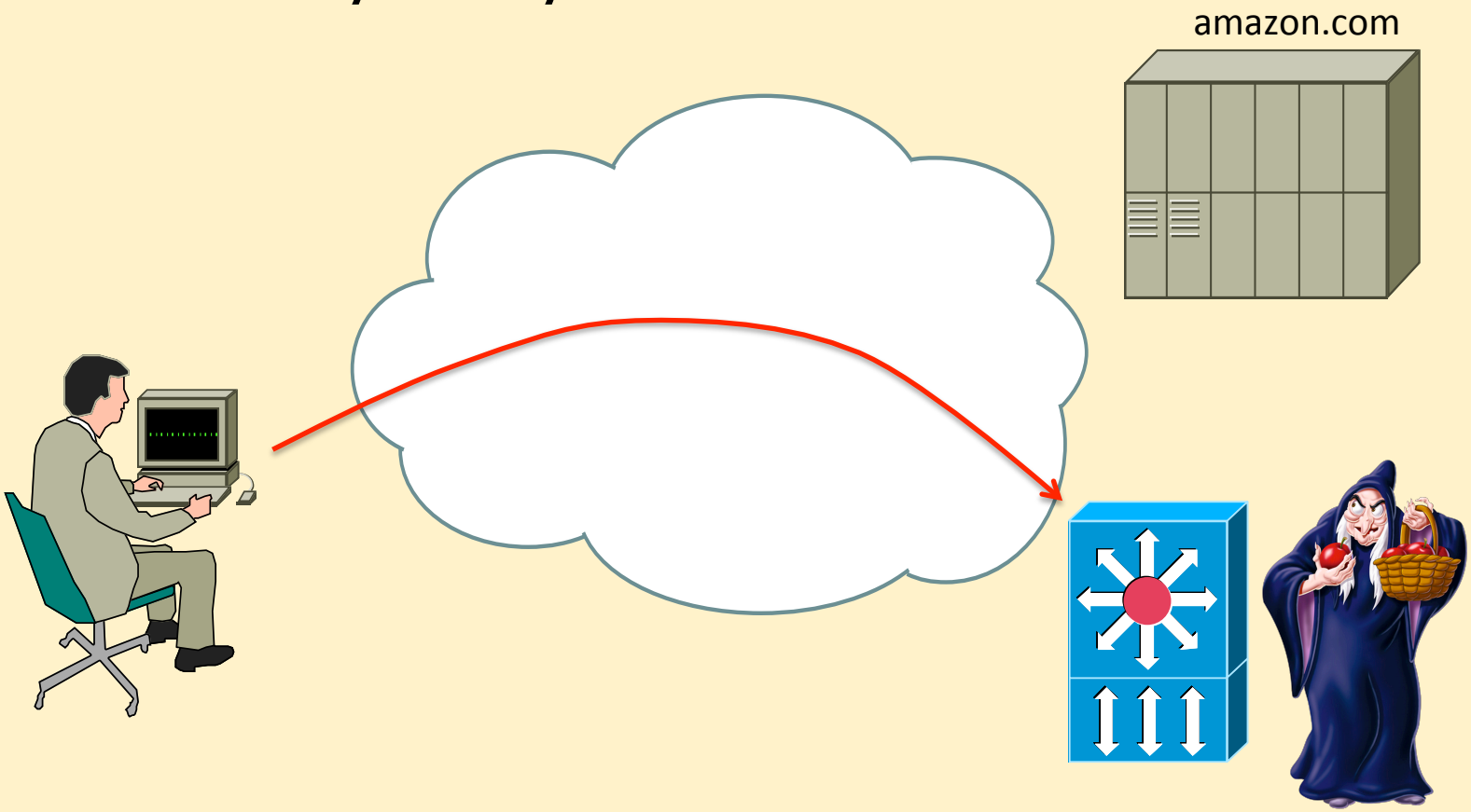
Eavesdropping: sensitive information carried in HTTP messages can be read by intruders.



Bad guy snoops traffic somewhere along the way, records account details.

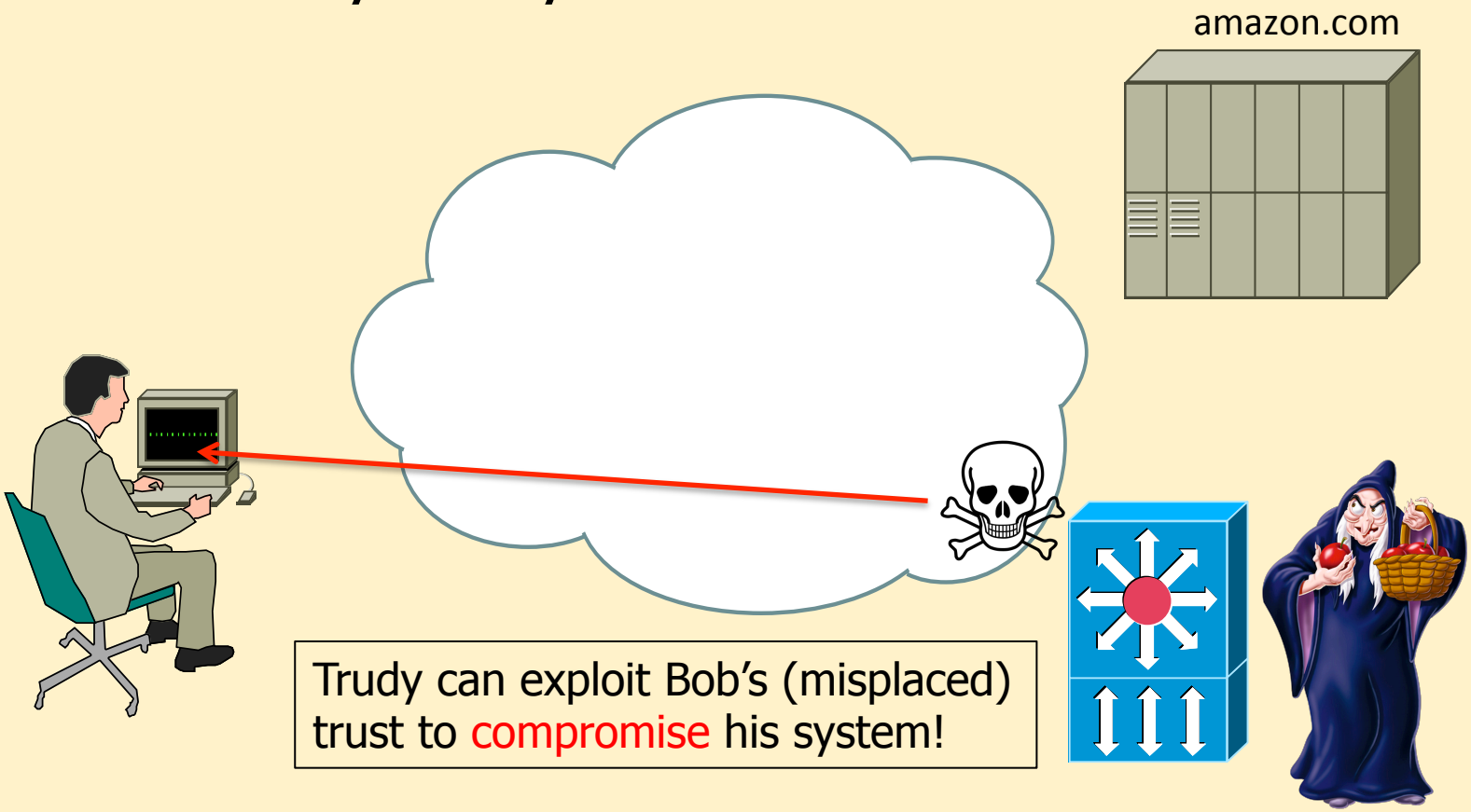
Threats:
What are we afraid of?

