The X.509 PKI for the WWW

Ralph Holz

Network Architectures and Services
Technische Universität München

November 2012

WWW is secured with SSL/TLS and X.509

SSL/TLS

- The backbone protocols for securing the WWW (and e-mail)
- We will talk about the exact protocol flow later
- Goals: authentication, confidentiality, integrity
- Employs public-key cryptography

X.509: Public Key Infrastructure standard

- Part of the X.500 family of standards
- X.500 vision: global directory to store and retrieve entity information
- Certification = digital signature: $Cert(X) = Sig(id_X, pubkey_X)$
- The idea of certificates is much older than SSL

X.509 and SSL/TLS

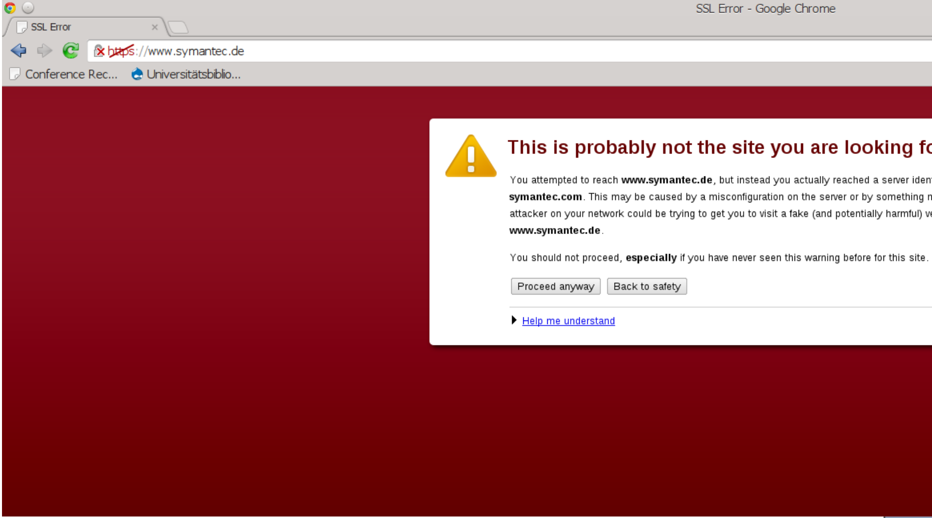
SSL includes certificate-based authentication

- Original design of SSL by Netscape (Mozilla!)
- Goal: protect sensitive information like cookies, user input (e.g., credit cards)
- The attack model in mind was more a criminal attacker, less a state-level attacker

X.509 seemed to fit the bill

- X.509 is an ASN.1-based certificate specification
- X.500 vision: global directory to identify entities
- Trusted Certification Authorities (CAs) issue certificates
- Certification by digital signature: $Cert(X) = Sig_{CA}(id_X, pubkey_X)$

Something went wrong, somewhere



Scope of this lecture

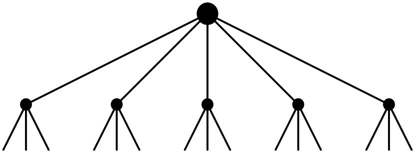
Lectio est divisa in partes tres:

- Part 1:
  Comprehensive overview of X.509 for the WWW
  (relevant for exam)
- Part 2:
  Results of the past 2 years investigating X.509 PKI deployment (not relevant for exam)
- Part 3:
  Several approaches to replace or improve the current PKI
  (relevant for exam)

Part 1:
X.509 for the WWW

Ideal PKI

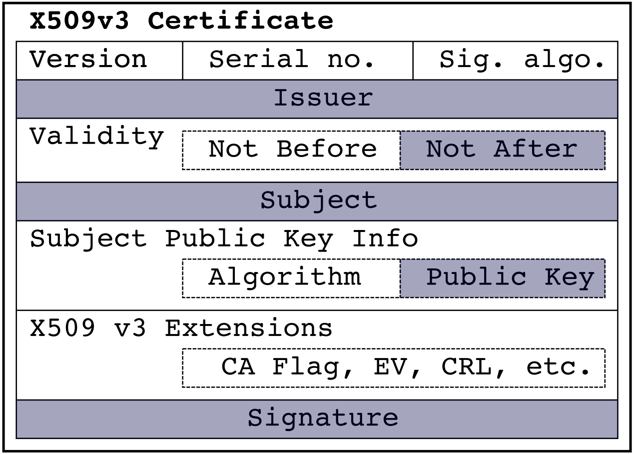
Globally responsible Certification Authority, certificate chains



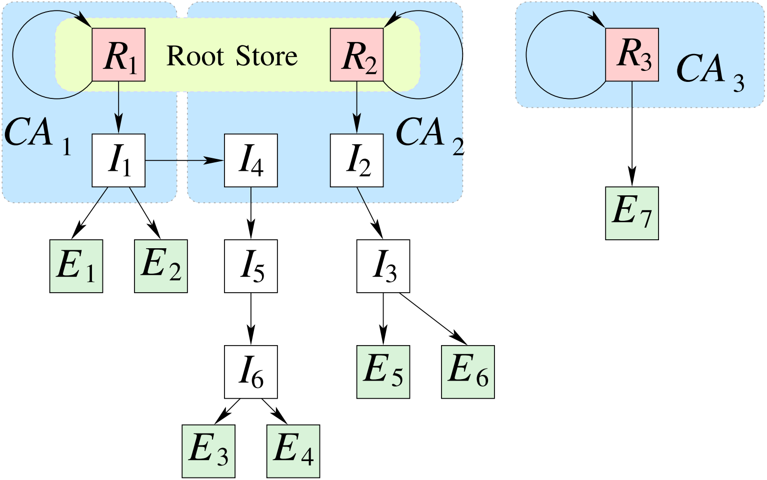
Scalability

- Large number of DNS domains (`.com` $>$ 100M alone)
- Who should be/run the Global CA? There is no universally trusted entity.
- Commercial CAs have become responsible for issuing certificates.

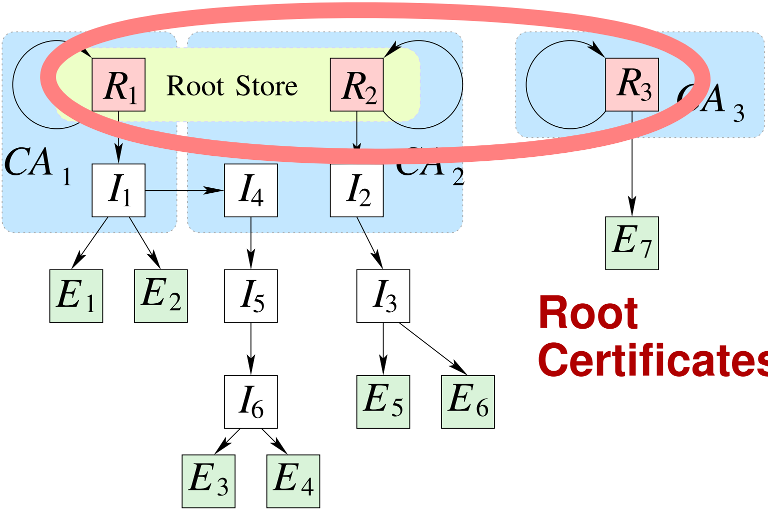
An X.509 Certificate



CAs in X.509 for the WWW



CAs in X.509 for the WWW



Root Certificates

Root stores

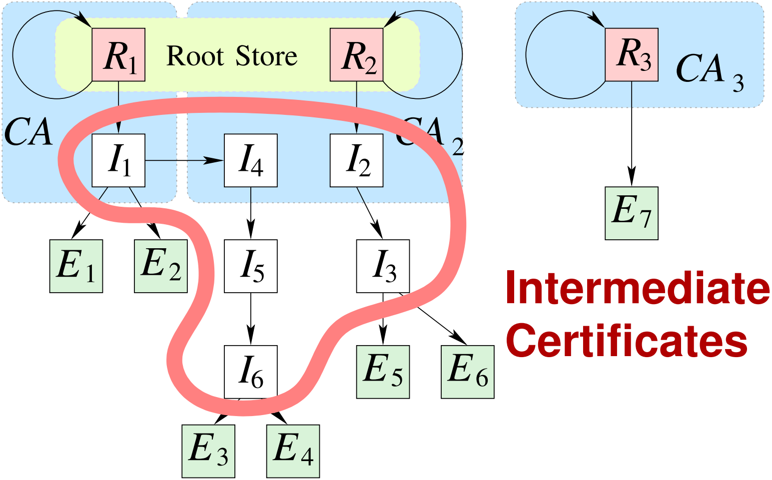
Root stores: certificates of trusted CAs

- ‘Trusted’ = trusted to issue certificates to the correct entities
- *Every* application that uses X.509 has to have a root store
- Operating Systems have root stores: Windows, Apple, Linux
- Browsers use root stores: Mozilla ships their own, IE uses Windows’ root store, etc.

