



CSC574 - Computer and Network Security

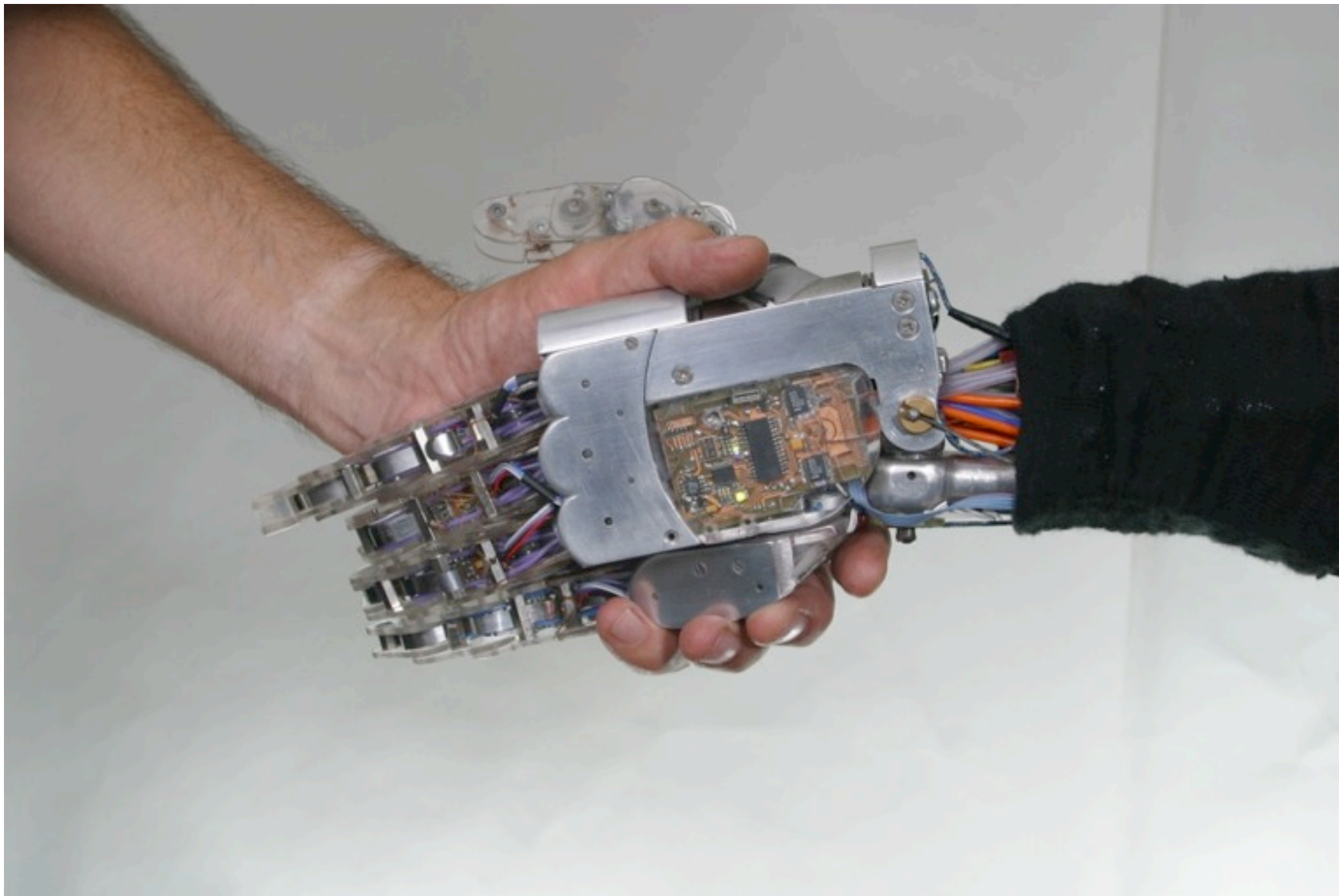
Module: Public Key Infrastructure

Prof. William Enck
Spring 2013

Meeting Someone New

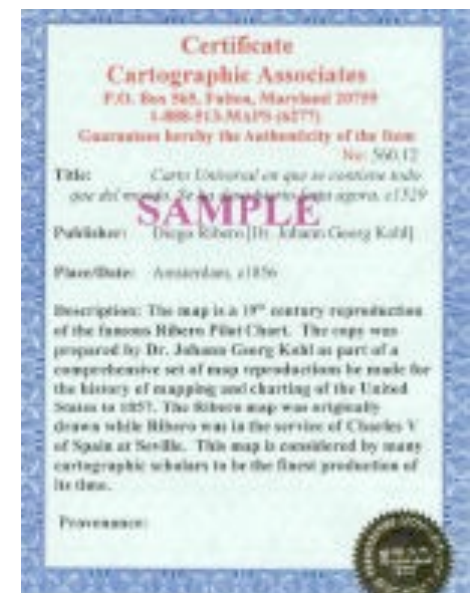


- Anywhere in the Internet



What is a certificate?

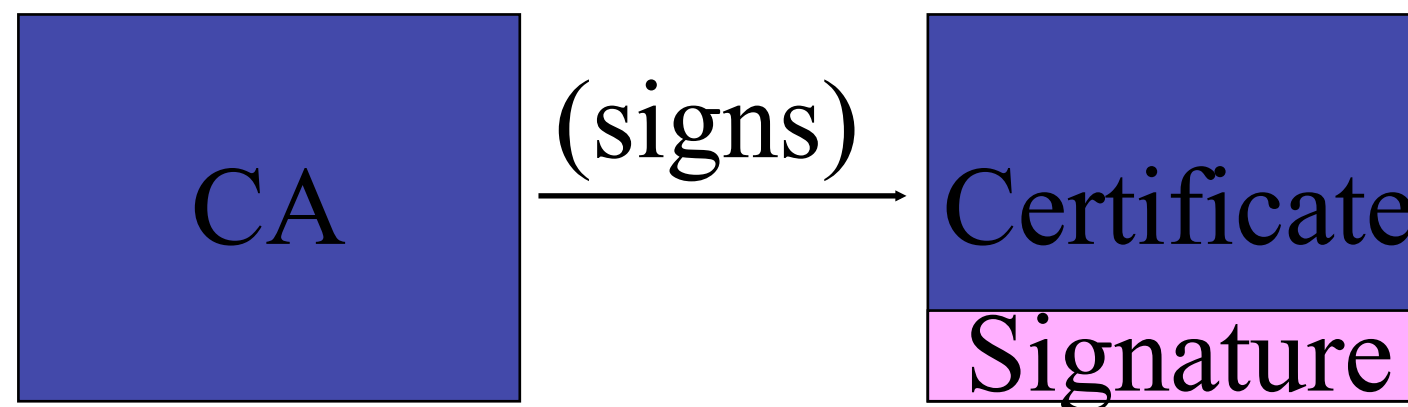
- A certificate ...
 - ▶ ... makes an association between a user identity/job/attribute and a private key
 - ▶ ... contains public key information $\{e,n\}$
 - ▶ ... has a validity period
 - ▶ ... is signed by some *certificate authority* (CA)
 - ▶ ... identity may have been vetted by a *registration authority* (RA)
- Issued by CA for some purpose
 - ▶ Verisign is in the business of issuing certificates
 - ▶ People trust Verisign to vet identity



Why do I trust the certificate?