Impersonation: Bob thinks he is talking to Amazon, but it's really Trudy's fake site.



Threats:
What are we afraid of?

Impersonation: Bob thinks he is talking to Amazon, but it's really Trudy's fake site.

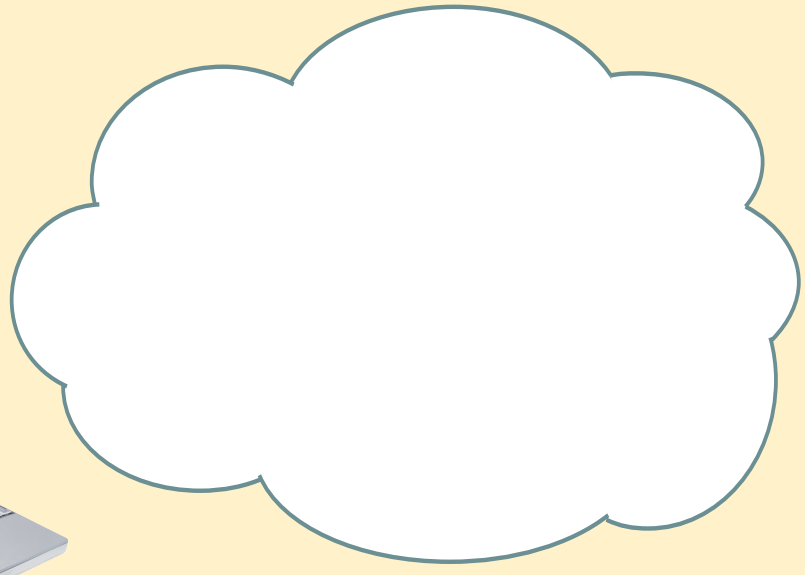


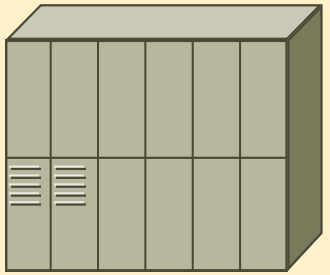
Outline

1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

Countermeasures

Confidentiality: end-to-end encryption prevents eavesdropping

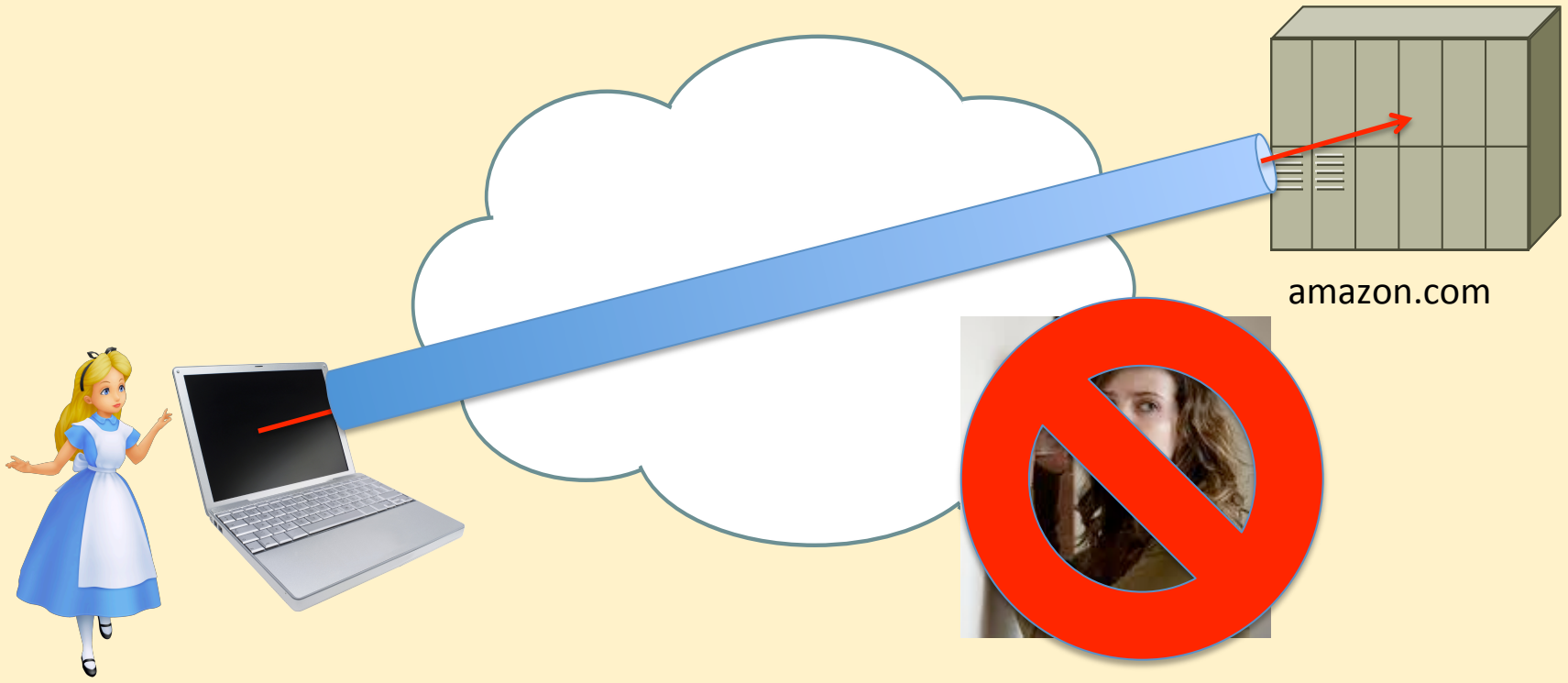




amazon.com

Countermeasures

Confidentiality: end-to-end encryption prevents eavesdropping



Countermeasures

Authentication: Bob can tell if he is talking to the real Amazon.com. (More precisely: his browser can.)

