DIMACS Security & Cryptography Crash Course, Day 2
Public Key Infrastructure (PKI)

Prof. Amir Herzberg

Computer Science Department, Bar Ilan University

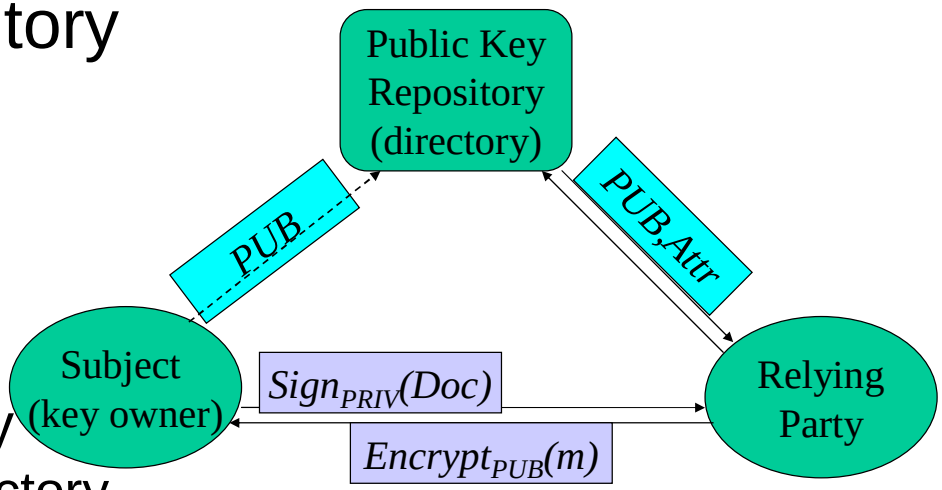
http://amir.herzberg.name

Lecture Outline: Public Key Infrastructure

- Public Key Certificates
- Identity in certificates – different approaches:
  - Must contain unique identifier(s) – X.509 Distinguished Names
  - Use (also) certificates with no identifier or non-global identifiers
- X.509 Public Key Certificates
- Certificate authorities, hierarchies and cross certification
- Certificate issuing and registration authorities
- Certificate (Path) Validation
- Certificate Revocation
- Conclusions

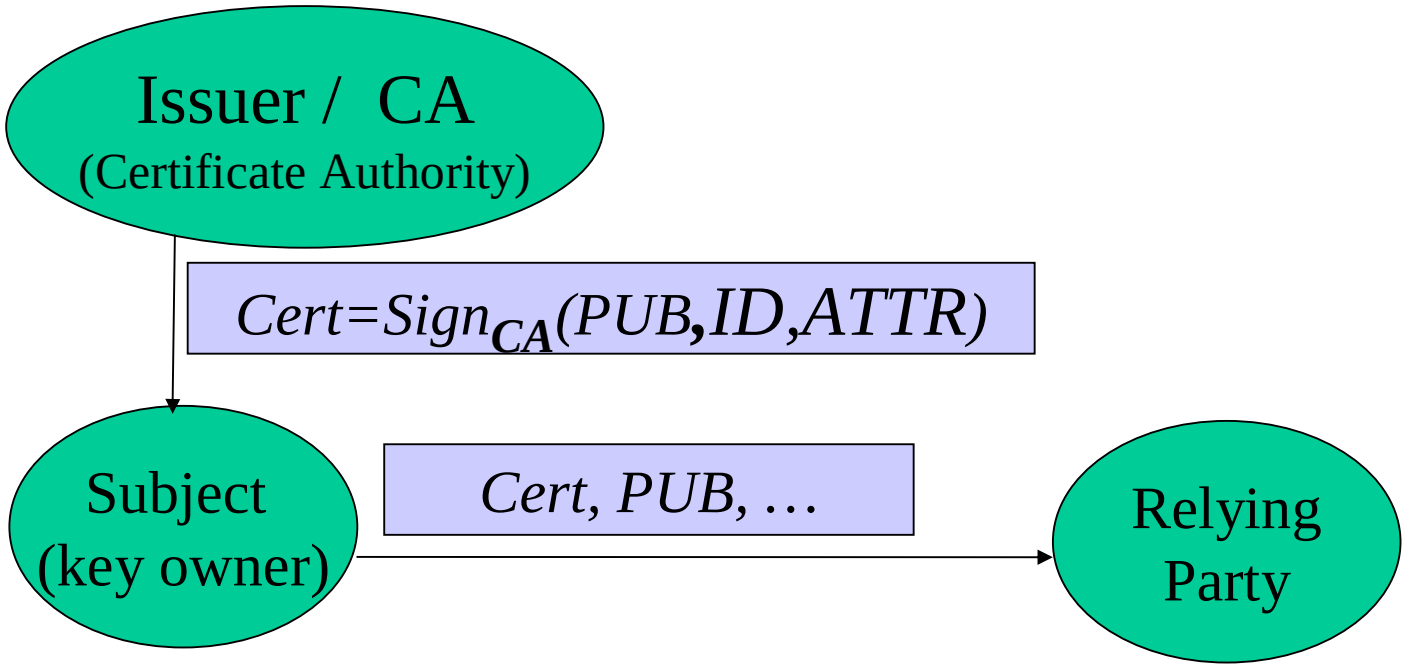
Relying on Public Keys

- Public keys are very useful
  - Encrypting data and keys
  - Signing documents
- How do we know the public key?
- Initial approach: public keys will be registered in (e.g. X.500) directory/repository
  - Trusted, centralized
  - Subject uniquely identified
  - Directory matches subject to public key
  - Public key authenticated btw directory and relying party
    - Using MAC or signed by directory
  - Possibly other attributes in directory



Public Key Certificates

- *Issuer* (or *Certification Authority – CA*) signs *certificate* binding public key to ID, attributes
- Issuer/CA *identifies subject* and/or her *attributes*
- *Relying party* validates *certificate* signed by *issuer*
- What are the attributes?
  - Should help the relying party decide on subject
  - Issuer (CA) should be able to validate them (liability!)



What are Certificate Attributes?

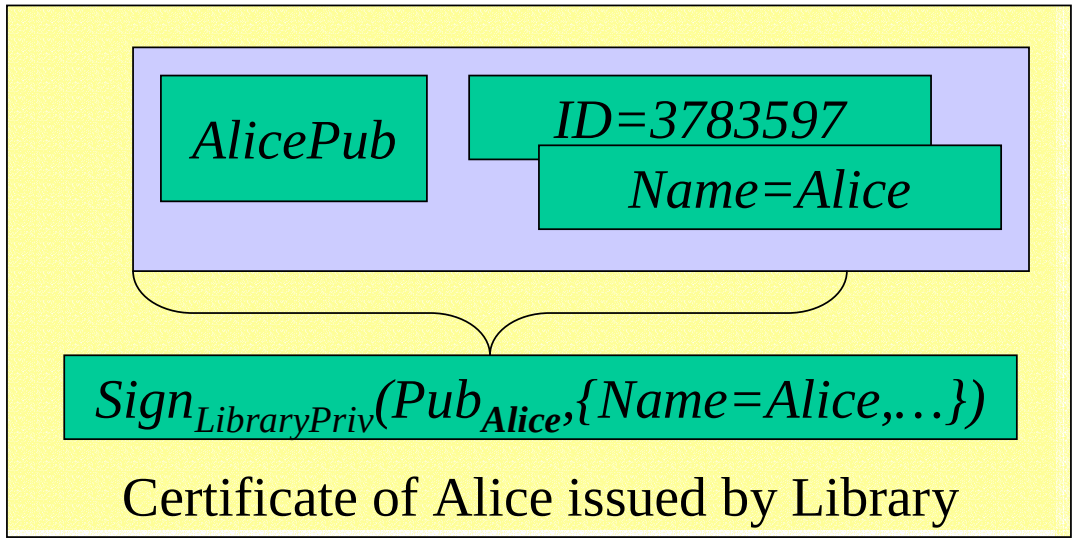
- Let *AttrNames* be the set of *attribute names*
- Let *Values* be the set of *attribute values*
- Let *A=AttrNames* x *Values* be the set of *attributes*
  - Each attribute is a pair <*n* ∈ *AttrNames, v* ∈ *Values*>
  - These are called `name-value pairs`.
- Attributes can be :
  - *Identifiers* (e.g. <email,jon@tau.il>; <ID,5724567>)
  - Other properties of *subject* (e.g. <grade,A>;<job,cop>)
  - Properties of the *certificate* (e.g. <valid,1-6/2001>)

What is a Public Key Certificate?

- Let *K* be the set of (public) keys
- A *public key certificate (PKC)* is a 4-tuple: $<Issuer_{pub}, Subject_{pub}, Attrs, Sign>$, where:
  - $Issuer_{pub}$, $Subject_{pub}$ are public keys
  - $Attrs \in A^{+}$ (Attributes; WLOG assume non-empty)
  - *Sign* is a signature using $Issuer_{priv}$ over $Subject_{pub}$ and *Attrs*; namely…
  - The certificate is *valid* if $Valid_{IssuerPub}(Sign,\{Subject_{pub}, Attrs\})$

Example: library certificate and card

- Card allows identification of Alice in person – linking to her record (e.g. by ID)
- Certificate allows validation of requests from Alice via network (signed with her public key)
- This is an *Identity PKC (Public Key Certificate)*
  - Natural; similar to Identity cards, passport, etc.



