CSE 3400 - Introduction to Computer & Network Security (aka: Introduction to Cybersecurity)

Lecture 12
Public Key Infrastructure – Part I

Ghada Almashaqbeh

UConn

From Textbook Slides by Prof. Amir Herzberg

UConn

Outline

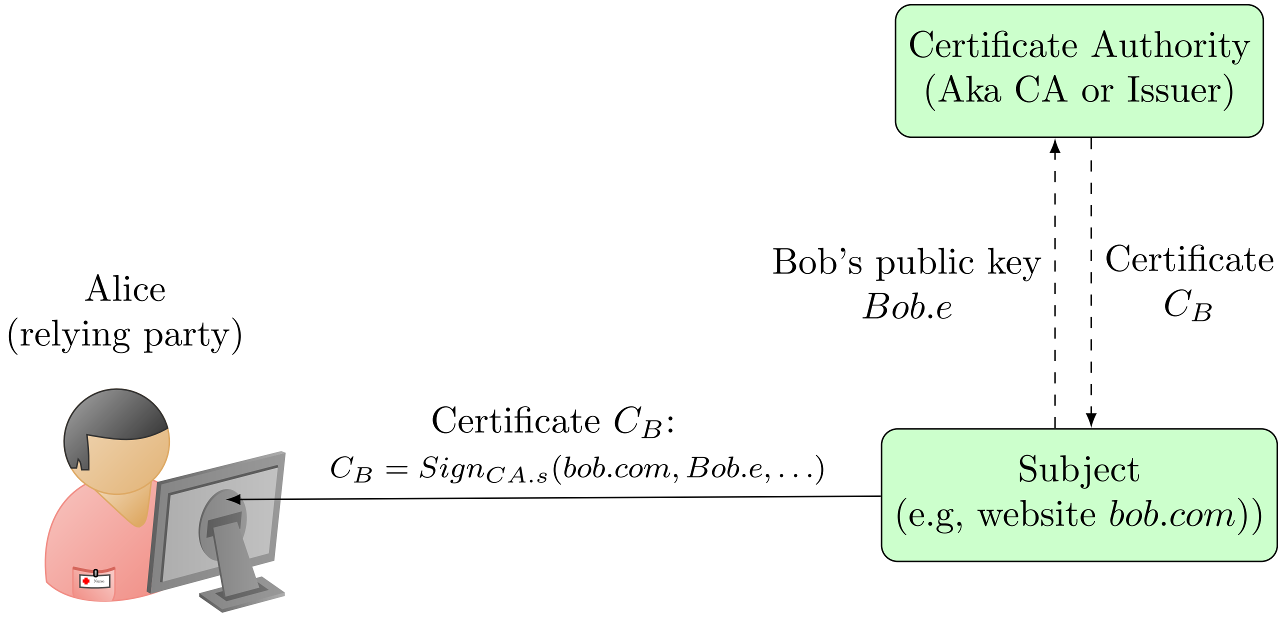
- Public key infrastructure (PKI) components.
- PKI goals.
- X.509 PKI concepts.
- Intermediate CAs and trust path verification.

Public keys are very useful…

- Secure web connections
- Software signing (against malware)
- Secure messaging, email
- Cryptocurrency and blockchains.
- But …
  - How do we know the PK of an entity?
    - Mainly: signed by a **trusted** **Certificate Authority**
    - E.g., in TLS, browsers maintain list of ‘root CAs’

Public Key Certificates & Authorities

- *Certificate:* signature by Issuer / Certificate Authority (CA) over subject's public key and attributes
- Attributes: identity (ID) and others…
  - Validated by CA (liability?)
  - Used by **relying party** for decisions (e.g., use this website?)


Certificate Authority (Aka CA or Issuer)
Bob's public key $Bob.e$
Certificate C_B
Alice (relying party)
Certificate C_B:
$C_B = Sign_{CA.s}(bob.com, Bob.e, \ldots)$
Subject (e.g, website $bob.com$))

Main application: Web-PKI

PKI deployed by **TLS/SSL, browsers, web-servers**

Browsers contain keys of **Root CAs** (trust anchors)

Root CAs defined by (four) **root programs**
(of Google, MS, Mozilla, Apple)

Root CA certifies **Intermediate CAs (ICA)**

Subject (website) certs issued by **intermediate CA**

Certificates are all about **Trust**

- Certificate: $C_{Bob} = Sign_{CA.s}(Bob.com, Bob.e, \ldots)$
  - CA attests that Bob’s public key is $Bob.e$
- Do we **trust** this attestation to be true?
- Special case of **trust management**
  - Important problem far beyond PKI… still not resolved !

Rogue Certificates

- Rogue cert: equivocating or misleading (domain) name
- Attacker goals:
  - Impersonate: web-site, phishing email, signed malware..
  - Equivocating (same name): circumvent name-based security mechanisms, such as *Same-Origin-Policy (SOP), blacklists, whitelists*, *access-control* …
  - Name may be misleading even if not equivocating
- Types of misleading names ('cybersquatting'):
  - Combo names: bank.com vs. accts-bank.com, bank.accts.com, …
  - Domain-name hacking: accts.bank.com vs. accts-bank.com, … or accts-bank.co
  - Homographic: paypal.com [l is L] vs. paypal.com [i is I]
  - Typo-squatting: bank.com vs. banc.com, baank.com, banl.com,…

PKI Failures

- Although the signature over the certificate verifies correctly, there is still a failure and the certificate must be revoked.
  - This is called a PKI failure.
- PKI failures include:
  - Subject key exposure.
  - CA failure.
  - Cryptanalysis certificate forgery.
    - Find collisions in the hash function used in the HtS paradigm,
    - or exploit some vulnerability in the digital signature scheme used for signing.