- A collections of “root” CA certificates
 - ▶ ... baked into your browser
 - ▶ ... vetted by the browser manufacturer
 - ▶ ... supposedly closely guarded (yeah, right)
- Root certificates used to validate certificate
 - ▶ Vouches for certificate’s authenticity



Public Key Infrastructure

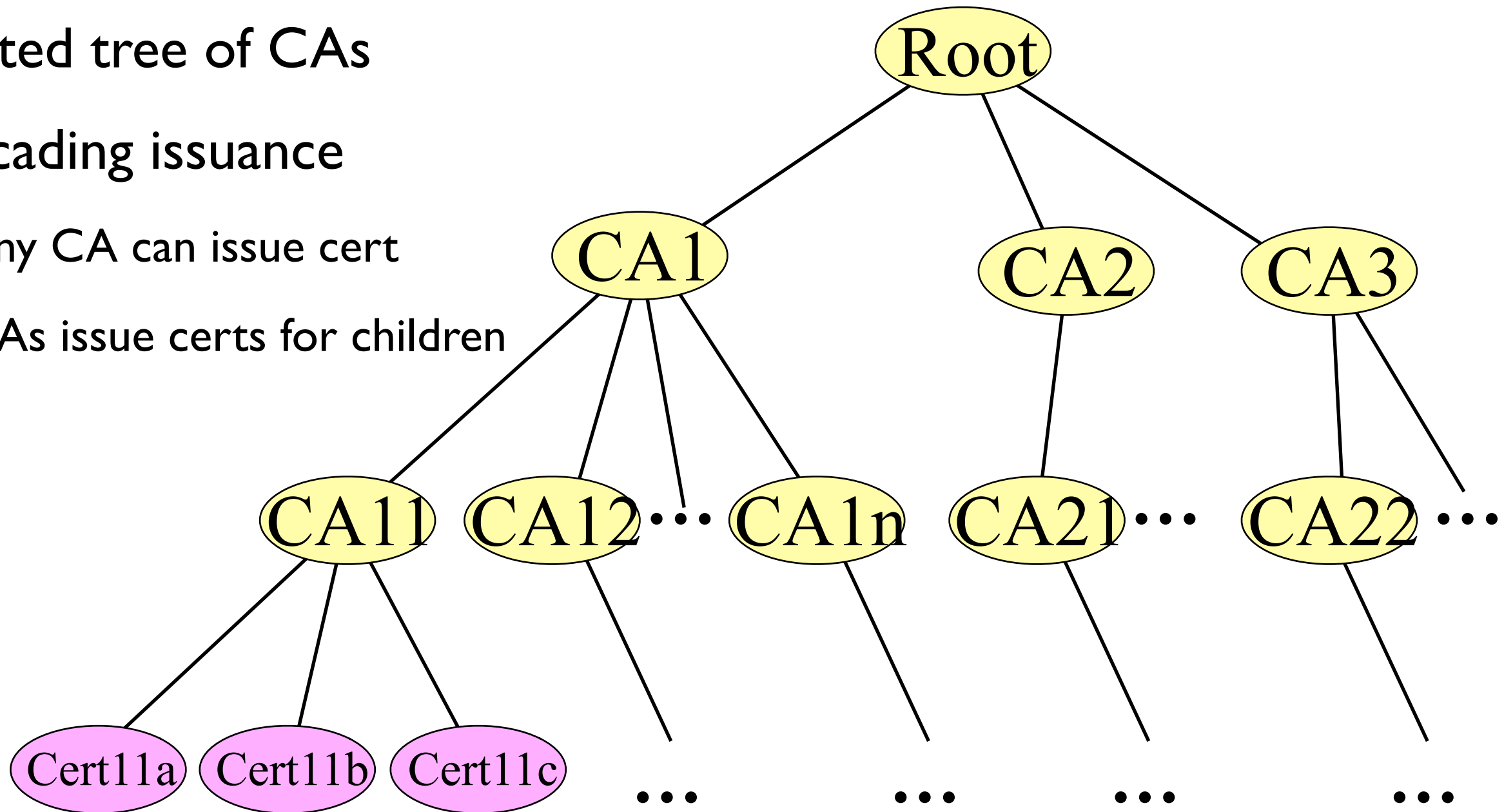


- System to “*securely distribute public keys (certificates)*”
 - Q: Why is that hard?
- Terminology:
 - Alice signs a certificate for Bob’s name and key
 - Alice is **issuer**, and Bob is **subject**
 - Alice wants to find a path to Bob’s key
 - Alice is **verifier**, and Bob is **target**
 - Anything that has a public key is a **principal**
 - Anything trusted to sign certificates is a **trust anchor**
 - Its certificate is a **root certificate**

