Public Key Infrastructure Fundamentals

Bart Preneel

Katholieke Universiteit Leuven

February 2011

Thanks to Paul van Oorschot

Context

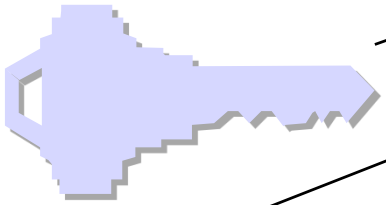
- 1. Cryptology: concepts and algorithms
- 2. Cryptology: protocols
- **3. Public-Key Infrastructure principles**
- 4. Networking protocols
- 5. New developments in cryptology
- 6. Cryptography best practices
- 7. Hash functions

How to establish public keys?

- point-to-point on a trusted channel
  - *mail business card, phone*
- direct access to a trusted public file (registry or database)
  - *authentication trees*
- on-line trusted server (bottleneck)
  - OCSP*: Online Certificate Status Protocol*
- off-line servers and certificates
  - *PKI: Public Key Infrastructure*
- implicit guarantee of public parameters
  - *identity based and self-certified keys*

What is a Certificate?

DN: cn=Planckaert
o=VTM, c=BE
Serial #: 8391037
Start: 1/3/10 1:00
End: 28/2/11 1:01
CRL: cn=CRL2,
o=VRS, c=US
Key:



CA DN: o=GLS, c=BE

Unique name of owner

Unique serial number

Period of validity

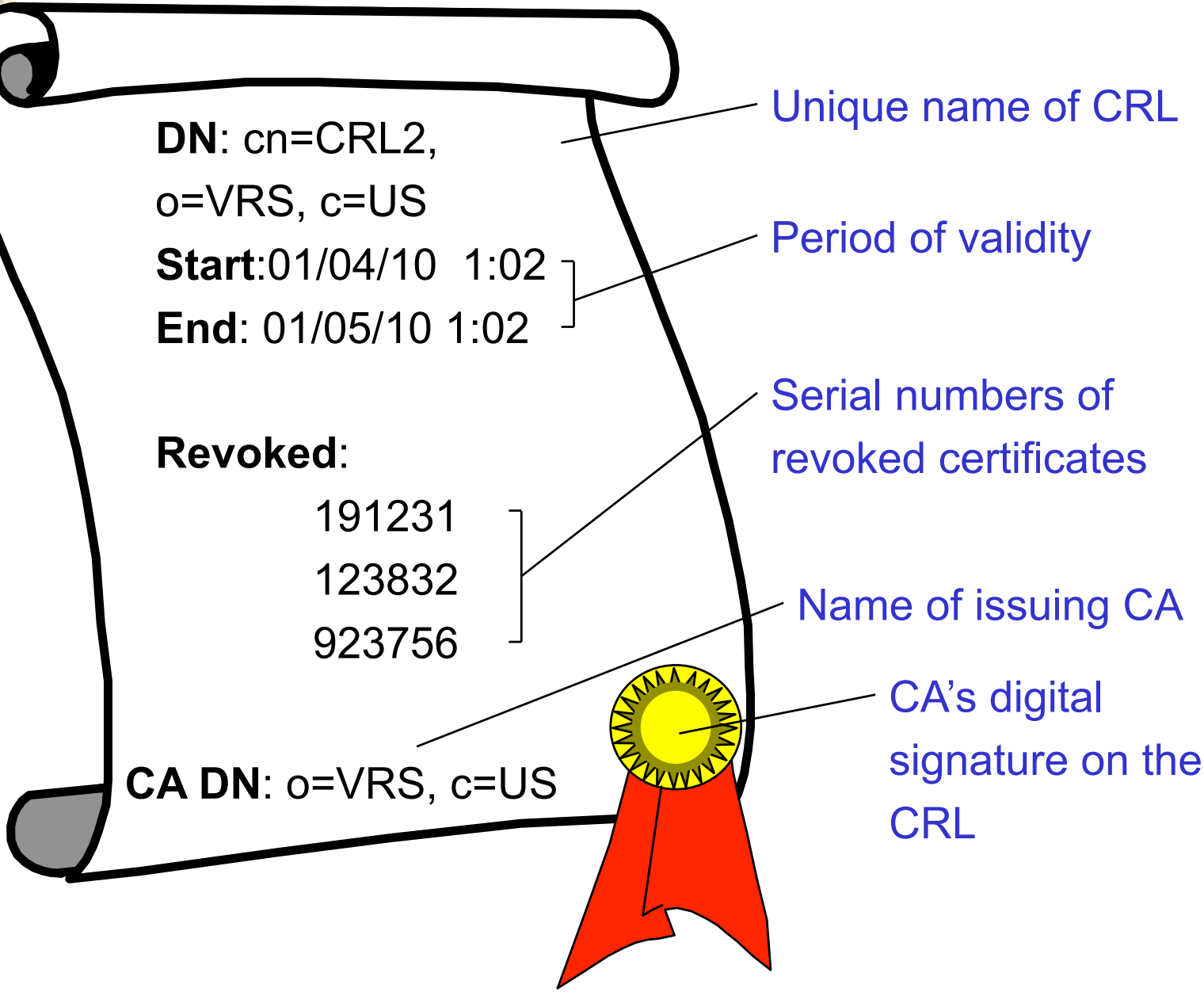
Revocation information

Public key

Name of issuing CA

CA's digital signature on the certificate

What is a Certificate Revocation List?



PKI Overview

1. Background:
   Keys and Lifecycle Management

2. PKI components (“puzzle pieces”)

