Certificates, Certificate Authorities, Certification Paths

István Zsolt BERTA
istvan@berta.hu

PKI lectures

1. Public key cryptography primitives
2. **Certificates, Certificate Authorities, Certification Paths**
3. Electronic signatures: signature creation & validation
4. Information security management at CAs
5. PKI Business

Certificates, CAs, Certification paths…

1. What is a Digital Certificate?
2. Contents of a Digital Certificate
3. Certificate Authority
   – registration, certificate issuing
   – revocation, CRL, OCSP
   – certificate policy
4. Certification Path

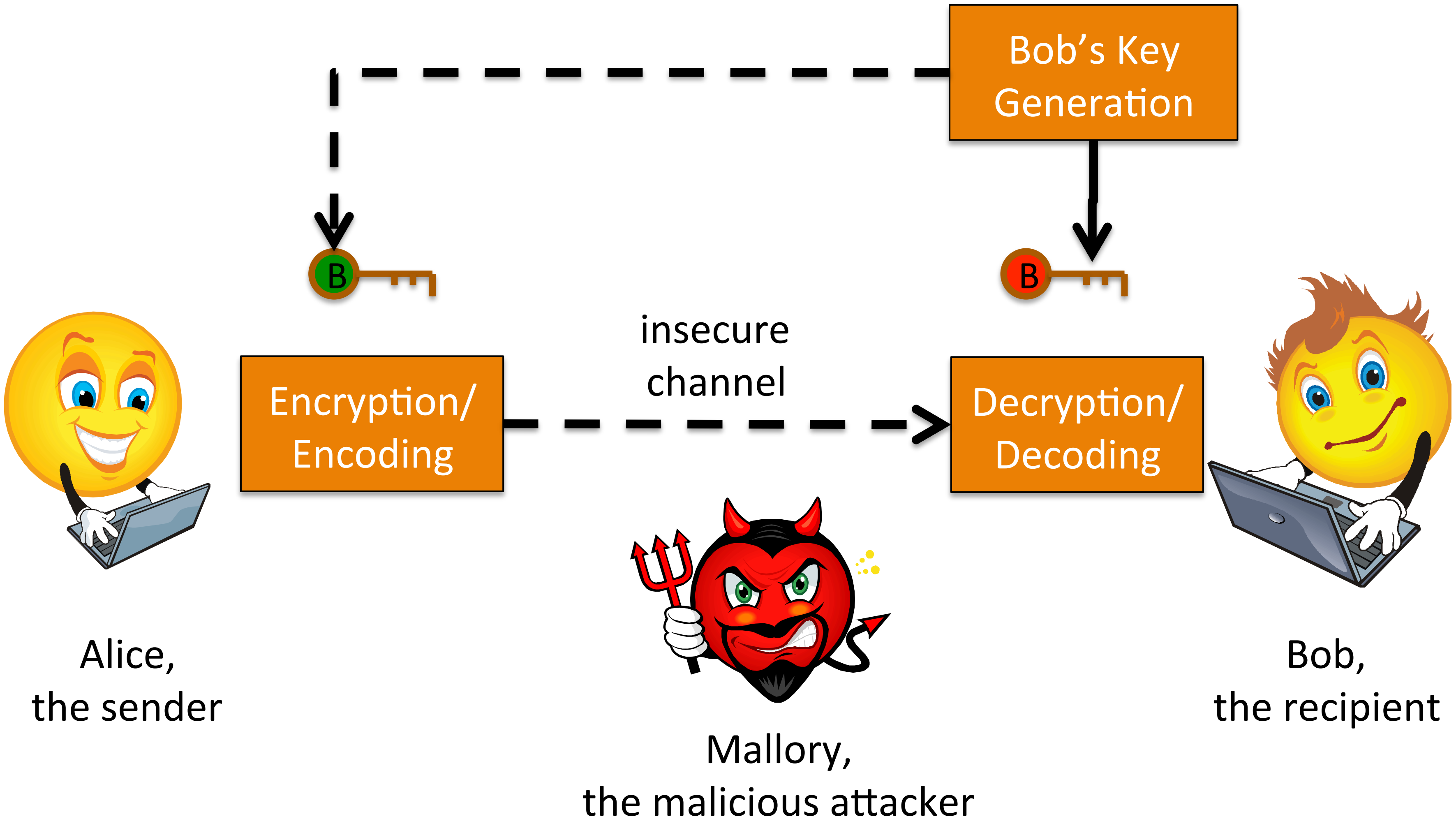
What is a digital certificate?

Authenticity of the public key

- Using someone's public key, we can secure our messages the given party
  - we can send encrypted messages to them
  - we can verify their signatures
- How can I obtain someone's public key?
- How do we know that we have the right public key?

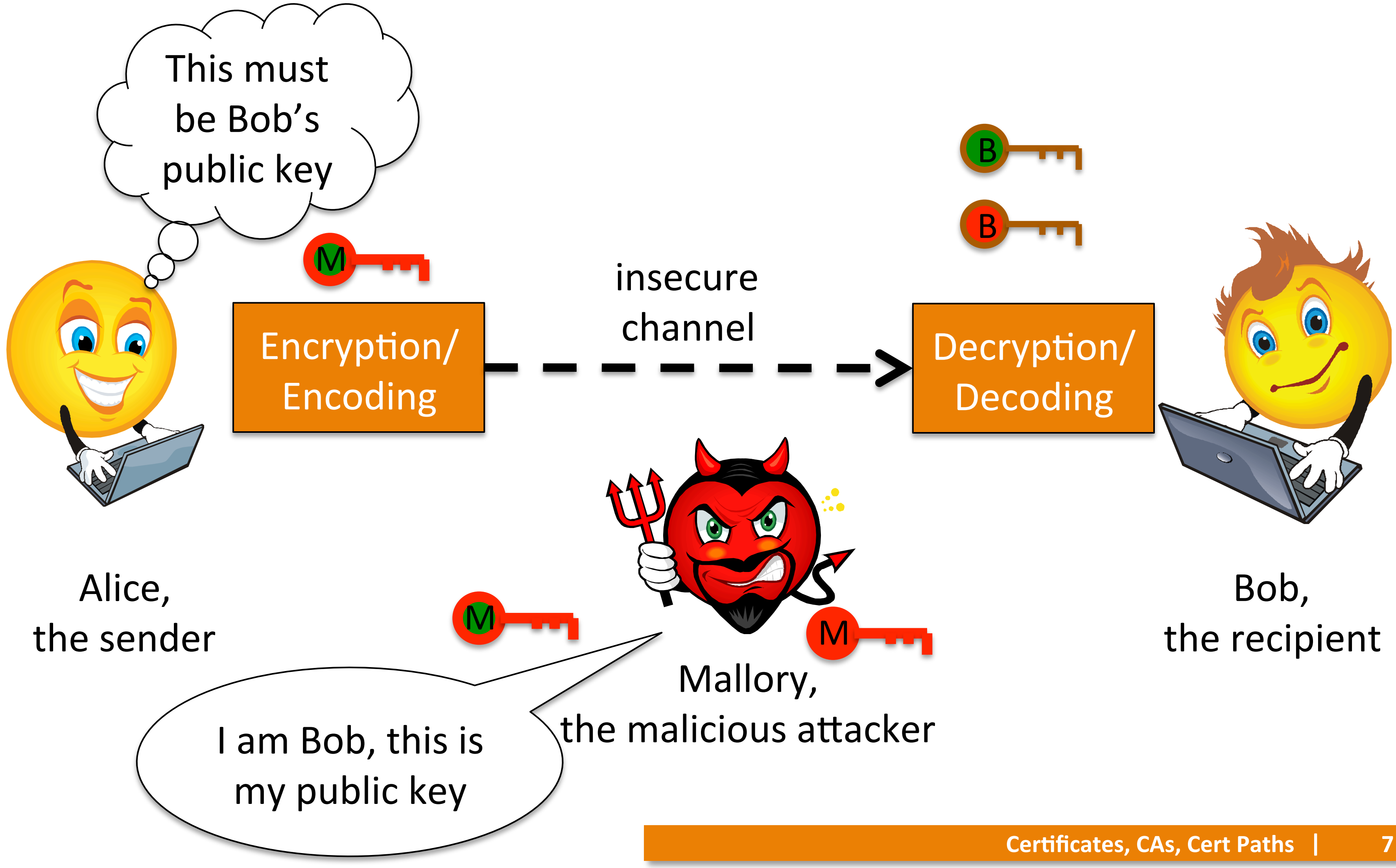
Public Key Cryptography

a.k.a. Asymmetric Key Cryptography



Man-in-the-middle attack

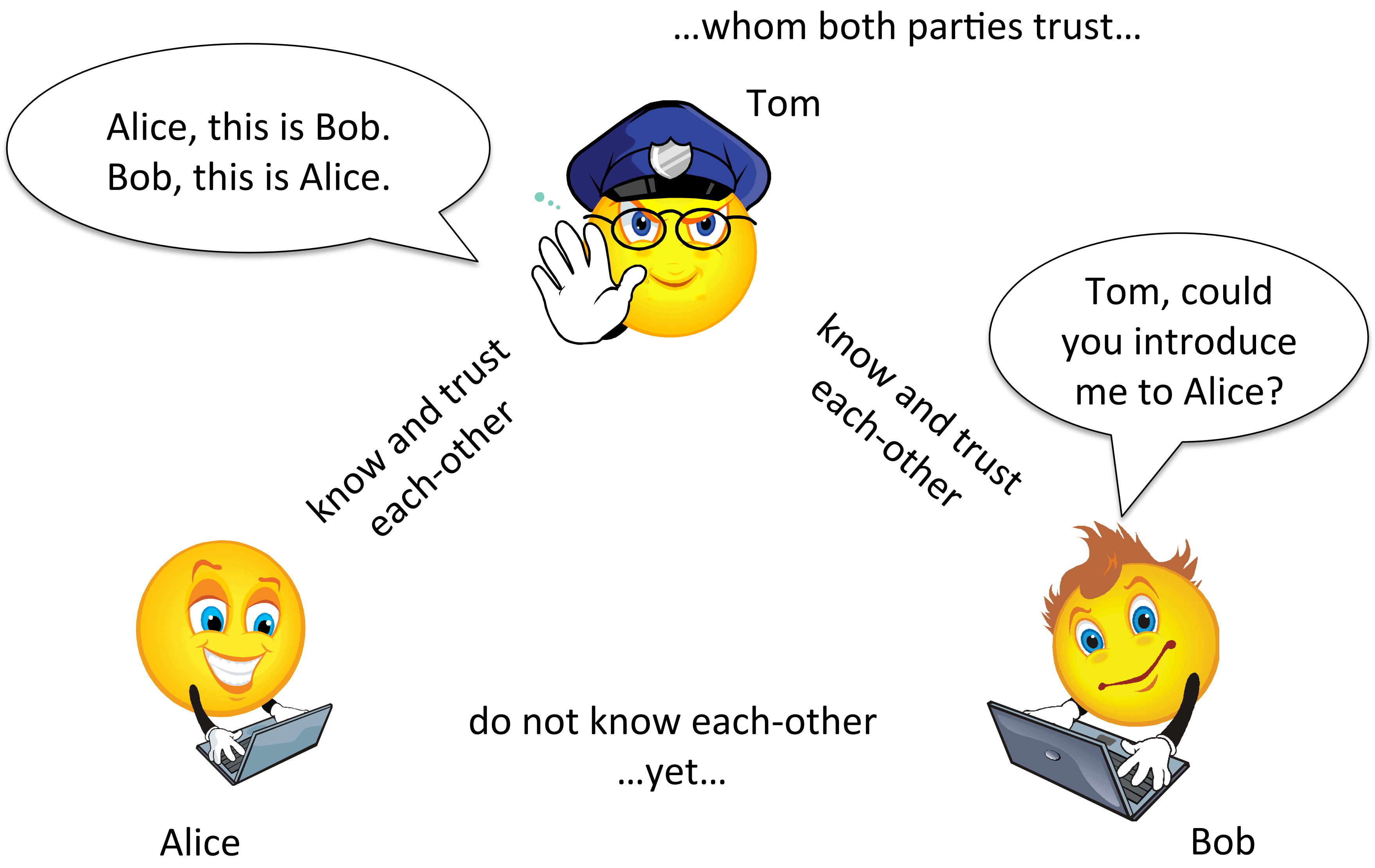
Mallory tricks Alice into using his public key in lieu of Bob's



Why is the authenticity of public key critical?