Some Infamous PKI Failures

2001	VeriSign: attacker gets code-signing certs
2008	Thawte: email-validation (attackers' mailbox)
2008,11	Comodo not performing domain validation
2011	DigiNotar compromised, 531 rogue certs (discovered); a rogue cert for _*.google.com_ used for MitM against 300,000 Iranian users.
2011	TurkTrust issued intermediate-CA certs to users
2012	Trustwave issued intermediate-CA certificate for eavesdropping
2013	ANSSI, the French Network and Information Security Agency, issued intermediate-CA certificate to MitM traffic management device
2014	India CCA / NIC compromised (and issued rogue certs)
2015	CNNIC (China) issued CA-cert to MCS (Egypt), who issued rogue certs. Google and Mozilla removed CNNIC from their root programs.
2013-17	Audio driver of Savitech install root CA in Windows
2015,17	Symantec issued unauthorized certs for over 176 domains, causing removal from all root programs.
2019	Mozilla, Google _browsers_ block _customer-installed_ Kazakhstan root CA (Qaznet)
2019	Mozilla, Google revoke intermediate-CA of DarkMatter, and refuse to add them to root program

PKI Goals/Requirements

Trustworthy issuers: Trust anchor/root CAs and Intermediary CAs; Limitations on Intermediary CAs (e.g., restricted domain names)

Accountability: identify issuer of given certificate

Timeliness: limited validity period, timely **revocation**

Transparency: public log of all certificate; no ‘hidden’ certs!

Non-Equivocation: one entity – one certificate

Privacy: why should CA know which site I use?

X.509 Certificates

Part of the X.500 Global Directory Standard

The X.500 Global Directory Standard

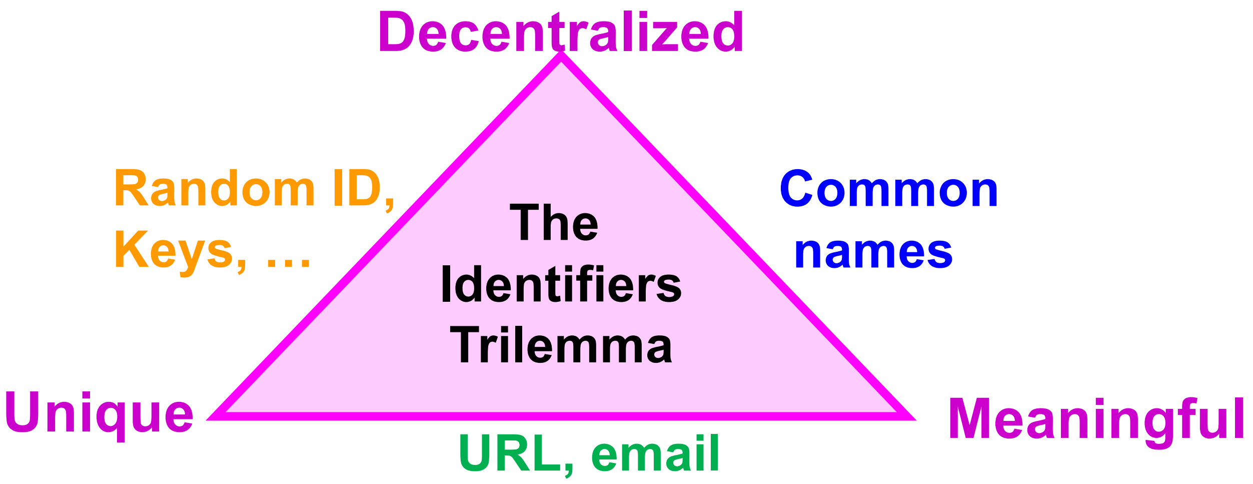
- X.500: an ITU standard, first issued 1988
  - ITU: International Telecommunication Union
- Idea: trusted global directory
  - Operated by hierarchy of trustworthy telcos companies and providers.
  - Never happened
    - Too complex, too revealing, too trusting of telcos
- Directory binds identifiers to attributes
  - Standard attributes (including public key)
  - Standard identifiers: Distinguished Names

Distinguished Names or Identifiers in Certificates

- Most certificates contain identifiers
  - Aka identity-certificates
- Basic goals of identifiers:
- Meaningful (to humans)
  - Memorable, reputation, off-net, legal
- Unique identification of entity (owner)
- Decentralized - with Accountability: assigned by trusted (certificate) authorities
  - Accountability: identification of the signing authority

The Identifiers Trilemma

- Achieving the three goals: Meaningful, Unique, Decentralized, seems very challenging!
- Examples of achieving any two of the goals:
  - Unique + Meaningful: URL, email
  - Meaningful + Decentralized: common name
  - Unique + Decentralized: hash of key