Identity Public Key Certificates

- *Identity PKC:* contains identifier(s)
  - Like most traditional certificates/credentials
- *Establish trust and reputation* using recognized or reviewed identities (of corporations and individuals)
- *Allocate liability and penalize undesirable actions* by identifying, suing and blacklisting the signer
  - Typically: identifier has legal or commercial meaning (off-net) – use existing (off-net) legal/reputation mech
  - *Distinguished name:* X.509 term for unique, well defined, legally binding identifier

X.500, X.509 and Distinguished Names

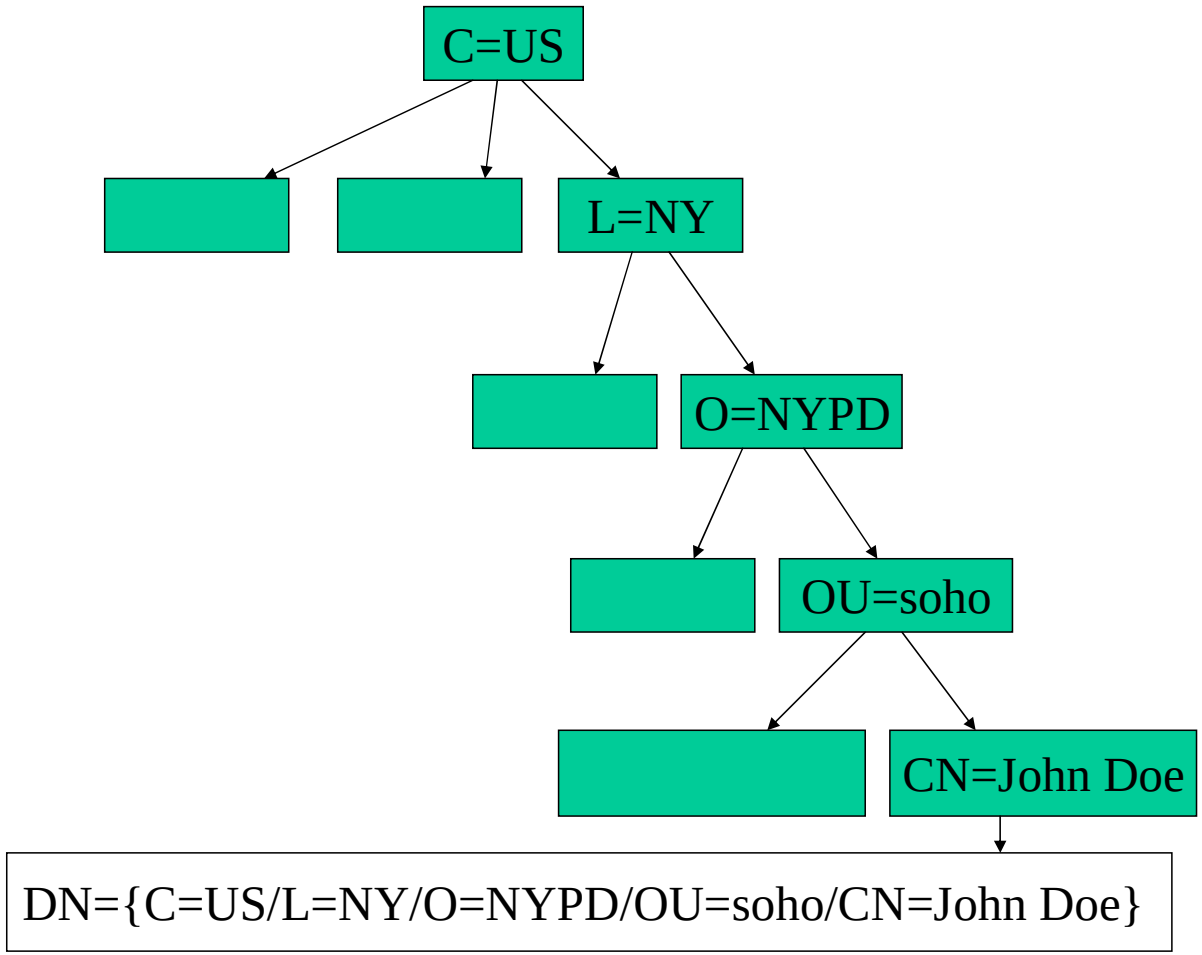
- X.500: ITU's recommendation (standard) for global, distributed, trusted, on-line directory (phone book)
  - Unique identifier: *Distinguished Name (DN)*
  - Operated by hierarchy of trustworthy directories
  - Never happened – too complex, too revealing
  - Different attributes, including public key
- X.509: authentication related to X.500
  - Initially: Authenticate entity to Directory (PW, Pub key)
    - To maintain entity's record
  - Identity certificates binding public key, name (DN)
  - Established: IETF PKIX, SSL, PGP, S/MIME, IP-Sec, …

Distinguished Names (DN)

- Single, globally unique names that everyone could use when referring to an entity – legally meaningful.
- Ordered sequence of pre-defined keywords, and a string value for each of them.
- Distributed directory, responsibility →*hierarchical DN*

Keyword	Meaning
C	Country
L	Locality name
O	Organization name
OU	Organization Unit name
CN	Common Name

Distinguished Name Hierarchy



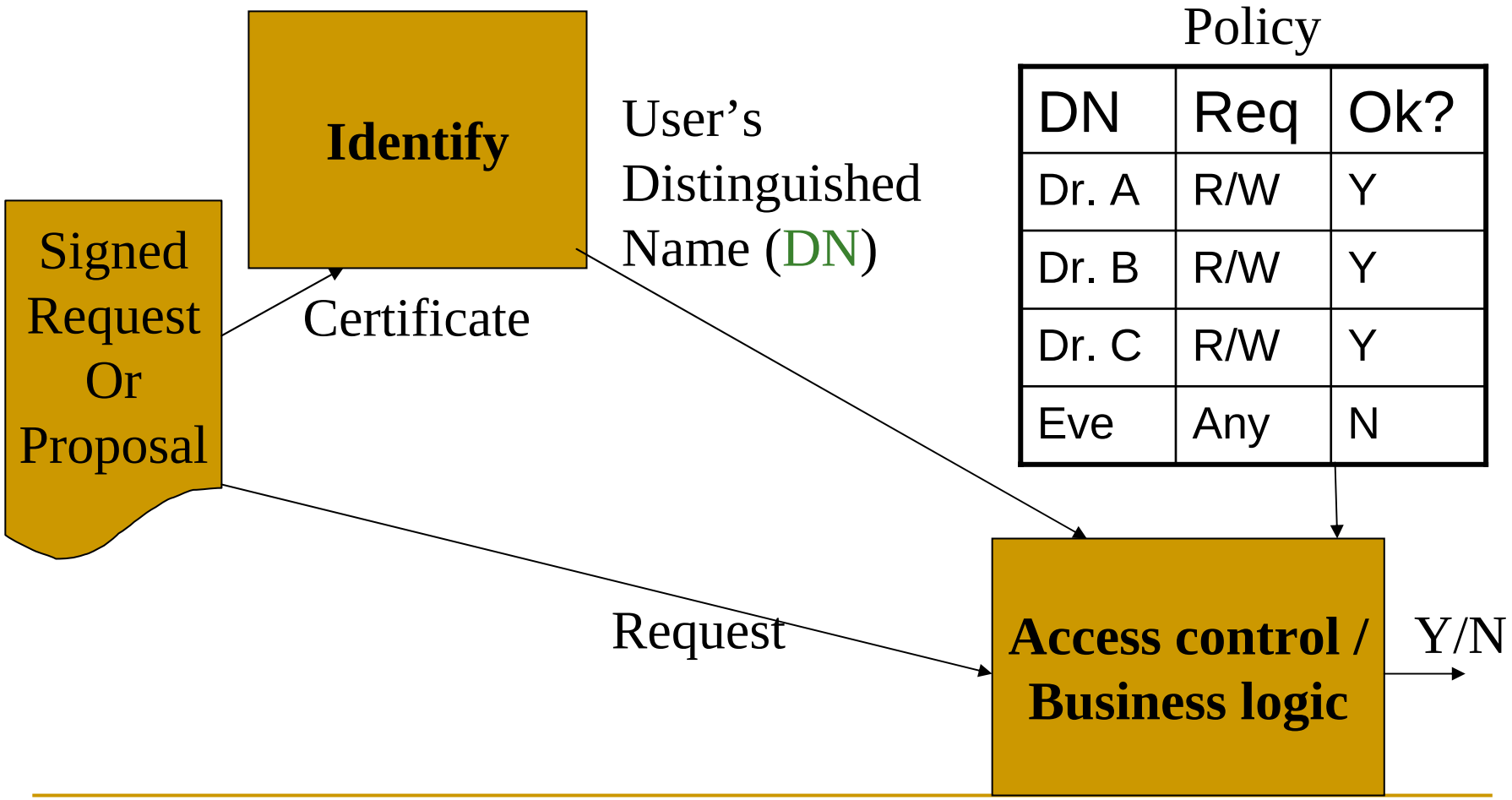
Distinguished Names - Problems

- Goal: unambiguous, unique, legal binding identification
- Names are not unambiguous; other identifiers (e.g. serial number, SSN) are not universally understood.
- Same entity can get multiple Distinguished Names.
- Providing & validating details is expensive & intrusive.
- Distinguished Name fields may expose
  - Organizational sensitive information (e.g. position)
  - Privacy, possibly allowing identity theft
- DN keywords hierarchy not well defined
- People are mobile; should your DN change when you change work? Should your public key?