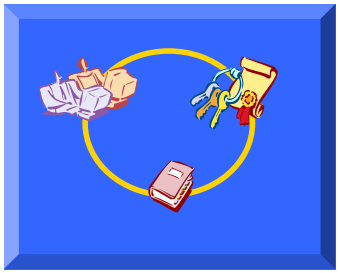
3. PKI Architectural View

4. Trust Models

Background:

Keys and Lifecycle Management

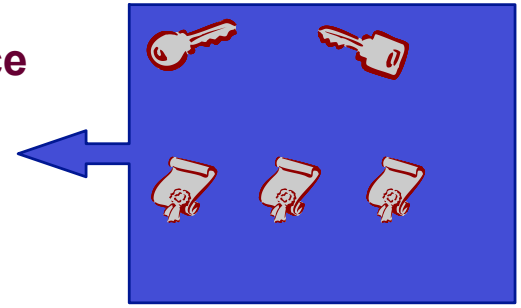
Sending secure e-mail



Encrypting/Signing...

Alice

Bob



Stored key material

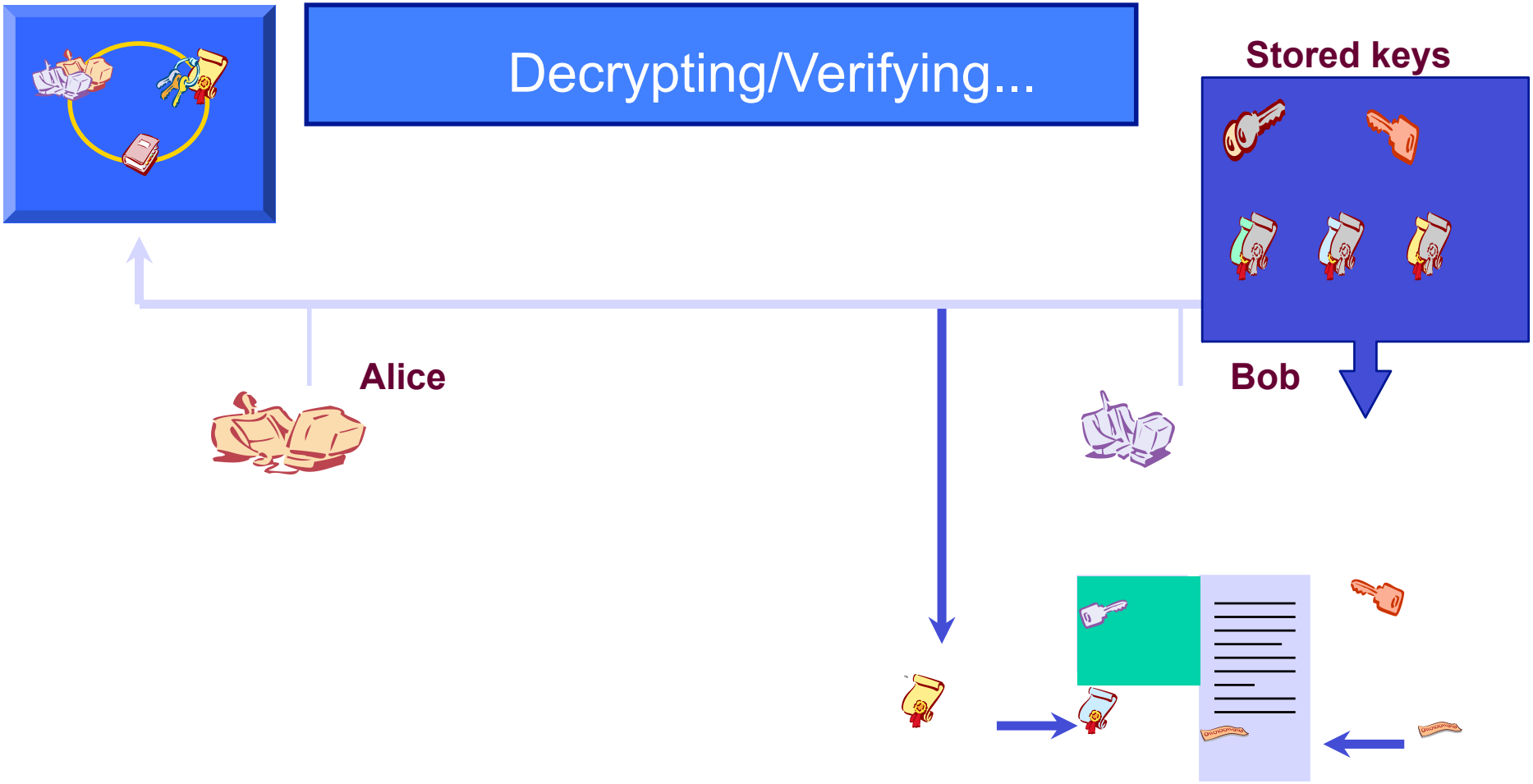
Abcdefalsdasdfasdf asdfasdfAsdfasdfas dfasdfasdfasdfasdfa sdfpolaskjflieaseifjas leifjalsiejf;