Root store processes

- Every root store vendor has their own process to determine if a CA is added or not
- A CA’s *Certification Policy Statements (CPS)* are assessed
- Mozilla: open discussion forum (but very few participants)
- Commercial vendors (Microsoft, Apple): little to no openness

Intermediate Certificates



Intermediate Certificates

Intermediate certs: part of a certificate chain, but neither a root certificate nor an end-entity certificate.

There are two primary reasons to use intermediate certificates:

- To delegate signing authority to another organisation: sub-CA
- Protect your main root certificate:
  - Intermediate cert is operated by the same organisation
  - Allows to store root cert in the root store, but private key may remain offline in some secure location
  - Online day-to-day operations can be done using the private key of the intermediate cert
  - Also makes it very easy to replace the intermediate cert in case of compromise, or crypto breakthroughs (e.g. hash algorithms) etc.

Hazards of Intermediate Certificates

Intermediate certs have the same signing authority as root certs:

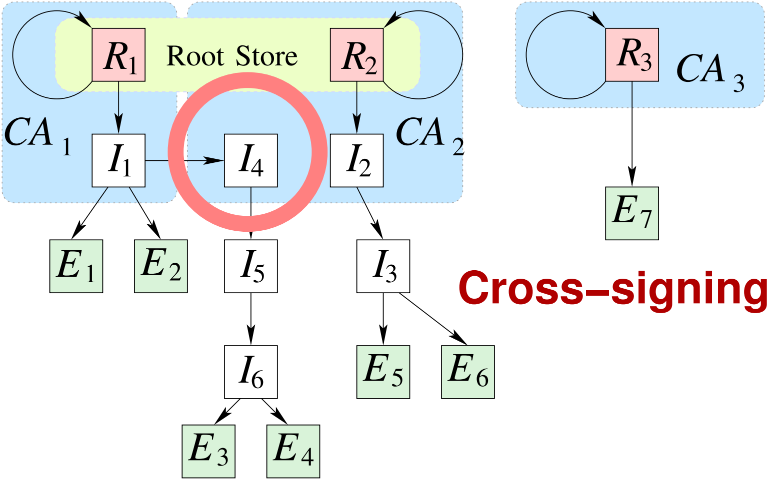
- There are no technical restrictions on what they can sign (e.g., DNS limitations)
- N.B.: DNS restrictions are in the standard, but little used
- The restriction must be supported by the client, too

Hazards of Intermediate Certificates

Some companies/organisations have SSL proxies

- They monitor their employees' traffic
- May make sense in order to avert things like industrial espionage
- However, some CAs have issued intermediate certs to be used as sub-CAs in proxies or added to client root stores
- This allows transparent rewriting of certificate chains– a classic Man-in-the-middle attack
- Worst: the holder of the sub-CA is suddenly as powerful as all CAs in the root store
- Since outing of first such CA, Mozilla requires practice to be disclosed, and stopped

Cross-signing

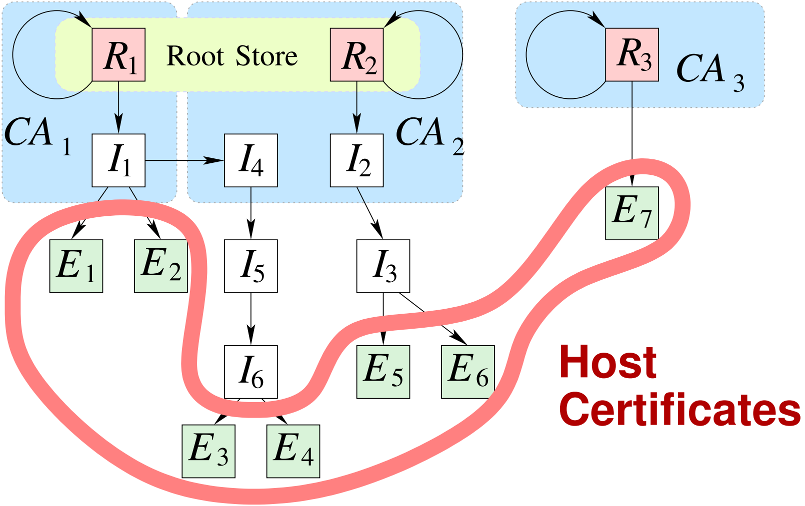


Cross-signing

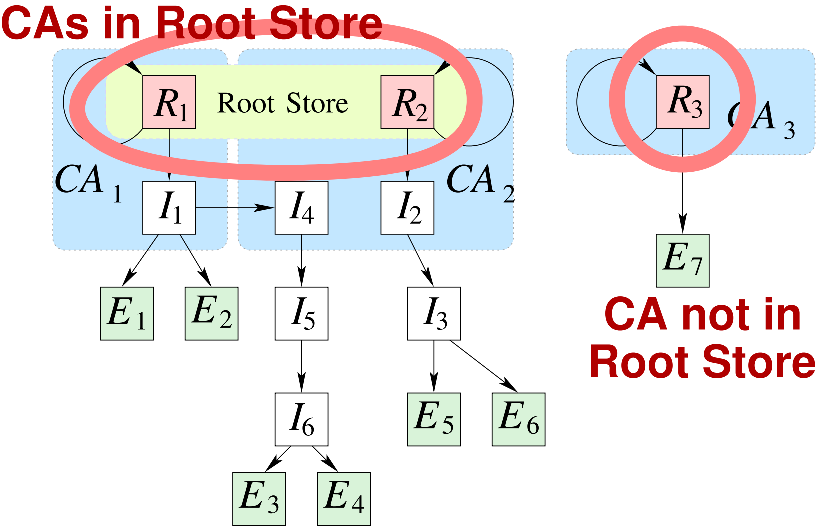
A CA signs a root or signing certificate of another CA

- A special case of intermediate cert
- In a business-to-business model, this makes sense:
  - Two businesses wishing to cooperate cross-sign each other
  - Makes it easy to design business processes that access each others' resources via SSL/TLS
- For the WWW, it completely breaks the root store model
- A new CA can be introduced, subverting control of the root store vendor
- This has happened. CNNIC (Chinese NIC) was cross-signed by Entrust, long before they became part of the root store in Mozilla
- Inclusion of CNNIC caused outrage anyway

End entities in X.509: DNS host name

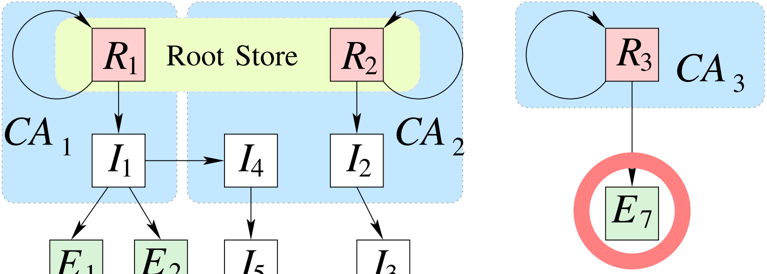


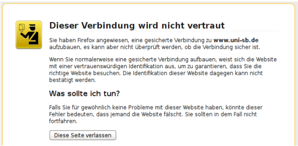
A CA is not in your root store?