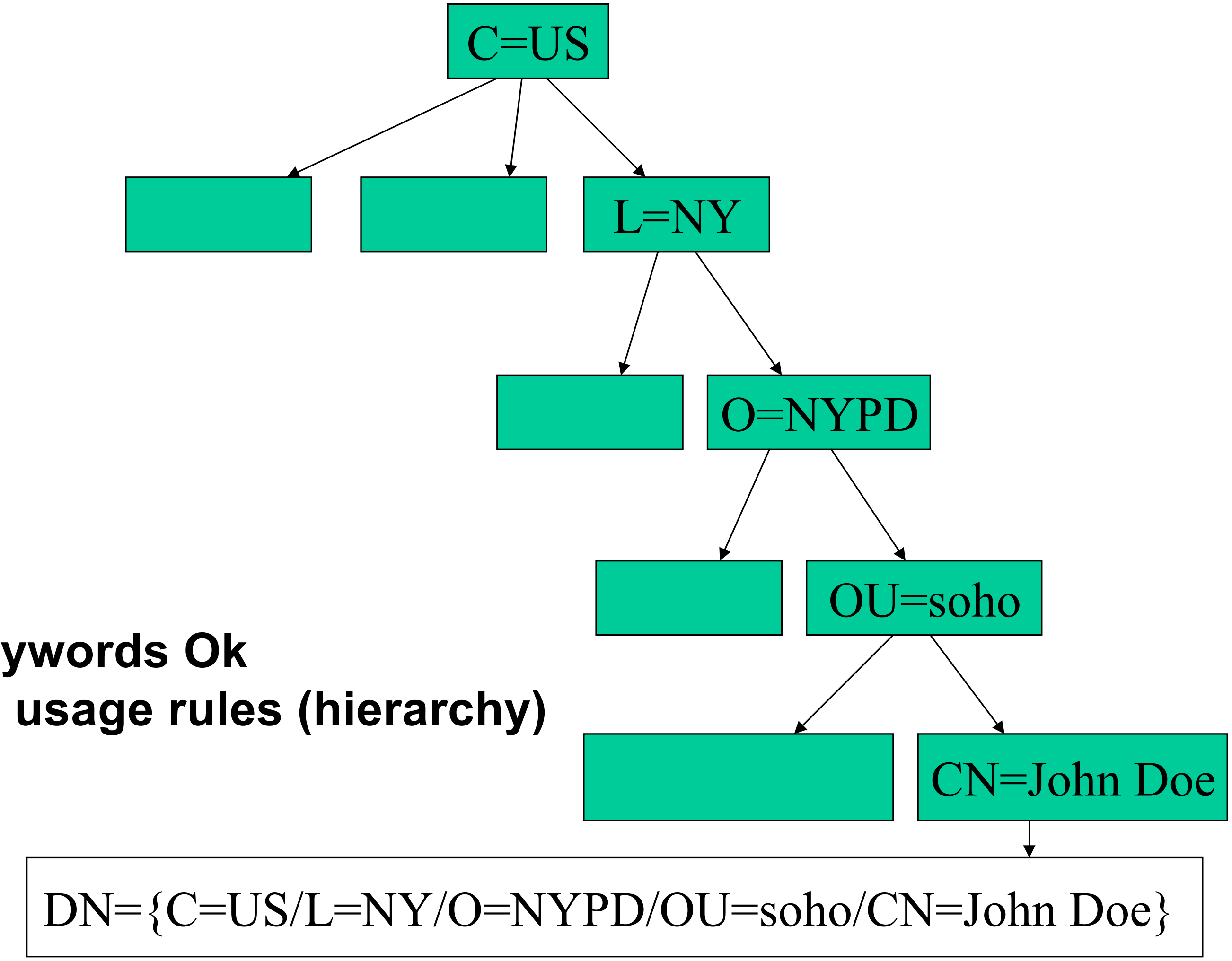


X.500 Distinguished Names (DN)

- Sequence of keywords, 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 (DN) Hierarchy


C=US

L=NY

O=NYPD

OU=soho

CN=John Doe

DN={C=US/L=NY/O=NYPD/OU=soho/CN=John Doe}

Comments:

1. **Other keywords Ok**
2. **No strict usage rules (hierarchy)**

Distinguished Names - Evaluation

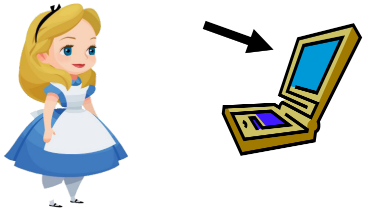
- **Decentralized?**
  - Separate name spaces
- **Unique** ?
  - Could be, if each name space has one issuer
  - TLS reality: browsers trust 100s of CAs for **every name**
- **Meaningful**?
  - Usually: Julian Jones/UK/IBM
  - But not always: Julian Jones2/UK/IBM
    - Added ‘counter’ to distinguish → mistakes, loss of meaning

Distinguished Names – More Problems

- Distinguished Name fields may compromise privacy
  - E.g., expose organizational sensitive information (e.g. unit)
- Handling changes in position, organizations
- Multiple, related hierarchies:
  - International organizations, divisions…
    - Julian Jones/UK/IBM or Julian Jones/IBM/UK ?
    - Julian Jones/Research/UK/IBM or Julian Jones/UK/Research/IBM ?
  - DNs are not usable; users do not know DNs.