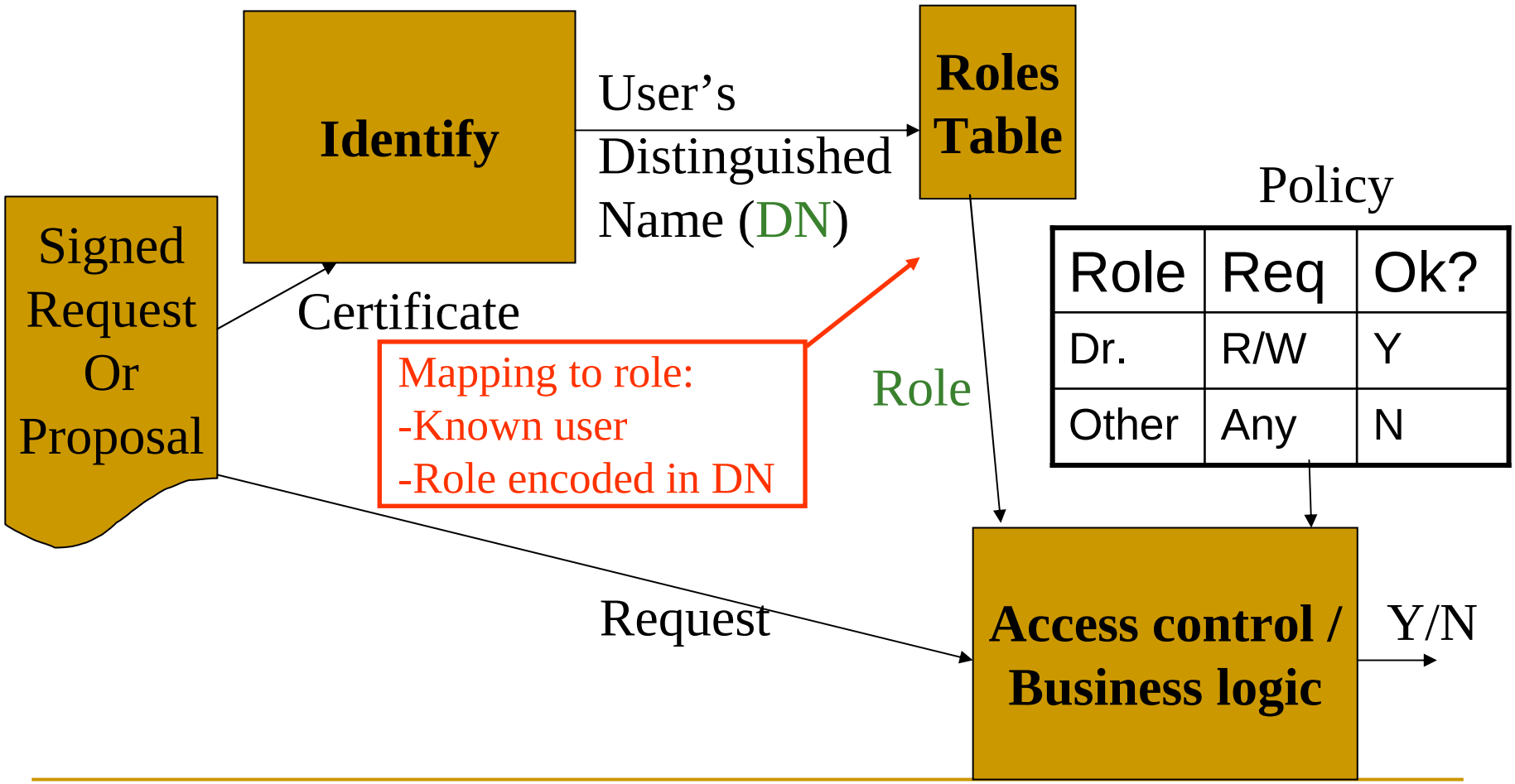
Distinguished Names – in Practice

- Legally acceptable identifiers in some countries.
- To ensure uniqueness, issuers often place a random string, serial number as part of the DN.
- As of Version 2, X.509 certificates contain additional `unique identifiers` for the subject and issuer
- As of Version 3, X.509 certificates allow general extensions, that are often used to add identifiers

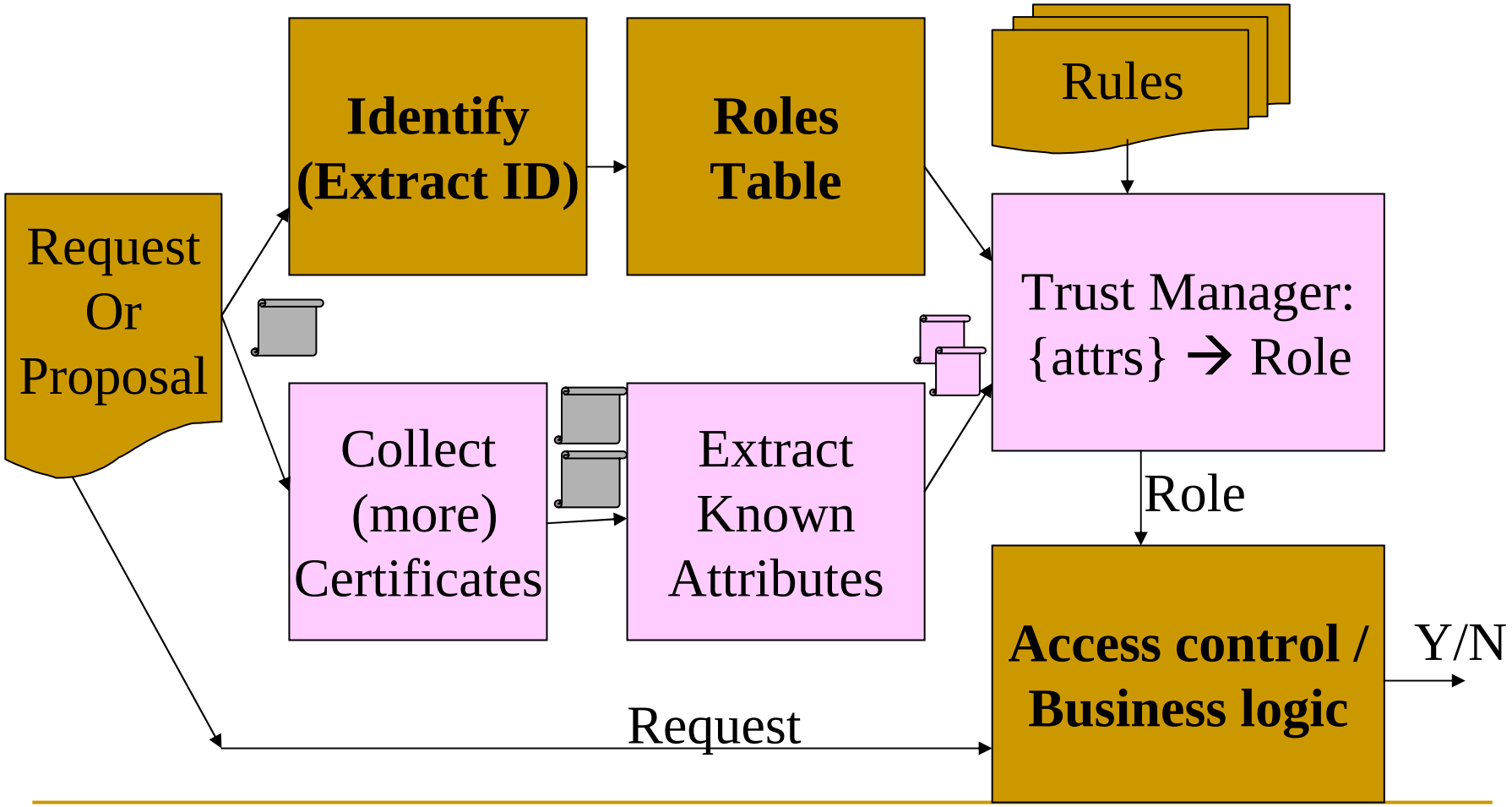
Relying Party Use of (X.509) Identity Certificates: Identity-based Access Control