One source of WWW errors

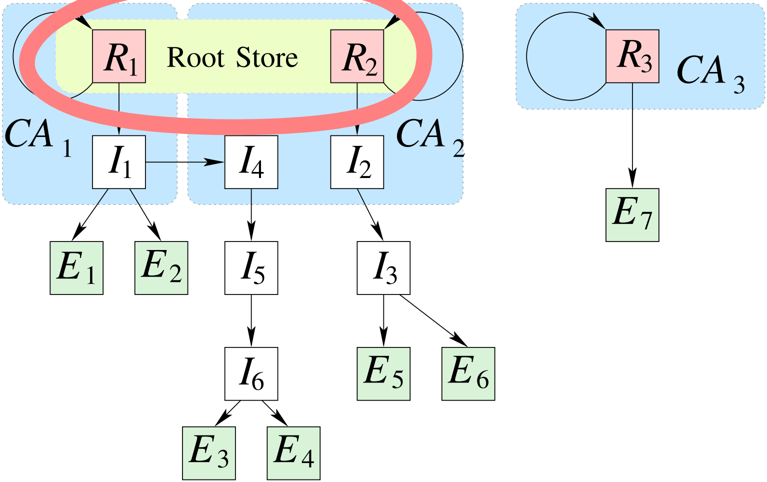
Root certificate not in Root Store





Root Stores Contain CA Certificates

CAs in Root Store



Browser (Client) Root Stores

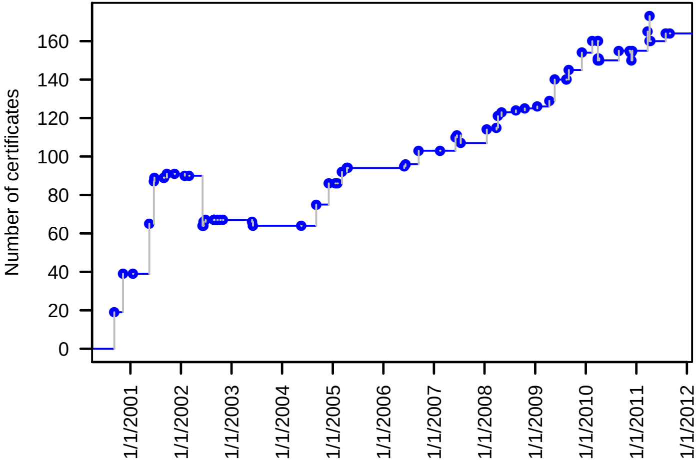
Remember:

- Your browser or your OS chooses the ‘trusted CAs’. Not you.
- All CAs have equal signing authority (there are efforts to change this)
- Any CA may issue a certificate for any domain.
- DNS path restrictions are a possibility; must be set by the CA in their signing cert
- A globally operating CA cannot feasibly set such restrictions in their root cert

The weakest CA determines the strength of the whole PKI. This is also true if the CA is a sub-CA.

Development of Mozilla Root Store

At times, more than 150 trustworthy Root Certificates



Certificate Issuance

How is a certificate issued in practice?

- Domain Validation:
  - Send e-mail to (CA-chosen) mail address with code
  - Confirmed ownership of mail address = ownership of domain
- Organisational Validation (OV, rare)
- Extended Validation (later, rare)

Some argue from an economical persepctive: ‘race to the bottom’

- CAs have only incentive to lower prices
- That translates into incentive to reduce costs = do less checks, faster

Certificate Revocation

Why do we need revocation?

- In theory, no certificate should be considered valid without a revocation check
- There are several cases when an already issued certificate must be withdrawn. Examples:
  - Corresponding private key compromised
  - CA compromised
  - Certificate owner does not operate service any longer
  - Key ownership has changed
- Full list in RFC 5280
- In these cases, there are two options: CRLs and OCSP

Certificate Revocation Lists (CRLs)

A CRL is a list of certificates that are considered revoked

- They are (should be) issued, updated and maintained by every CA
  - Certificates are identified by serial number
  - A reason for revocation can be given
  - Every CRL must be timestamped and signed
- There are further entries, like time of next update
- Technically, a browser (client) should download CRL (and update it after the given time), and lookup a host certificate every time it connects to a server

Problems with CRLs

CRLs have a number of problems

- Intermediate certs should be checked, too – induces load and network activity
- There is a time interval between two updates (window for attack)
- The update time is the same for all clients – peak loads on CAs self-induced
- CRLs can grow large (several mega-bytes) – unsuitable for checks during an SSL handshake
  - Response to this: Delta CRLs that contain only latest updates
  - Requires server side support – so far, very rarely used
- Downloads of CRLs can be blocked by a Man-in-the-middle
- For these reasons, browsers have never activated CRLs by default

Online Certificate Status Protocol (OCSP)

OCSP allows live revocation checks over the network

- Query-response model
- Query = lookup of a certificate in a server-side CRL-like data structure
  - Query by several hash values and cert's serial number
  - Replay protection with nonces
  - Query may be signed
  - Does not require encryption
- Response:
  - Contains cert status: `good`, `revoked`, `unknown`
  - Must be signed

Problems with OCSP

There are a number of issues with OCSP:

- Lookups go over the network – induces latency
- OCSP information must be fresh. Not just from CRLs.
- `unknown` is not clearly enough defined in standard: Is cert not known to the CA? Or is it just not in the CRL?
- Compare this to the model of credit-card authorisation: the only responses are `accepted` and `denied`
- OCSP servers must have high availability
- OCSP can be blocked by a Man-in-the-middle
- Privacy! OCSP servers know which sites users access
- Browsers ‘accept as good’ if no OCSP response received
- *“[OCSP was] designed as a fully bug-compatible stand-in for CRLs”* – P. Gutmann

OCSP Stapling

Addresses several problems of OCSP

- Problems addressed: latency of lookup, load on CA
- The idea is thus that servers request fresh OCSP ‘proof’ from CA: ‘this certificate is still considered valid’
- This can be done at regular intervals
- The ‘proof’ is ‘stapled’ to the certificate that the server sends in the SSL/TLS handshake
- Reduces load on CA
- Although around for a long time, the idea is only now gaining traction
- Solves privacy problem

Revocation: lessons learned

Revocation is crucial, but one Achilles heel of X.509 PKI

- It is probably safe to say that CRLs never worked, and are of very limited use
- OCSP checks are expensive, too (latency, load)
- OCSP stapling is an improvement
- There is an ongoing argument whether revocation (CRL, OCSP) is fatally flawed or not
- Revocation is not a solved problem

Part 2:
Recent results –
or: the sorry state of X.509

How This Got Our Interest (1)

PKI weaknesses in 2008

- Early December 2008:
  - ‘Error’ in Comodo CA: no identity check
  - Reported by Eddy Nigg of StartSSL (a CA)
  - A regional sub-seller just took the credit card number and gave you a certificate
  - No real reaction by Mozilla
- Late December 2008: whitehat hacks StartSSL CA
  - Technical report: simple flaw in Web front-end
  - Certificate for `mozilla.com` issued
  - Caught by 2nd line of defence: human checks for high-value domains

How This Got Our Interest (1)

PKI weaknesses in 2008

- Early December 2008:
  - ‘Error’ in Comodo CA: no identity check
  - Reported by Eddy Nigg of StartSSL (a CA)
  - A regional sub-seller just took the credit card number and gave you a certificate
  - No real reaction by Mozilla
- Late December 2008: whitehat hacks StartSSL CA
  - Technical report: simple flaw in Web front-end
  - Certificate for `mozilla.com` issued
  - Caught by 2nd line of defence: human checks for high-value domains

How This Got Our Interest (1)

PKI weaknesses in 2008

- Early December 2008:
  - ‘Error’ in Comodo CA: no identity check
  - Reported by Eddy Nigg of StartSSL (a CA)
  - A regional sub-seller just took the credit card number and gave you a certificate
  - No real reaction by Mozilla
- Late December 2008: whitehat hacks StartSSL CA
  - Technical report: simple flaw in Web front-end
  - Certificate for `mozilla.com` issued
  - Caught by 2nd line of defence:
    human checks for high-value domains

How This Got Our Interest (2)

PKI weaknesses in 2009

- February 2009
  - New ‘easy’ attack on MD5 (‘MD5 considered harmful today’)
  - Demonstrated by issuing valid but fake CA certificate
  - ‘Fast’ reaction by vendors: MD5 to be disabled for signatures by 2012
- Spring 2009
  - J. Nightingale of Mozilla writes crawler to traverse HTTPs sites
  - Goal: determine number of MD5-signed certificates (11%)
  - This piece of software was made public, it’s our starting point

How This Got Our Interest (2)

PKI weaknesses in 2009

- February 2009
  - New ‘easy’ attack on MD5 (‘MD5 considered harmful today’)
  - Demonstrated by issuing valid but fake CA certificate
  - ‘Fast’ reaction by vendors: MD5 to be disabled for signatures by 2012
- Spring 2009
  - J. Nightingale of Mozilla writes crawler to traverse HTTPs sites
  - Goal: determine number of MD5-signed certificates (11%)
  - This piece of software was made public, it’s our starting point