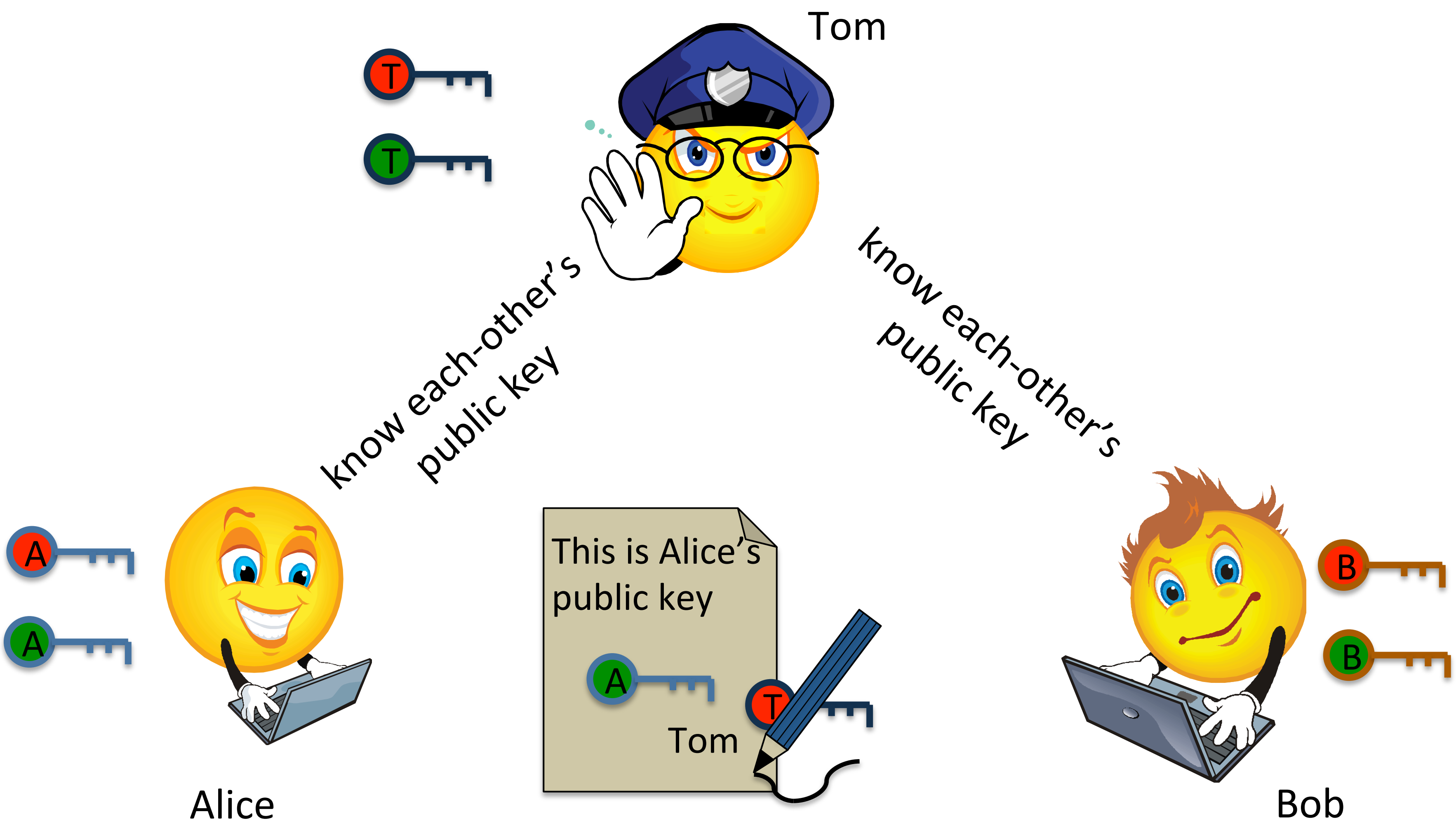
- If we encrypt with the attacker's public key, the attacker will be able to decrypt our message
- If we verify signed messages with the attacker's public key, we shall consider messages signed by the attacker as authentic
- Before using a public key, we need to make sure that it belongs to the person/party we want to communicate with

Trusted Third Party


…whom both parties trust…
Tom
Alice, this is Bob. Bob, this is Alice.
Tom, could you introduce me to Alice?
know and trust each-other
know and trust each-other
do not know each-other …yet…
Alice
Bob

Trusted Third Party

…whom both parties trust…

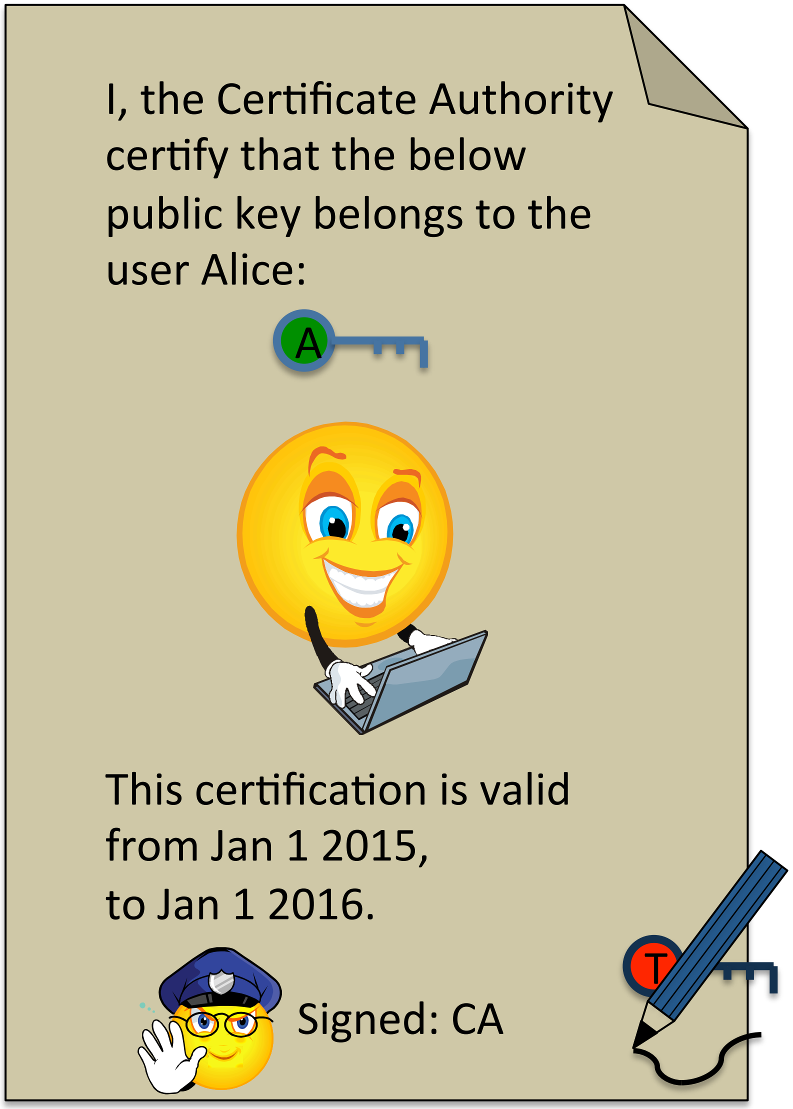


Tom
T
T
know each-other's public key
know each-other's public key
A
A
This is Alice's public key
A
T
Tom
B
B
Alice
Bob

Public Key Infrastructure

- Public Key Infrastructure (PKI) is the set of organizations, crypto algorithms, standards, laws, assumptions etc. that can be used for gaining assurance that a certain public key belongs to a certain individual.

Digital Certificate



I, the Certificate Authority certify that the below public key belongs to the user Alice:

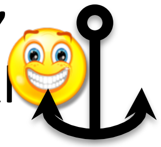
This certification is valid from Jan 1 2015, to Jan 1 2016.

Signed: CA

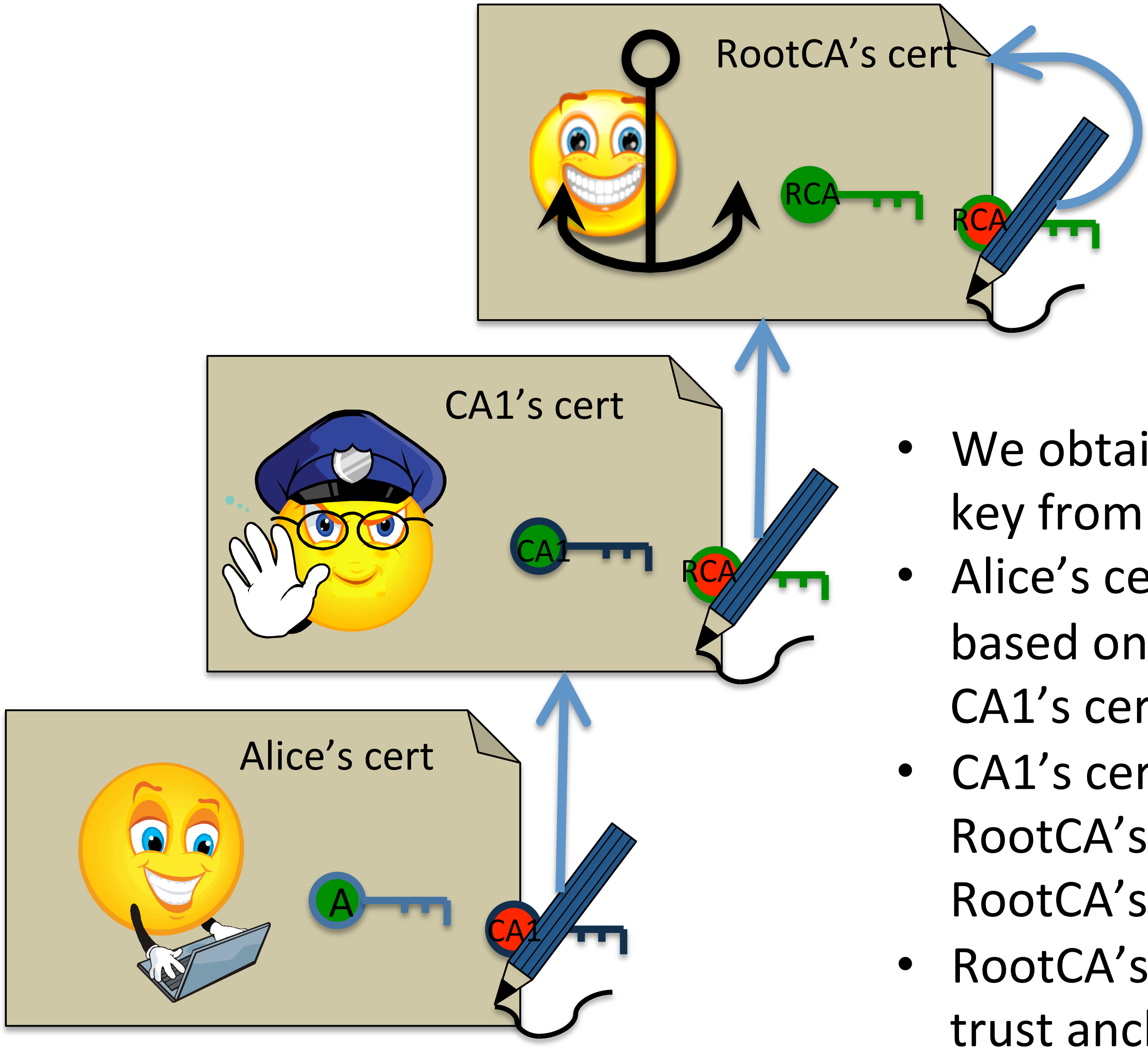
- Contains the 'name' of the subject (user/organization)
- Contains the public key of the subject
- Valid for a certain period of time only
- May contain further information
- Signed with the private key of a certificate authority (CA), can be verified by the CA's public key
- Standardized format (X.509)

PKI Terminology

- Certificate Authority (CA): An organization who issues certificates
- End-entity: A user / machine / organization / etc.
  who does NOT issue certificates (but may have one)
- Subject: The party for whom the certificate is issued for;
  the Subject can be either a CA or an End-entity
- Subscriber: The one who pays for the CA's service
- Relying party: The one who uses a certificate for obtaining the public key of a Subject in a secure manner
- Trust anchor: A public key + name we use as a basis for verification;
  trust anchors are obtained and verified out-of-band, and not via PKI
- Root CA: The CA whose public key / cert is used as a trust anchor – at a given time, by a given relying party



Certification Path



- We obtain user Alice's public key from Alice's cert
- Alice's cert can be verified based on CA1's public key in CA1's cert
- CA1's cert can be verified by RootCA's public key in RootCA's cert
- RootCA's key/cert is a trust anchor

Contents of a Certificate

Key fields

- Serial number
- Subject (→ distinguished name, DN)
- Issuer (→ distinguished name, DN)
- Validity: notBefore, notAfter
- Certificate Policies
- QCStatement (for qualified certs)
- Transaction Limit (for qualified certs)
- Availability of revocation information
- Key usage
- Public key of subject, CA’s signature, key identifiers, identifiers of algorithms used, etc

Structure of a Certificate (RFC 5280)

tbsCertificate

- Version (X.509 v3)
- Serial Number
- Signature (identifier of algorithm)
- Issuer (DN)
- Validity → notBefore, notAfter (UTC)
- Subject (DN)
- Subject Public Key Info (identifier of algorithm)
- (unique ID for subject/issue keys, should not be present)
- Extensions → …, …, …, …

signatureAlgorithm

signatureValue

ITU-T X.509 is the core specification, RFC5280 is for internet certs.