Securing the Web

- Secure Sockets Layer (Netscape) and Transport Layer Security (IETF) were developed (ca. 1995-6) to secure the channel between client and server
  - Confidentiality: Prevent eavesdropping
  - Authentication: Detect impersonation
- These are general protocols, designed for use by any application running over TCP
  - HTTPS = Hypertext Transfer Protocol over SSL/TLS

    Both SSLv3 and TLSv1-3 are in common use, but only TLS is still being updated with new ciphersuites
- Both use public key cryptography to authenticate the server and establish confidentiality
- Authentication turns out to be the main challenge

Outline

1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

Public Key Cryptography

- Basic idea:
  - Keys come in pairs
    - one key is public (known to anyone)
    - one key is private (known only to Bob)
  - Basic operations:

    signature = sign(message, private key)

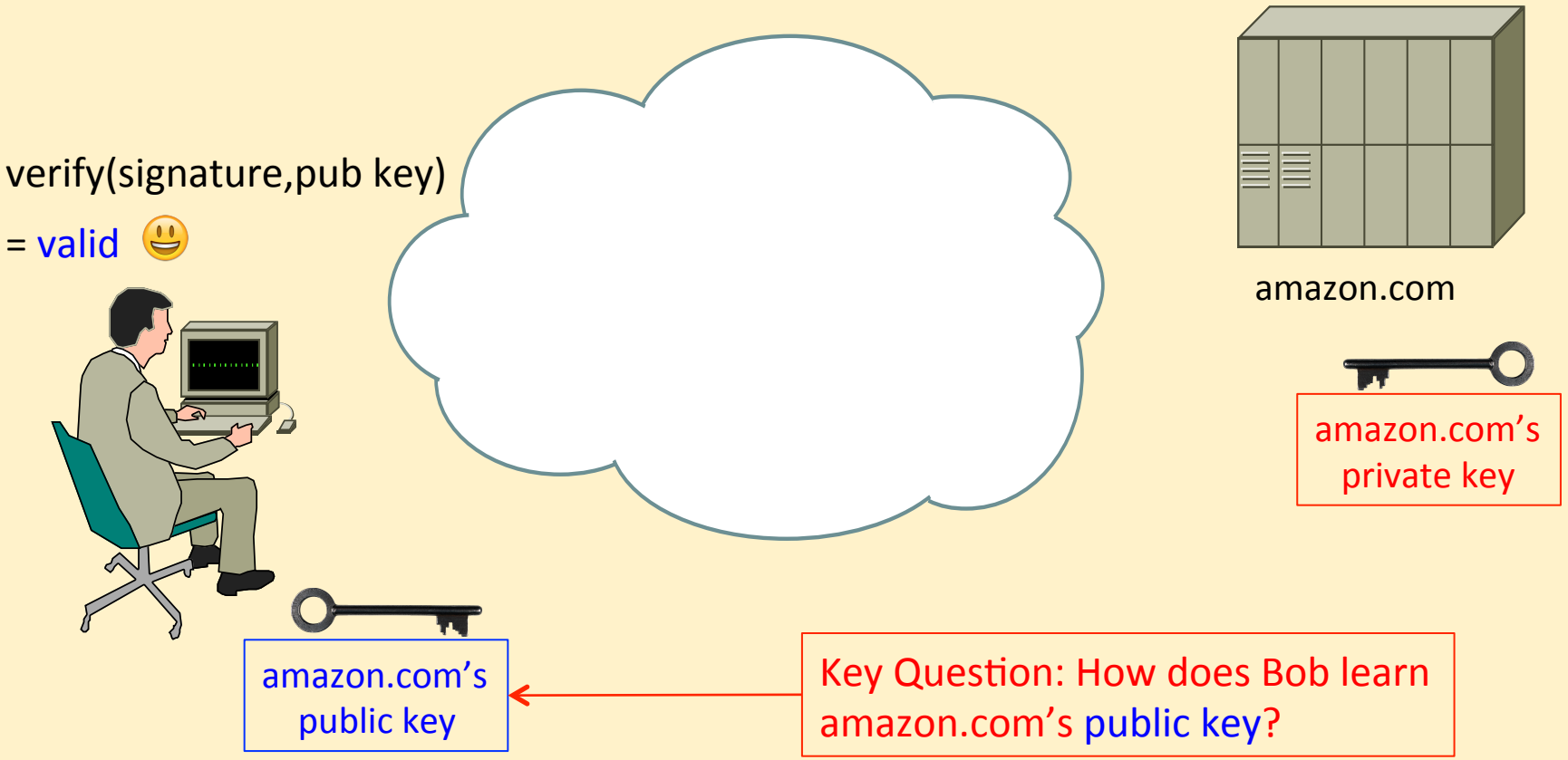
    verify(signature, public key) ⟶ valid | invalid
- Mathematics of the algorithm (plus assumptions about hardness of certain problems) ensures that a valid signature cannot be created without knowledge of the private key.

Public Key Authentication

Authentication: Bob can tell if he is talking to the real Amazon.com. (More precisely: his browser can.)

Public Key Authentication

Authentication: Bob can tell if he is talking to the real Amazon.com. (More precisely: his browser can.)



Public Key Distribution

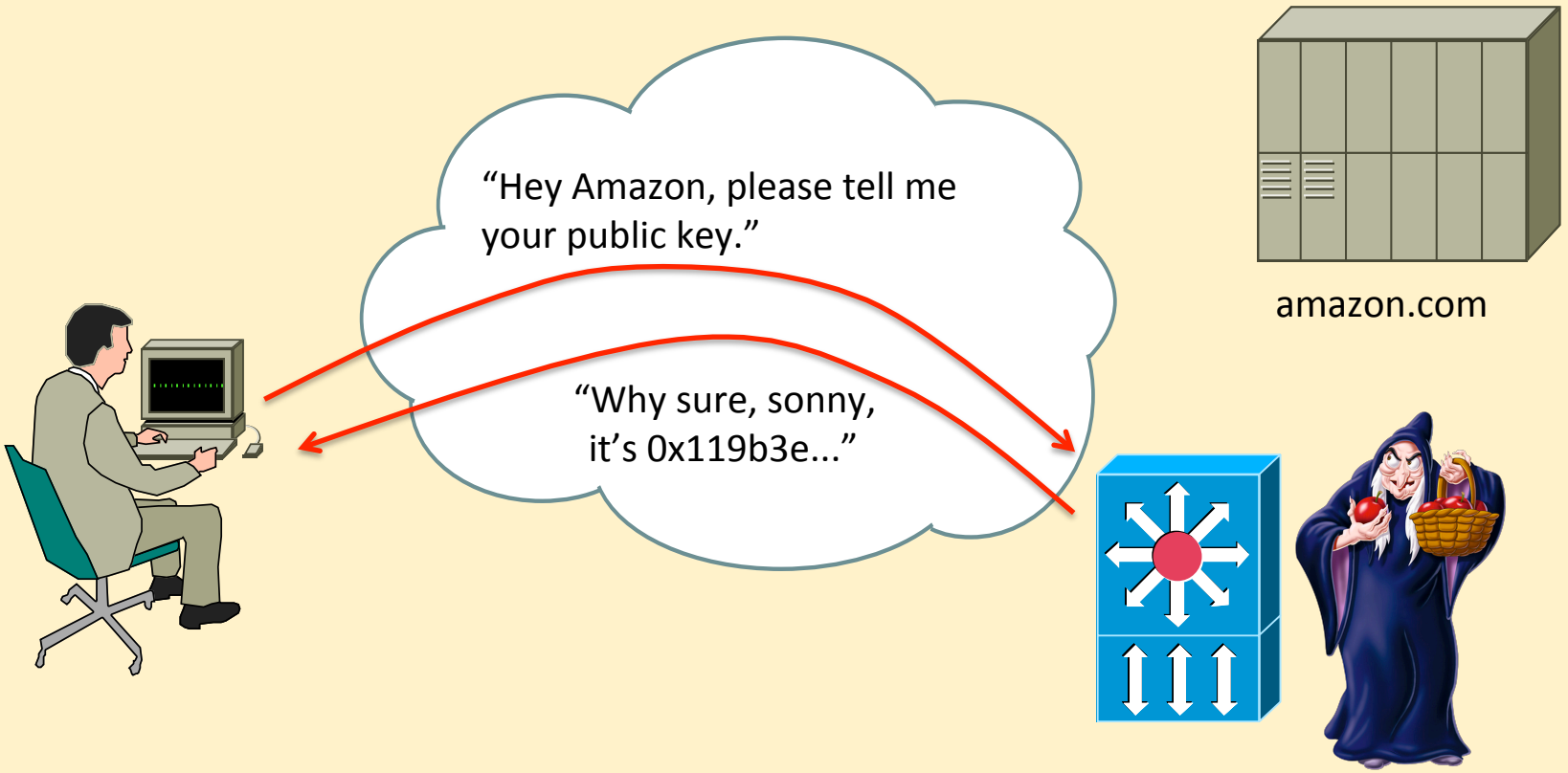
Ways for Bob to learn Amazon.com's public key:

- Ask the server?
  - No: This is begging the question!