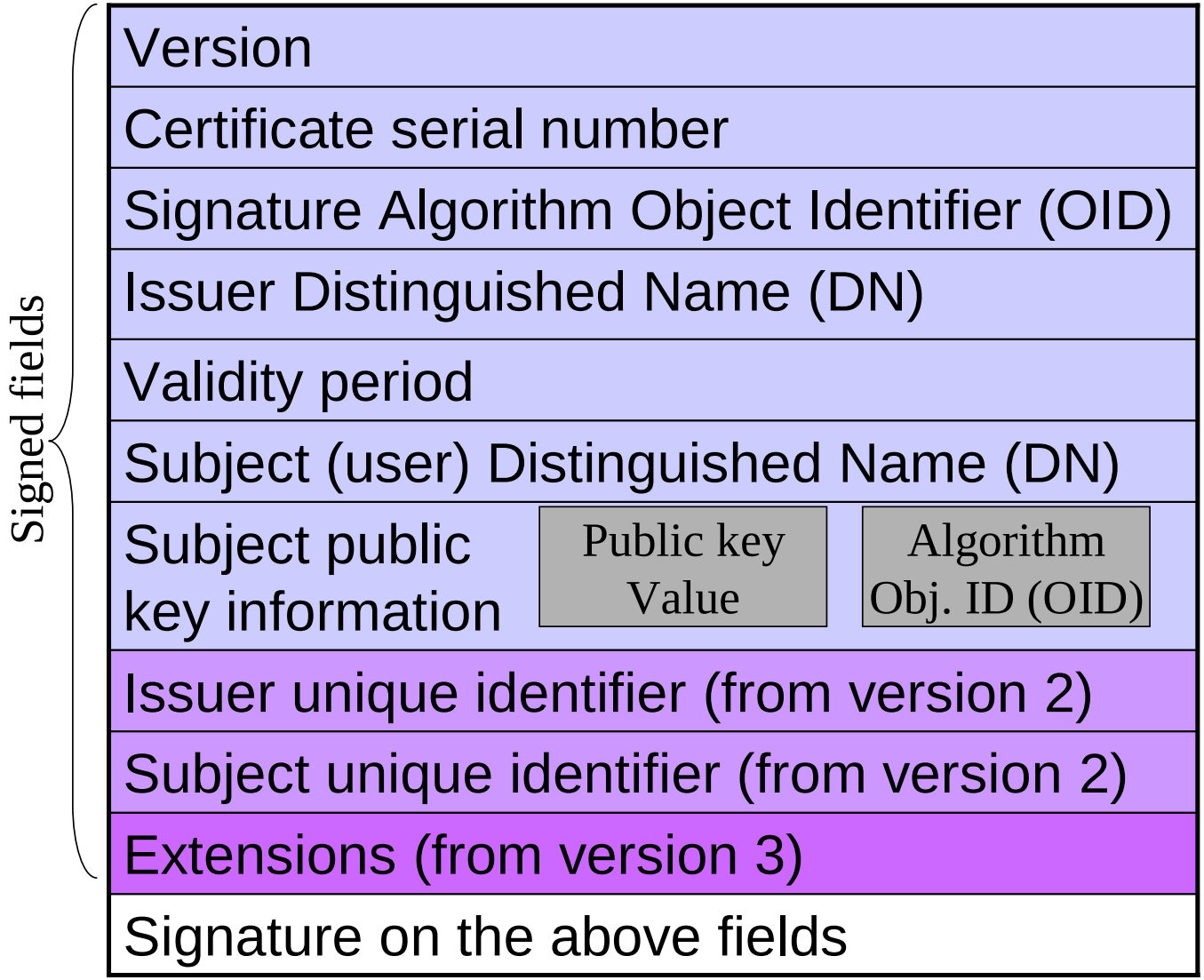
Relying Party Use of (X.509) Identity Certificates: Role-based Access Control



Using also non-identity certificates…



X.509 Public Key Certificates



Object Identifiers (OID)

- From Abstract Syntax Notation (ASN.1) standard
- Global, unique identifiers, e.g. for algorithms
- Sequence of numbers, e.g.: 1.16.840.1.45.33
- Top level numbers: 0 – ITU, 1 – ISO, 2 – joint
- Each organization assigns lower-level identifiers
- X.509 use: identify algorithms and extensions.

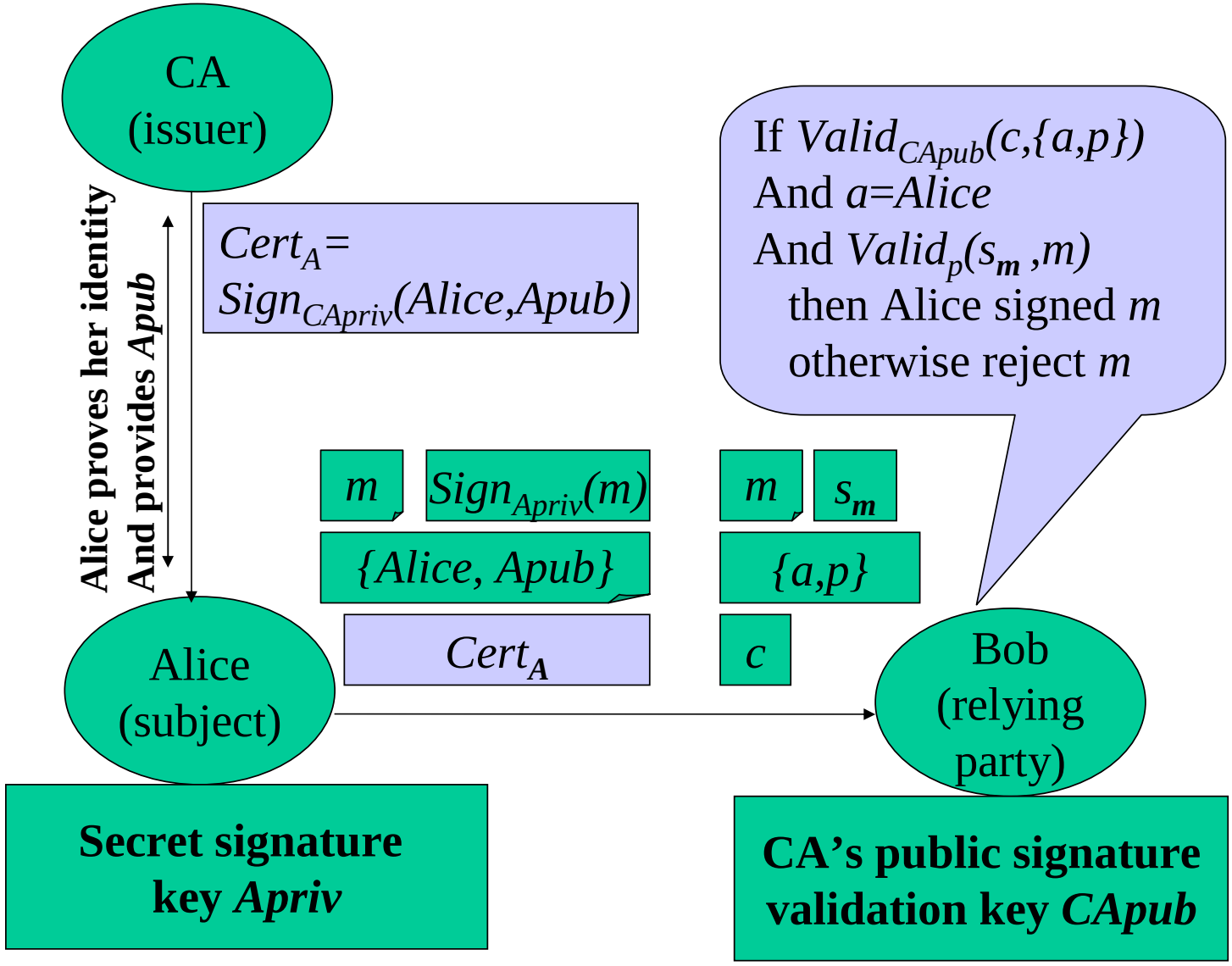
X.509 Extensions Mechanism

- Used for certificates and Certificate Revocation Lists (CRL)
- Each extension contains:
  - Extension identifier (OID)
  - Criticality indicator
    - If critical, relying parties MUST understand extension to use certificate
    - If non-critical, Ok to use certificate anyway
  - Extension value