Some extensions

- Detailed description of how the cert can be used
  - certificate policies
  - QCStatements (for qualified certificates)
- Restrictions on key usage
  - key usage
  - extended key usage
- Constraints on cert usage
  - basic constraints
  - naming constraints / policy constraings / etc
- Alternative names of the subject
- Availability of revocation information
  - CRL distribution point
  - freshest (delta) CRL
  - authority information access
- …

Subject Distinguished Name (DN)

- Distinguished Name, as per LDAP
  - Common Name, Surname, GivenName stb.
  - Title
  - Organization, Organization Unit
  - Locality
  - Country
  - Pseudonym
  - Email Address
  - Serial Number, Distinguished Name Qualifier
  - ...
- Pseudonyms → can a cert be pseudonymous?
- What does an organization in a certificate mean?
- How does the CA verify the above information?
- RFC 4043: Permanent Identifier

Examples for pseudonyms

- Donald Duck – obvious pseudonym, makes no sense
- someone else's name – could be misleading
- different spelling – e.g. Kovacs Janos instead of Kovács János
- preserving privacy of the subject
  - CN = managing director
  - O = XYZ Plc.
  - C = HU
- which part of the DN is 'pseudo' and which part is 'real'?

Key Usage extension

```
KeyUsage ::= BIT STRING {
          digitalSignature        (0),
          nonRepudiation          (1),
          keyEncipherment         (2),
          dataEncipherment        (3),
          keyAgreement            (4),
          keyCertSign             (5),
          cRLSign                 (6),
          encipherOnly            (7),
          decipherOnly            (8)  }
```

- Further restrictions can be imposed via ExtendedKeyUsage (EKU), e.g.:
  - clientAuthentication, serverAuthnetication
  - e-mail protection
  - timestamping, code signing

Typical key usage combinations

- Signature
  - nonRepudiation (+ digitalSignature) – EU standards
  - digitalSignature alone – US standards
- Encryption
  - keyEncipherment + dataEncipherment + secureEmail EKU
- Authentication
  - digitalSignature + keyAgreement + e.g. clientAuthentication EKU
- Certificate Authorities
  - keyCertSign + cRLSign

Types of certificates

- X.509 defines attributes
- Webserver (TLS/SSL) certs
  - Domain Validated (DV) certs – CA verifies if subject controls the domain
  - Organization Validated (OV) certs – CA verifies that the subject organization
  - Extended Validation (EV) certs
- Purpose of the cert can be
  - authentication (e.g. TLS)
  - encryption
  - signature / qualified signature
- Subject can be
  - natural person
  - organization
  - device

Signature, Encryption, Authentication

- EU standards require a different certificate & key pair for signature, encryption and authentication
  - authentication: a random challenge is encoded with the private key – one could trick the user into sign a document or decrypting a message
  - encryption: decryption key should be escrowed, signature keys (and authentication keys) MUST NOT be escrowed
  - signature: in many EU countries (e.g. Germany, Hungary) law also prohibits the signing key to be used for anything else

  (reason: qualified signatures / controls over encryption)
- US standards do not mandate such a strong separation (as there is no term like qualified signatures)

Certificate Authority

Certificate Authority (CA)

- The organization that certifies that a given public key belongs to a given subject
- via issuing digital certificates
- The CA operates a secure system, so that it may act as a 'trusted third party' and relying parties may accept their certificates
- The aim of this system is to ensure the secure usage of the CA private key, and the integrity of the CA's registries
- The CA may undergo various audits so that it may prove the above to relying parties
  - EU: governmental supervision (e-signature legislation)
  - US: commercial audits
- CA is called Certification Service Provider (CSP) in EU legislation

What does the CA do?

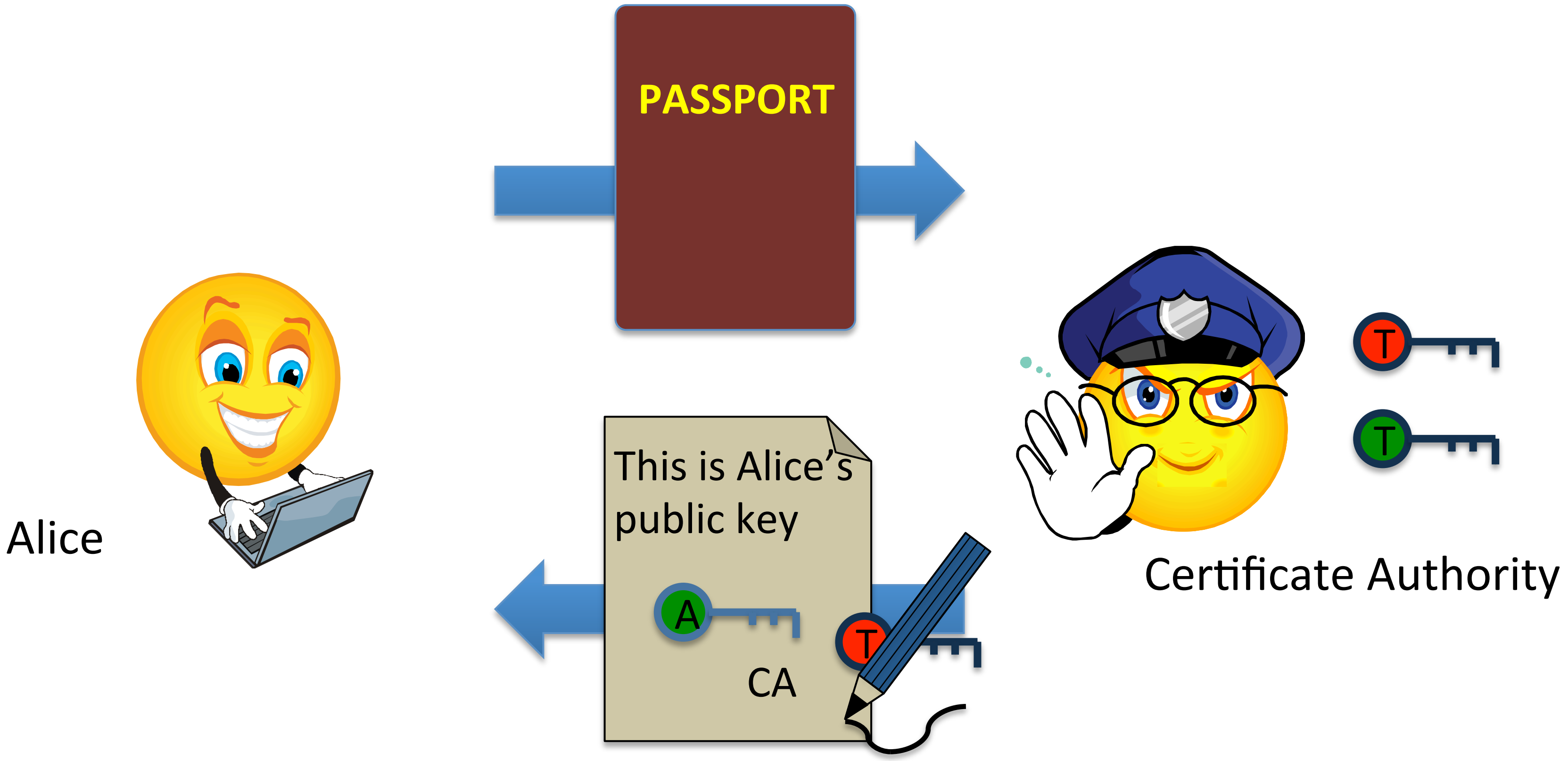
- Registration: The CA verifies the information to appear in the certificate and/or the identity of the subject
- The CA ensures that the public key to appear in the certificate belongs to the subject, i.e. the private key is solely controlled by the subject
- The CA revokes the certificate if:
  - unauthorized parties may have accessed the subject's private key
  - any information in the certificate changes
- The CA retains information obtained at registration
- The CA is liable for the certificate it issues
- The CA should operate according to public Certificate Policies
- The CA may charge money for the above services

Certificate Policy (CP)

- A document that contains requirements that apply to the cert, e.g.:
  - how the subject was registered, and how their DN was determined
  - how the CA verified that the given public key belongs to the subject
  - how the certificate is revoked, and how long this takes
  - if the CA is responsible for the certificate, with any limitations
  - what can the certificate be used for
  - what legal effect the certificate may have
  - how long registration information is retained
  - …
- The CA should indicate in the Certificate Policies extension of the cert the OIDs of the policy/policies that apply to the certificate
- CPs should follow the structure of RFC3647 so that they can be compared
- Multiple CAs may apply the same CP, they can indicate in their Certificate Practice Statement *how* the generic requirements of the CP are implemented; → Relying Party Agreement (RPA)

Registration

- The process where the identity / name of the user is verified her public key is obtained called ‘registration’

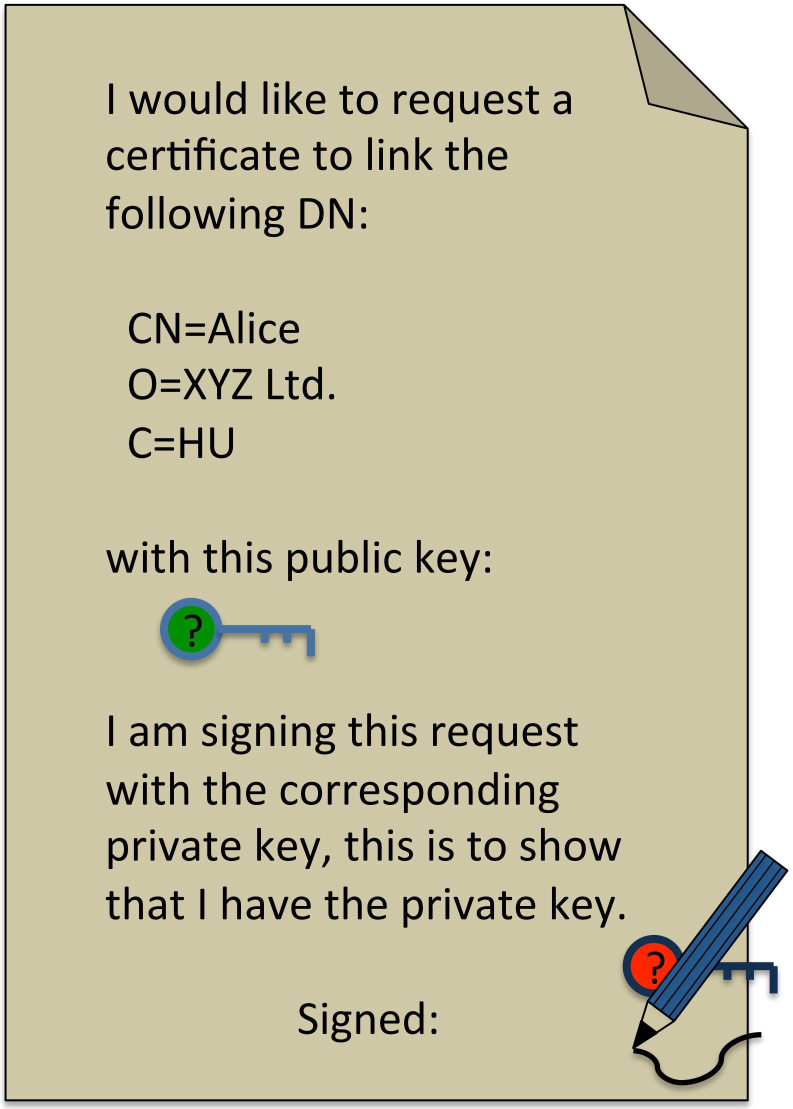


Registration (cont'd)

- The CA may check public registries to verify that the subject individual or organization exists
- Certain certs may be issued remotely
- Certain certs – including qualified certs (EU law) – need to be issued after face-to-face registration only

- The CA may have a separate Registration Authority (RA), who performs the registration; the CA may be performing technical steps only in this case
- Liability of CA vs liability of RA
  $\rightarrow$ problematic case, e.g. in Hungary it does not work
- Sometimes RAs are just resellers

Certificate Request (PKCS#10)



I would like to request a certificate to link the following DN:

CN=Alice
O=XYZ Ltd.
C=HU

with this public key:

I am signing this request with the corresponding private key, this is to show that I have the private key.