How This Got Our Interest (2)

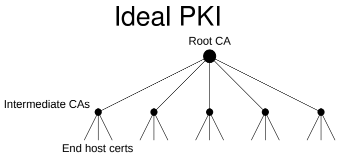
PKI weaknesses in 2009

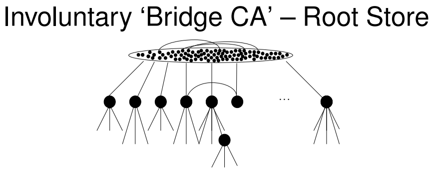
- February 2009
  - New ‘easy’ attack on MD5 (‘MD5 considered harmful today’)
  - Demonstrated by issuing valid but fake CA certificate
  - ‘Fast’ reaction by vendors: MD5 to be disabled for signatures by 2012
- Spring 2009
  - J. Nightingale of Mozilla writes crawler to traverse HTTPs sites
  - Goal: determine number of MD5-signed certificates (11%)
  - This piece of software was made public, it’s our starting point

How This Got Our Interest (3)

State of Mozilla Root Store

- Mozilla 2009: "Does anyone know who owns this root cert?"
- It turned out there were root certs that no-one could remember
- No-one could remember when they were accepted, or on which grounds





Kurt Seifried vs. RapidSSL

How to hijack a Web mailer in 3 easy steps

- Step 1: register e-mail address:
  `ssladministrator@portugalmail.pt`
- Step 2: ask RapidSSL for certificate for portugalmail.pt, giving this address as your contact
- Step 3: Watch 'Domain Validation by e-mail probe' fail

Kurt succeeded. It cost him $<$ 100 USD.

Main failure here:

- Web mailers and CAs have not agreed on 'protected' addresses
- This issue is now in Mozilla's 'Problematic practices'

How This Got Our Interest (4)

In 2011, the foundations of X.509 were rocked.

- March 2011: Comodo CA hacked (a sub-seller, again)
  - Attacker claims to come from Iran
  - $\approx$ 10 certificates for high-value domains issued
  - Browser reaction: blacklisting of those certificates *in code*
  - Neither CRLs nor OCSP trusted enough to work for victims
- July 2011: DigiNotar CA hacked
  - Attacker claims to be the same one as in March
  - 531 fake certificates, high-value domains
  - E.g., Google, Facebook, Mozilla, CIA, Mossad, Skype
  - Some hints pointed at Man-in-the-middle attack in Iran
  - The Netherlands' PKI was operated by DigiNotar...
  - For the first time, a Root CA is removed from a browser for being compromised

How This Got Our Interest (4)

In 2011, the foundations of X.509 were rocked.

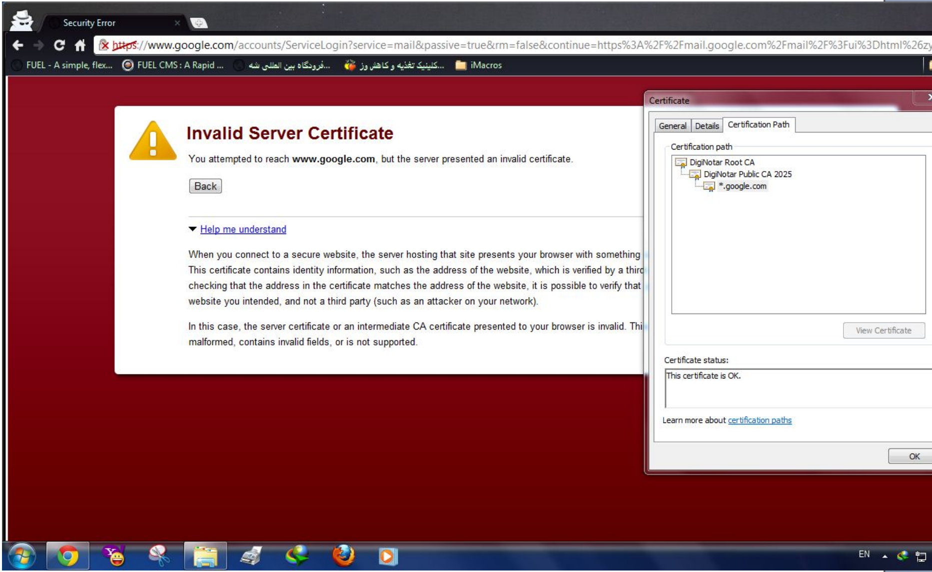
- March 2011: Comodo CA hacked (a sub-seller, again)
  - Attacker claims to come from Iran
  - $\approx$ 10 certificates for high-value domains issued
  - Browser reaction: blacklisting of those certificates *in code*
  - Neither CRLs nor OCSP trusted enough to work for victims
- July 2011: DigiNotar CA hacked
  - Attacker claims to be the same one as in March
  - 531 fake certificates, high-value domains
  - E.g., Google, Facebook, Mozilla, CIA, Mossad, Skype
  - Some hints pointed at Man-in-the-middle attack in Iran
  - The Netherlands' PKI was operated by DigiNotar...
  - For the first time, a Root CA is removed from a browser for being compromised

How This Got Our Interest (4)

In 2011, the foundations of X.509 were rocked.

- March 2011: Comodo CA hacked (a sub-seller, again)
  - Attacker claims to come from Iran
  - $\approx$ 10 certificates for high-value domains issued
  - Browser reaction: blacklisting of those certificates *in code*
  - Neither CRLs nor OCSP trusted enough to work for victims
- July 2011: DigiNotar CA hacked
  - Attacker claims to be the same one as in March
  - 531 fake certificates, high-value domains
  - E.g., Google, Facebook, Mozilla, CIA, Mossad, Skype
  - Some hints pointed at Man-in-the-middle attack in Iran
  - The Netherlands' PKI was operated by DigiNotar...
  - For the first time, a Root CA is removed from a browser for being compromised

DigiNotar vs. Iran?



Can We Assess the Quality of this PKI?

A good PKI should

- ... allow HTTPs on all WWW hosts
- ... contain only valid certificates
- ... offer good cryptographic security
  - Long keys, only strong hash algorithms, ...
- ... have a sensible setup
  - Short validity periods (1 year)
  - Short certificate chains (but use intermediate certificates)
  - Number of issuers should be reasonable (weakest link!)

Acquiring Our Data Sets

Active scans to measure *deployed* PKI

- Scan hosts on Alexa Top 1 million Web sites
- Nov 2009 – Apr 2011: scanned 8 times from Germany
- March 2011: scans from 8 hosts around the globe

Passive monitoring to measure *user-encountered* PKI

- Munich Research Network, monitored all SSL/TLS traffic
- Two 2-week runs in Sep 2010 and Apr 2011

EFF scan of IPv4 space in 2010

- Scan of 2-3 months, no *domain* information

In the meantime...

EFF scan presented at 27C3

- Focuses on CA certification structure
- Scan of IP addresses:
  does not allow to check match of host names
- No temporal distribution
- EFF project: SSL Observatory

Ivan Ristic of Qualys presents similar scan

- Smaller data basis
- Data set not published as raw data
- No temporal distribution
- Could not include it in our analysis

Our Data Sets

Active Scans — Passive Monitoring — EFF IPv4 scan

Location	*Time (run)*	*Type*	*Certificates*
Tuebingen, DE	November 2009	Active scan	833,661
Tuebingen, DE	December 2009	Active scan	819,488
Tuebingen, DE	January 2010	Active scan	816,517
Tuebingen, DE	April 2010	Active scan	816,605
Munich, DE	September 2010	Active scan	829,232
Munich, DE	November 2010	Active scan	827,366
Munich, DE	April 2011	Active scan	829,707
Munich, DE	April 2011	Active scan with SNI	826,098
Shanghai, CN	April 2011	Active scan	798,976
Beijing, CN	April 2011	Active scan	797,046
Melbourne, AU	April 2011	Active scan	833,571
İzmir, TR	April 2011	Active scan	825,555
São Paulo, BR	April 2011	Active scan	833,246
Moscow, RU	April 2011	Active scan	830,765
Santa Barbara, US	April 2011	Active scan	834,173
Boston, US	April 2011	Active scan	834,054
Munich, DE	September 2010	Passive monitoring	183,208
Munich, DE	April 2011	Passive monitoring	989,040
EFF servers	March–June 2010	Active IPv4 scan	11,349,678

25 million certificates to evaluate.