    We don't know we're really talking to Amazon!

Public Key Distribution

Why Bob can't just ask the server for its public key...



Public Key Distribution

Ways for Bob to learn Amazon.com's public key:

- Ask the server?
  - No: This is begging the question!

    We don't know we're really talking to Amazon!
- The key comes pre-installed in the browser?
  - No: This doesn't scale!

    Millions of sites need HTTPS; new ones may arise every day.

Public Key Distribution

Ways for Bob to learn Amazon.com's public key:

- Ask the server?
  - No: This is begging the question!

    We don't know we're really talking to Amazon!
- The key comes pre-installed in the browser?
  - No: This doesn't scale!

    Millions of sites need HTTPS; new ones may arise every day.
- Trusted 3rd parties certify the binding between entity and public key (by signing the binding)
  - Browser comes equipped with the public keys of a limited number of these Certificate Authorities

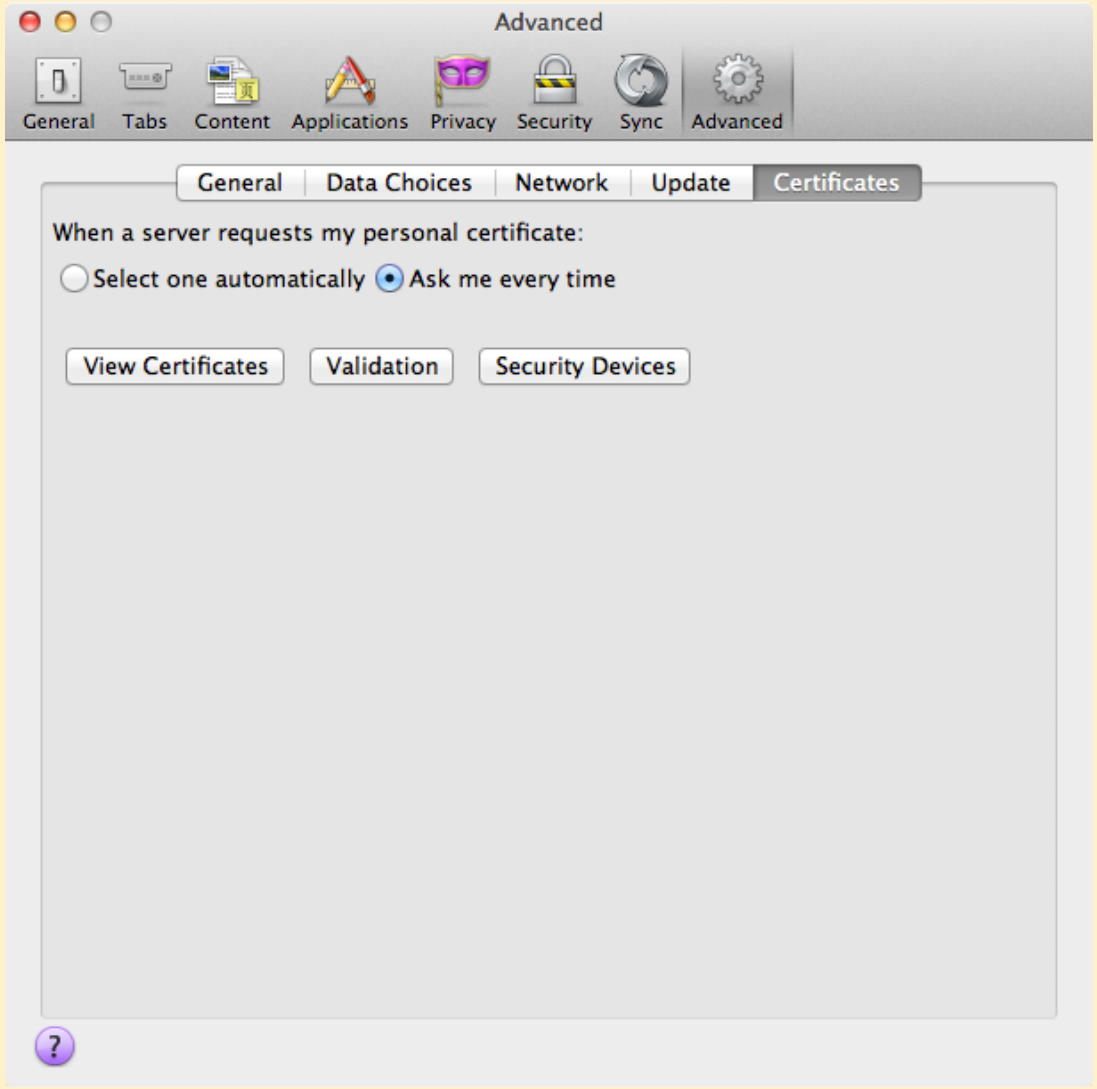
Public Key Certificates

- A trusted 3rd party attests to Amazon's public key
- Reduces the problem:
  1. Get the CA's public key.
  2. Given a server's (amazon.com's) certificate (issued by that CA), verify the CA's signature on the cert.
  3. Use the certified public key to verify the server's identity
- CA public key is a root of trust
  - CA can sign keys of other CAs and/or end users (amazon)
  - Scales (as usual) by adding hierarchy