Signed:

- A PKCS#10 cert request contains the DN for the cert and the public key, and it is signed with the corresponding private key
- It proves that someone at some point of time requested a cert for the given DN and key, and that party was in possession of the private key
- This is a typical way to communicate a public key to a CA

Key generation

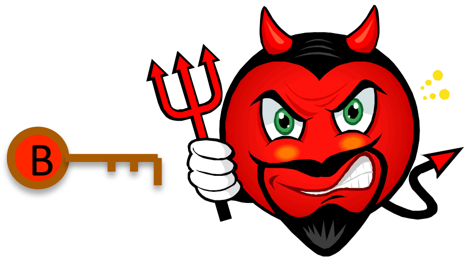
- Case 1: If the key pair is generated by the Subject
  - The subject creates a PKCS#10 and sends it to the CA
  - The CA never receives the private key so cannot abuse it
  - The CA issues the cert based on the PKCS#10 and the registration information
- Case 2: If the key pair is generated by the CA
  - The CA issues a certificate based on the registration information
  - Internally the CA probably also uses a PKCS#10
  - The CA must ensure not to use the private key (for anything else than signing the PKCS#10 …)
  - The CA must hand over the private key to the subject, and destroy any other copies so that it cannot abuse them
  - Hardware crypto tokens (smart card / USB) can ensure this
  - Key escrow service for encryption certs can be an exception for this

Where is the private key?

- Soft keys – key is stored in a file (database, registry, etc) on a computer
  - pro: easy to use, can be used on multiple machines, backups can be made
  - con: easy to use, can be used on multiple machines, backups can be made
- Smart cards and personal hardware tokens
  - secure microcomputer
  - they generate the key and the private key cannot be exported
  - wherever my token is, there is my private key ☺
- Hardware security modules (HSM) as enterprise solutions
  - ~smart cards: wherever my token is, there is my private key ☺
  - high performance, many private key operations as bulk
  - “n of m” access to the key, possibility to make backups
  - CA signing key must be protected via HSM
- Keys in the cloud
  - my key is at a cloud service provider (probably in an HSM)
  - I don’t need to bother with a token, but I am at the mercy of the provider

Revocation

- If the attacker obtains the private key of the user, he may abuse it: the attacker may sign documents or decrypt messages on the user's behalf

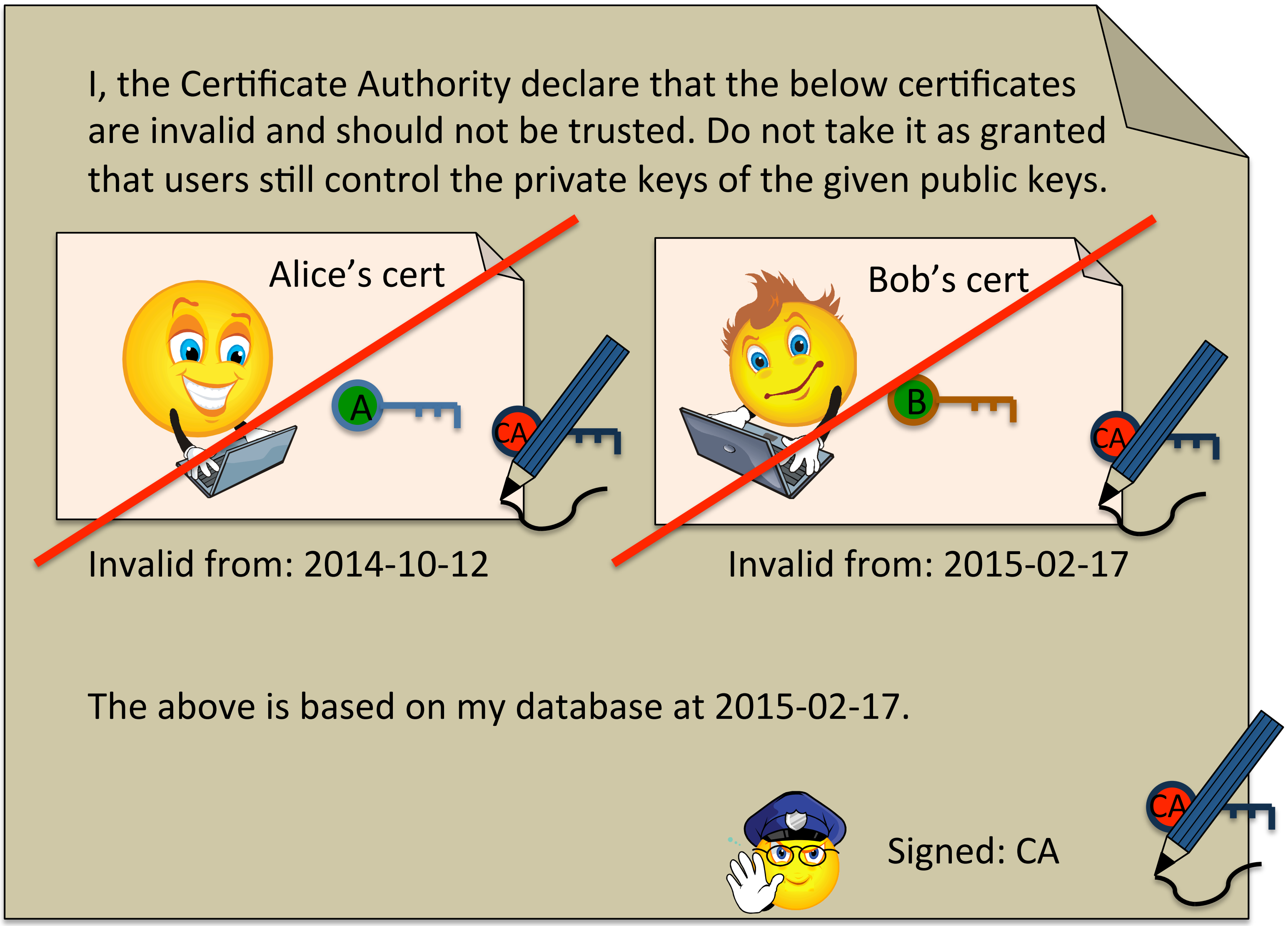


- A certificate is valid within a given time period only
  - expired certificates are invalid
- A certificate can become invalid asynchronously if it is revoked
  - revocation – permanent
  - suspension (on hold) – suspended certs can be reinstated → …
- Revocation / revocation status is published, usually by the issuing CA

Revoked certs…

- Revocation means: the cert is invalid, its private key is not necessarily controlled by the Subject
- Encryption and Authentication certs:
  - DO NOT USE THE PUBLIC KEY IN THE CERT
- Signature certs:
  - new signatures should not be accepted, but
  - old signatures should remain valid…
  - …it gets tricky…

Certificate Revocation List (CRL)



I, the Certificate Authority declare that the below certificates are invalid and should not be trusted. Do not take it as granted that users still control the private keys of the given public keys.

Invalid from: 2014-10-12

Invalid from: 2015-02-17

The above is based on my database at 2015-02-17.

Signed: CA

Structure of a CRL (RFC 5280)

tbsCertList

- Version (X.509 v3)
- Signature (identifier of algorithm)
- Issuer (DN, typically same as IssuerDN in certs)
- thisUpdate (information relates to this time)
- nextUpdate (that time I will have a new CRL)
- revoked certificates → (cert serial, invalidity date, reason)
- Extensions → …, …, …, …

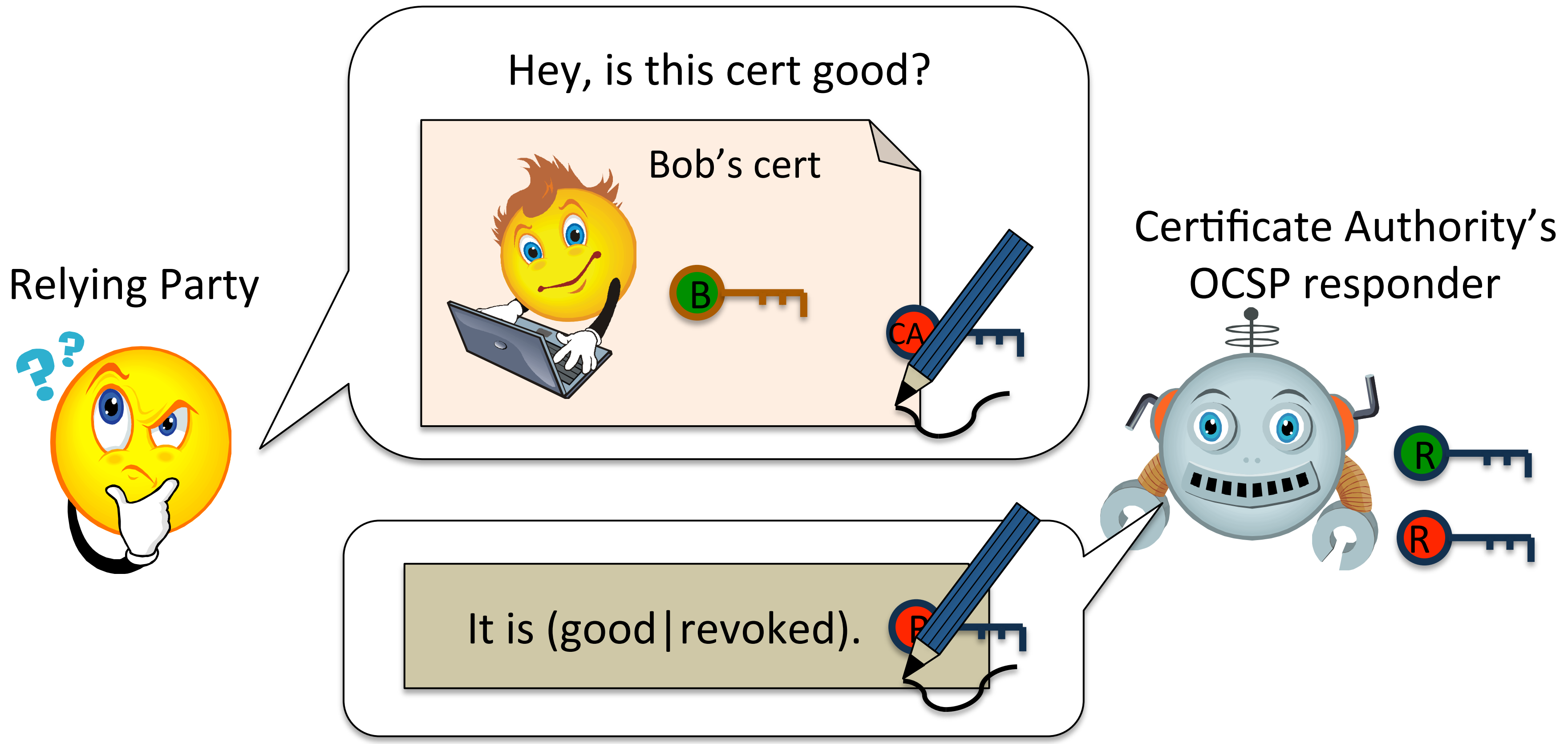
signatureAlgorithm

signatureValue

Revocation Reason

```
CRLReason ::= ENUMERATED {
       unspecified                 (0),
       keyCompromise               (1),
       cAComplomise                (2),
       affiliationChanged          (3),
       superseded                  (4),
       cessationOfOperation        (5),
       certificateHold             (6),
            -- value 7 is not used
       removeFromCRL               (8),
       privilegeWithdrawn          (9),
       aACompromise               (10)  }
```

Online Certificate Status Protocol (OCSP)



- Online question, immediate answer; answer may be fresh or pre-generated
- CAs prefer OCSP for significantly lower bandwidth
- OCSP is **strongly** preferred for electronic signatures
- RFC 6960 (formerly: RFC 2560)

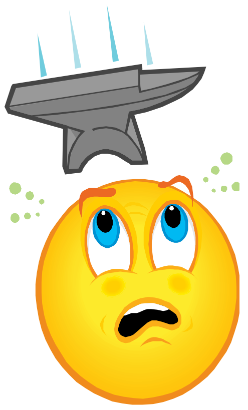
CA Liability

- A CA should be liable for any damages resulting from its services (it EU law, this is explicitly stated)
- Damages may result from:
  - Issuing the cert to the wrong person or with invalid information e.g. cert issued to Alice, where the private key is controlled by Mallory
  - Failure to process revocation request ‘in time’ (and the attacker abuses the certificate)
  - CA breached by attacker, issuing arbitrary certs ☹
- Both the Subject/Subscriber and the Relying Party may suffer damage

- There are no court cases, no precedents
- DigiNotar is the only CA ever punished

CA Breaches / Events

- Comodo – attackers obtained fraudulent certs by hacking a reseller of Comodo
- Certigna – private key published by mistake (for ssl cert?)
- DigiNotar – CA fully breached in 2011 → eliminated
- GlobalSign – no breach, turned out to be a false alarm
- Duqu malware – targeted against CAs? → ask Levente ☺
- KPN – found DDOS tools on its webserver…
- DigiCert – banned for issuing 512-bit RSA keys
- StartCom – attacked but could deflect breach
- Verisign – hacked in 2010, but it was not the CA?
- Trustwave – issued a CA cert to a non-CA
- Turktrust – mistakenly issued CA cert



CA prices

- Significant differences in prices, ranging
  from $1000 (Symantec/Verisign) to free certs (e.g. Startcom)
- Clients have no way to differentiate between products
- CAs may charge based on prestige
- The market is not driven by costs

- CAs are trying to find a way to charge more for the same product (e.g. EV certs)

- Comparison of prices:
  – https://www.sslshopper.com/
  – http://www.whichssl.com/compare-ssl-certificates.html

Which cert to choose?