DN={c=US/L=com/O=Bank}

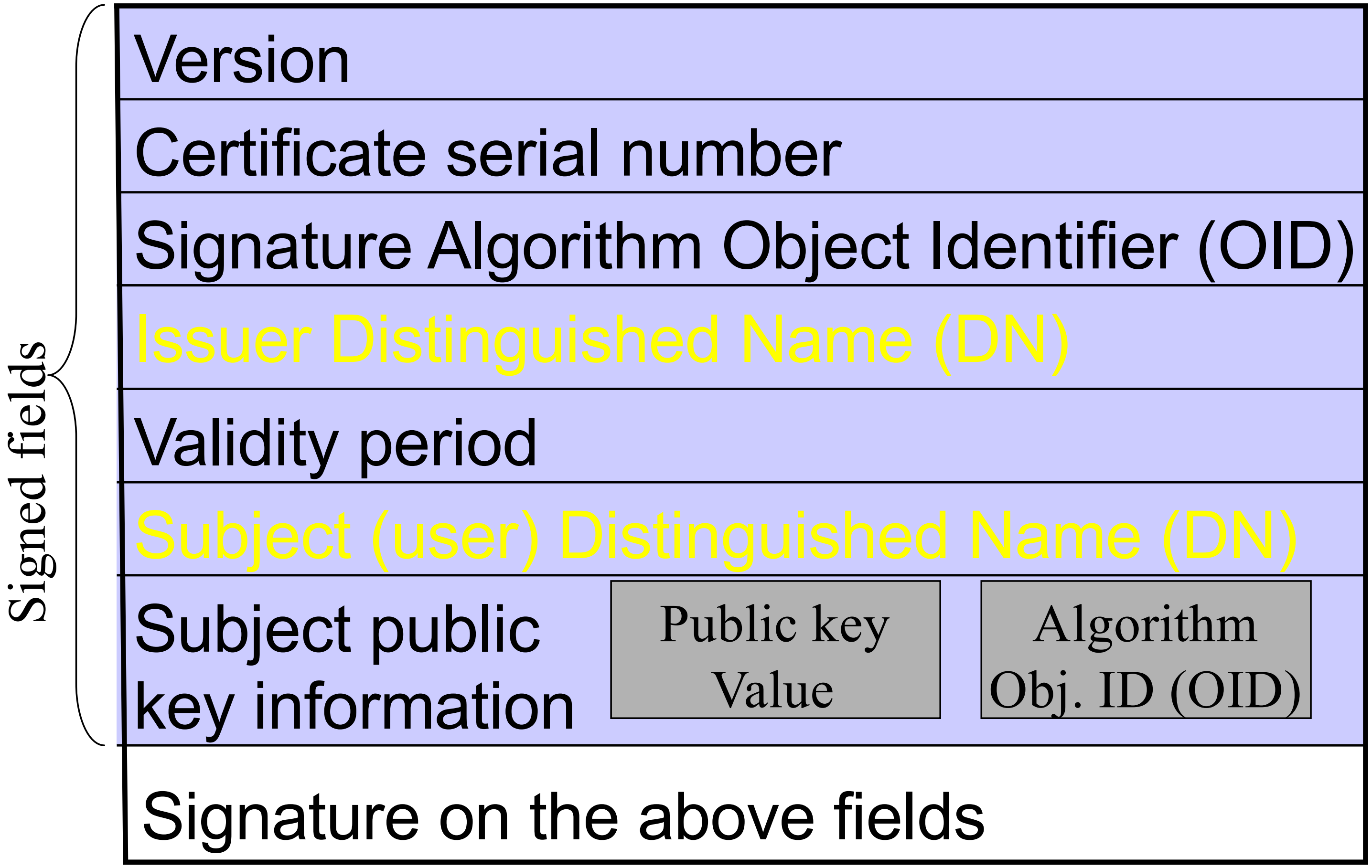
https://bank.com



X.509 public key certificates

- X.509: authentication mechanisms of X.500
- Initially: Authenticate to Directory (Password-based authentication)
  - To maintain entity’s record
- Later (and now): **X.509 public key certificate**
  - Signature binds **public key** to distinguished name (DN) and to other attributes
    - Some defined in X.509 standard, others in `extensions`
- Used widely in spite of complaints about its complexity.
  - SSL / TLS, code-signing, PGP, S/MIME, IP-Sec, …

Original (V1) X.509 Certs Format



Version

Certificate serial number

Signature Algorithm Object Identifier (OID)

Issuer Distinguished Name (DN)

Validity period

Subject (user) Distinguished Name (DN)

Subject public key information

Public key Value

Algorithm Obj. ID (OID)

Signature on the above fields

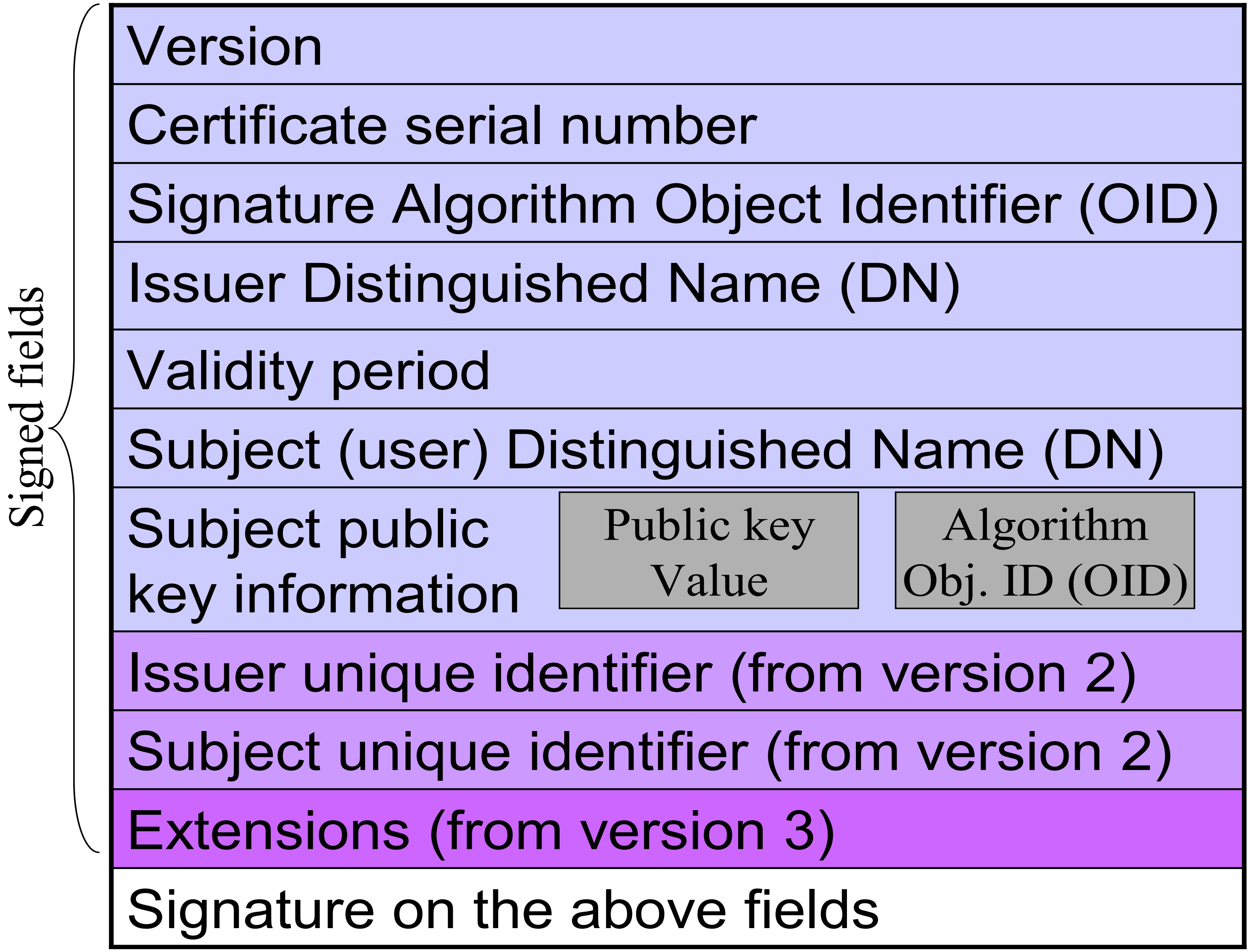
Object Identifiers (OID):

- Global, unique identifiers
- Sequence of numbers, e.g.: 1.16.840.1.45.33
  - Hierarchical

X.509 Certs & Subject Identifiers

- V1: Distinguished Name (for subject & issuer)
- V2: unique identifiers (for subject & issuer)
- V3: extensions (used in practice)
  - Some defined in X.509, others elsewhere
  - PKIX: IETF standard extensions profile
    - Widely adopted, including in SSL/TLS (& https)
  - Example: SubjectAltName extension
    - Including DNSname: identify website by domain name
- [V4: not covered, not widely deployed]

X.509 Public Key Certificates



Signed fields

Version

Certificate serial number

Signature Algorithm Object Identifier (OID)