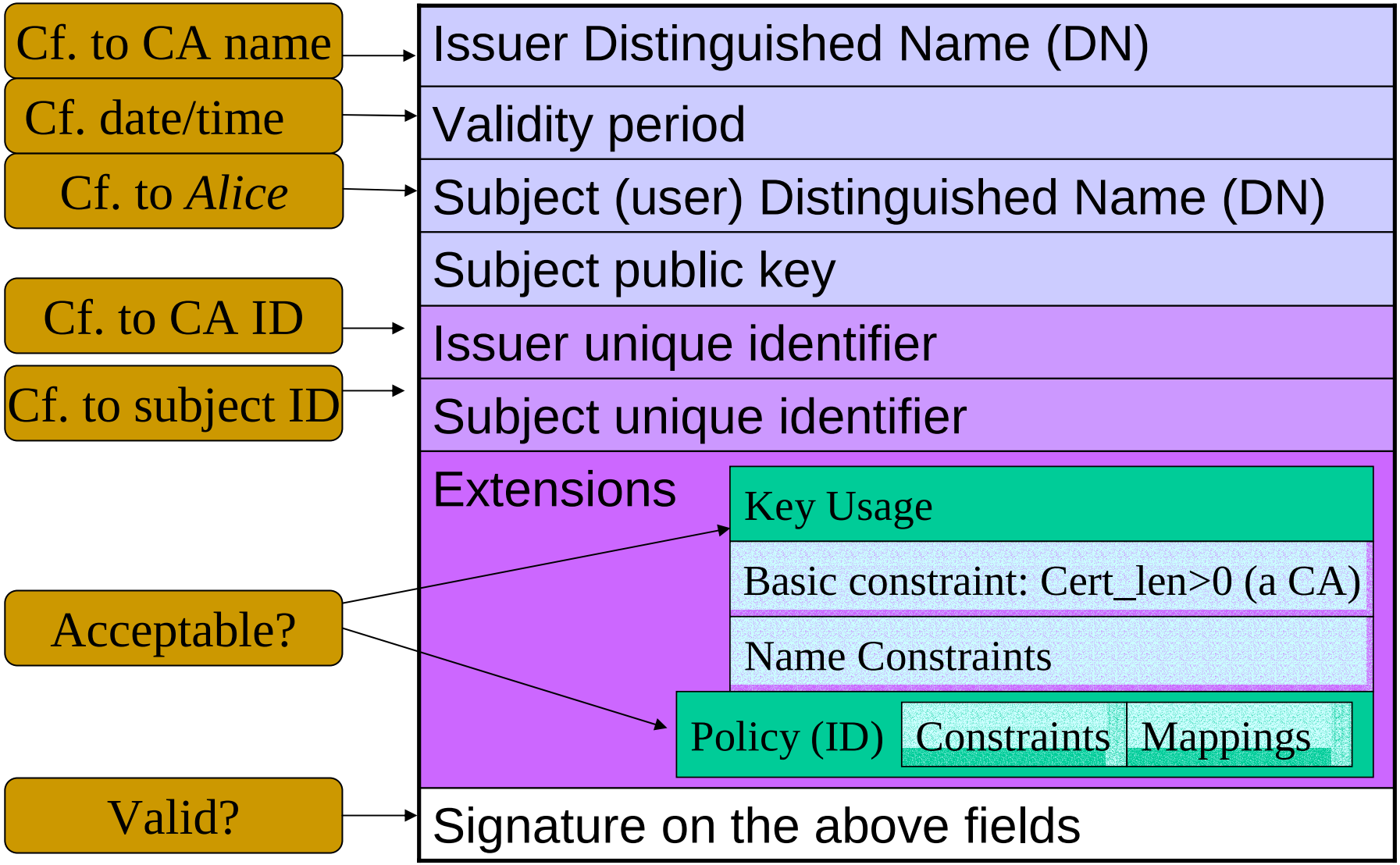
X.509v3 Standard Extensions

- Key identifier and usage (signing, encryption, etc.)
- Subject and issuer alternative names (e.g. e-mail)
  - E.g.: subject dNSName in subjectAltName extension
- Certificate policy identifier and qualifiers
  - What is the policy of the CA (and disclaimers)
- Certification path constraints
  - Basic constraint: CA or end entity, path length
  - Name and policy constraints (on certs issued by subject)
- Policy mappings
  - How to interpret attributes in certificates issued by subject (if CA)
- Certificate Revocation List (CRL) extensions
  - More on revocation later…

Certification Authority (CA)