Our Data Sets

Active Scans — Passive Monitoring — EFF IPv4 scan

Location	*Time (run)*	*Type*	*Certificates*
Tuebingen, DE	November 2009	Active scan	833,661
Tuebingen, DE	December 2009	Active scan	819,488
Tuebingen, DE	January 2010	Active scan	816,517
Tuebingen, DE	April 2010	Active scan	816,605
Munich, DE	September 2010	Active scan	829,232
Munich, DE	November 2010	Active scan	827,366
Munich, DE	April 2011	Active scan	829,707
Munich, DE	April 2011	Active scan with SNI	826,098
Shanghai, CN	April 2011	Active scan	798,976
Beijing, CN	April 2011	Active scan	797,046
Melbourne, AU	April 2011	Active scan	833,571
İzmir, TR	April 2011	Active scan	825,555
São Paulo, BR	April 2011	Active scan	833,246
Moscow, RU	April 2011	Active scan	830,765
Santa Barbara, US	April 2011	Active scan	834,173
Boston, US	April 2011	Active scan	834,054
Munich, DE	September 2010	Passive monitoring	183,208
Munich, DE	April 2011	Passive monitoring	989,040
EFF servers	March–June 2010	Active IPv4 scan	11,349,678

25 million certificates to evaluate.

Our Data Sets

Active Scans — Passive Monitoring — EFF IPv4 scan

Location	*Time (run)*	*Type*	*Certificates*
Tuebingen, DE	November 2009	Active scan	833,661
Tuebingen, DE	December 2009	Active scan	819,488
Tuebingen, DE	January 2010	Active scan	816,517
Tuebingen, DE	April 2010	Active scan	816,605
Munich, DE	September 2010	Active scan	829,232
Munich, DE	November 2010	Active scan	827,366
Munich, DE	April 2011	Active scan	829,707
Munich, DE	April 2011	Active scan with SNI	826,098
Shanghai, CN	April 2011	Active scan	798,976
Beijing, CN	April 2011	Active scan	797,046
Melbourne, AU	April 2011	Active scan	833,571
İzmir, TR	April 2011	Active scan	825,555
São Paulo, BR	April 2011	Active scan	833,246
Moscow, RU	April 2011	Active scan	830,765
Santa Barbara, US	April 2011	Active scan	834,173
Boston, US	April 2011	Active scan	834,054
Munich, DE	September 2010	Passive monitoring	183,208
Munich, DE	April 2011	Passive monitoring	989,040
EFF servers	March–June 2010	Active IPv4 scan	11,349,678

25 million certificates to evaluate.

Our Data Sets

Active Scans — Passive Monitoring — EFF IPv4 scan

Location	*Time (run)*	*Type*	*Certificates*
Tuebingen, DE	November 2009	Active scan	833,661
Tuebingen, DE	December 2009	Active scan	819,488
Tuebingen, DE	January 2010	Active scan	816,517
Tuebingen, DE	April 2010	Active scan	816,605
Munich, DE	September 2010	Active scan	829,232
Munich, DE	November 2010	Active scan	827,366
Munich, DE	April 2011	Active scan	829,707
Munich, DE	April 2011	Active scan with SNI	826,098
Shanghai, CN	April 2011	Active scan	798,976
Beijing, CN	April 2011	Active scan	797,046
Melbourne, AU	April 2011	Active scan	833,571
İzmir, TR	April 2011	Active scan	825,555
São Paulo, BR	April 2011	Active scan	833,246
Moscow, RU	April 2011	Active scan	830,765
Santa Barbara, US	April 2011	Active scan	834,173
Boston, US	April 2011	Active scan	834,054
Munich, DE	September 2010	Passive monitoring	183,208
Munich, DE	April 2011	Passive monitoring	989,040
EFF servers	March–June 2010	Active IPv4 scan	11,349,678

25 million certificates to evaluate.

Our Data Sets

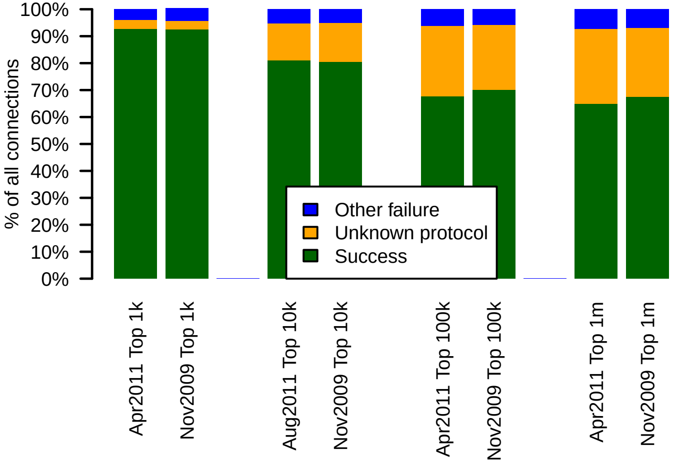
Active Scans — Passive Monitoring — EFF IPv4 scan

Location	*Time (run)*	*Type*	*Certificates*
Tuebingen, DE	November 2009	Active scan	833,661
Tuebingen, DE	December 2009	Active scan	819,488
Tuebingen, DE	January 2010	Active scan	816,517
Tuebingen, DE	April 2010	Active scan	816,605
Munich, DE	September 2010	Active scan	829,232
Munich, DE	November 2010	Active scan	827,366
Munich, DE	April 2011	Active scan	829,707
Munich, DE	April 2011	Active scan with SNI	826,098
Shanghai, CN	April 2011	Active scan	798,976
Beijing, CN	April 2011	Active scan	797,046
Melbourne, AU	April 2011	Active scan	833,571
İzmir, TR	April 2011	Active scan	825,555
São Paulo, BR	April 2011	Active scan	833,246
Moscow, RU	April 2011	Active scan	830,765
Santa Barbara, US	April 2011	Active scan	834,173
Boston, US	April 2011	Active scan	834,054
Munich, DE	September 2010	Passive monitoring	183,208
Munich, DE	April 2011	Passive monitoring	989,040
EFF servers	March–June 2010	Active IPv4 scan	11,349,678

25 million certificates to evaluate.

Errors in TLS Connection Setup

Scans from Germany, Nov 2009 and Apr 2011

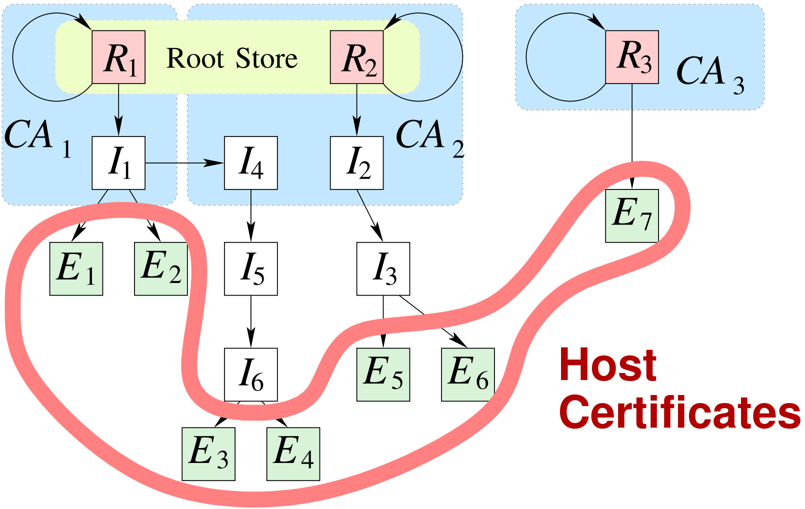


Errors in TLS Connection Setup

`UNKNOWN PROTOCOL`

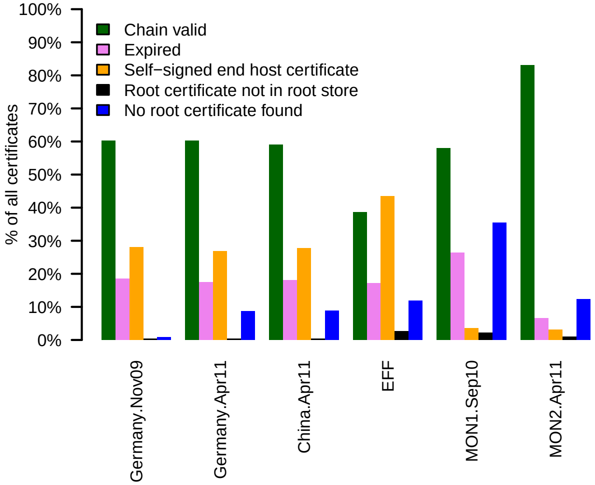
- Rescanned those hosts and manual sampling
- Always plain HTTP...
- ... and always an `index.html` with HTML 2 ...
- Hypothesis: old servers, old configurations
- More likely to happen in the lower ranks