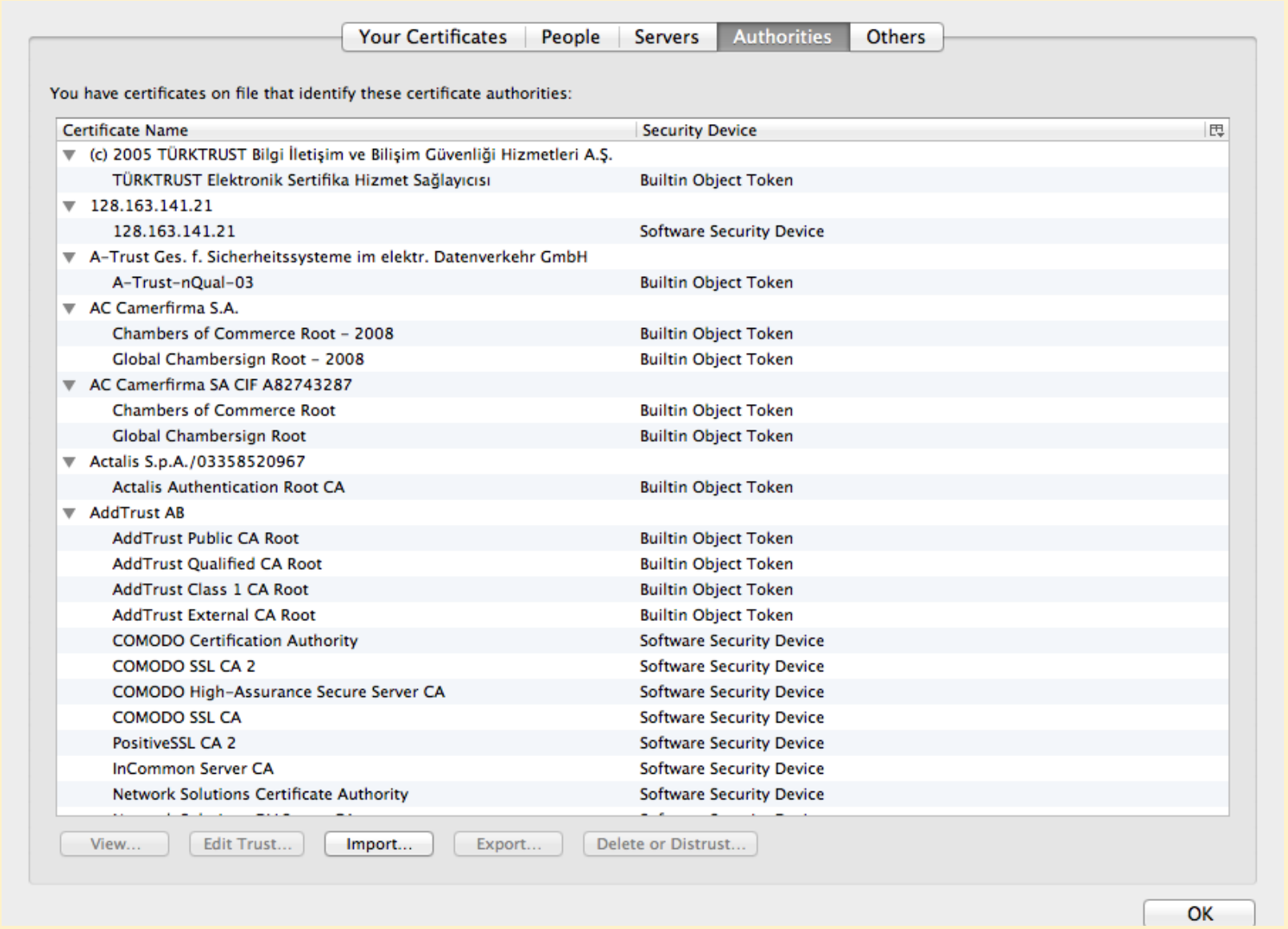
Certificate Authorities (CAs)

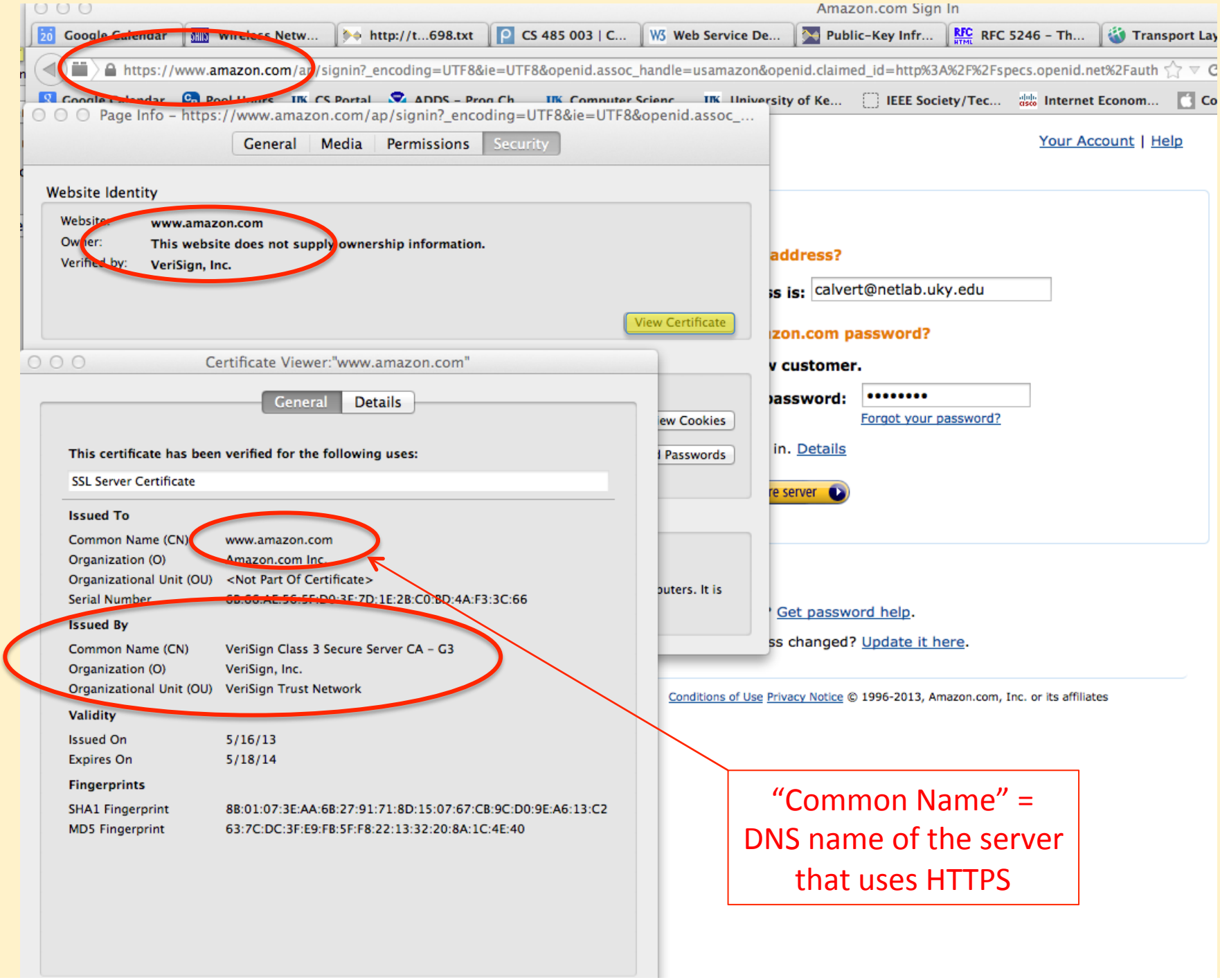
- CA Public keys are distributed out-of-band
  - Usually in the form of a self-signed certificate
  - Browsers come preconfigured with CA certs
- In general, the job of a CA is to make sure that it only issues certificates that are legitimate.
  - What should you have to do to get a certificate?
    - Tradeoff: ease of acquiring vs. ease of impersonation