Abcdefalsdasdfasdf asdfasdfAsdfasdfas dfasdfasdfasdfasdfa sdfpolaskjflieaseifjas leifjalsiejf;lasiejfflasij efj;lialakjsdf asd ;laksdjfladksjflak sjdflkasjdlfjsald;jf;lak jaslkdjjfasdfasdfasdl kj

aslkdjf;laskjdflasjdlfj ks;ldkjfsalkjlkj;lkjasf

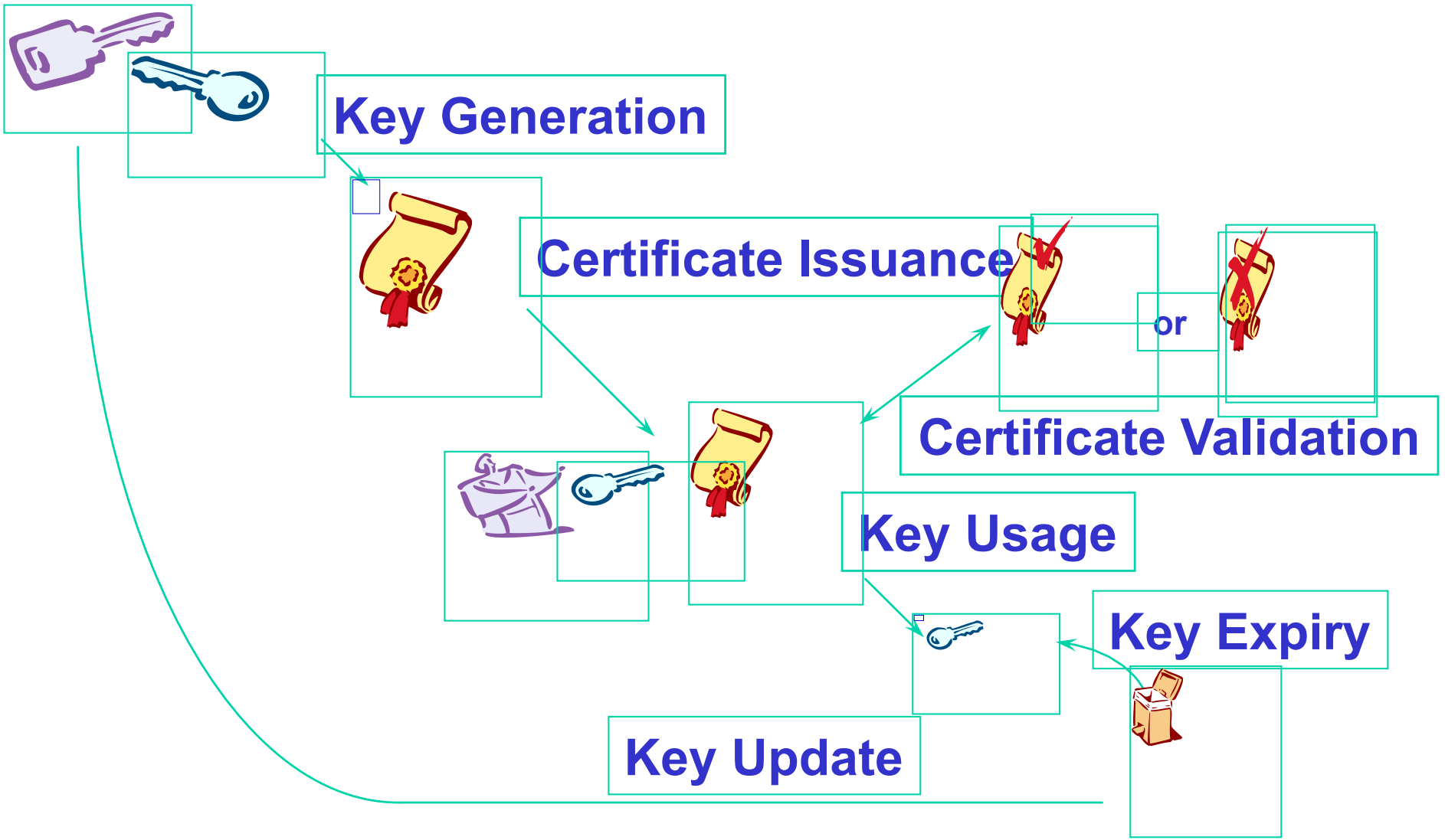
- **Alice composes a message for Bob**

Receiving secure e-mail



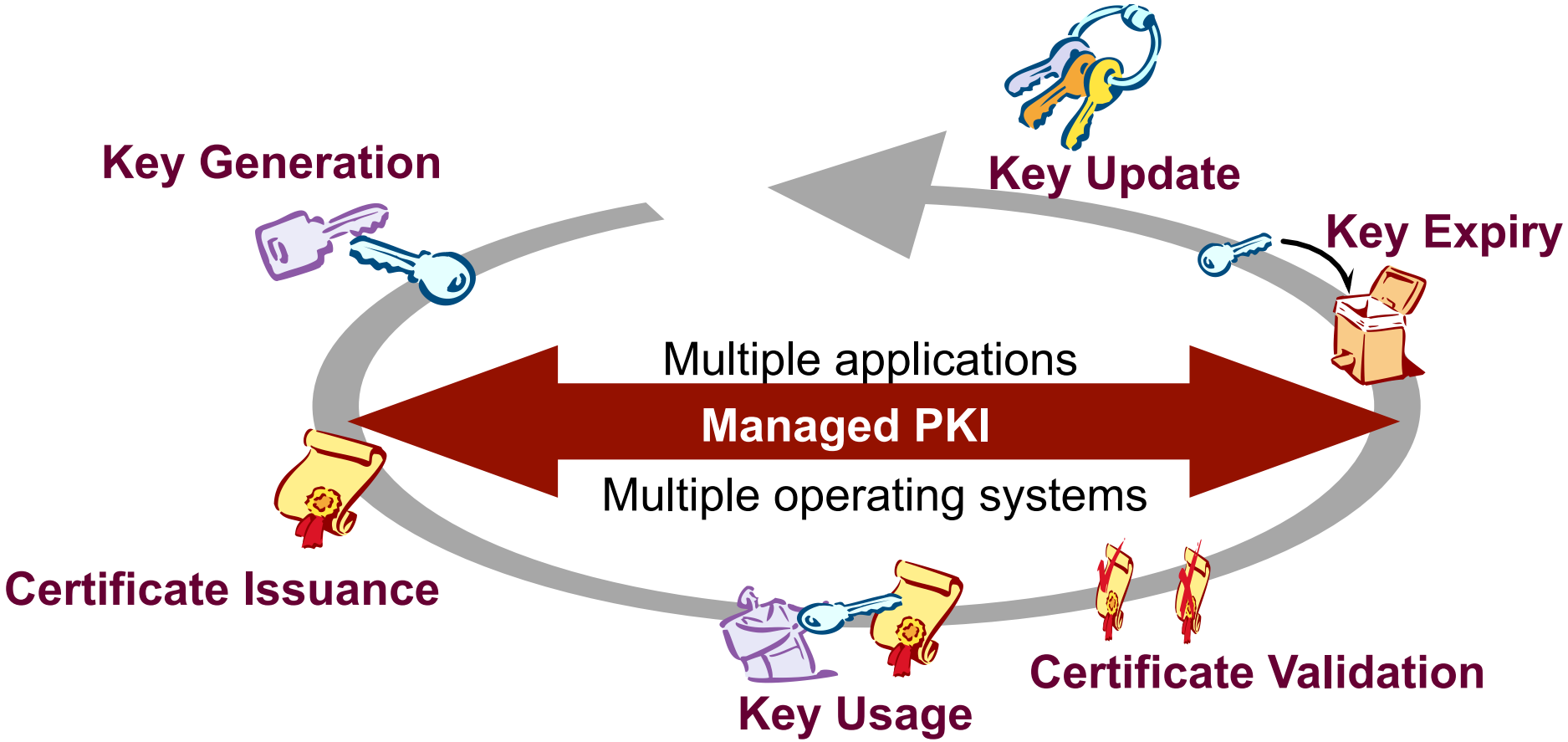
- **Bob uses the one-time symmetric key to retrieve the message text and signed hash**