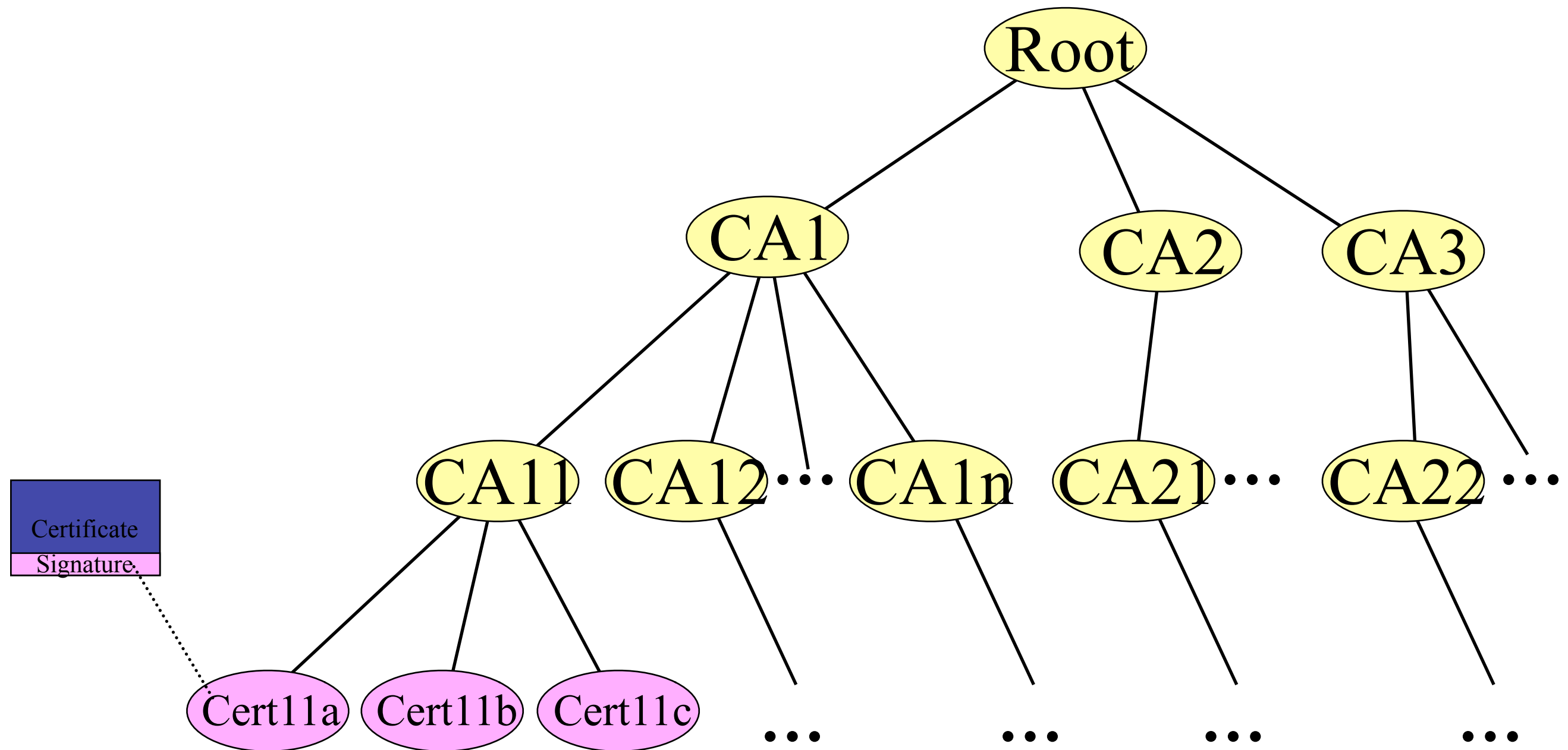
What is a PKI?



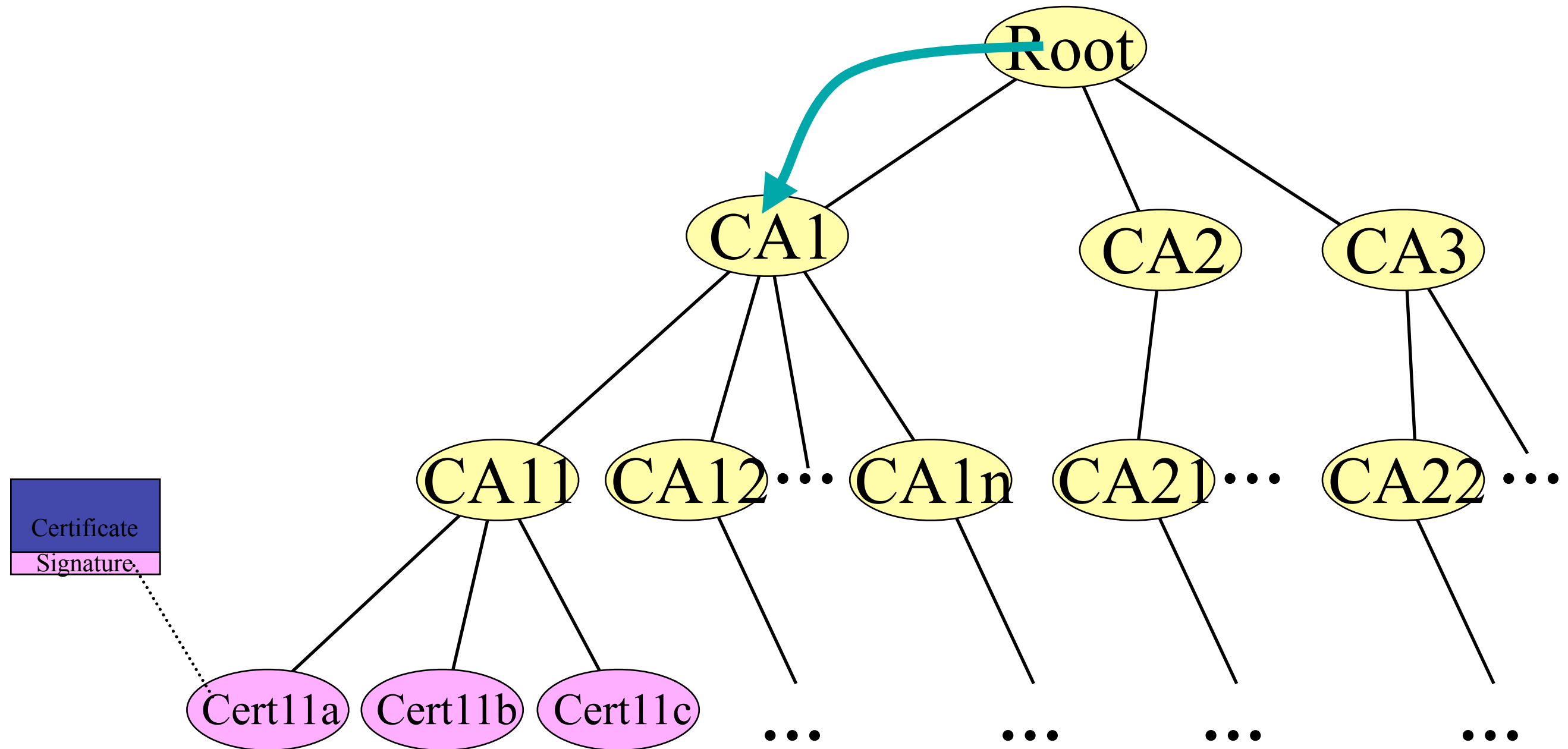
- Rooted tree of CAs
- Cascading issuance
 - Any CA can issue cert
 - CAs issue certs for children