CA Public Keys in Browsers

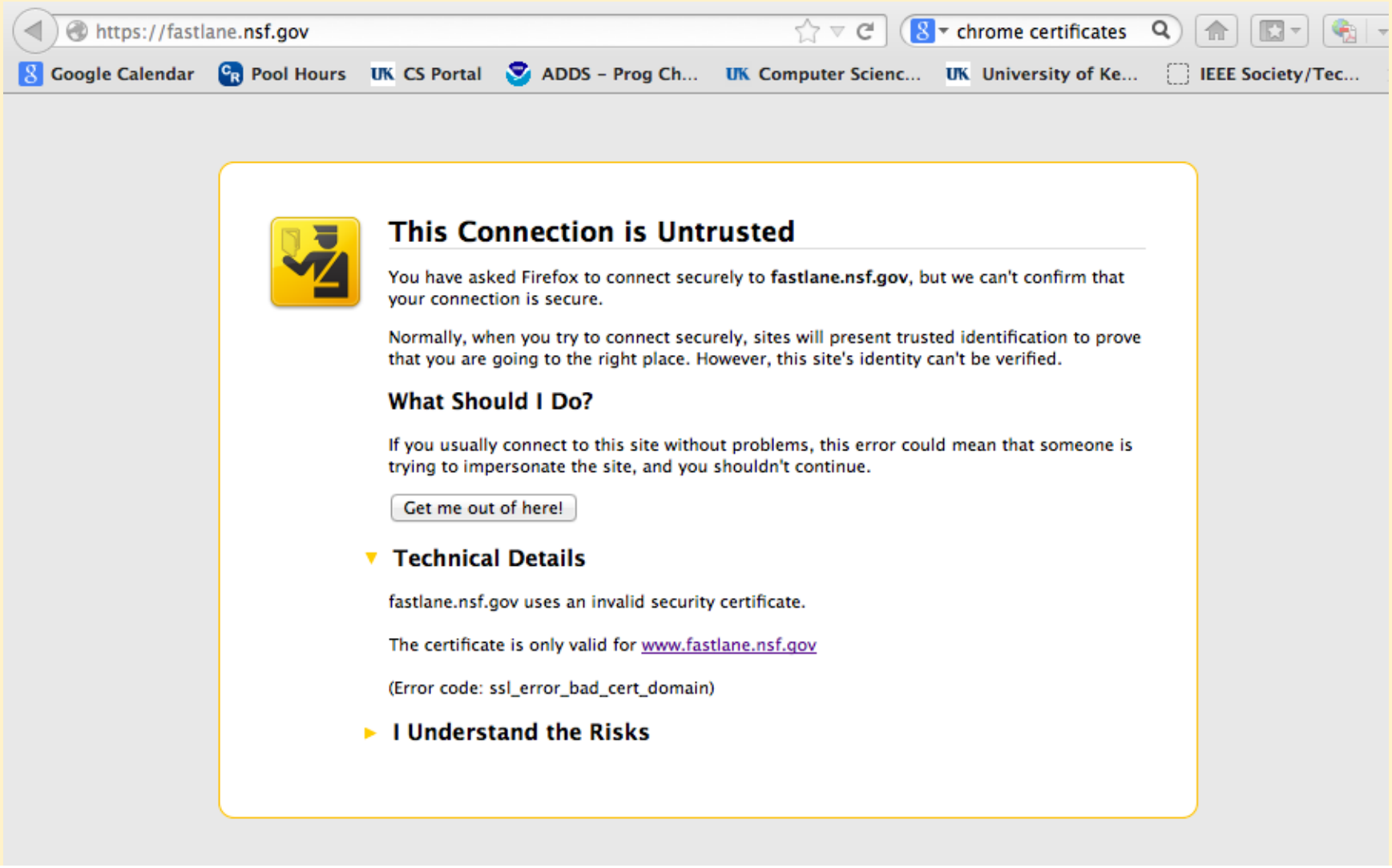


CA Public Keys in Browsers

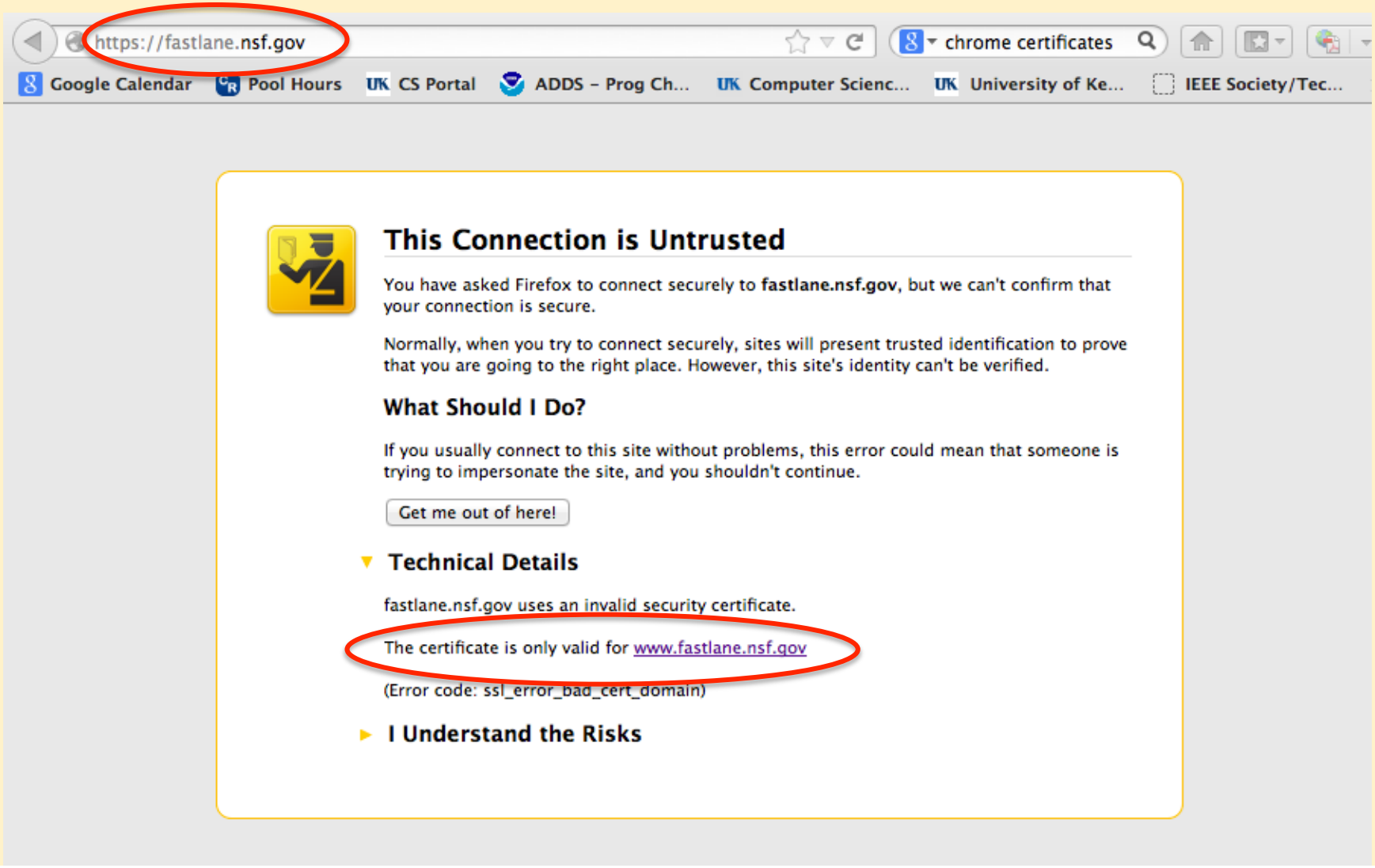



Amazon.com Sign In
https://www.amazon.com/ap/signin?_encoding=UTF8&ie=UTF8&openid.assoc_handle=usamazon&openid.claimed_id=http%3A%2F%2Fspecs.openid.net%2Fauth
Page Info – https://www.amazon.com/ap/signin?_encoding=UTF8&ie=UTF8&openid.assoc_...
General
Media
Permissions
Security
Website Identity
Website: www.amazon.com
Owner: This website does not supply ownership information.
Verified by: VeriSign, Inc.
View Certificate
Certificate Viewer:"www.amazon.com"
General
Details
This certificate has been verified for the following uses:
SSL Server Certificate
Issued To
Common Name (CN) www.amazon.com
Organization (O) Amazon.com Inc.
Organizational Unit (OU) <Not Part Of Certificate>
Serial Number
Issued By
Common Name (CN) VeriSign Class 3 Secure Server CA - G3
Organization (O) VeriSign, Inc.
Organizational Unit (OU) VeriSign Trust Network
Validity
Issued On 5/16/13
Expires On 5/18/14
Fingerprints
SHA1 Fingerprint 8B:01:07:3E:AA:6B:27:91:71:8D:15:07:67:CB:9C:D0:9E:A6:13:C2
MD5 Fingerprint 63:7C:DC:3F:E9:FB:5F:F8:22:13:32:20:8A:1C:4E:40
Your Account | Help
calvert@netlab.uky.edu
Forgot your password?
Get password help.
Update it here.
Conditions of Use Privacy Notice © 1996-2013, Amazon.com, Inc. or its affiliates
"Common Name" = DNS name of the server that uses HTTPS

If Certificate Validation Fails...



Certificate Validation Fails...what to do?



CA Public Keys in Browsers

- Firefox comes with 130+ roots of trust (CAs public keys) pre-installed
  - Other browsers similar, but...
- Roots of trust may vary with browser and platform

Trust Structures