Validity of End-Hosts Certificates



Validation of Certificate Chains

Just check chains, not host names



Correct Domain Name in Certificate

Now also check host names

- Look in Common Name (CN) and Subject Alternative Name (SAN)
- Munich, April 2011, only valid chains:
  - 12.2% correct CN
  - 5.9% correct SAN

Only 18% of certificates are fully verifiable

- Positive ‘trend’: from 14.9% in 2009 to 18% in 2011

Unusual Host Names

CN=plesk or similar

- Found in 7.3% of certificates
- Verified: Plesk/Parallels panels

CN=localhost

- 4.7% of certificates
- Very common: redirection to HTTP after HTTPs

Host Names in Self-signed Certificates

Self-signed means:

- Issuer the same as subject of certificate
- Requires out-of-band distribution of certificate

Active scan

- **2.2%** correct Common Name (CN)
- **0.5%** correct Subject Alternative Name

Top 3 most frequent CNs account for $>$ 50%

- `plesk` or similar in 27.3%
- `localhost` or similar in 25.4% – standard installations?

Certificate Occurrences

Many certificates valid for more than one domain

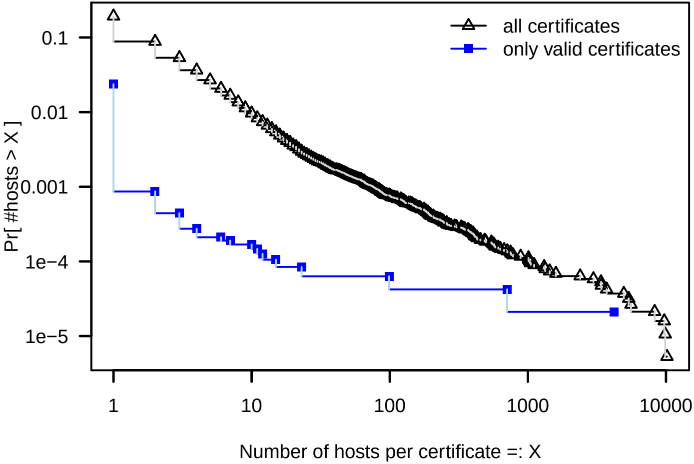
- Domains served by same IP
- Some certificates issued for dozens of domains
- Certificate reuse on multiple machines increases options for attacker

Often found on hosters

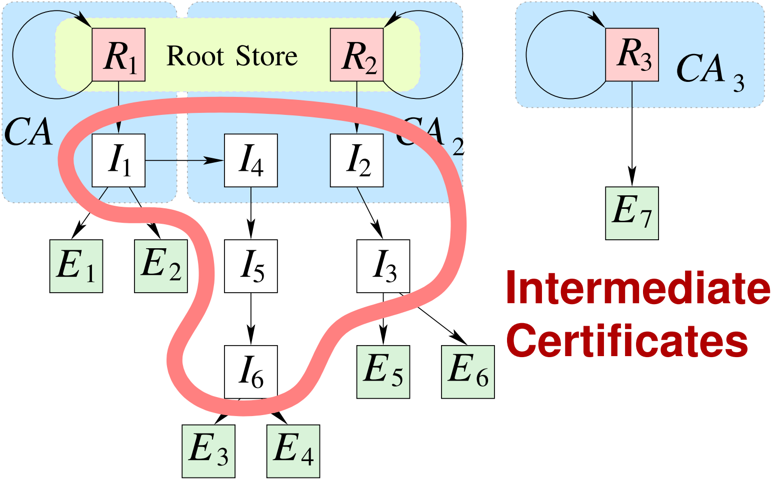
- E. g. `*.blogger.com`, `*.wordpress.com`

Certificate Occurrences

How often does a certificate occur on X hosts?

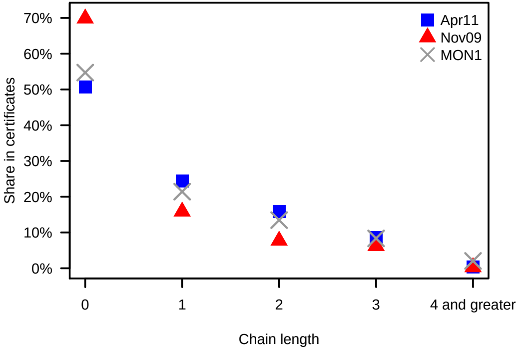


Certificate Chains



Intermediate Certificates

Certificate Chain Lengths

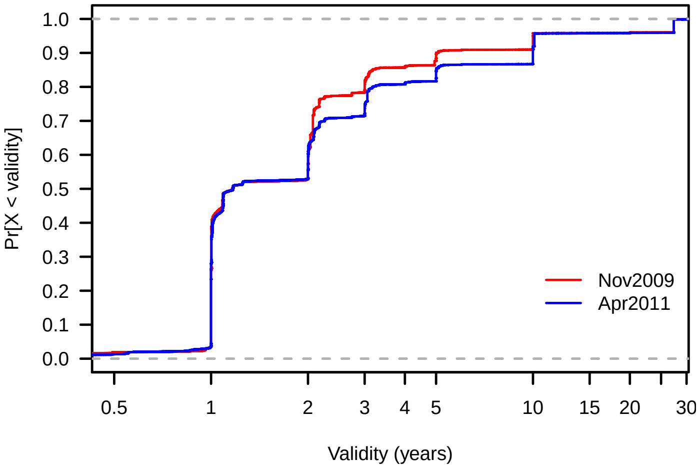


Finding more positive than negative:

- Trend to use intermediate certificates more often
- Allows to keep Root Certificates offline
- But chains still reasonably short

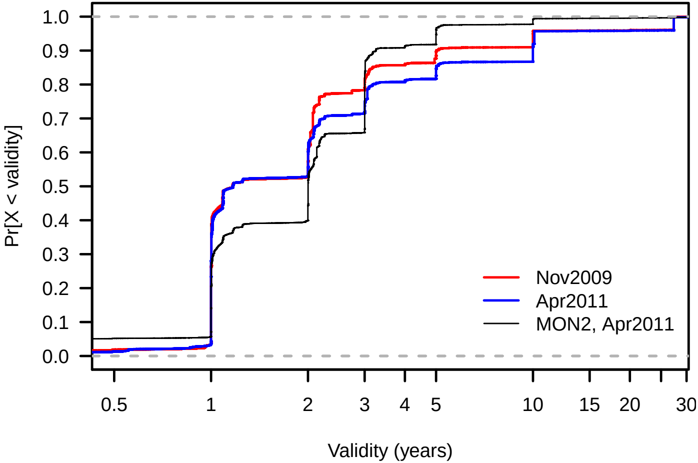
Validity Periods

CDF of validity periods, active scans



Validity Periods

CDF of validity periods, scans and monitoring



Public Key Properties

Key types

- RSA: 99.98% (rest is DSA)
- About 50% have length 1,024 bit
- About 45% have length 2,048 bit
- Clear trend from 1,024 to 2,048 bit

Weird encounters

- 1,504 distinct certificates that share another certificate's key
- Many traced to a handful of hosting companies
- Nadiah Henninger's work: Embedded devices, poor entropy!
- `www.factorable.net`

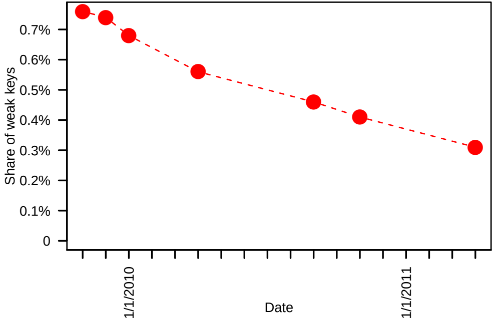
Debian Weak Keys (1)