Key Lifecycle Management



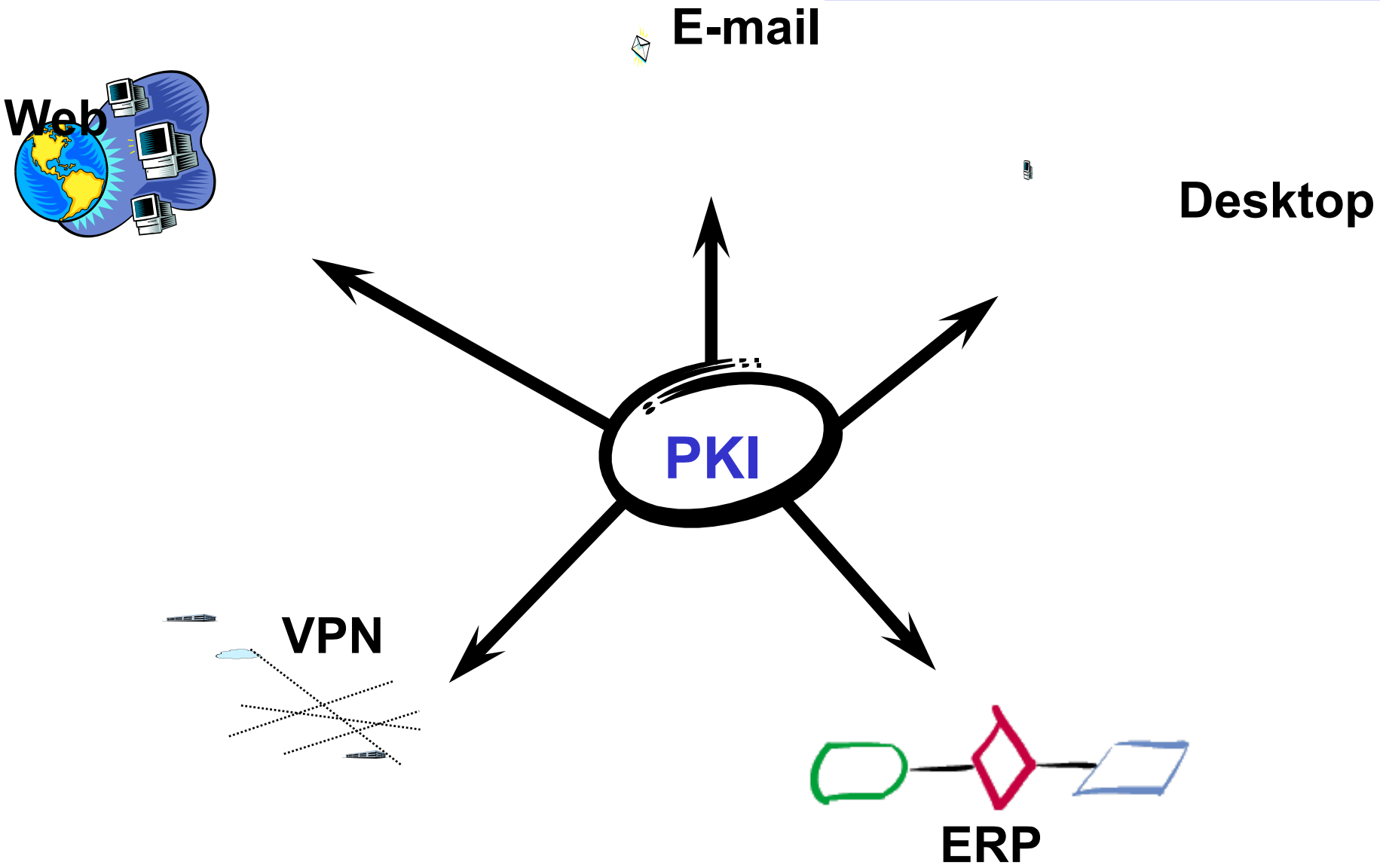
Fundamental PKI features

- Automated and transparent key and certificate lifecycle management
- Consistent behavior across applications

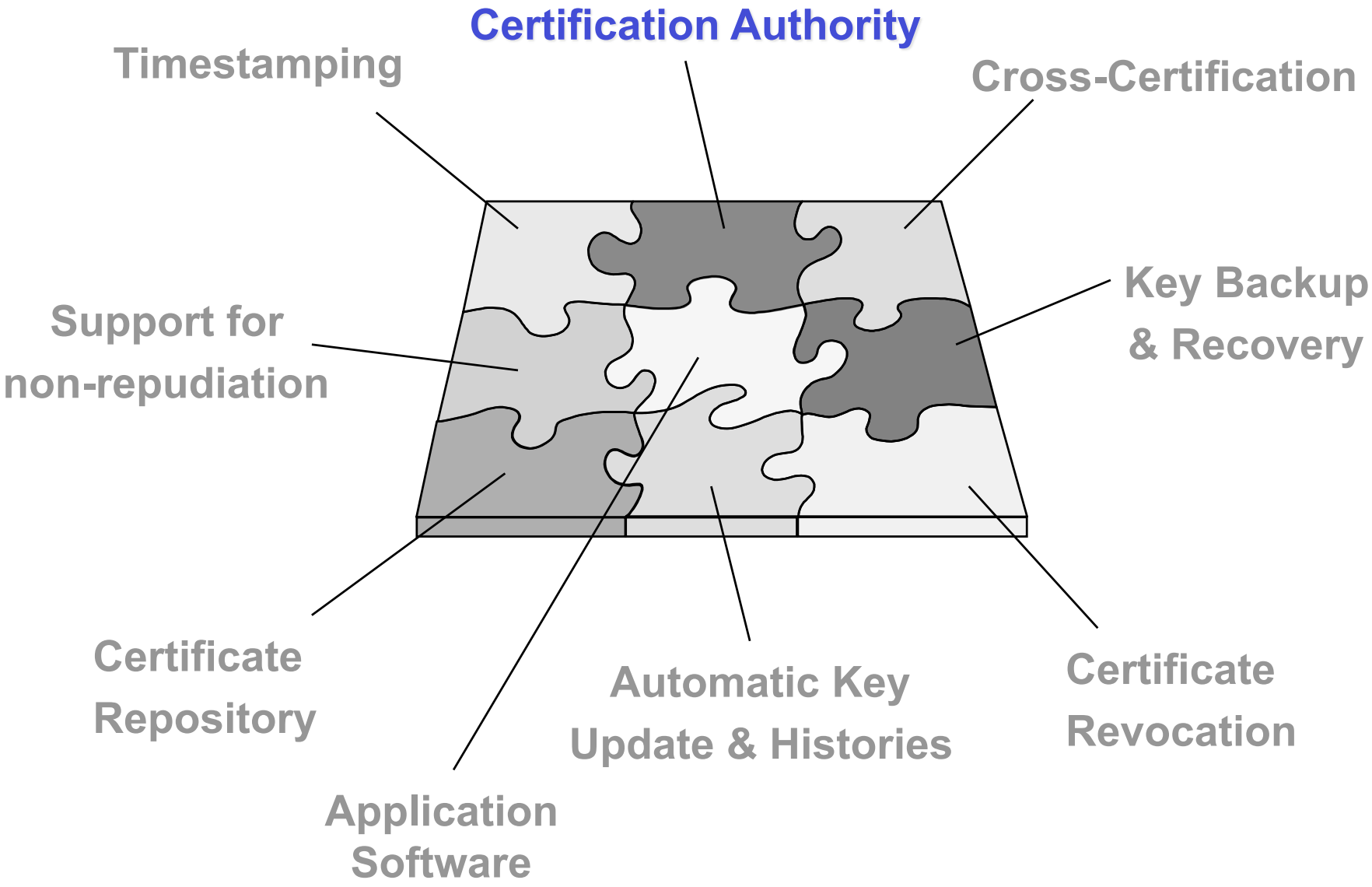


PKI provides unified security

This grand vision from 1997 has never materialized!