Issuer Distinguished Name (DN)

Validity period

Subject (user) Distinguished Name (DN)

Subject public key information

Public key Value

Algorithm Obj. ID (OID)

Issuer unique identifier (from version 2)

Subject unique identifier (from version 2)

Extensions (from version 3)

Signature on the above fields

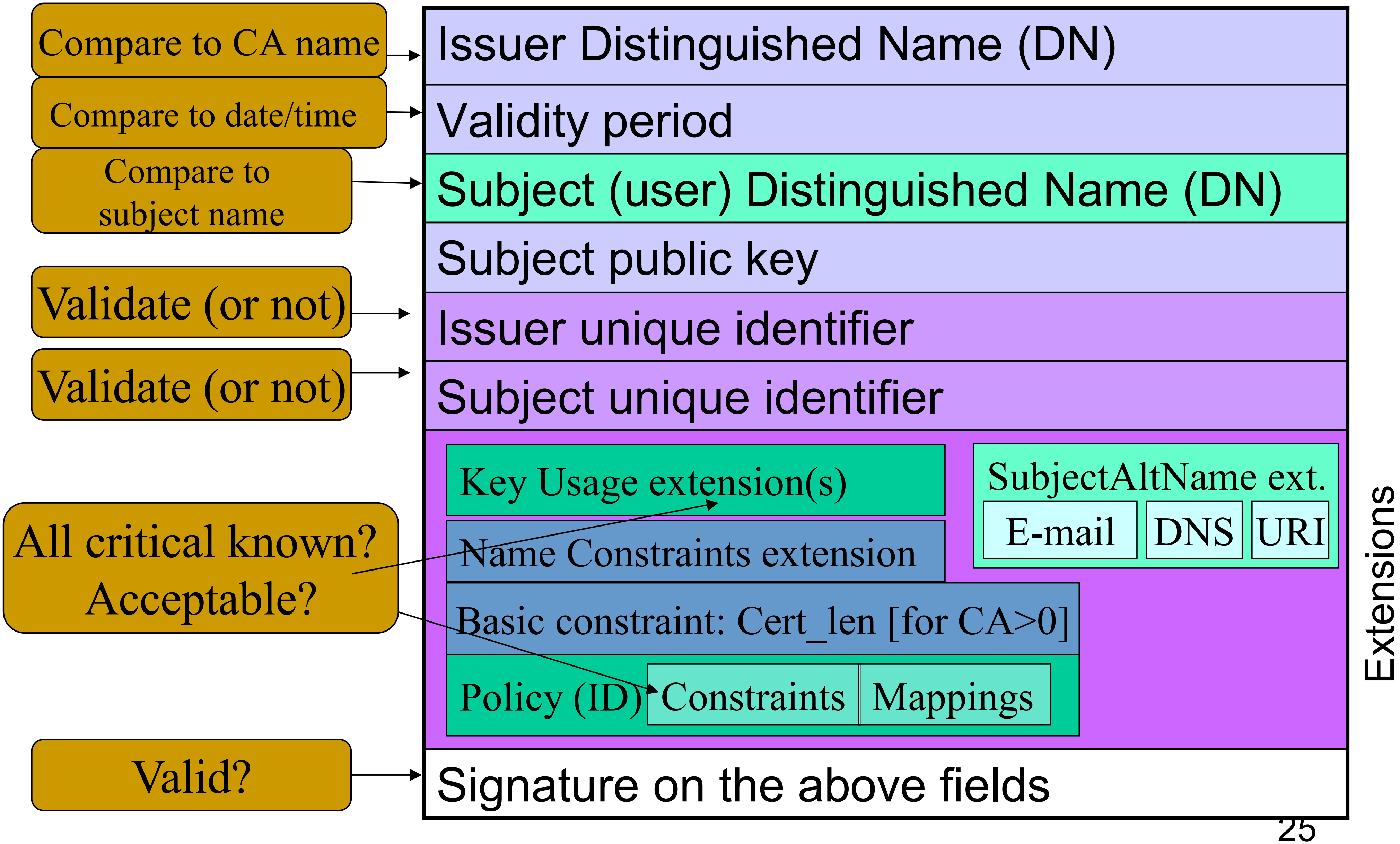
X.509 V3 Extensions Mechanism

- Each extension contains:
- **Extension identifier**
  - As an OID (Object Identifier)
  - E.g. `key usage`
- **Extension value:** arbitrary string serve as a value for the extension.
  - E.g. use the key for encryption, or `Permit C=GB' for name constrains extension.
- **Criticality indicator**
  - **If critical**, relying parties MUST NOT use a certificate with any unknown critical extension
  - **If non-critical:** use certificate with unknown non-critical extensions; **ignore** unknown (non-critical) extensions and accept the certificate.

Key Usage, Identifier Extensions

- Key-usage extension.
  - X.509: may be critical
- Use of the public key being certified
    - Encrypt, verify-signature, verify-certificate, …
- Extended key usage extension
  - Additional optional use of the key: **Non-critical**
  - Details/restrictions related to `key usage' : **Critical**
- Subject/authority key identifier
  - Used when subject/CA has many keys; non-critical

X.509 Certificate Validation (simplified)



SubjectAltName (ESN) Extension

- Bound identities to the subject
  - In addition/instead of Subject Distinguished Name
  - Same extension may contain multiple ESNs
- Goal: unique and meaningful names
  - Common: DNS name (dNSName), e.g., a.com
    - TLS/SSL allows wildcard domains (*.a.com)
  - Or: email address, IP address, URI, other
- IssuerAltName (IAN) extensions
  - Similar – for issuer

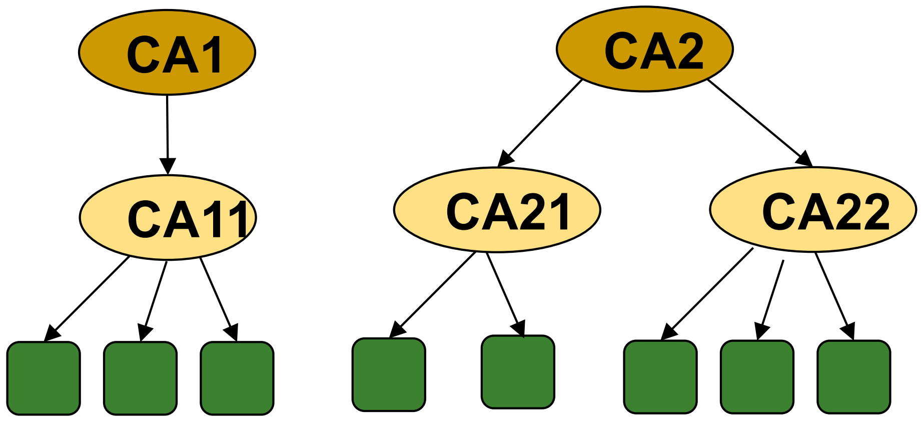
Intermediate CAs and Path Verification

Why Intermediate CAs?