- *"Any CA can usurp a certificate issued by any other CA"*
- *"Overall security is that of the least trustworthy CA"*

- ***"Anyone who selects a public CA on a factor other than price fails to understand the trust models that underlie today's use of CAs"***

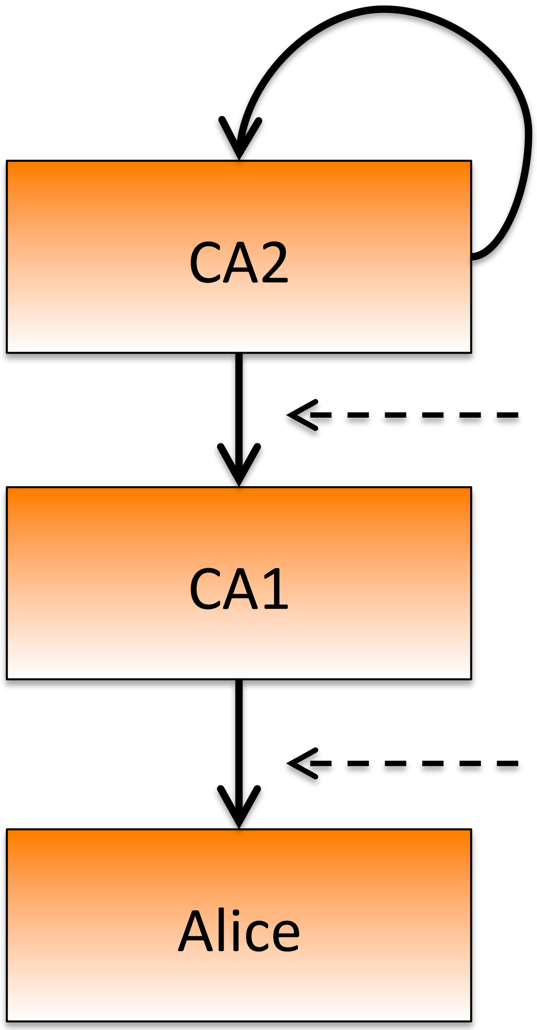
Source: [Peter Gutmann]()

Notes:

- Choose based on price, but take note of what your relying parties accept. (EV might be a factor.)
- Customer service can also be a factor.
- 100% agree: There is no point in buying certs from a secure CA; there is point in accepting secure CAs only as a relying party

Certification Paths

Notations



entity CA2 and its key pair

self-signed certificate of CA2

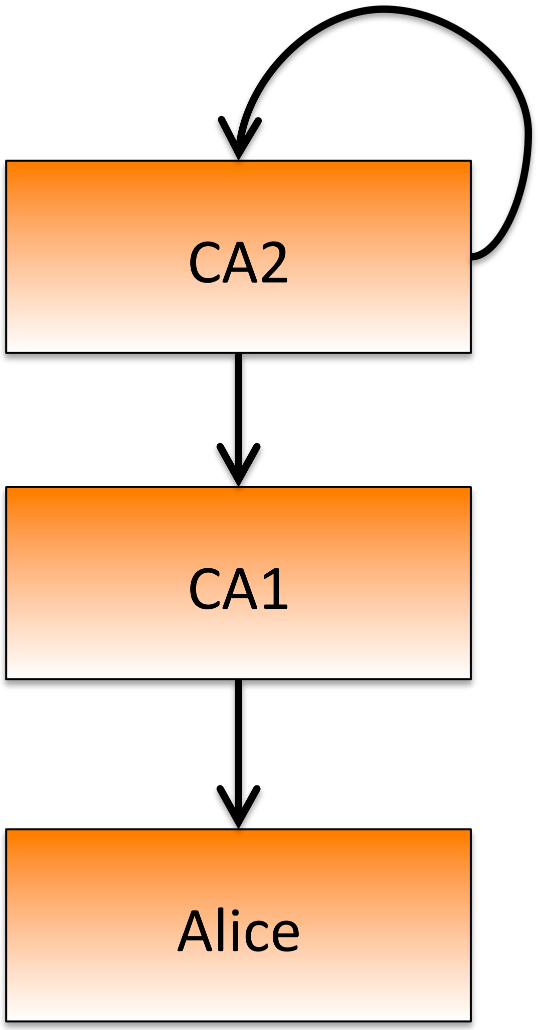
CA1's certificate issued by CA2

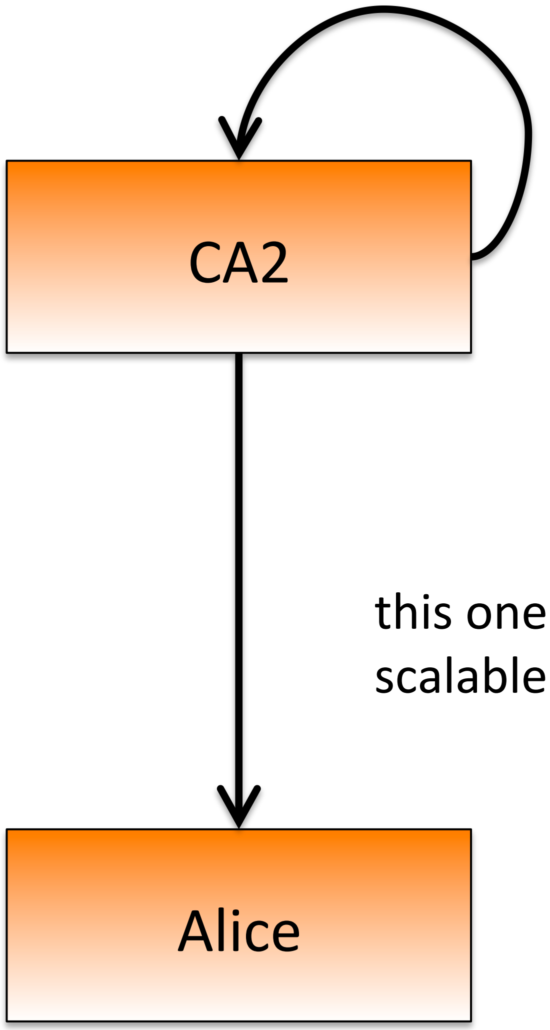
entity CA1 and its key pair

the end-entity's certificate issued by CA1

the end-entity and her key pair

Basic chains





this one is less scalable

Root CA / trust anchor

- Any certificate validation is based on trust anchors
- Typically, certain CA's certain self-signed certificates are used as trust anchors – root CA, root CA certificate
- There is no single word root
- Each PKI community has their own root/roots, e.g.:
  - EU roots
  - NATO root
  - roots accepted by Widows
  - roots accepted by Mozilla
  - roots accepted by Apple, …
  - roots connected to US Federal Bridge CA
  - …

How to obtain a trust anchor?

- Gives it to you when you subscribe to its services
  → this did not evolve this way
- Some CAs publish the hash of their root certs in printed media
  → not much use, but nice tradition
- Obtained via other CAs or trust services lists…
- **Comes pre-installed with your OS**; major systems have root certificate programs
  - Microsoft
  - Mozilla
  - Apple
  - Google
  - Opera
  - Adobe

What makes a CA ‘root’?

- A self-signed certificate?
- It handles its private key in a safe and secure way?
- It must be used by law?
- It provides good quality revocation status information?
- It is widespread?
- It takes responsibility?
- Users trust it?

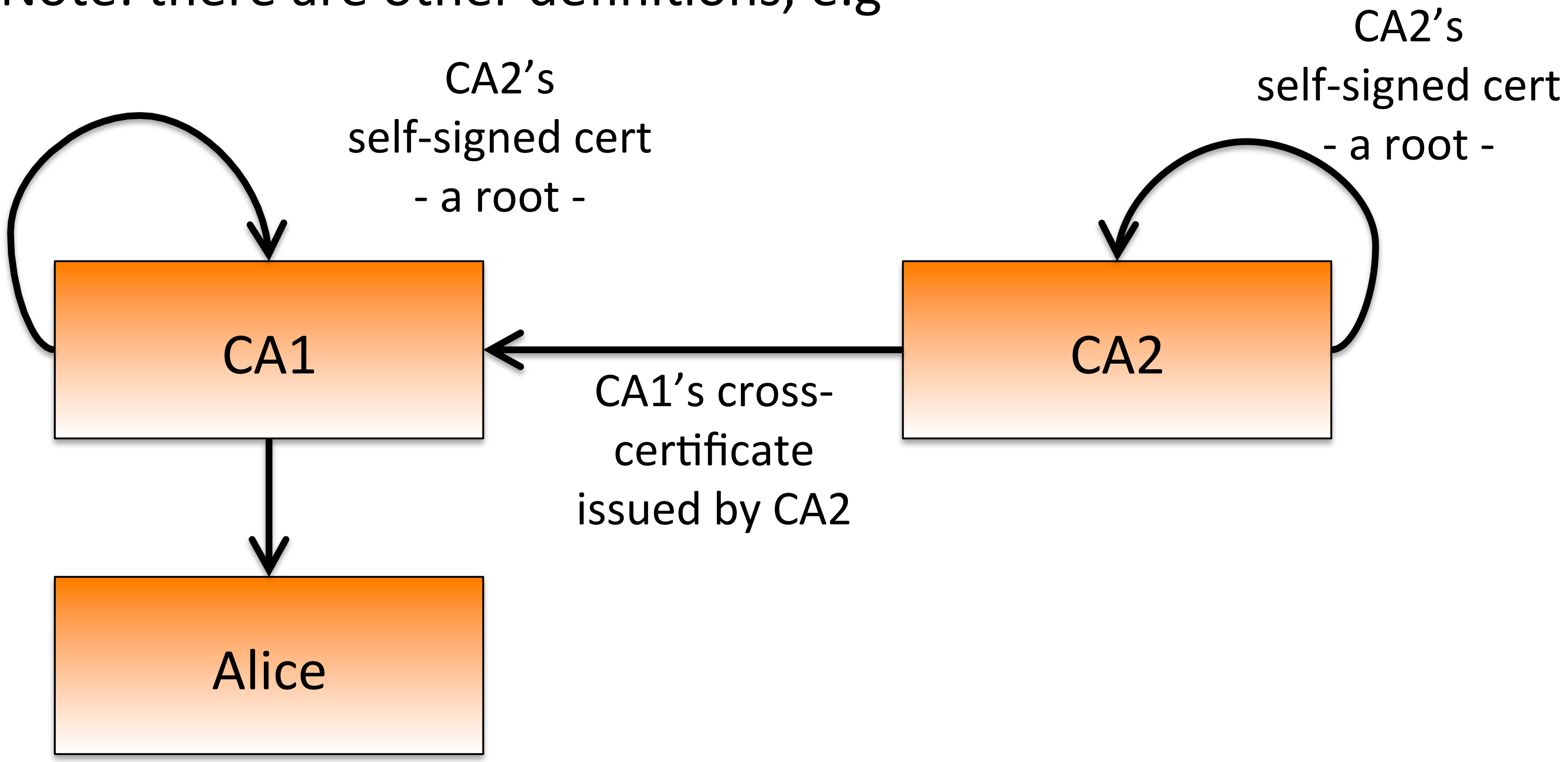
- **Its certificate is used by a wide range of users in a certain community as a trust anchor**