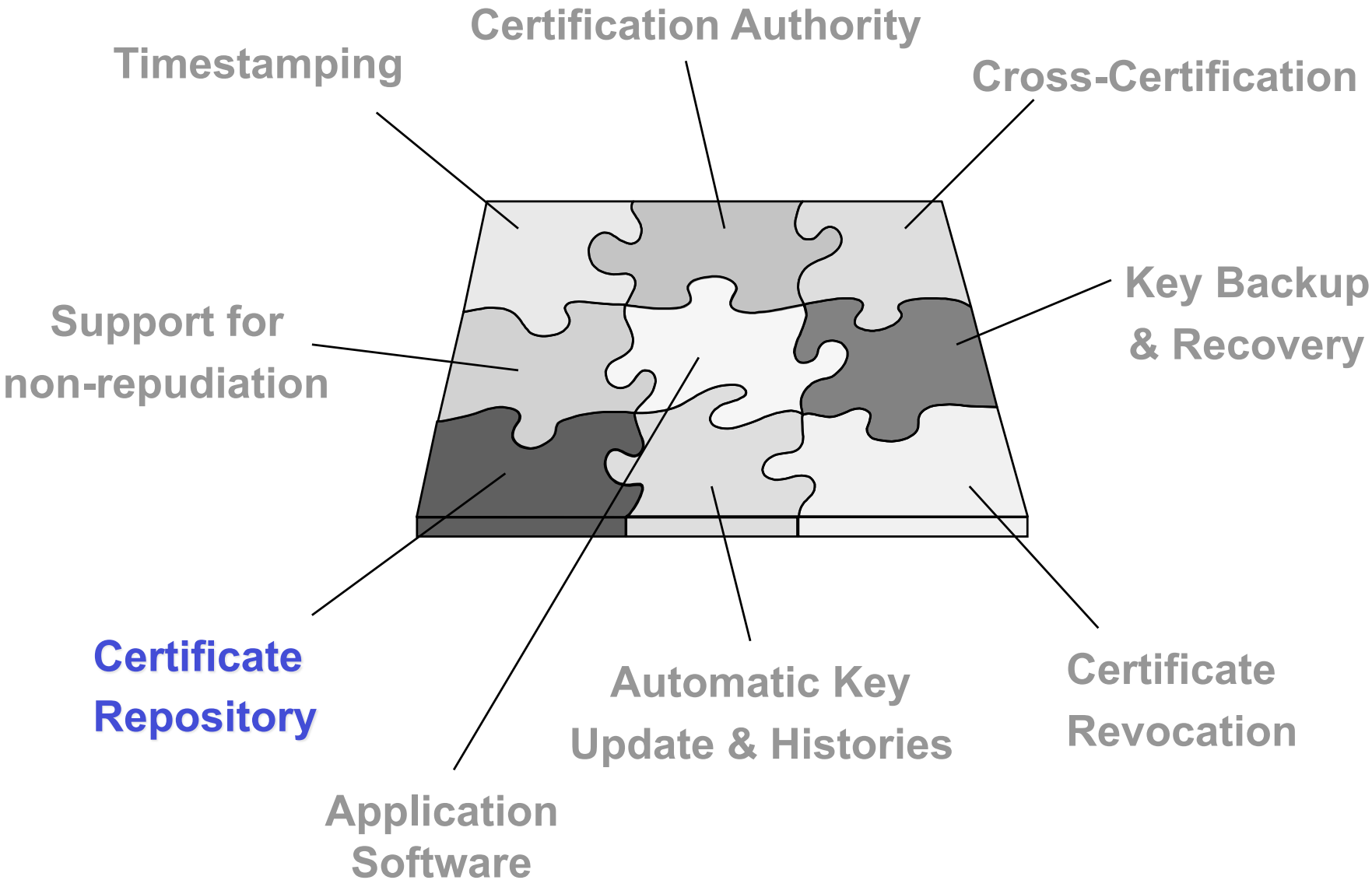
Certification Authority



Certification Authority

- Issue certificates for all entities / devices (for multiple applications) from a single CA
  - single system saves h/w, s/w, training, personnel
- Flexible certificate policy / security policy
  - tailor to needs of environment, application or entity (e.g. certificate lifetime, crypto algorithms, keylengths, password rules, ...)