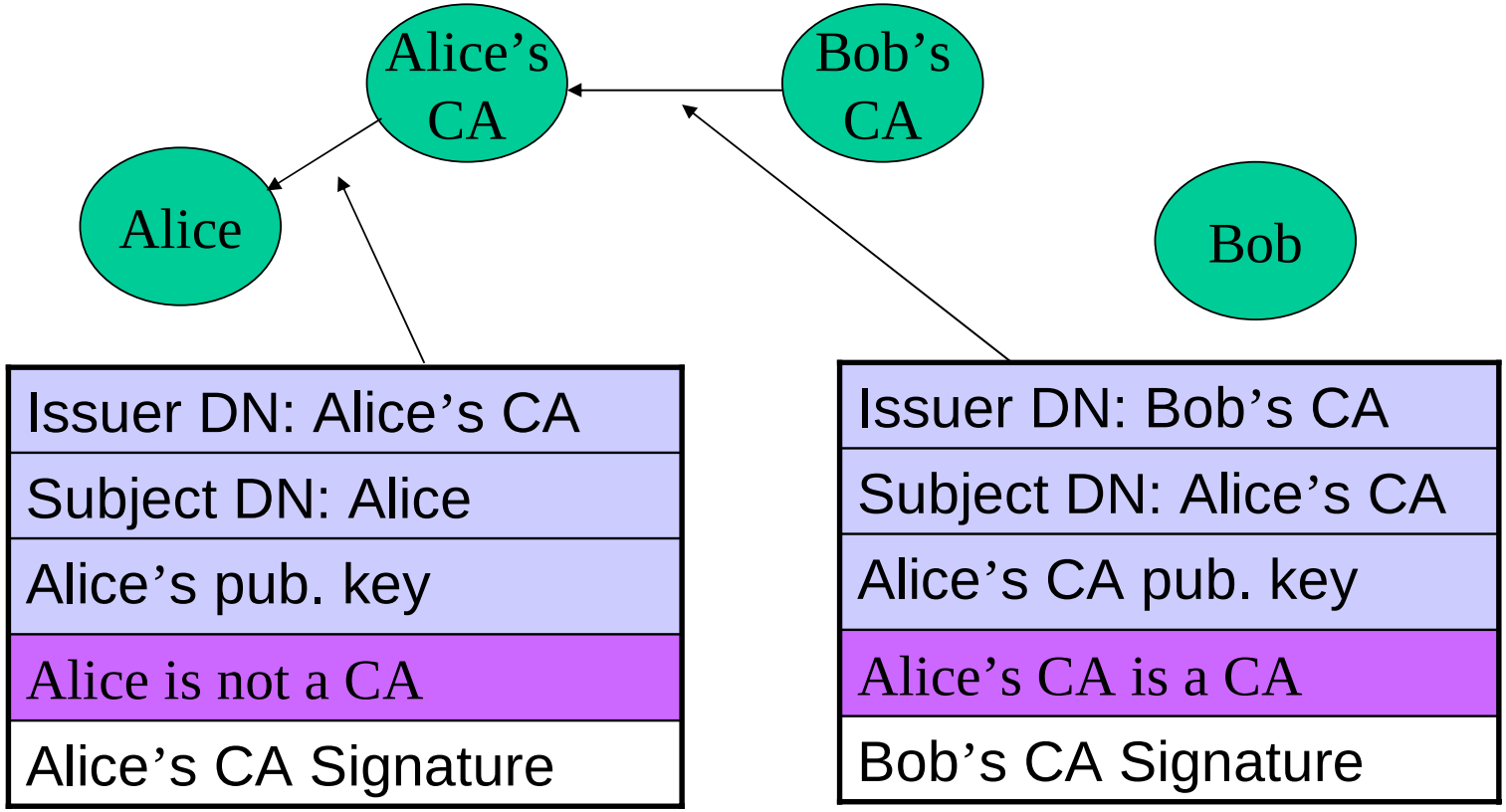


Certificate Validation $Valid_{CApub}(c,\{a,p\})$

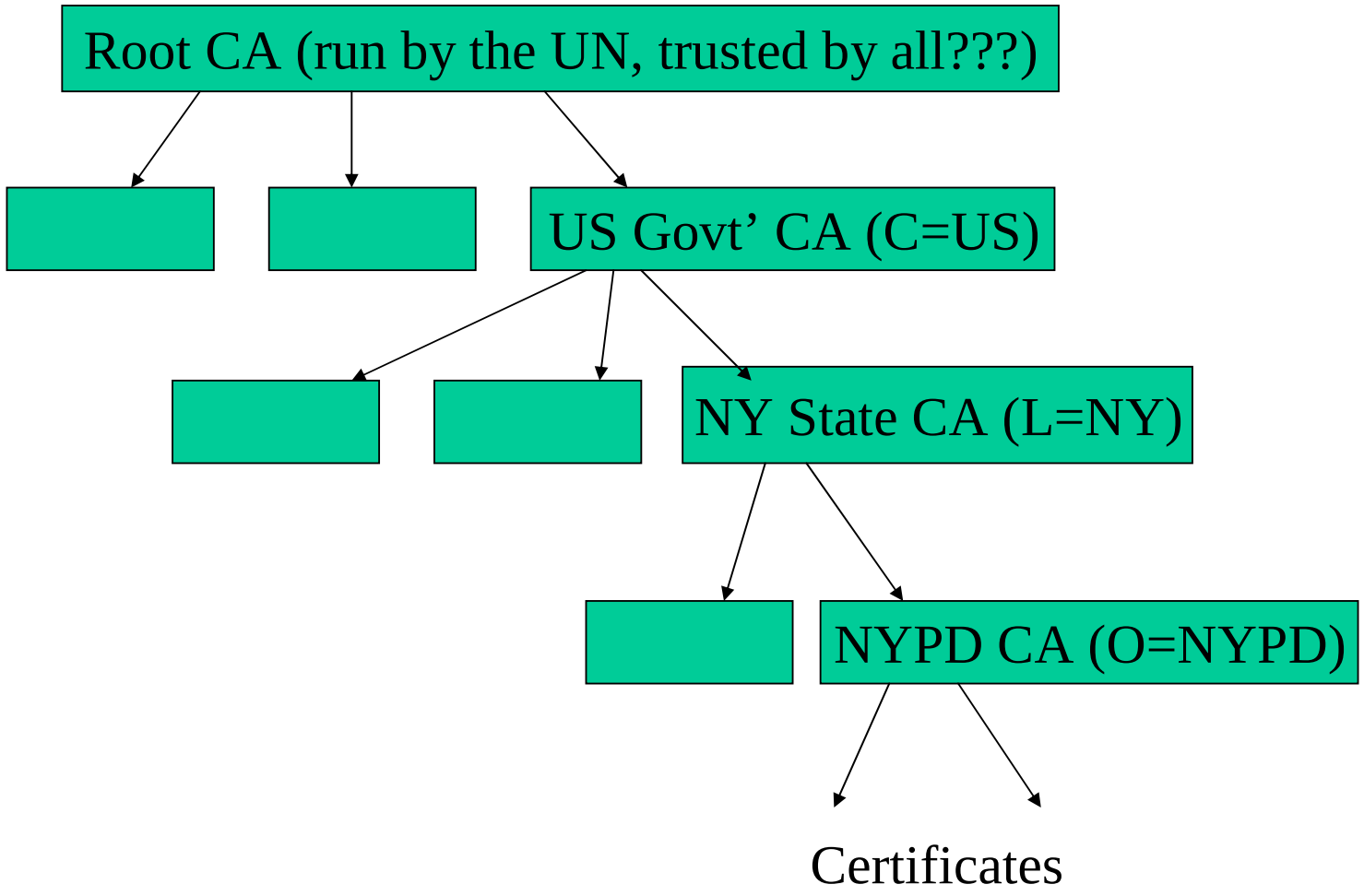


Certificate Path

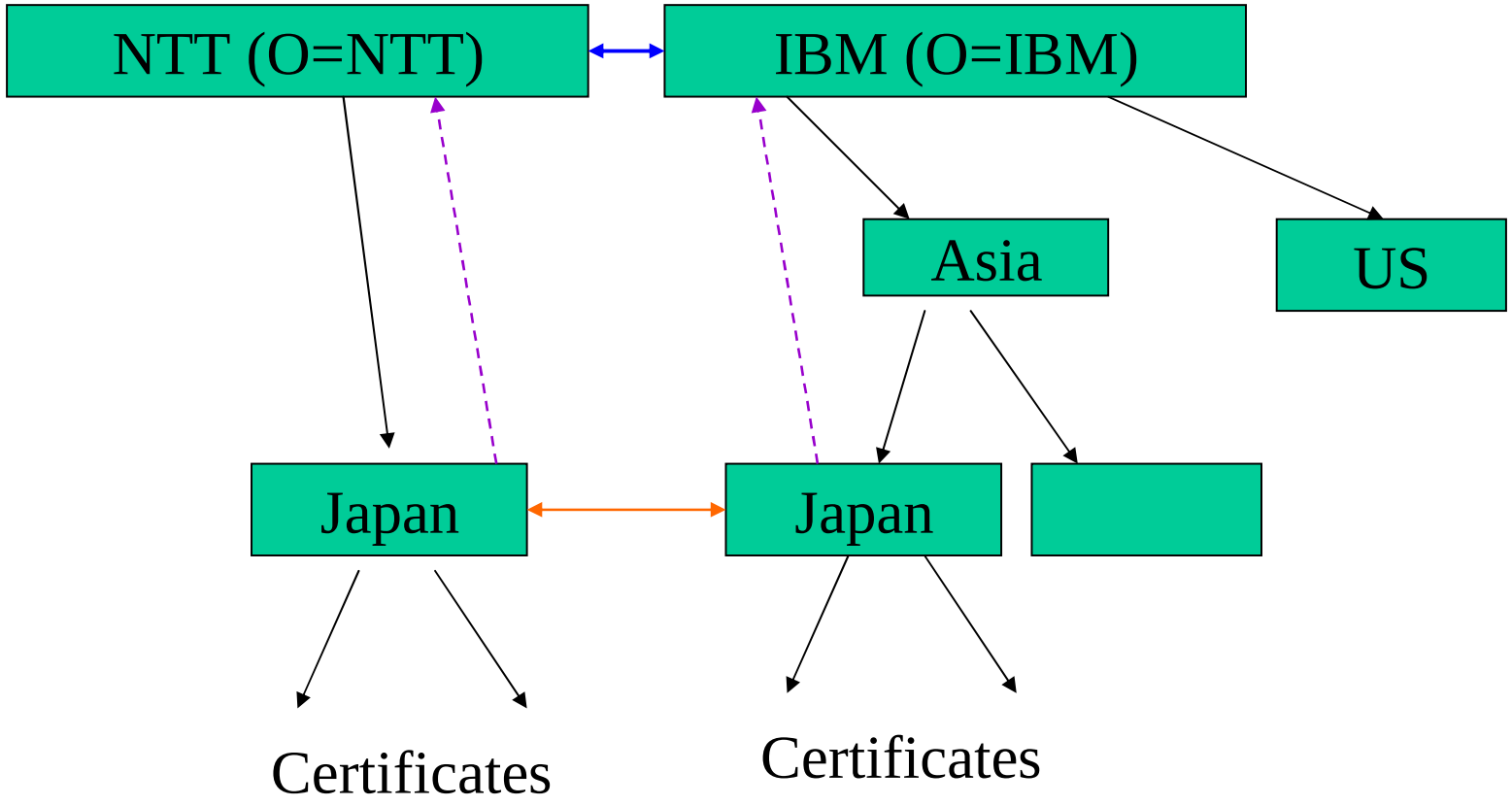
- What if Bob does not know Alice's CA?
- Solution: Certificate Path – a CA known and trusted by Bob certifies Alice's CA



Global X.509 CA Hierarchy



X.509 Cross Certification



Certificate Path Validation

- By relying party or a *trusted path validation service*
- Local validation of each certificate
  - Validity periods, key usage, revocations.
- Verify chain of distinguished names and identifiers
- Verify each certificate:
  - Signed by previous public key
  - Certificate path below all `basic constraint` length limits
  - Verify name constraints (permissible name space)
  - Perform any policy mapping and verify policy constraints

Certificate Path Discovery

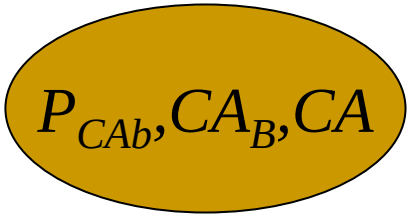
- Offline problem: given set of (locally valid) public key certificates, is there a valid certificate path to Alice's certificate?
- Online problem: same, but collect more certificates as needed.