Connecting PKI communities

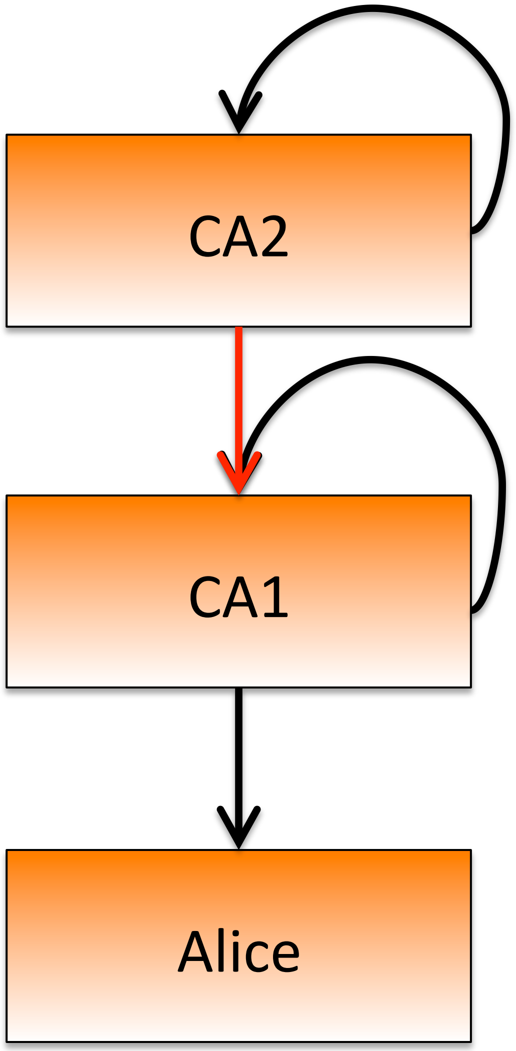
- PKI communities can be connected by e.g. community A’s root CA issuing a certificate for the root of community B
- Thus can users of community A accept all certificates in community B without (hopefully) any changes in their configuration

Cross-certification

- Cross-certification: A CA issues a certificate for another CA.
- Note: there are other definitions, e.g

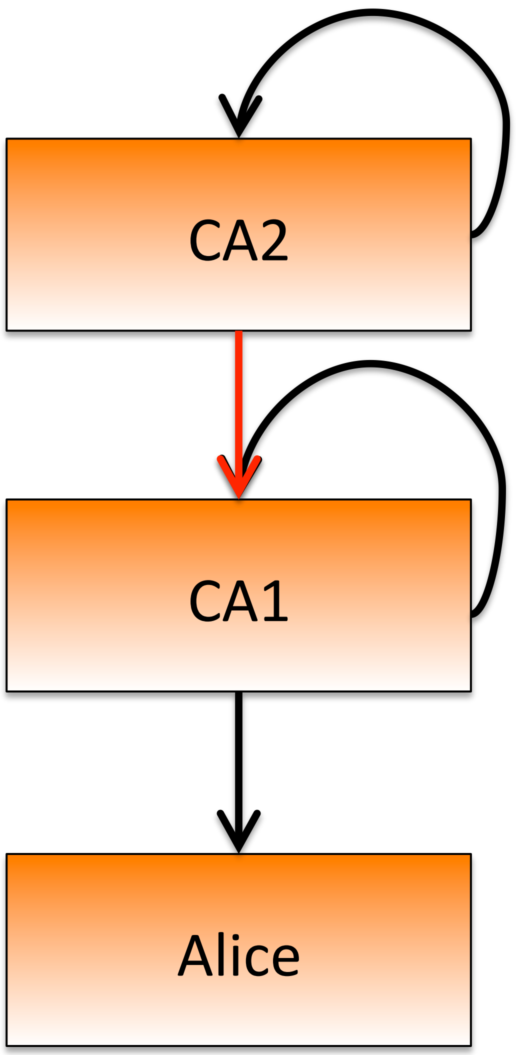


What exactly does this mean?



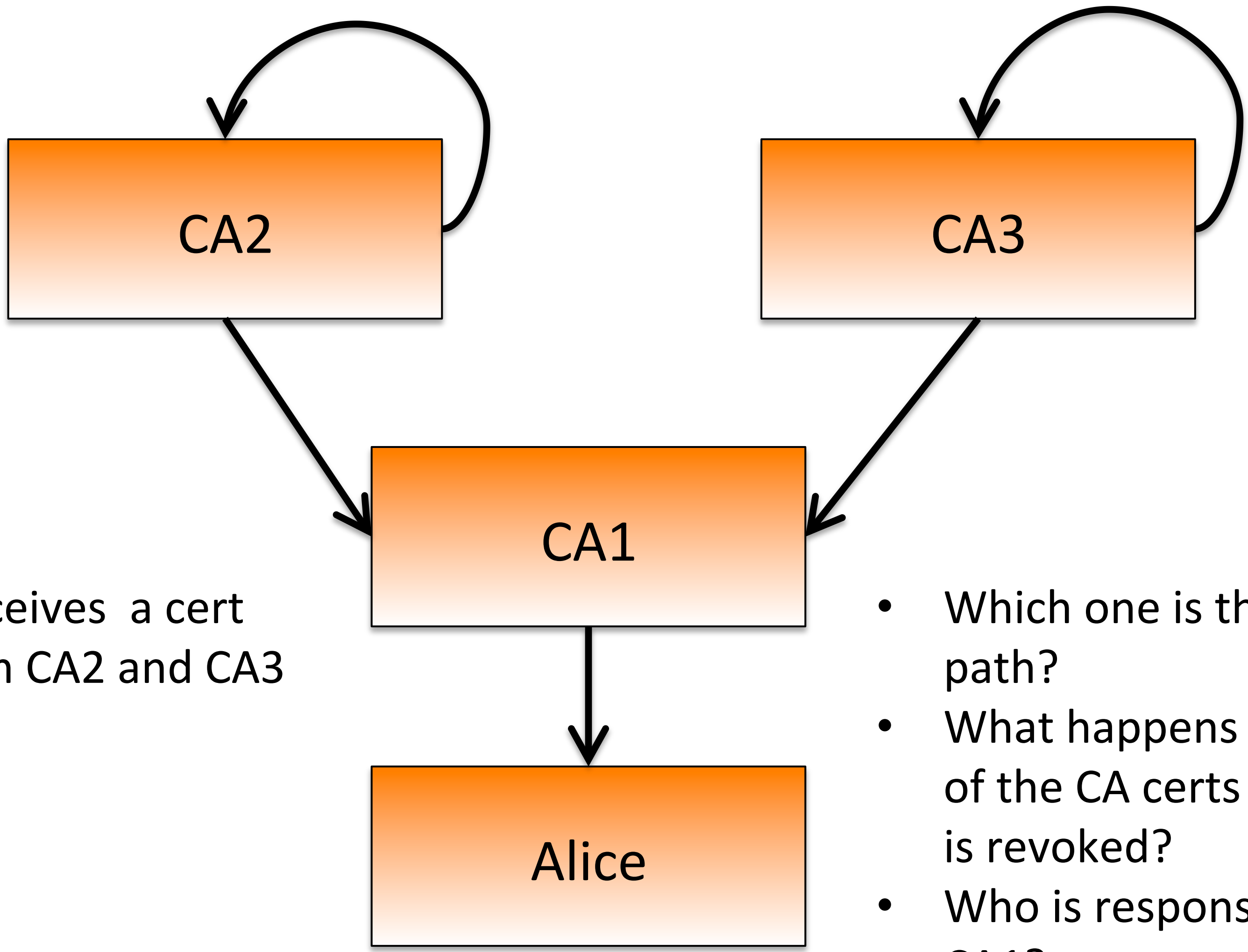
- Does this mean only that CA2 certifies that the given public key belongs to CA1?
- Does CA2 certify that CA1 behaves correctly?
- Does this mean that CA1 follows the same certificate policies as CA2?
- Does this mean that CA1 discloses and follows a given certificate policy?
- Is CA2 responsible for the activity of CA1? Is CA2 fully responsible?

What exactly does this mean? (cont'd)



- There are no clear answers
- Users trusting CA2 will also be accepting certs of CA1 automatically
- There shall 'probably' be some responsibility of CA2 over CA1
- This is rare if CA2 and CA1 belong to two different organizations
- There are few examples, and almost no court cases