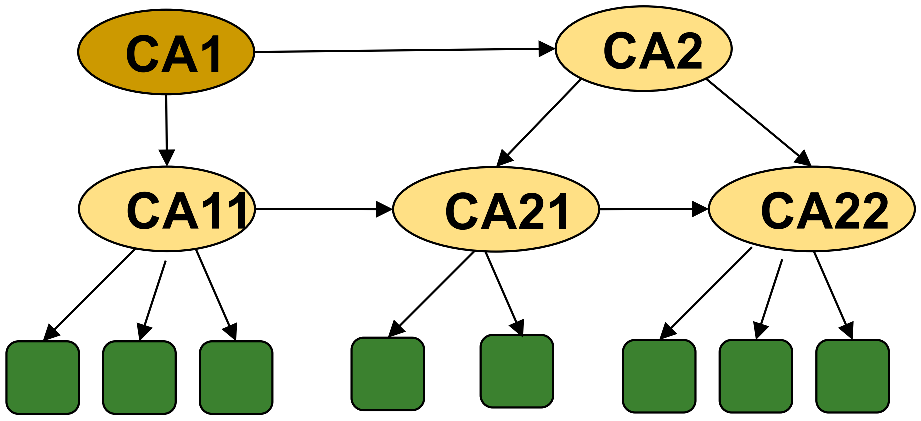
- Relying parties rely on trust anchor CA(s) to establish trust in a certificate.
- Large number of subjects to certify.
  - One (or a few) trust anchor CAs cannot handle all the load.
- An anchor or root CA certifies other CAs to become intermediate CAs.
  - So the root A certifies intermediate B, then B will sign certificates for subjects (B is an issuer).
- Certificate path validation allows validating such certificates that are issued by intermediate CAs.
  - Like tracing them back to a trust anchor.

Certificate paths in different PKIs

- **Web/TLS PKI:** ‘root CAs’+‘intermediate CAs’:
  - Root CA issues cert for intermediate CAs

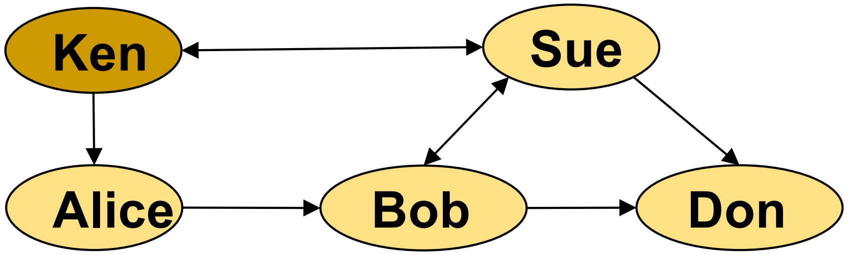


- **Web-of-Trust PKIs:**
  - Directed graph, not tree
  - Different variants/policies



Web of Trust PKI

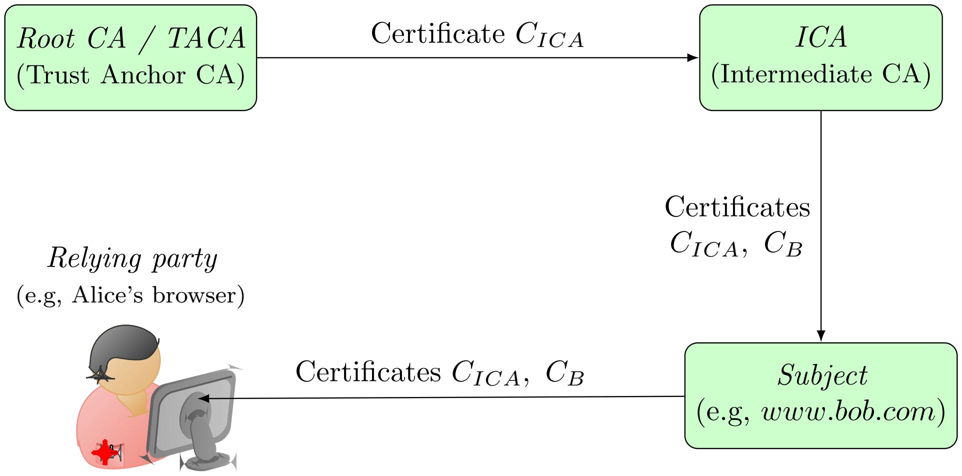
- PGP's friends-based Web-of-Trust:
  - Everyone is subject, CA and relying party
  - As a CA, certify (pk, name) for `friends'
  - As a subject, ask friends to sign for you
  - As a relying party, trust certificates from friends
    - Or also from friends-of-friends? Your policy….
    - Should you trust all your friends (equally)?