Offline Certificate Path Discovery

- Given set of PKCs, is there a valid certificate path to Alice's certificate?
- Recall: A *public key certificate (PKC)* is a 4-tuple: <*IssuerPK, SubjectPK, Attrs, Sign*>
- Simplify:
  - All PKCs locally valid
  - No path length constraints
  - Name, policy constraints – none/trivial/ignored
  - Only remaining relevant attributes are:
    - DN = Distinguished Name
    - CA = Y if this is a CA, N if not a CA
- Defines a graph…

Certificate Path Graph

- Vertices *V:* <*pub_key*, *DN*, *CA_flag*>
  - *CA_flag=CA* for a CA, *N* – just end-entity
- Edges *E:* connect from <*p*,*n*,*CA*> to <*p'*,*n'*,*f*> if there is a certificate:Signed by *p,* issuer DN = *n,* subject DN = *n',* subject PK = *p',* CA flag = *f*
- Example: Bob initial graph contains only the public key and Distinguished Name of CA_B, the CA Bob trusts:

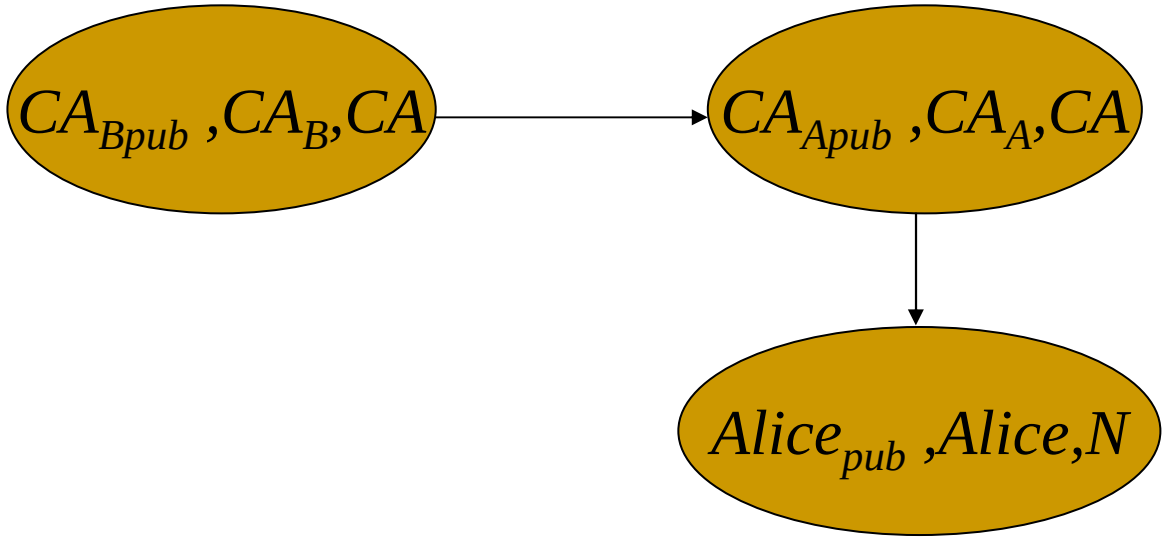


Certificate Path Graph

- Vertices *V: <pub_key, DN, CA_flag>*
  - *CA_flag=1* for a CA, 0 – just end-entity
- Edges *E:* connect from *<p,n,CA>* to *<p',n',f>* if there is a certificate:Signed by *p,* issuer DN = *n,* subject DN = *n',* subject PK = *p',* CA flag = *f*
- Example: After Bob receives also cross-certificate signed by his CA for Alice's CA, with properties: *{DN= CA_A, CA=Y}*:

Certificate Path Graph

- Vertices *V: <pub_key, DN, CA_flag>*
  - *CA_flag=1* for a CA, 0 – just end-entity
- Edges *E:* connect from *<p,n,1>* to *<p',n',f>* if there is a certificate:Signed by *p,* issuer DN = *n,* subject DN = *n',* subject PK = *p',* CA flag = *f*
- Example: After Bob receives also the certificate from Alice's CA to Alice, with properties *{DN= Alice, CA=N}* :



Certificate Path Graph

- Vertices *V: <pub_key, DN, CA_flag>*
  - *CA_flag=CA* for a CA, N – just end-entity
- Edges *E:* connect from *<p,n,CA>* to *<p',n',f>* if there is a certificate:
  - Signed by public key *p*
  - With issuer DN = *n*
  - With subject DN = *n'*
  - With subject PK = *p'*
  - With CA flag = *f*
- Question: let *V'⊆V* be trusted CA's. Is there a path from a vertex in *V'* to *<p,n,f>?* Find shortest path!
- Answer: use BFS; work=*O(|E|).*

Observations

- Length constraint usually not used / ignored
  - If used, need to be reflected in graph → more complex
  - Relying party should decide acceptable length
  - Select shortest path (BFS)
- This graph/policy is *monotonic* – more certificates only add validity
- `Internal` distinguished names and other identifiers are not significant!
- Can support arbitrary CA trust relationships
- Originally X.509 focused on the simple global hierarchy

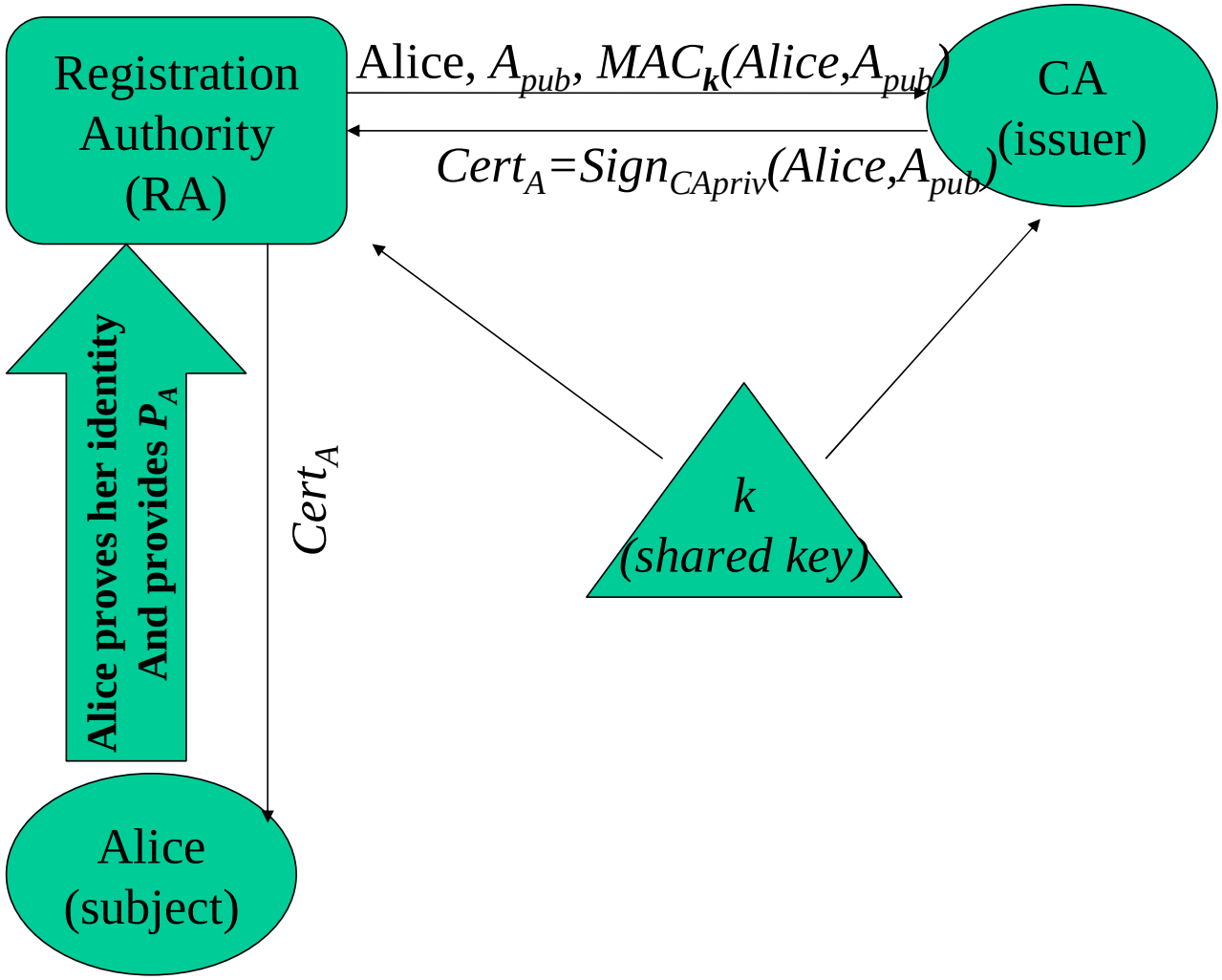
Realities of CA hierarchies

- Used mostly within an organization
  - E.g. with Lotus Notes
- CA interoperability is difficult
  - Motivation, liability, different policies
  - Effort: US Federal Bridge CA
- Relying parties often simply trust each CA; example –list of CA's in browsers