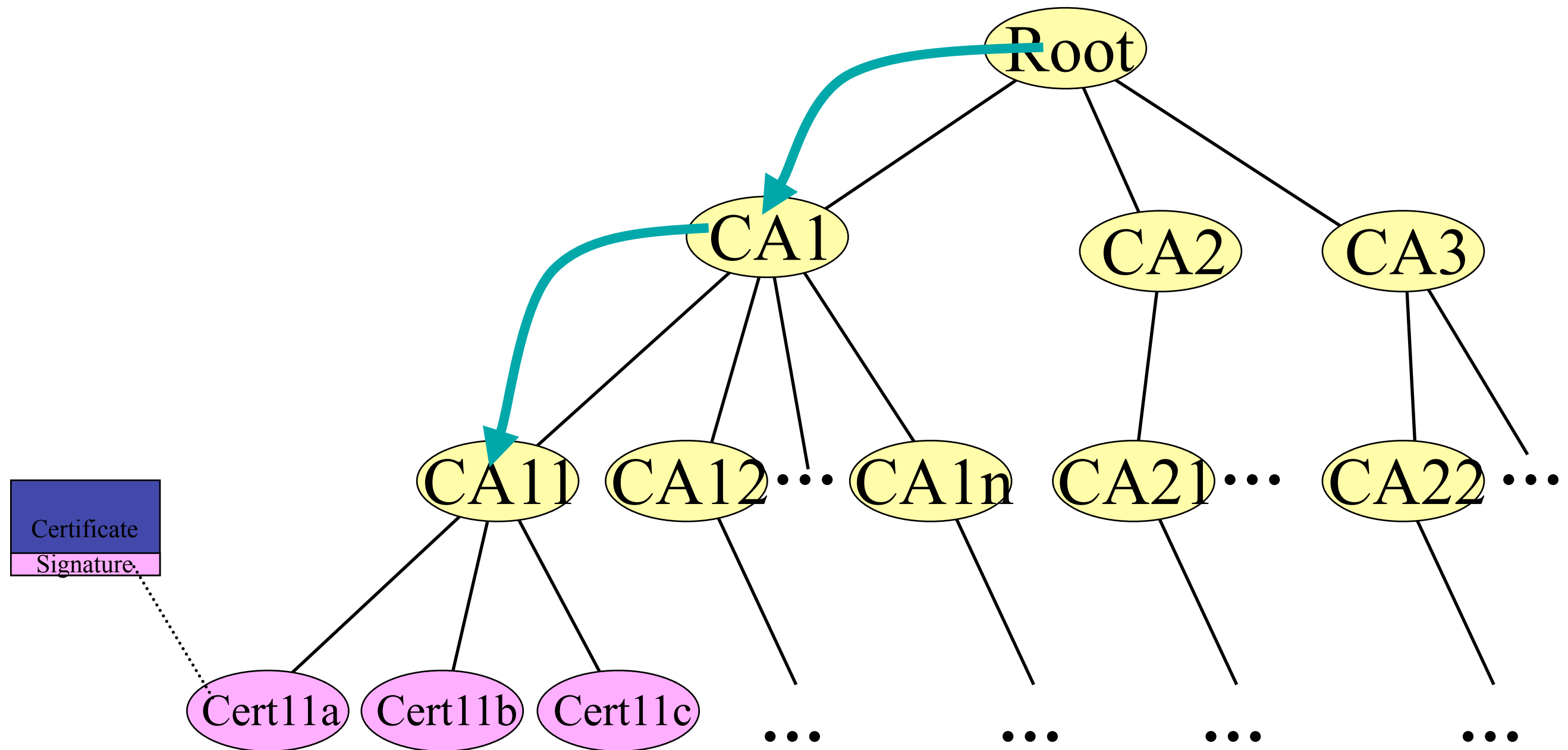
Certificate Validation



Certificate Validation



Certificate Validation



PKI and Revocation

- Certificate may be revoked before expiration
 - ▶ Lost private key
 - ▶ Compromised
 - ▶ Owner no longer authorized
- Revocation is hard ...
 - ▶ The “anti-matter” problem
 - ▶ Verifiers need to check revocation state
 - Loses the advantage of off-line verification
 - ▶ Revocation state must be authenticated



Certificate Revocation List (CRL)

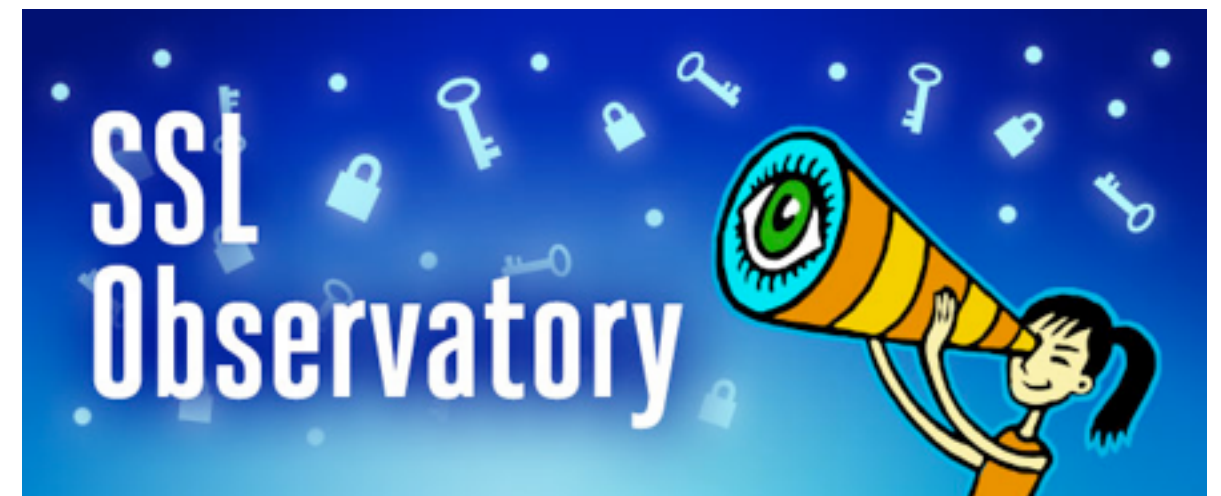


- A Certificate Revocation List (CRL) contains a list of revoked certificates.
 - ▶ Why are CRLs issued periodically even if no certificates are revoked?
 - ▶ How frequent should CRLs be issued?
 - ▶ If a CRL is maintained, why associate an expiration time with certificates?
- Delta CRL: includes list of changes from last complete CRL
 - ▶ Delta CRL issued more frequently than full CRL
- In reality, browsers don't make use of CRLs.