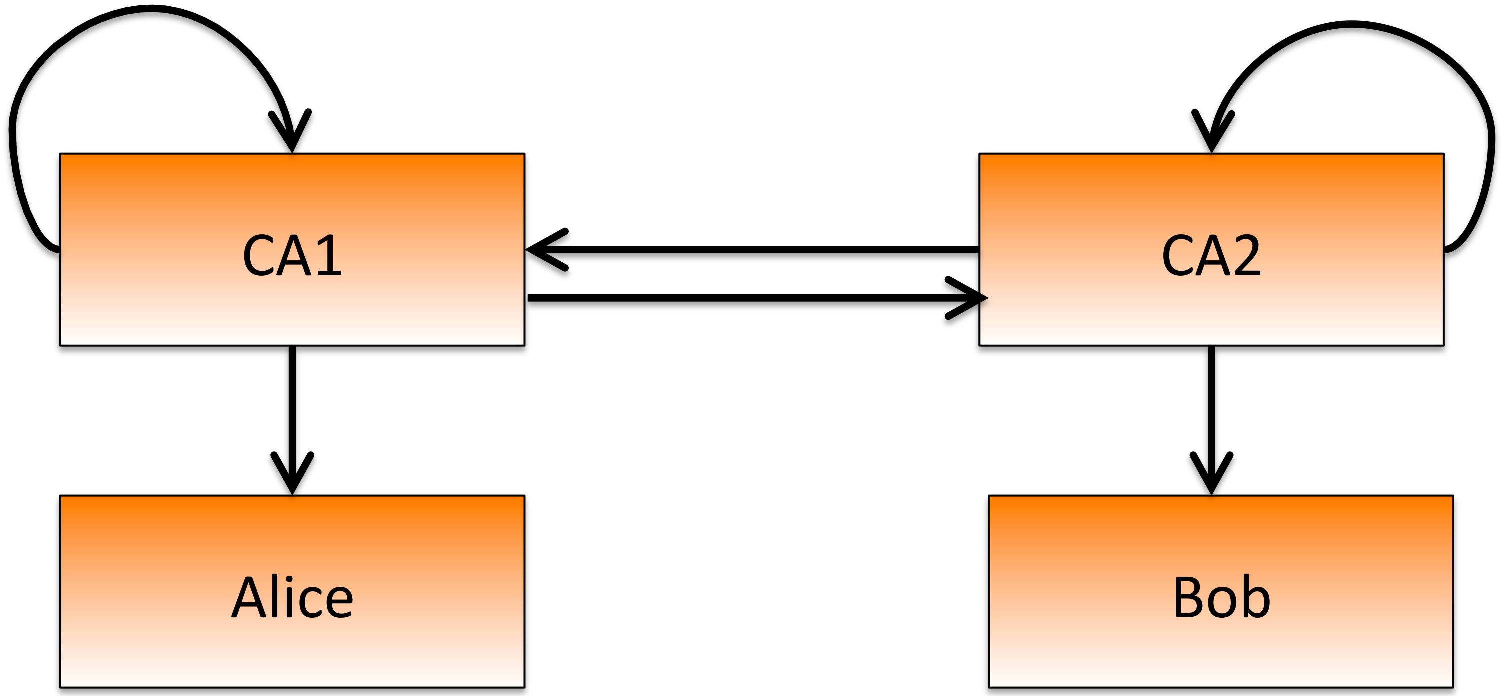
Fork in the certification path



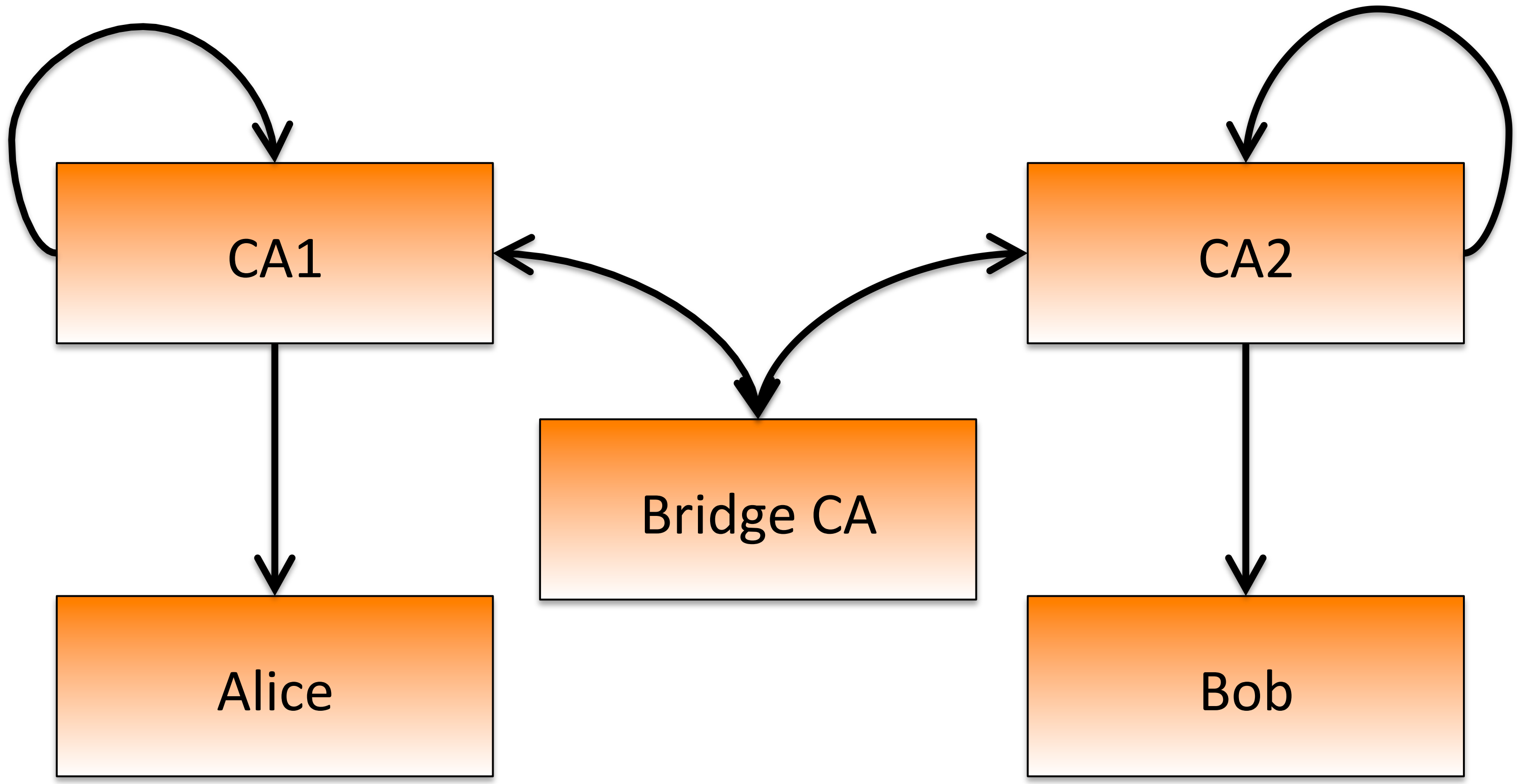
CA1 receives a cert both from CA2 and CA3

- Which one is the ‘real’ path?
- What happens if one of the CA certs of CA1 is revoked?
- Who is responsible for CA1?

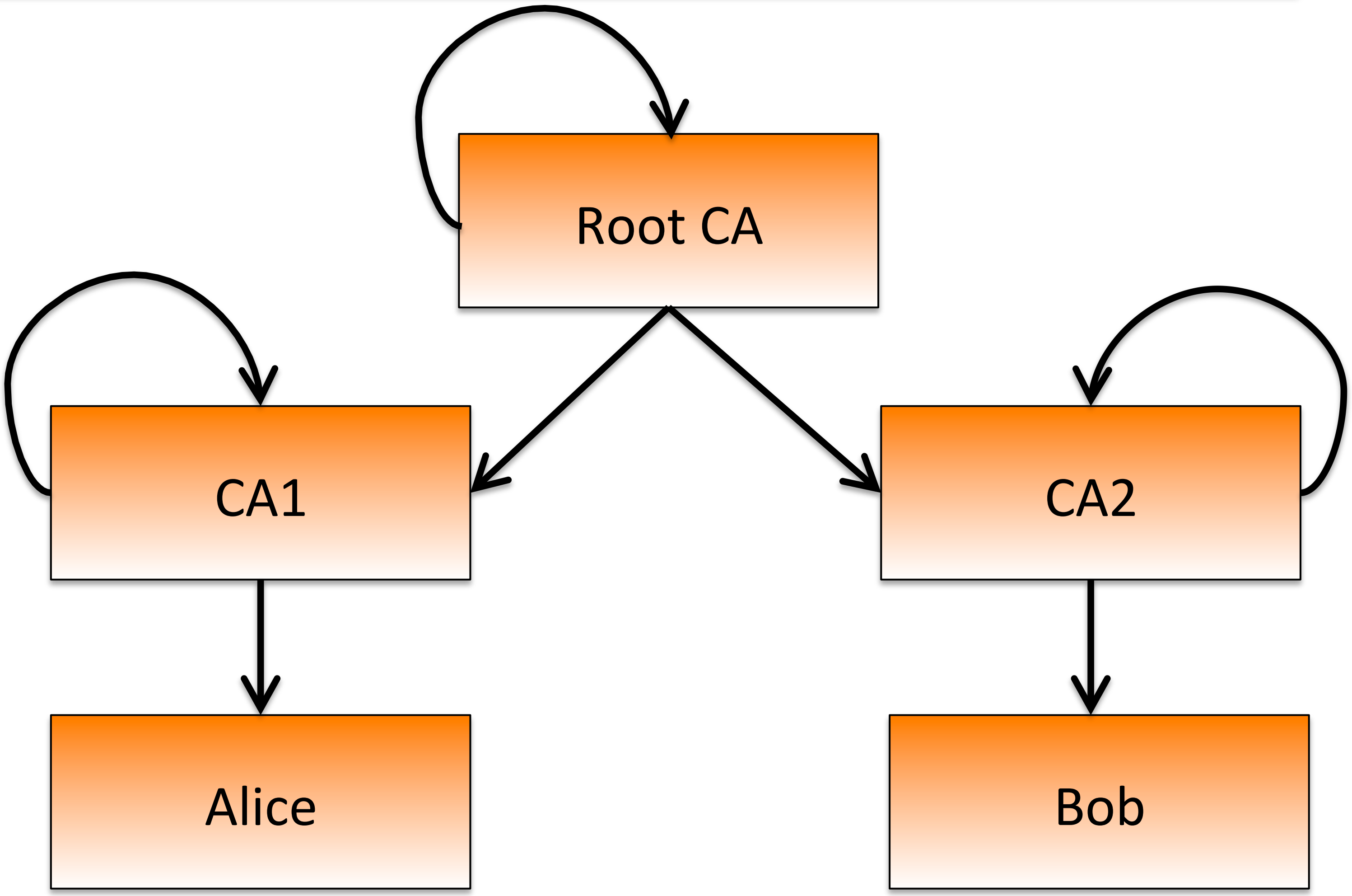
Bi-directional cross-certification



Bi-directional cross-certification



Super Root



Which one is the ‘real’ path for cert X?

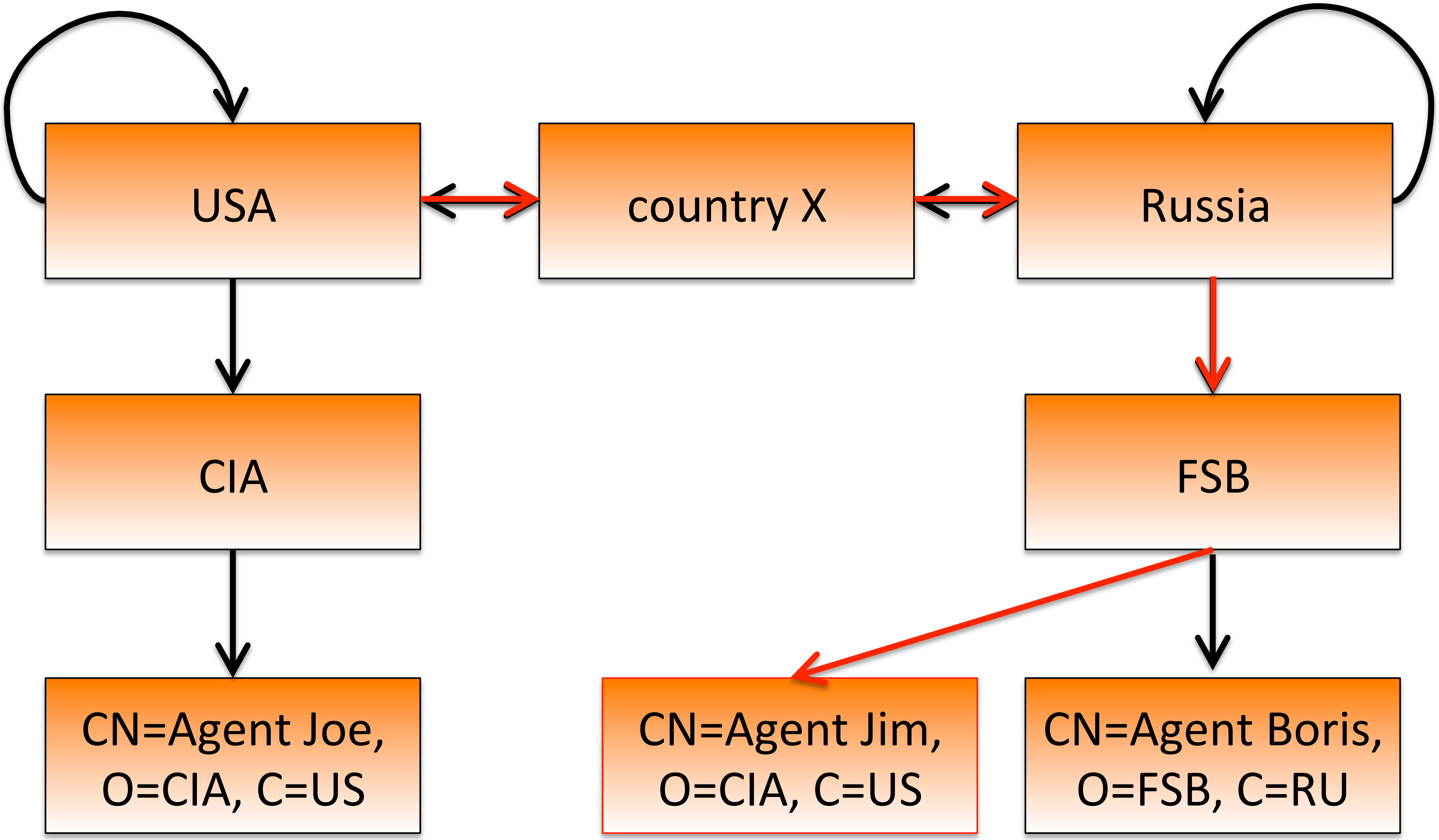
- There is no such thing as a ‘real’ path
- Any validation is performed with respect to a certain policy
- This policy shall contain what roots and what path-building algorithms are acceptable

- **PKI validation is not objective, a cert/signature is valid/invalid with respect to this policy only**

Problems with complex PKI structures

- There is a certification path, but our application does not find it
- Unintended certification path

Unintended certification path


USA
country X
Russia
CIA
FSB
CN=Agent Joe, O=CIA, C=US
CN=Agent Jim, O=CIA, C=US
CN=Agent Boris, O=FSB, C=RU