Bug of 2008

- Generation of random numbers weak (bad initialisation)
- Only 2^{16} public/private key-pairs generated
- Allows pre-computation of private keys
- Debian ships blacklist of keys

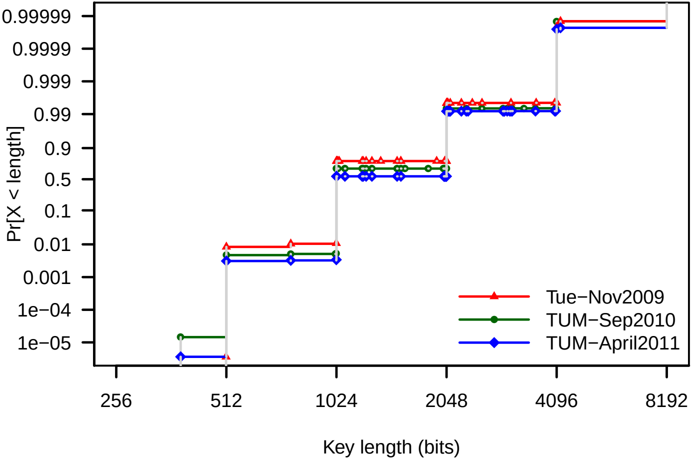
Debian Weak Keys (2)

Weak randomness in key generation – serious bug of 2008



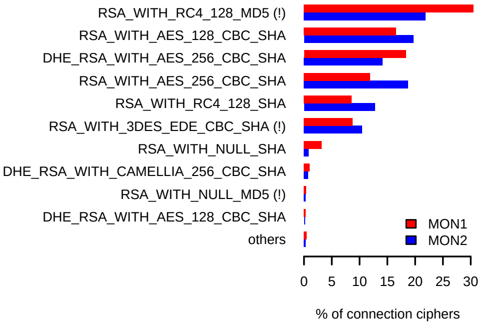
Public Key Lengths

CDF for RSA key lengths – double-log Y axis



Symmetric Ciphers

Results from monitoring

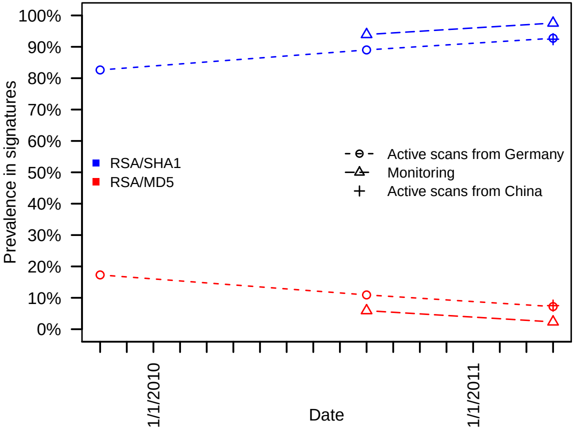


(Mostly) in line with results from 2007 by Lee et al.

- Order of AES and RC4 has shifted, RC4-128 most popular

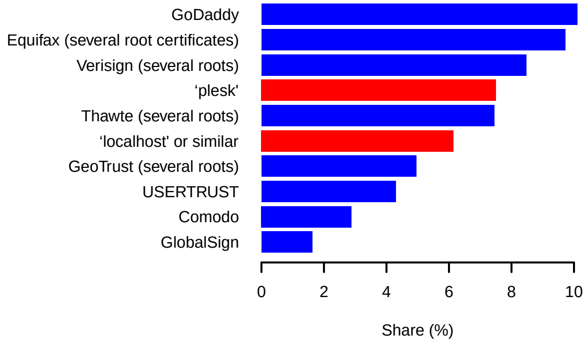
Signature Algorithms

MD5 is being phased out



Certificate Issuers

Very few CAs account for $>$ 50% of certificates



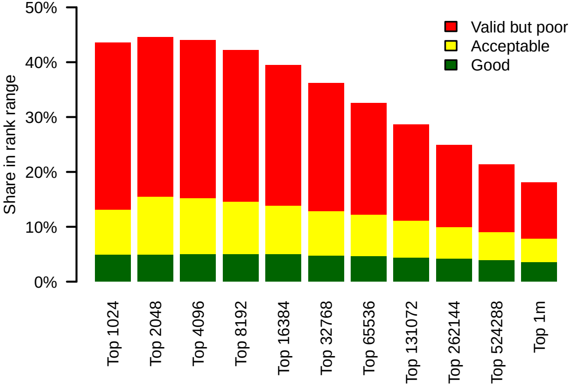
But there are 150+ Root Certificates in Mozilla.

Certificate Quality

We defined 3 categories

- 'Good':
  - Correct chains, correct host name
  - Chain ≤ 2
  - No MD5, strong key of > 1024 bit
  - Validity ≤ 13 months
- 'Acceptable'
  - Chain ≤ 3, validity ≤ 25 months
  - Rest as above
- 'Poor': the remainder

Certificate Quality



Validity correlates with rank

- Share of ‘poor’ certificates higher among high-ranking sites

Conclusion

In great part, the X.509 PKI is in a sorry state

- Only 18% of the Top 1 Million Web sites show fully valid certificates
- Invalid chains
  - Expired certificates are common
  - Often no recognisable Root Certificate
  - Lack of correct domain information information
- Frequent sharing of certificates between hosts is problematic
- Much carelessness

Conclusion

Certification practices are very poor. But crypto OK.

Some positive developments

- Very slight trend for fully valid certificates
- Chains short, intermediate certificates used
- Key lengths OK
- Weak MD5 algorithm is being phased out

Part 3:
Proposals to enhance or replace X.509

Can X.509 be reinforced? Or replaced?

No 'silver bullet' known that would resolve all issues

- Attacker model of SSL/TLS + X.509 $\approx$ protect credit card numbers
- State-scale attacks were not in scope back in the 1990s

Several recent proposals:

- Hardening certification
- Pinning Information
- Use of DNSSEC
- Notary Principle
- Public Logs

A word of warning

All of these concepts are very recent

- Very few have passed IETF and are RFCs
  - E.g. DNS-based authentication of names entities (DANE), RFC 6698
- Others may yet enter an IETF track:
  - Certificate Transparency: BoF
  - TACK is written up in form of an RFC
- Many are still incomplete

But the *underlying ideas* are *very relevant*.

Hardening certification

Extended Validation (EV)

- Already deployed
- CAs require state-issued documents before certification
- Certificates carry special OID that browsers evaluate to show the ‘green bar’
- More expensive, rarely bought by customers

Base Line Requirements

- CA/Browser forum standard
- Absolute minimum requirements for validation
- Audit-based, rules for audits

Pinning

Concept:

- On connecting to a host via SSL/TLS, the client stores one or more identifying values:
  - Hash value of certificate (‘Certificate pinning’)
  - Hash of public key of host (‘Key pinning’, more flexible)
  - Hash of cert of used CA (‘CA pinning’)
  - ...or a hash of the CA’s public key
- Upon reconnect to host: verify that identifier is still the same
- Warn on change

Pinning

Advantages

- Raises barriers for attackers
- Practical usefulness demonstrated in DigiNotar incident

Issues

- No defence when client makes first contact to host
- False alarms may occur:
  - Legitimate changes to certificates (and public keys) not detected
  - Some sites use several certificate chains (Citibank, Facebook...)
  - Some sites exchange their certificates frequently (Google)

Pinning variants (examples)

Shipped with client

- Google Chrome has pinned several sites of high relevance (Google, Gmail, Tor, $\cdots$)
- Browser's auto-update mechanism might be useful here?

Trust Assertions for Certificate Keys

- RFC draft by Moxie Marlinspike, Trevor Perrin
- Idea: servers have TACK key, sign their certificates with it
- Clients are meant to pin to the TACK key
- Introduces some flexibility to pinning – work-in-progress

DANE: DNS-based authentication of named entities

DNS is a distributed global database containing records about hosts

- DNSSEC is a technology to integrity-protect and origin-authenticate DNS queries/responses
- DNSSEC is a hierarchical PKI with records under control of DNS registries (TLD)
- We will discuss DNSSEC later in the lecture
- Verification from root zone down to leaf zones

DANE adds support for new DNS record:

- TLSA record to store full certificate information or a digest ('Subject Public Key Info')
- TLSA records can store information about end-host cert, intermediate cert, CA cert, etc.