Basic Question: what authorities do I trust?

- Monopoly
  - Single root of trust, everybody knows its key, which never changes
  - Obvious problems
- Hierarchy of CAs
  - Root certifies "child" CAs, which may certify other CAs or regular users
  - Benefit: easier to get to a CA near you
  - Drawback: still a single root of trust

Trust Structures

- Web of Trust
  - Individuals (Alice, Bob) sign keys of people they trust
  - I collect public keys of people I know
  - When presented with a new public key, try to find a chain of people I trust, ending with someone who signed it
  - This is used in Zimmerman's PGP ("Pretty Good Privacy")
  - Issue: scalability, reliability
- "Oligarchy" (name due to Kaufman, Perlman and Speciner)
  - Multiple roots of trust, each signs certificates
  - Trust only public keys signed by one of these CAs
  - How to choose a CA?

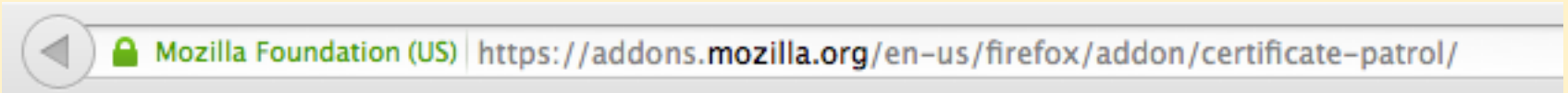
Outline

1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

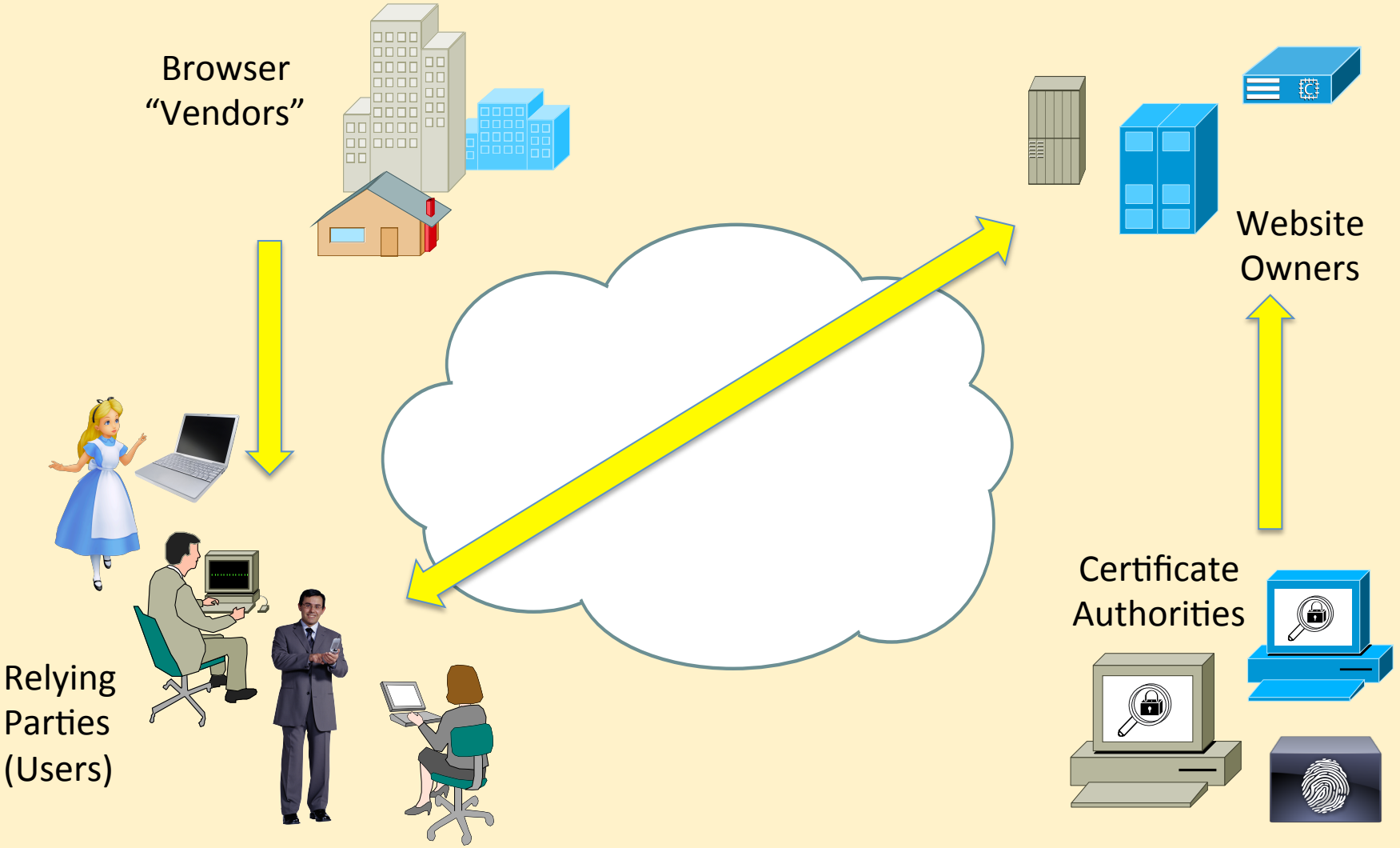
Levels of Certification

Certificates come in different "levels":

- Domain Validated (DV)
  - Issuing CA verifies "control" of the domain name
    - In practice: answer an email to the address listed in the SOA record of the DNS zone (WHOIS database)
  - Process can be automated ⇒ fast turnaround
- Organization Validated (OV)
  - No standards for what this means
  - Typical: verify organization's contact information via third party source (Secretary of State, telephone directory, ...)
- Extended Validation (EV)
  - More extensive validation process
  - More expensive
  - Browser indication: "green bar"

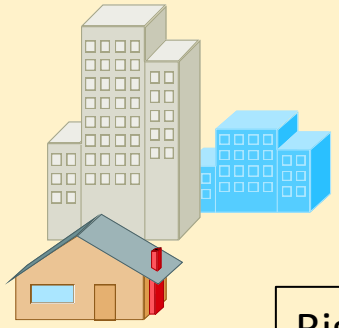


Who are the Stakeholders?