Certificate-Path Constraints Extensions

- Basic constraints:
  - Is the subject a CA? (default: FALSE)
  - Maximal length of additional CAs in path
    - pathLengthConstraint
- Policy constraints:
  - Require certificate-policies along path
  - Allow/forbid `policy mappings’
  - Details in textbook (or RFC)
- Name constraints
  - Constraints on DN and SubjectAltName
    - in certs issued by subject
    - Only relevant when subject is a CA !
  - ‘Permit’ and ‘Exclude’

Certificate-Path Constraints - Example



Root CA / TACA
(Trust Anchor CA)

Certificate C_{ICA}

ICA
(Intermediate CA)

Certificates
$C_{ICA},\ C_B$

Relying party
(e.g, Alice's browser)

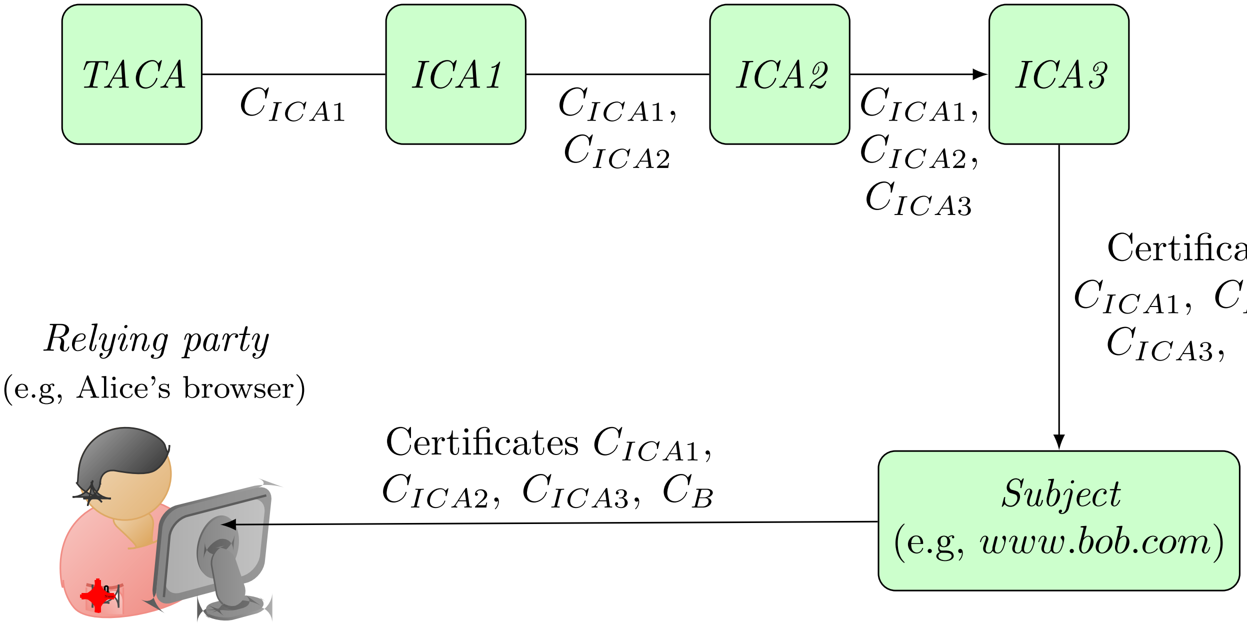
Certificates $C_{ICA},\ C_B$

Subject
(e.g, *www.bob.com*)

	C_{ICA} constraints extensions					C_B valid?
	Basic		Name		Policy	
	cA	pathLen	Permit	Exclude	Req. Policy	
1	No	(any)	(any)	(any)	(any)	No
2	Yes	(any)	*bob.com*	none or *x.bob.com*	none or > 1	Yes
3	Yes	(any)	*cat.com*	(any)	(any)	No
4	Yes	(any)	*bob.com*	*www.bob.com*	(any)	No
5	Yes	(any)	(any)	(any)	0	No
6	Yes	(any)	(none)	*bob.com*	(any)	No

Here the certificate has
no policy extensions.

Certificate-Path Constraints - Example



TACA

C_{ICA1}

ICA1

C_{ICA1}, C_{ICA2}

ICA2

C_{ICA1}, C_{ICA2}, C_{ICA3}

ICA3

Certificates C_{ICA1}, C_{ICA2}, C_{ICA3}, C_B

Relying party
(e.g, Alice's browser)

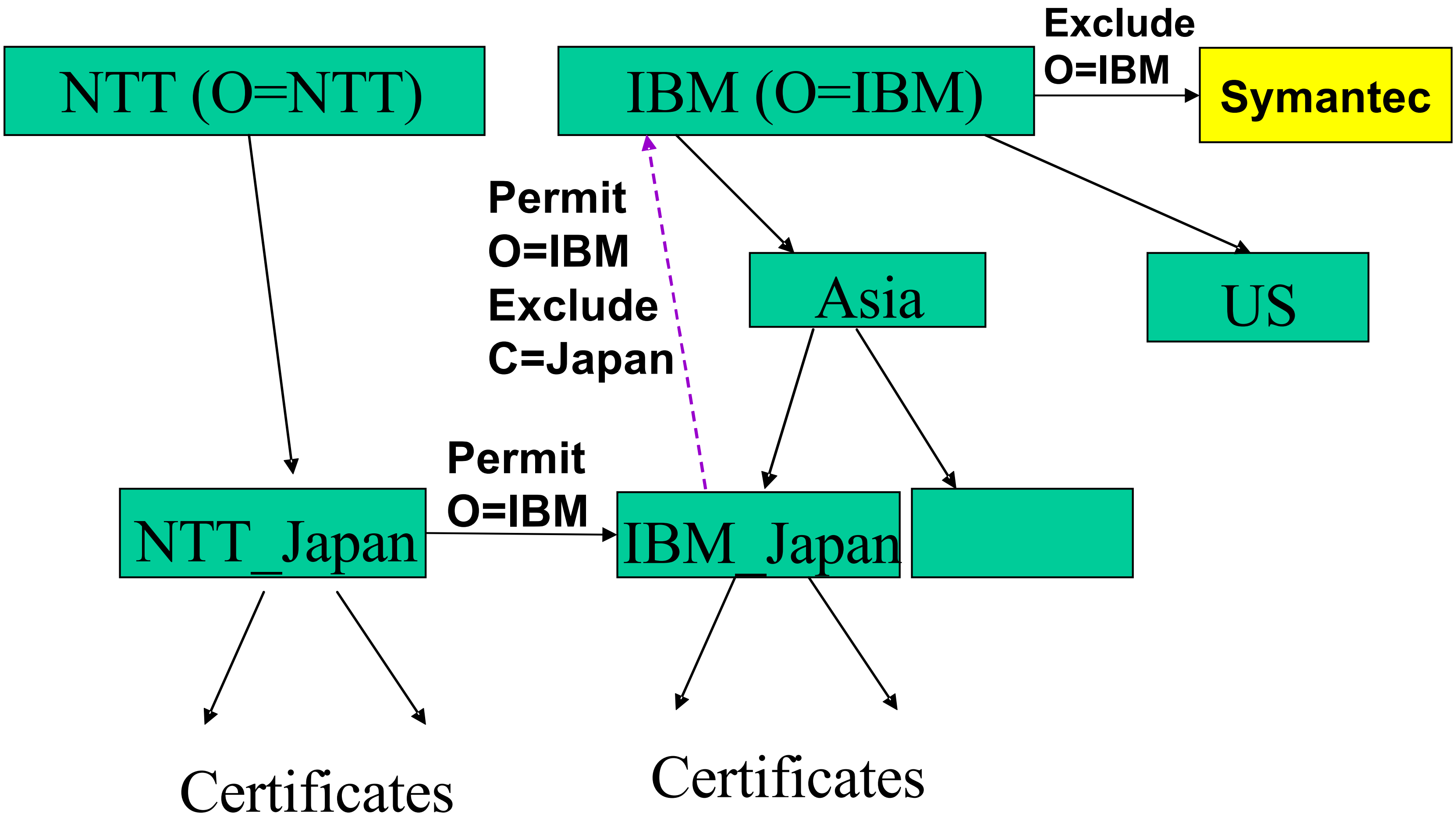
Certificates C_{ICA1}, C_{ICA2}, C_{ICA3}, C_B