EFF SSL Observatory



- Project started to understand who is using SSL certificates and how CA's are being used
- Scanned HTTPS for entire IPv4 space
- Results are available for download
- <https://www.eff.org/observatory>

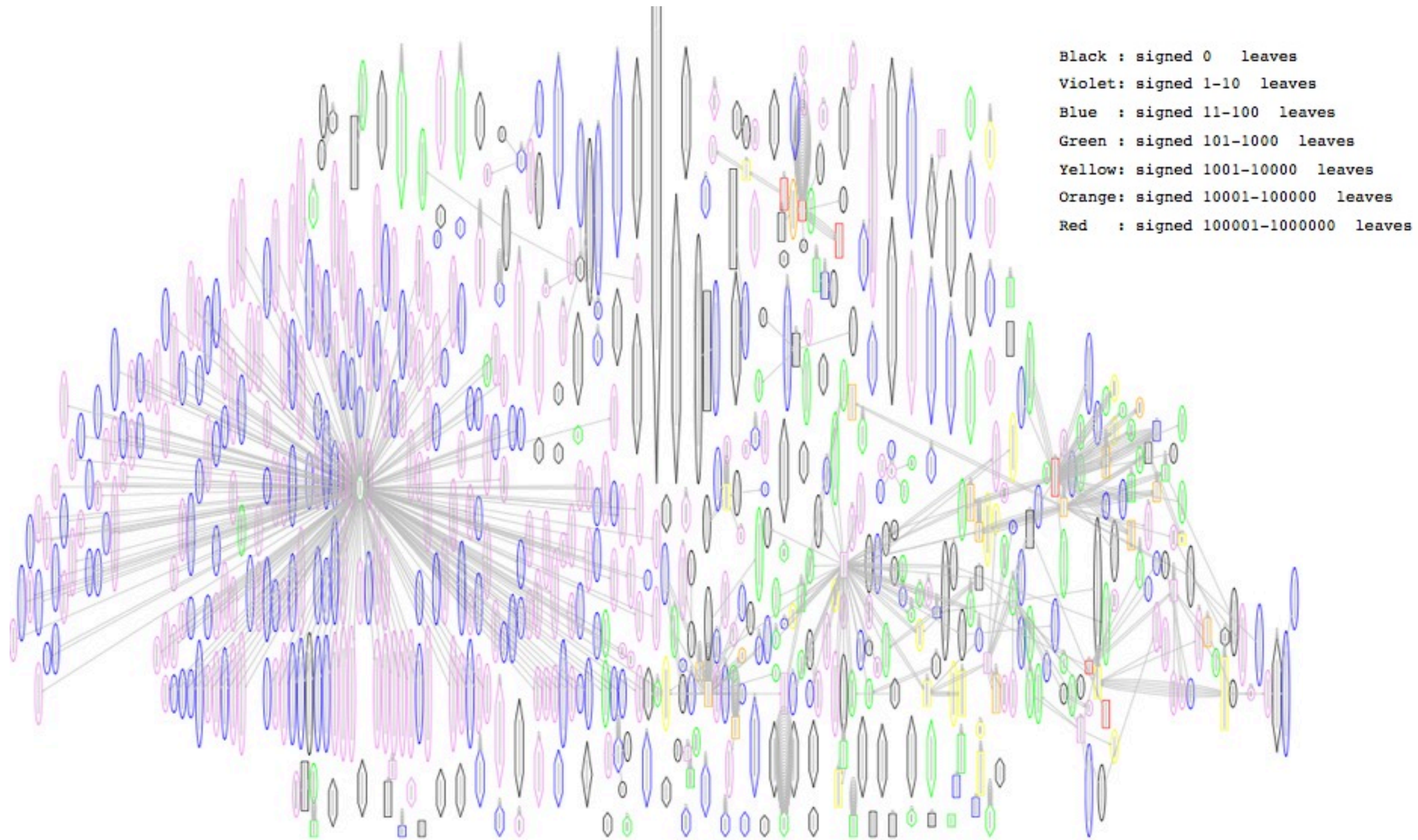


Number of Trusted CAs



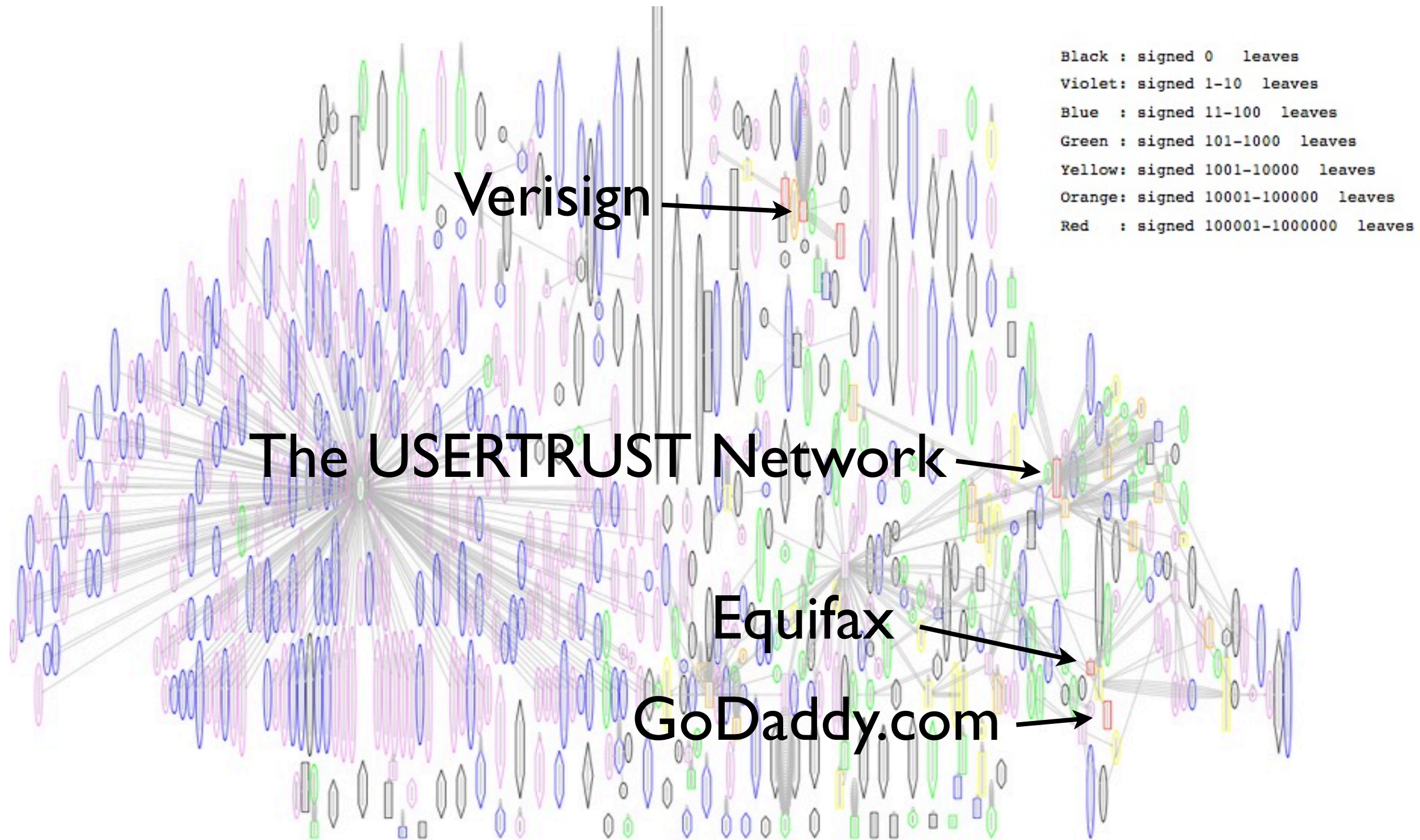
- The SSL Observatory project 2010 study looked at trusted signers
- In the browser:
 - ▶ Mozilla: 124 trust roots (~60 orgs)
 - ▶ Microsoft: 19 trust roots (Win7), but on-demand updating makes it 300+ (100+ orgs)
- Chained trusted signers:
 - ▶ 1,482 CA certificates trustable by Windows or Firefox (651 orgs)

PKI (Circa 2010)



http://www.eff.org/files/colour_map_of_CAs.pdf

PKI (Circa 2010)



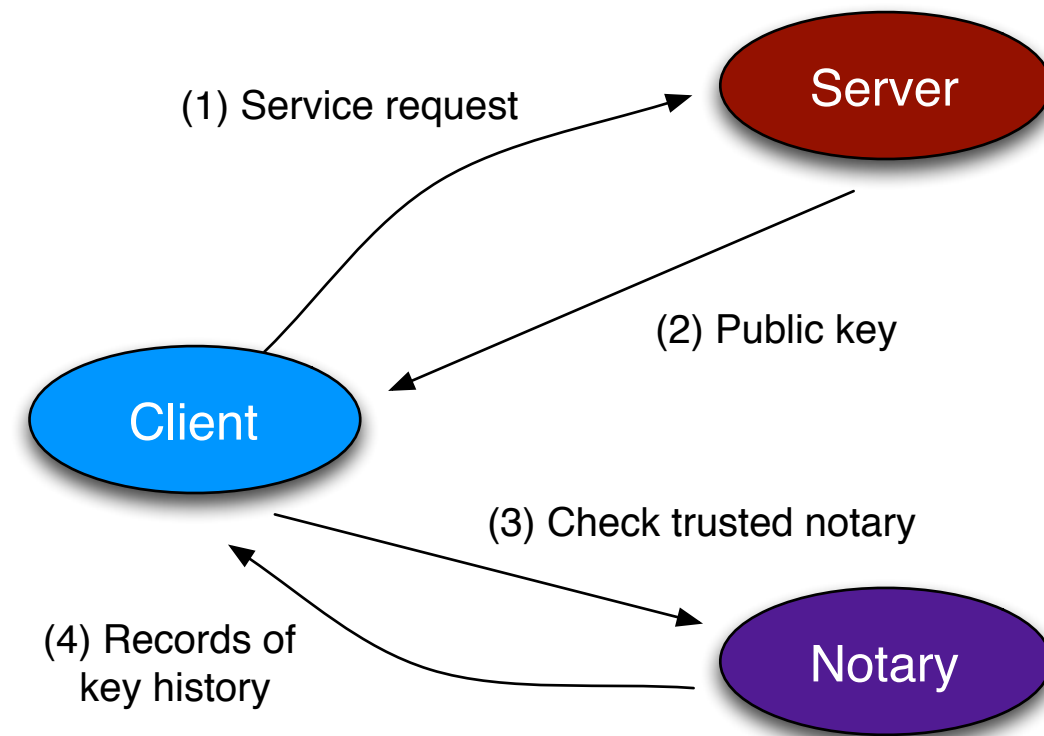
http://www.eff.org/files/colour_map_of_CAs.pdf

Compromised CAs



- When a trusted CA is compromised in some way, it affects all domains
 - ▶ CAs are not restricted in what domains they sign
- Compromised CAs are now a legitimate threat
 - ▶ 2001: VeriSign issued two certs claiming to be “Microsoft Corporation”
 - ▶ 2011: fraudulent certs obtained from Comodo and DigiNotar (allegedly by Iranian hackers)
 - ▶ 2012: Trustwave issued subordinate root cert that was used for MiTM
- Political tension is also on the rise ...

- What's the alternative to a PKI model?
- Currently all-or-nothing: trust the CA and all of the services it certifies, or don't trust anything
- Alternative: make decision on case-by-case basis based on observed server behavior
- Perspectives: check a set of *network notaries* to see what key has been offered in past
- *Spatial* & *temporal* redundancy



Convergence



- Moxie Marlinspike: Perspectives-like design
- Choose whether you want all notaries to agree or just majority agreement (configurable)
- Coined phrase: *trust agility*
- Questions:
 - ▶ How do you know the notaries are secure?
 - ▶ Which notaries should you trust?
 - ▶ What if you get two different views on what the public key should be?




10 Risks of PKI (circa 2000)



- This is an overview of one of many perspectives of PKI technologies
 - ▶ PKI was, like many security technologies, claimed to be a panacea
 - ▶ It was intended to solve a very hard problem: build trust on a global level
 - ▶ Running a CA -- “license to print money”
- Basic premise:
 - ▶ Assertion #1 - e-commerce does not need PKI
 - ▶ Assertion #2 - PKI needs e-commerce
- Really talking about a full PKI (everyone has certs.)