DANE is not without critiques

Positive comments

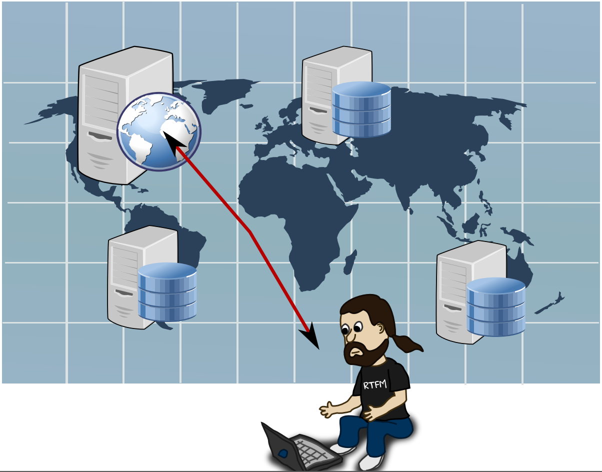
- A strong reassurance of certificate validity on a second channel

Negative comments

- DNS operators need to become PKI operators – same level of assurance like CA checks?
- Possible caching and performance issues due to DNSSEC?
- Countries are often in control of their TLDs – think of `bit.ly`
- This makes DANE susceptible to some forms of state-level attacks
- Countries like the USA have unproportionate influence on DNS governance

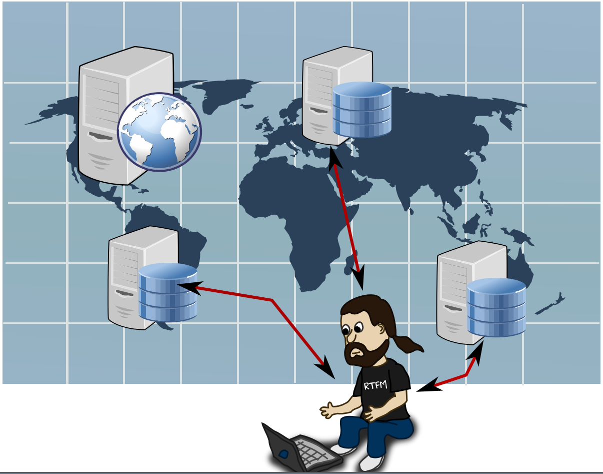
Notary-based systems

When connecting to a host and receiving the TLS certificate...



Notary-based systems

... connect to some special notaries elsewhere and double-check



Notary-based systems

Examples

- Perspectives (Carnegie-Mellon, 2009): browser plug-in
- Convergence (Moxie Marlinspike, 2011): browser plug-in
- The above are not very different; Convergence is more mature
- Crossbear (ourselves :-), 2011): attempts to locate and report Man-in-the-middle

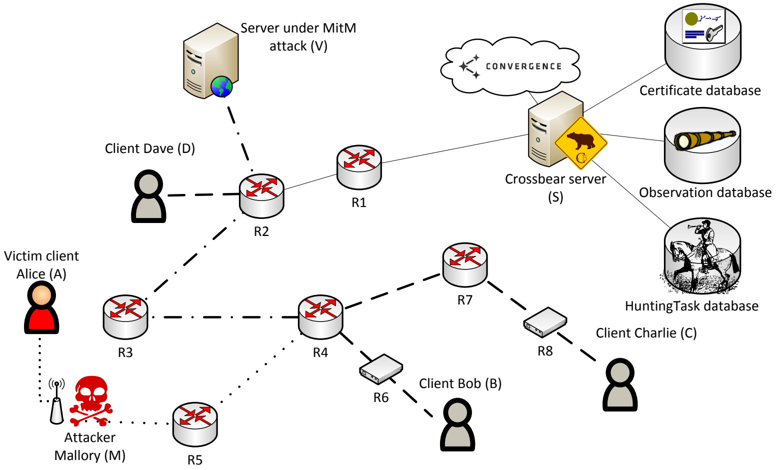
Notary-based systems

Discussion

- Detection works well as long as the attacker does not control all paths from notaries to server
- Attacker can drop traffic to notaries $\rightarrow$ detectable
- Privacy: notaries know where users surf $\rightarrow$ Convergence uses a simple form of onion-routing
- False positives may occur

Crossbear

The goal is *detection and localisation*, not user-friendliness



Public Log Schemes

Public Log: store information publicly and append-only

- Sovereign Keys
  - Sites use authoritative key to cross-sign their certificates
  - This key is then published in a public log
  - Result: cross-certification of keys
- Certificate Transparency
  - Store info about who is certified by whom in the Public Log
  - Goal: detect rogue CA issuing key for a site
  - Result: detect rogue CAs, get assurance abouyt key

Schemes are very new - end of 2011

Sovereign Keys (EFF)

Sites store information on $<$ 30 timeline servers

timestamp	name	key	protocols	evidence
`1322736203`	`A`	`0x427E8A`	`https, smtps`	$Sig_{CA}(A, \cdots)$
`1323254603`	`B`	`0x7389FB`	`https:8080`	$Sig_B(B, \cdots)$
`1323657143`	`C`	`0x49212A`	`imaps`	$Sig_C(C, \cdots)$
`1413787143`	`A`	`0x427E8A`	`https, smtps`	$Sig_{CA}(A, \cdots)$
...	...	...	...	...

Work-in-progress (alive)

- Timeline is auditable by clients
- Mirrors proposed
- `https://www.eff.org/sovereign-keys`

Sovereign Keys: Discussion

Advantages

- Does not need CA support
- Evidence can be based on DANE DNSSEC, CAs, ...
- Performance and bandwidth?

Issues

- Continous monitoring of timelines needed
- Entries are not space-efficient (linear in number of certs)
- Privacy (suggested remedy: TOR-like proxying)
- Loss of sovereign key can lead to loss of domain

Certificate Transparency (Google)

Store proof of certification in Public Log

timestamp	name	cert	evidence
1322736203	A	Cert chain by Verisign	$MSig(hashes)$
1323254603	B	Self-signed cert	$MSig(hashes)$
1323657143	C	Cert by CACert	$MSig(hashes)$
...	...	...	$MSig(hashes)$

Work-in-progress (alive)

- Timeline consistency can be monitored
- Roles: clients, auditors, monitors (on-behalf)

Certificate Transparency (Google)

Proof that a given cert is in log can be generated

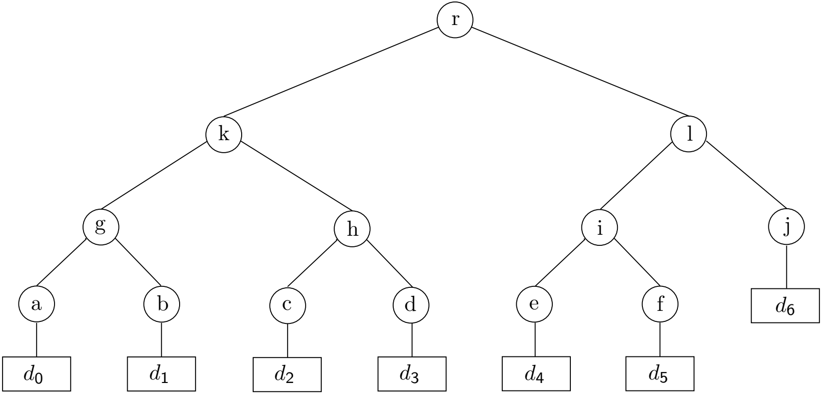


Figure : Log is a Merkle tree, d_i are new certificate chains.

Discussion of Certificate Transparency

Advantages

- Protects against rogue CAs
- Efficient data structure: proofs are in $O(\log n)$

Issues

- Requires continous monitoring of logs
- Monitors need full log at all times, act on behalf of others
- Proofs are in $O(\log n)$, but storage is linear

Attempt: summary of proposals

There is no candidate that solves all issues

- All proposals must gain vendor support
- DANE has done so, Certificate Transparency stands a chance
- Convergence, TACK, Sovereign Keys:
  - Different concepts, but allow to abolish the X.509 PKI altogether
  - Come with new drawbacks and have so far gained little support
- Pinning works well, but does not scale
- It seems that, in short- and mid-term, we have to live with band-aids rather than comprehensive solutions