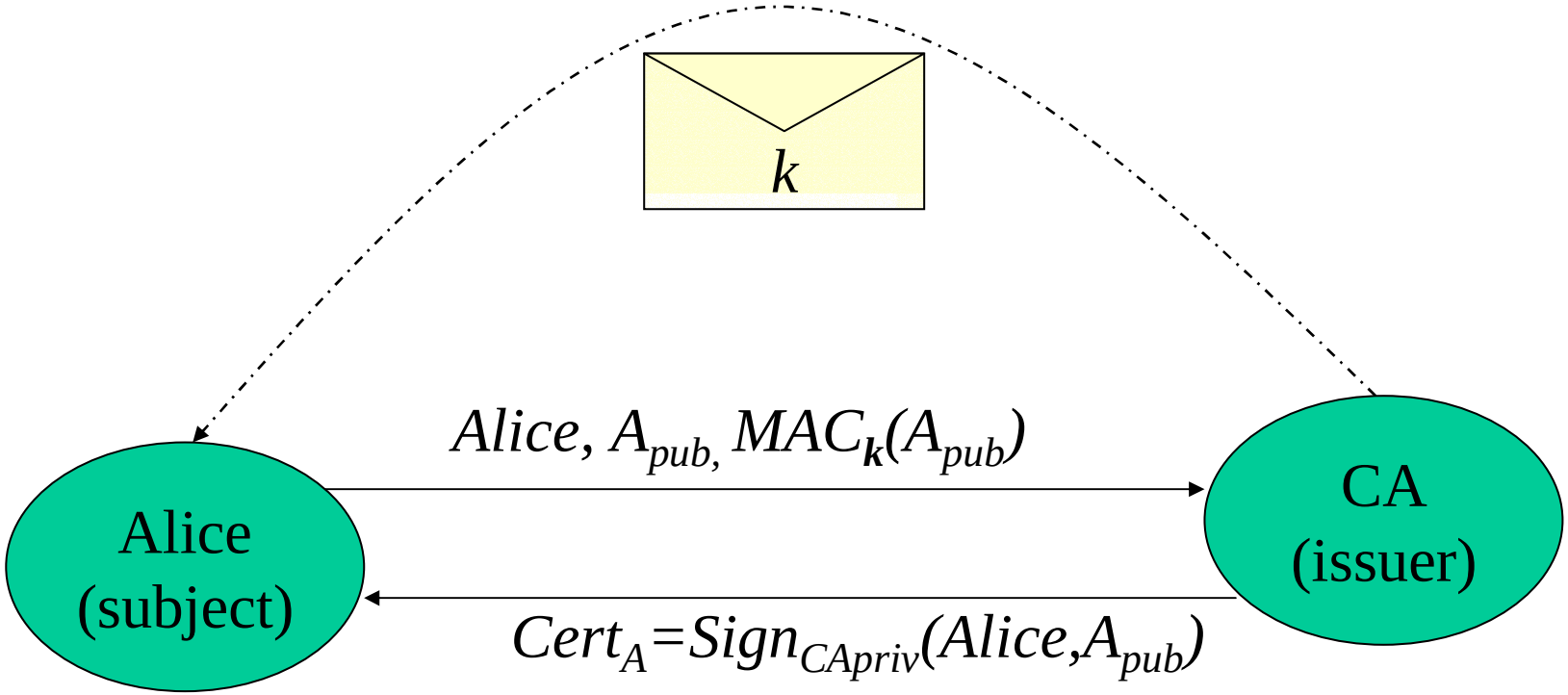
Identity PKC Risks

- To relying parties
  - Fraudulent identity
  - Disputes (claims of fraudulent identity)
- To identified entity - identity theft
- To CA – liability
  - Potentially unbounded – unknown applications
  - Limit by stating policy (of issuing, use, liability)
  - In reality, often extreme disclaimers of liability
- Main threats:
  - Exposure of the CA private signing key
  - Issuing certificate for false identity
    - Esp. a problem with remote issuing

Issuing Certificate With Registration Authority



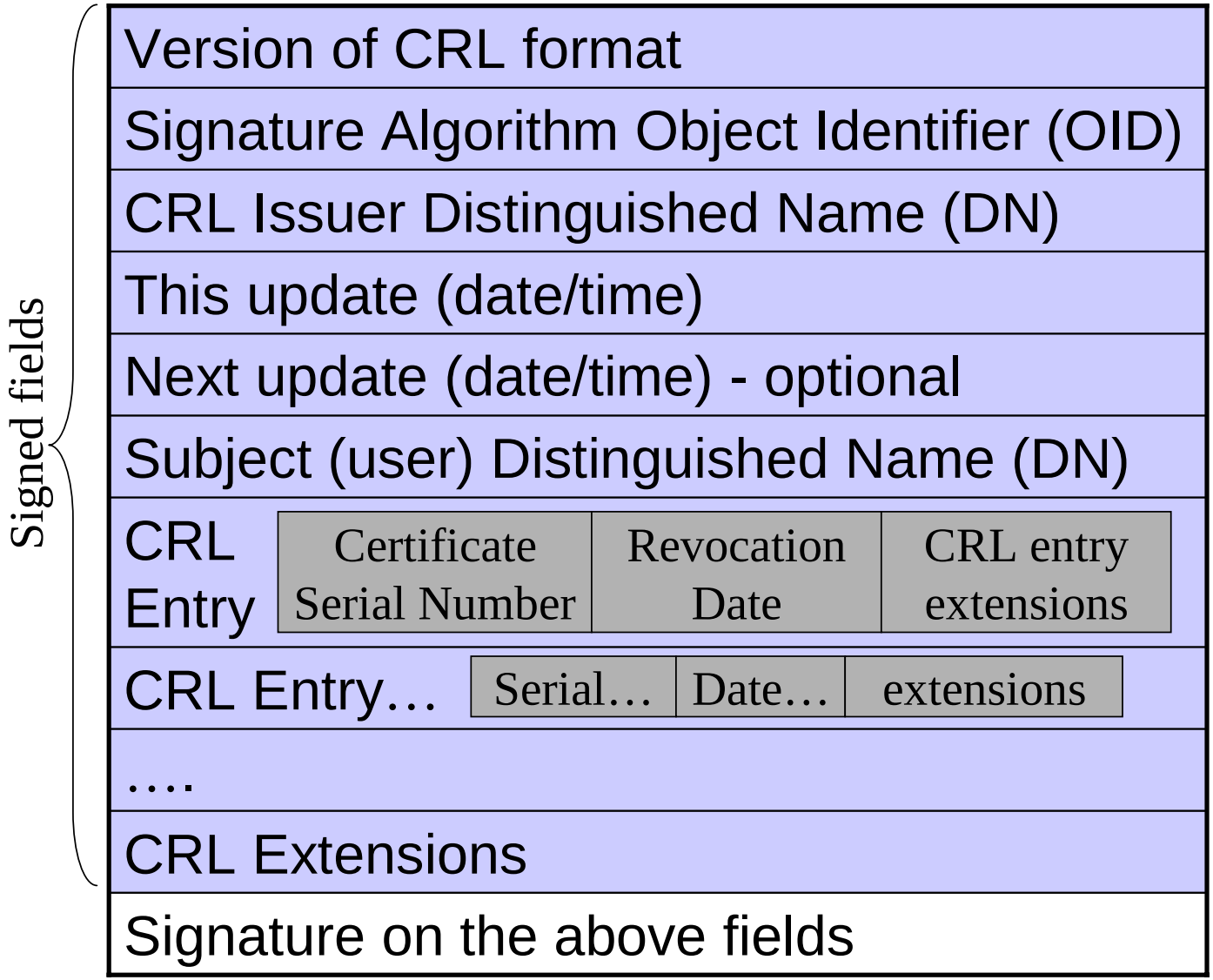
Issuing Certificate by Shared Initial Secret



Certificate Revocation

- Reasons for revoking certificate
  - Key compromise
  - CA compromise
  - Affiliation changed (changing DN or other attribute)
  - Superseded (replaced)
  - Cessation – not longer needed
- How to inform relying parties?
  - Do not inform – wait for end of (short?) validity period
  - Distribute *Certificate Revocation List (CRL)*
  - Ask - Online Certificate Status Protocol (OCSP)

X.509 CRL Format



Revocation is Difficult

- If CRLs contain all revoked certificates (which did not expire)… it may be huge!
- CRLs are (also) not immediate
  - Who is responsible until CRL is distributed?
  - What is the impact on non-repudiation?
- Solutions:
  - More efficient CRL schemes
    - Delta CRL – only new revocations since last `base CRL`
    - CRL distribution point – split certificates to several CRLs
    - Certificate Revocation Tree
    - Authorities Revocation List (ARL): listing only revoked CAs
  - Online Certificate Status Protocol (OCSP)
  - Very short validity for certificates – no CRLs

Short-Term Certificates

- Idea: very short validity period of certificates, so no need to revoke them
- Concern: overhead of signing many certificates each (short) period
- Solutions:
  - Sign multiple keys in one certificate - *hash tree*
  - Certificate includes a *hash chain*, e.g. $h^{365}(x)$; expose $h^{365-i}(x)$ to validate the certificate for the i^{th} day.

Conclusion

- Public Key Certificates link between a public key and (attributes of) its owner
- X.509 focus is on Identity PKC
- Identity PKC are natural – we are used to ID cards
- But even X.509 added non-identity attributes:
  - In extensions of the X.509 PKC
  - In attribute certificates (next lecture)
- Next two lectures – secure communication

What is an Identity PKC?

- Let *I* be the set of identifiers
  - Not of identities!
  - WLOG $I \subseteq Values$ (identities can be attribute values)
- Let $IN \subseteq AttrNames$ be the set of attribute names for identifiers (Identifier Names)
- An *identity* public key certificate is a PKC with an attribute $<n,v>$ s.t. $n \in IN$, $v \in I$.
- *Distinguished Name*: use unique, well defined, legally meaningful name identifiers
  - Allows using existing (off-Net) means of reputation, liability and judgment
  - Part of X.500 and X.509 ITU standards