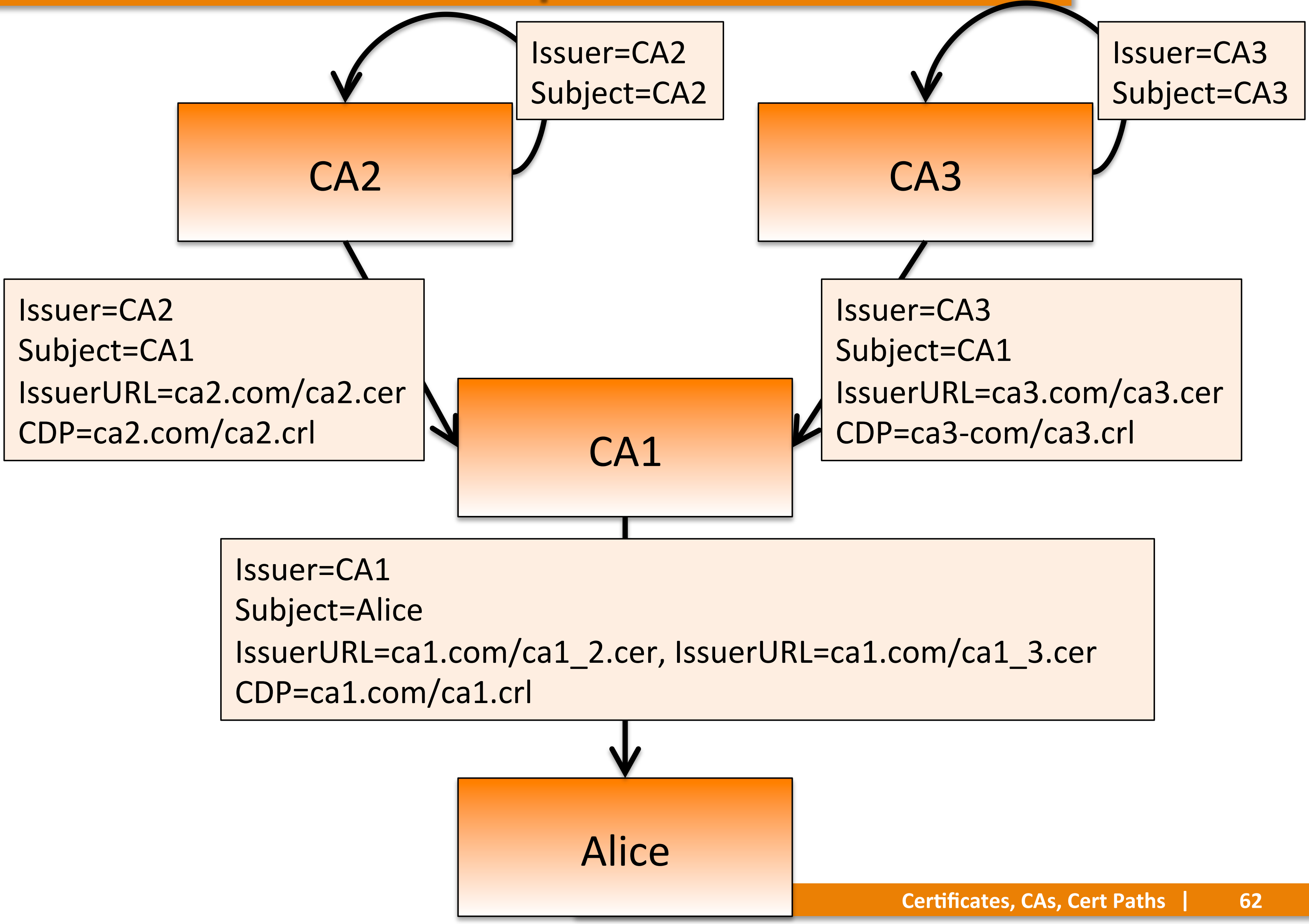
Solutions

- Use certificate policies, verify them
- PolicyConstraints, NamingConstraints
- PathlengthConstraints

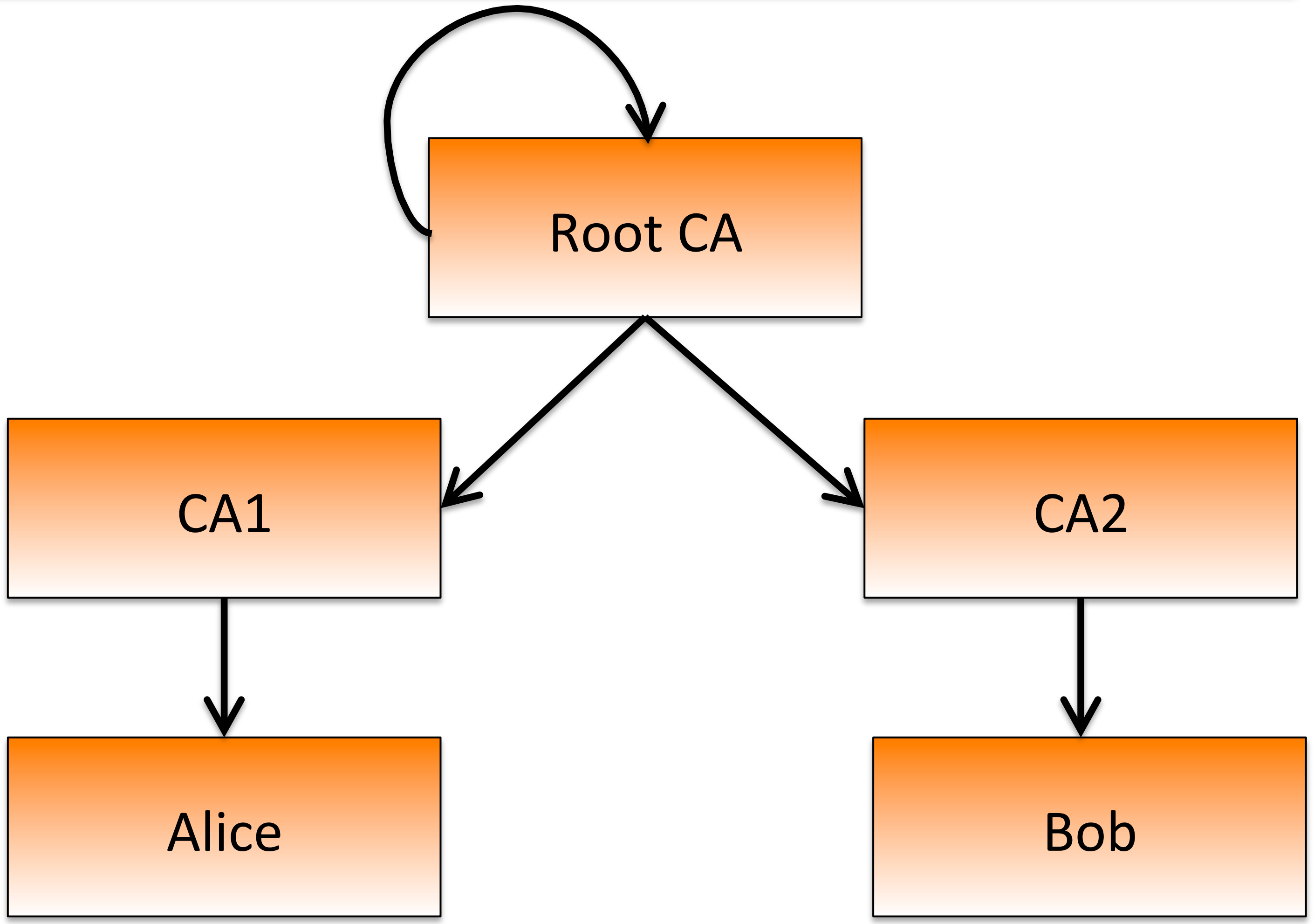
- Use a good application!

- Note: Problems are not on this level yet. A few years ago there was a problem with supporting basicConstraints
- Cross-certification is rare in practice

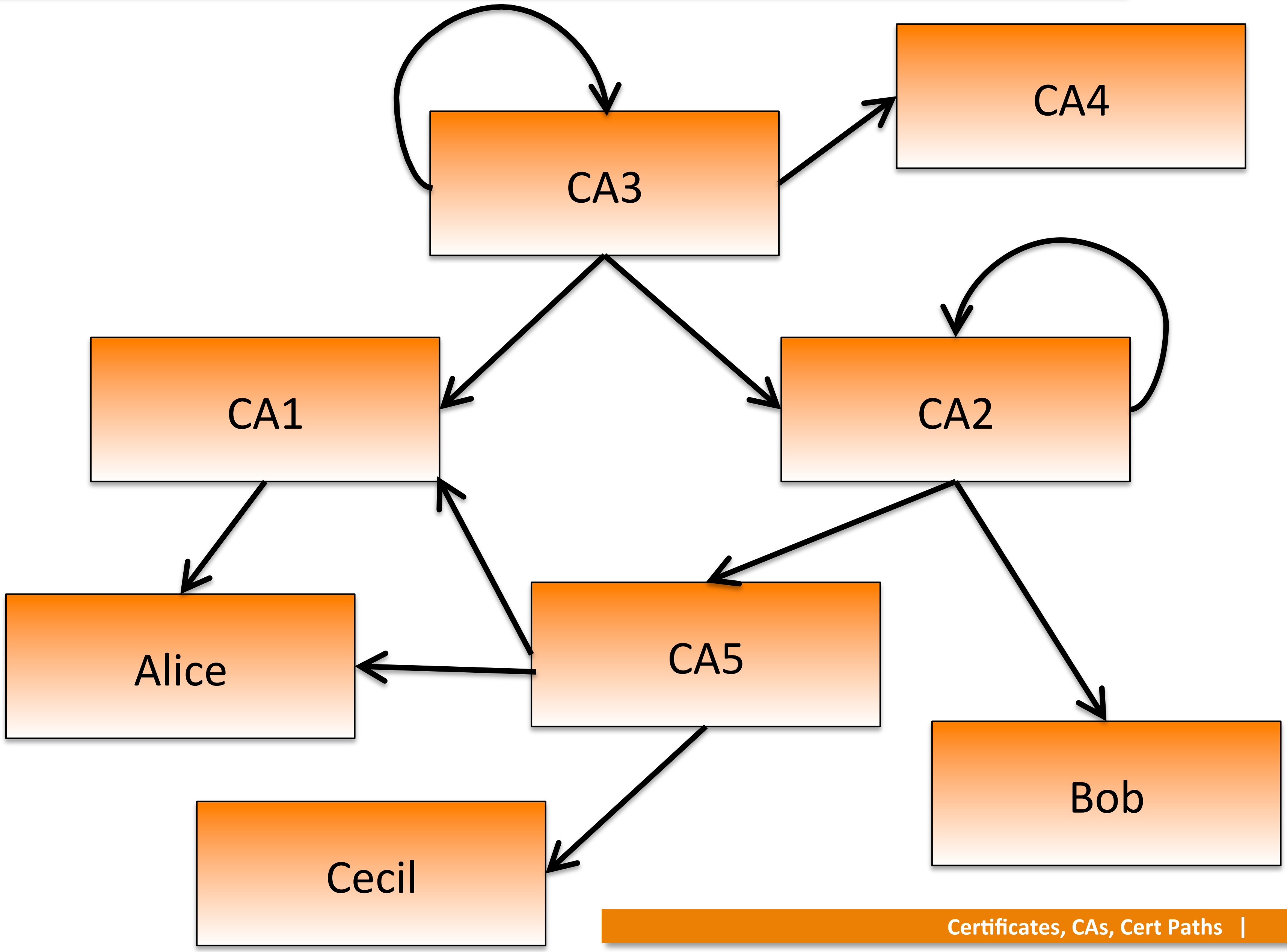
Fork in the certification path – cert contents



Hierarchical PKI



Mesh PKI



Hierarchical vs Mesh PKI

- A strictly hierarchical PKI exists in brand new systems and fairy tales only
- After a while you will end up with a mesh PKI anyway

- Prepare your application for a mesh environment
- Fallback solution: By limiting what intermediate certs an app can see, you can limit what environment it will see

PKI vs PGP

- In PKI certain parties are dedicated to issuing certificates, they are the CAs. End-entities cannot issue certificates (and others do not trust them).
- In PGP all parties can certify the public keys of others, this is not limited to CAs. → Web of trust

- (Of course, PGP uses different data structures…)

Summary

- Certificate: A signed statement from a Certificate Authority that links a public key to a given Subject
- The signature on a certificate can be validated by the CA's certificate, thus chaining up to a trust anchor (root CA); trust anchors can be obtained & validated out-of-band only
- The Certificate Authority registers the Subject, issues the certificate, revokes it when needed
- The CA maintains a secure operating environment in order to ensure the secure usage of its private key and the integrity of its registries

Recommended reading

- RFC 5280 – certificate and CRL format
- RFC 3647 – Certificate Policy framework
- Ten Risks of PKI
- Peter Gutmann's PKI writings:
  - PKI tutorial - Everything you Never Wanted to Know about PKI but were Forced to Find Out
  - Crypto Won't Save You Either – crypto is bypassed, almost never breached
  - X.509 Style Guide – note: scary HP Lovecraft quotations
- The "Certificate Authority" Trust Model for SSL: A Defective Foundation for Encrypted Web Traffic and a Legal Quagmire law paper, strong criticism – slides
- Security Collapse in the HTTPS Market

Note: all of the above a fully optional