Subject
(e.g, *www.bob.com*)

Here the certificate has no policy extensions. And all ICAs have CA flag true.

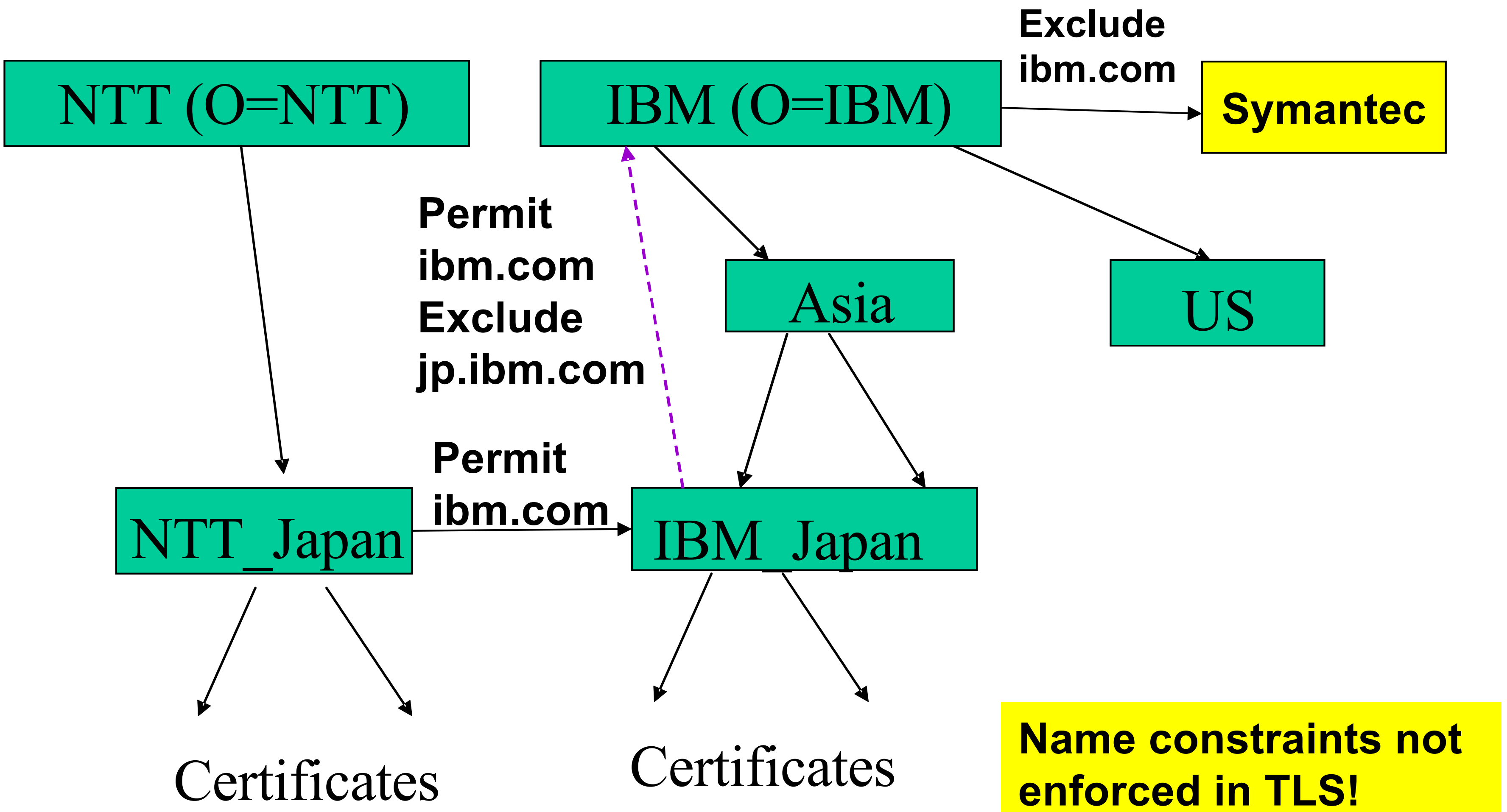
	C_{ICA1} constraints extensions					C_B valid?
	Basic		Name		Policy	
	cA	pathLen	Permit	Exclude	Req. Policy	
1	Yes	< 2	(any)	(any)	(any)	No
2	Yes	none or ≥ 2	*bob.com*	none or *x.bob.com*	none or > 3	Yes
3	Yes	(any)	(any)	(any)	≤ 3	No
4	Yes	(any)	*cat.com*	(any)	(any)	No
5	Yes	(any)	(none)	*bob.com*	(any)	No

Name constraints on DN



- NTT JP permits (allows) IBM JP to certify IBMers
- IBM JP permits IBM to certify all IBMers, except of IBM JP
- IBM trusts Symantec's certificates, except for O=IBM

Name constraints on dNSName



Name constraints not enforced in TLS!

Covered Material From the Textbook

- Chapter 8:
  - Sections 8.1, 8.2, and 8.3

Thank You!