Risk 1 - Who do we trust, and for what?

- Argument: CA is not inherently trustworthy
 - ▶ Why do/should you trust a CA?
 - ▶ In reality, they defer all legal liability for running a bad CA
 - ▶ Risk in the hands of the certificate holder
- 
- Counter-Argument: Incentives
 - ▶ Any CA caught misbehaving is going to be out of business tomorrow
 - ▶ This scenario is much worse than getting sued
 - ▶ Risk held by *everybody*, which is what you want
 - Everyone has reason to be diligent

Risk 2 - Who is using my key?



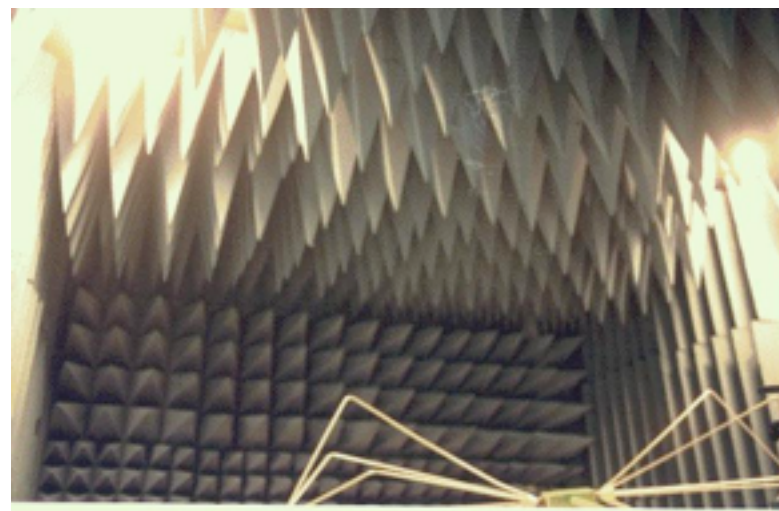
- Argument: key is basically insecure
 - ▶ Your key is vulnerable, deal with it
 - ▶ In some places, you are being held responsible after a compromise
- Counter-Argument: this is the price of technology
 - ▶ You have to accept some responsibility in order to get benefit
 - ▶ Will encourage people to use only safe technology
- Q: what would happen if same law applied to VISA?



Aside: TEMPEST



- Transient Electromagnetic Pulse Surveillance Technology
 - ▶ Monitor EMF emanations to reconstruct signal
 - ▶ For example, a video monitor normally exist at around 55-245 MHz, and can be picked up as far as one kilometer away.
 - ▶ ... or by a guy in a van across the street, e.g., steal private key.
- Generally, this is the domain of spy/national security issues
- Much classified work on signal eavesdropping and prevention



Risk 3 - How secure is the verif(ier)?



- Argument: the computer that verifies your credential is fundamentally vulnerable
 - ▶ Everything is based on the legitimacy of the verifier root public key (integrity of certificate files)
 - ▶ Browsers transparently use certificates
- Counter-Argument: this is the price of technology
 - ▶ You have to accept some *risk* in order to get benefit
 - ▶ Will encourage people to use only safe technology
- Q: What's in your browser?



Risk 4 - Which John Robinson is he?

- Argument: identity in PKI is really too loosely defined
 - ▶ No standards for getting credential
 - ▶ No publicly known unique identifiers for people
 - ▶ So, how do you tell people apart
 - ▶ Think about Microsoft certificate
- Counter-Argument: due diligence
 - ▶ Only use certificates in well known circumstances
 - ▶ When in doubt, use other channels to help
- Q: Is this true of other valued items (checks?)



Risk 5 - Is the CA an authority?



- Argument: there are things in certificates that claim authenticity and authorization of which they have no dominion
 - ▶ “rights” (such as the right to perform SSL) - this confuses authorization authority with authentication authority
 - ▶ DNS, attributes -- the CA is not the arbiter of these things



- Counter-Argument: this is OK, because it is part of the implicit charge we give our CA -- we implicitly accept the CA as authority in several domains

Risks 6 and 7



- 6 : Is the user part of the design?
 - ▶ Argument: too many things hidden in use, user has no ability to affect or see what is going on
 - Ex.: Hosted website has cert. of host(er), not page
 - ▶ Counter-Argument: too sophisticated for user to understand
- 7 : Was it one CA or CA+RA?
 - ▶ Argument: separation of registration from issuance allows forgery
 - e.g., RA handles vetting, CA makes certificates, so, you better have good binding between these entities or bad things can happen
 - ▶ Counter-Argument: this is an artifact of organization, only a problem when CA is bad (you are doomed anyway)



Risks 8 and 9



- 8 : How was the user authenticated?
 - ▶ Argument: CAs do not have good information to work with, so real identification is poor (as VISA)
 - ▶ Counter-Argument: It has worked well in the physical work, why not here?
- 9 : How secure are the certificate practices?
 - ▶ Argument: people don't use them correctly, and don't know the implications of what they do use
 - Point in fact: revocation and expiration are largely ignored in real system deployments
 - ▶ Counter-Argument: most are pretty good now, probably won't burn us anytime soon



Risk 9 - How secure cert. practices?



- Argument: certificates have to be used properly to be secure
 - ▶ Everything is based on the legitimacy of the verifier root public key, protection of its key
- Counter-Argument: this is the price of technology
 - ▶ You have to accept some *risk* in order to get benefit
 - ▶ Will encourage people to use only safe technology
- Q: What's in your browser?



Risk 10 - Why are we using PKI?



- Argument: We are trying to solve a painful problem: authenticating users.
 - ▶ However, certificates don't really solve the problem, just give you another tool to implement it
 - ▶ Hence, it is not a panacea
 - ▶ No delivered on its promises



- Counter-argument?

Burning question ...



- Can we solve the PKI problem with better crypto?