Who Risks What?

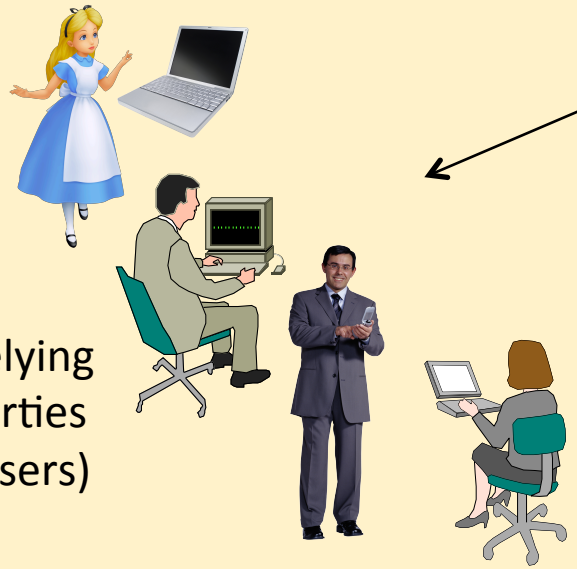
Browser "Vendors"



Website Owners

Risk: loss of business if users don't believe their site is secure

Risk: loss of sensitive information, time, $$ if they trust a rogue site

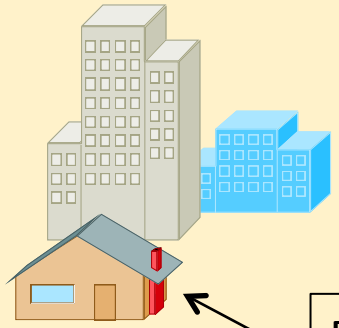


Relying Parties (Users)

Certificate Authorities

Who Risks What?

Browser "Vendors"



Website Owners

Risk: loss of market share if browser trusts a bad CA

Risk: bankruptcy if removed from browser trust chain

Relying Parties (Users)

Certificate Authorities

What Does the Market Look Like? [1]

- EFF's SSL Observatory project (December 2010)
  - Collected 1.5M valid certificates from around the web
  - Identified ~1100 issuing CAs
- Highly concentrated
  - 3 vendors account for more than ¾ of the market
    - Symantec (includes Verisign and Thawte)
    - GoDaddy
    - Comodo
- Widely varying prices

[1] "Security Economics in the HTTPS Value Chain" by Asghari, van eeten, Arnbak & van Eijk, 2013

Price Variations

Cert Type	Minimum Price	Maximum Price	Avg (Std. Dev.)
DV	$0	$249	$81 (74)
OV	$38	$1172	$258 (244)
EV	$100	$1520	$622 (395)

Market Share

Certificate Type	Market Leaders
DV	GoDaddy (40%), Symantec/GeoTrust (36%), Symantec/Thawte (10%)
OV	Symantec (54%), Comodo (21%), Entrust (6%), Network Solutions (5%)
EV	Symantec (68%), Comodo (7.9%), Godaddy (5.2%)

Observations

- This *should* be a commodity market:
  - Browsers do not distinguish between cert providers!
    - Certificates are "perfectly substitutable"
  - Buyers cannot distinguish between more/less secure sellers (CAs)!
  - High fixed costs, (very) low marginal costs
- Expect to see competition on price only

  "race to the bottom"
- Instead: price variability, market dominated by large players
  - What gives?

Competition Among CAs

What are CAs' customers buying?

- Brand reputation

  "Nobody ever got fired for buying Verisign [now Symantec]."

- Additional services
  - E.g., certificate management services
- Some CAs may be *"too big to fail"...*

Risks to the System - I

The DigiNotar Incident

- DigiNotar, a CA in the Netherlands
  - Served as CA for some Dutch government functions
  - Included as trust root in
- Hacked in July 2011
  - Attacker accessed root CA system and issued a wildcard certificate for google.com
    - Subsequently used in a large-scale MITM attack on 300K users in Iran
    - In the interim 531 certs for 53 domain names were issued
  - Incident did not become public until September 2011
  - After investigation, Dutch government took over DigiNotar
- Removed from browser CA lists shortly after
  - DigiNotar declared bankruptcy

Risks to the System - II

Other Incidents

- Larger CAs have also been hacked
  - Verisign (RSA) breach in 2010, not publically acknowledged until 2012
  - Comodo has reportedly been breached several times
- None have been removed from browser CA lists
  - Some CAs are likely *too big to fail.*

Outline

1. What are we afraid of?
2. Countermeasures: Securing the Web
3. Public-key Crypto and Certificate Authorities
4. A Look at the "CA ecosystem"
5. Problems and Solutions

Systemic Problems

- Any CA can issue a certificate for any site.

  That is: trustworthiness of amazon.com's cert does *not* depend (only) on the security practices of its issuer.

  ...also depends on the practices of all other CAs!

  Trustworthiness of the entire system cannot exceed the trustworthiness of its weakest component.

- Information asymmetry abounds.
  - Security practices of CAs are not visible to the stakeholders who are most at risk.

    "There seems to be wide consensus that the average end-user cannot reasonably be expected to exert control over the HTTPS ecosystem." [1]
  - CAs have strong incentives not to reveal security incidents.
- Risks to some large CAs are externalized.

[1] "Security Economics in the HTTPS Value Chain" by Asghari, van eeten, Arnbak & van Eijk, 2013

Potential Solutions

- DANE: DNS-based Authentication of Named Entities
  - Store cert-related information in DNS to increase trust
  - E.g., name of CA authorized to issue certs for amazon.com
  - In the limit: public key info
  - Requires DNSSEC deployment to secure the DNS info
- Convergence, Perspective (convergence.io)
  - Rely on consensus of a set of *Notaries* to determine reliability of a cert
  - Users set their own policies on which Notaries to trust
  - Anyone can be a trust Notary

Summary

- The current architecture of trust for HTTPS (indeed, anything using SSL/TLS) is broken.
  - Information asymmetry abounds
  - Brand reputation is about the only competitive factor
  - Incentives are unclear, even perverse
    - E.g., browser vendors consider everything in terms of performance
  - Some CAs are "too big to fail"
- Good technological solutions exist.
  - Most involve adding new sources of info/replacing CAs
  - But it will take a while for them to be deployed
- The real question is not "Should I trust the padlock?" but "Do I have a choice?"

More Stuff to Keep you Up at Night

- What should public keys actually be bound to?
  - Domain names?
  - Organizations?
  - People?
- Can you tell the difference between "KINKOS" and "KINKOS"?

  Fourth letter in the first is Unicode 0x4B, LATIN CAPITAL LETTER K; in the second it is 0x039A, GREEK CAPITAL LETTER KAPPA
  - What stops me from registering the second one under .com?