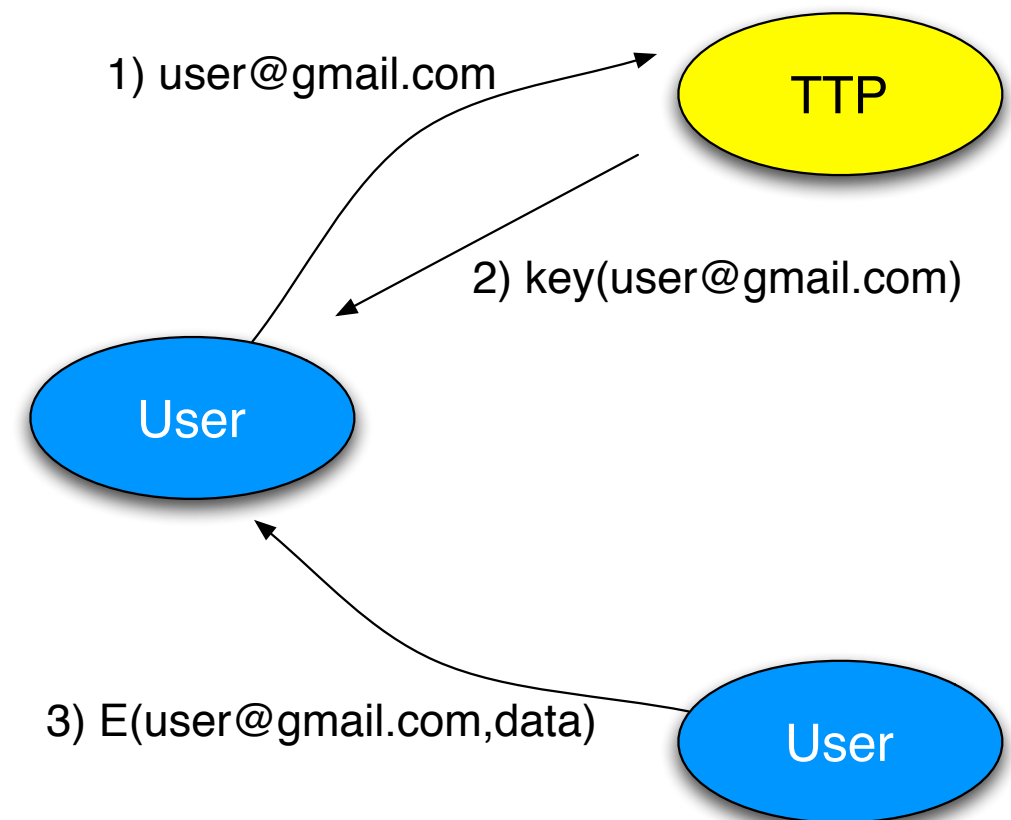


Identity Based Cryptography



- What if your email address was your public key?
 - ▶ E.g., $E(\text{whenck@ncsu.edu}, \text{data}) = \text{ciphertext?}$
 - ▶ E.g., $\text{Verify}(\text{signature}, \text{whenck@ncsu.edu})$
- 1984 - Shamir asked for such a system, but it (largely) remained out of reach until Boneh/Franklin 2001
 - ▶ The public key is any arbitrary key
 - ▶ Based on “Weil pairings” -- a new cryptographic device with lots and lots of uses (IBE among them)
 - ▶ Interested readers should see: Identity based encryption from the Weil pairing, SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003.
- Advances from theory community, few systems

- Functionally, you receive your privacy key from a *trusted third party* who is responsible for generating all keys in the system.
- Thereafter you (and others) can use the system as if you generated the private key yourself.
- Advantages
 - ▶ No public key distribution
 - ▶ No name binding problems (?)
 - ▶ Key space flexibility
 - ▶ Others?



Basic IBE Construction

- *Setup* (generate by TTP)

$$Global\ Parameters = G$$

$$Master\ Key = K_G$$

- *Extract* (by TTP for user, sting “str”)

$$Extract(G, K_G, Str) = K_{Str}^-$$

- *Encrypt* (by user)

$$E(G, Str, data) = ciphertext$$

- *Decrypt* (by user)

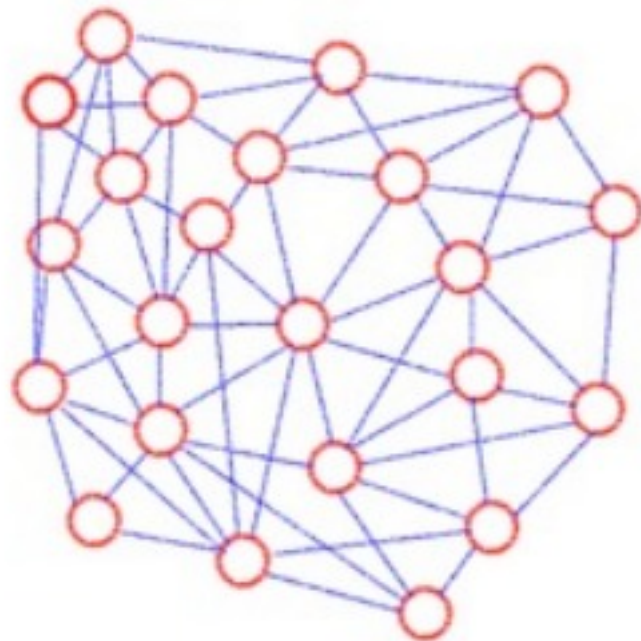
$$D(G, K_{Str}^-, ciphertext) = data$$

- Many thought that IBE would lead to a revolution in public key system (solve PKI problems), it didn't.
- Why - IBE moves the problems around
 - ▶ Is there any **TTP** that everyone trusts?
 - ▶ String ambiguity is still a problem? (John Robinson?)
 - ▶ Revocation is still a problem (potentially worse)
 - ▶ ... (see 10 reasons above)
- Fundamentally
 - ▶ IBE really does not solve the CA problem, as the TTP is fulfilling that role.
 - ▶ Having strings instead of obscure numbers does not get at the problems with PKI ...
 - ▶ Existence of certificates is not really the problem ...

A thought ...



- Can we build secure systems without a centralized authority?



- Two parallel efforts to solve this problem
 - ▶ SDSI: Simple Distributed Security Infrastructure
 - ▶ SPKI: Simple Public Key Infrastructure
 - ▶ Not widely used
- In SDSI/SPKI, principals/organizations declare groups of keys that they use for communicating with others
- To find a key for “Ashwin Shashidharan”, instead of searching for Ashwin’s key and seeing who claimed to sign it, you would look for “William Enck’s Keys” and find the listings for his colleagues and students.

Pretty Good Privacy



- PGP: Developed by Phil Zimmerman in 1992
 - ▶ Widely used
- Suppose Alice knows Bob. If Bob makes his key public, Alice can sign it after Bob proves his identity
- Charlie can look up Bob's key in a directory
 - ▶ If Charlie knows Alice's key, he can trust Bob's key
- PGP uses a web of trust: instead of PKI tree, we have a PKI graph
 - ▶ We can trust a key declaration if there is one or more trusted graph traversals

Problems with distributed models



- Key Distribution?
- Strangers?
- General Usability?