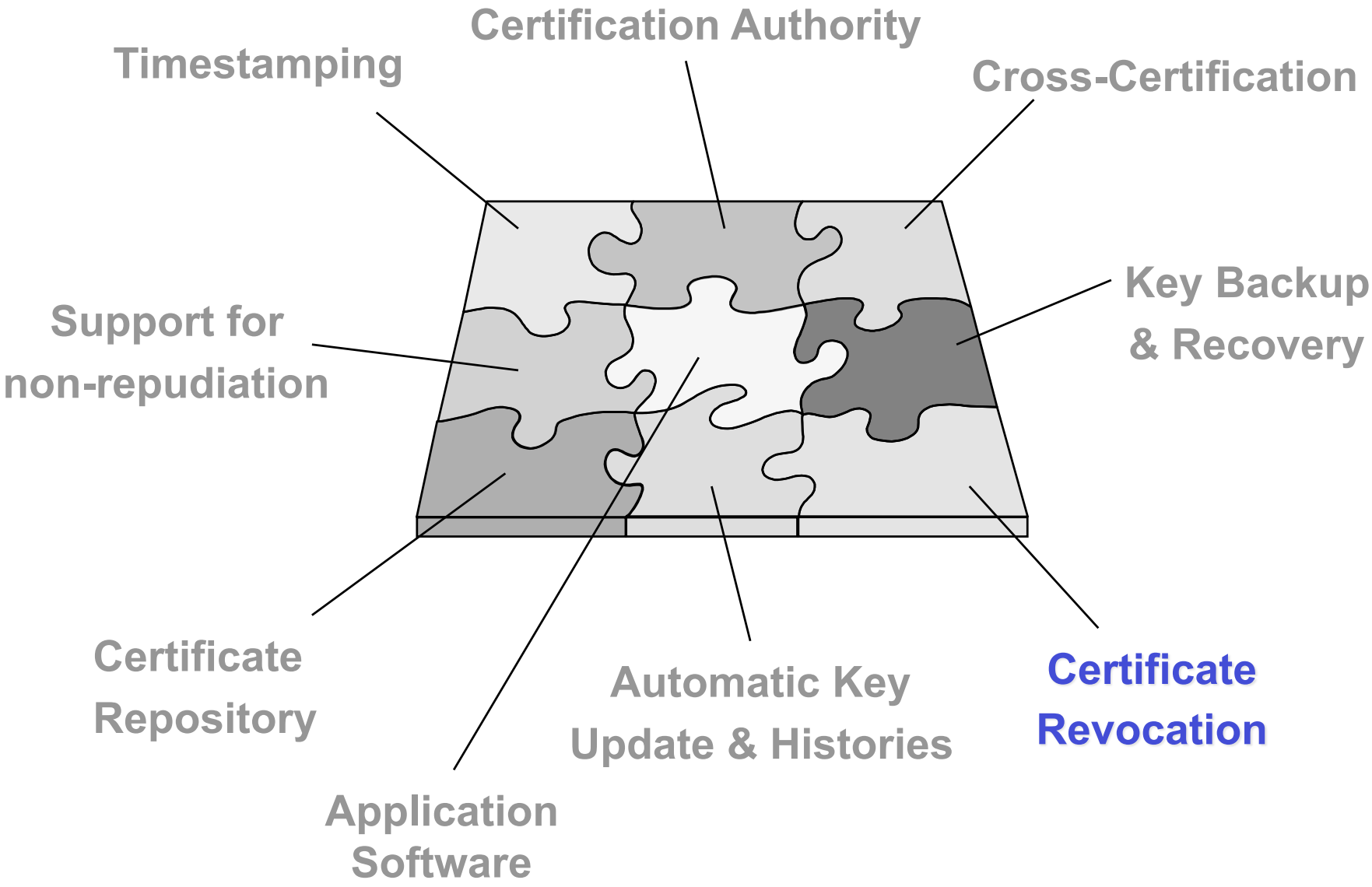
Certificate Repository



Certificate Repository

- LDAP-compliant directory stores certificates
  - standards-based for interoperability
- Directory products built specifically to address scalability issues
  - X.500 or proprietary schemes to replicate data (scales to millions of users)

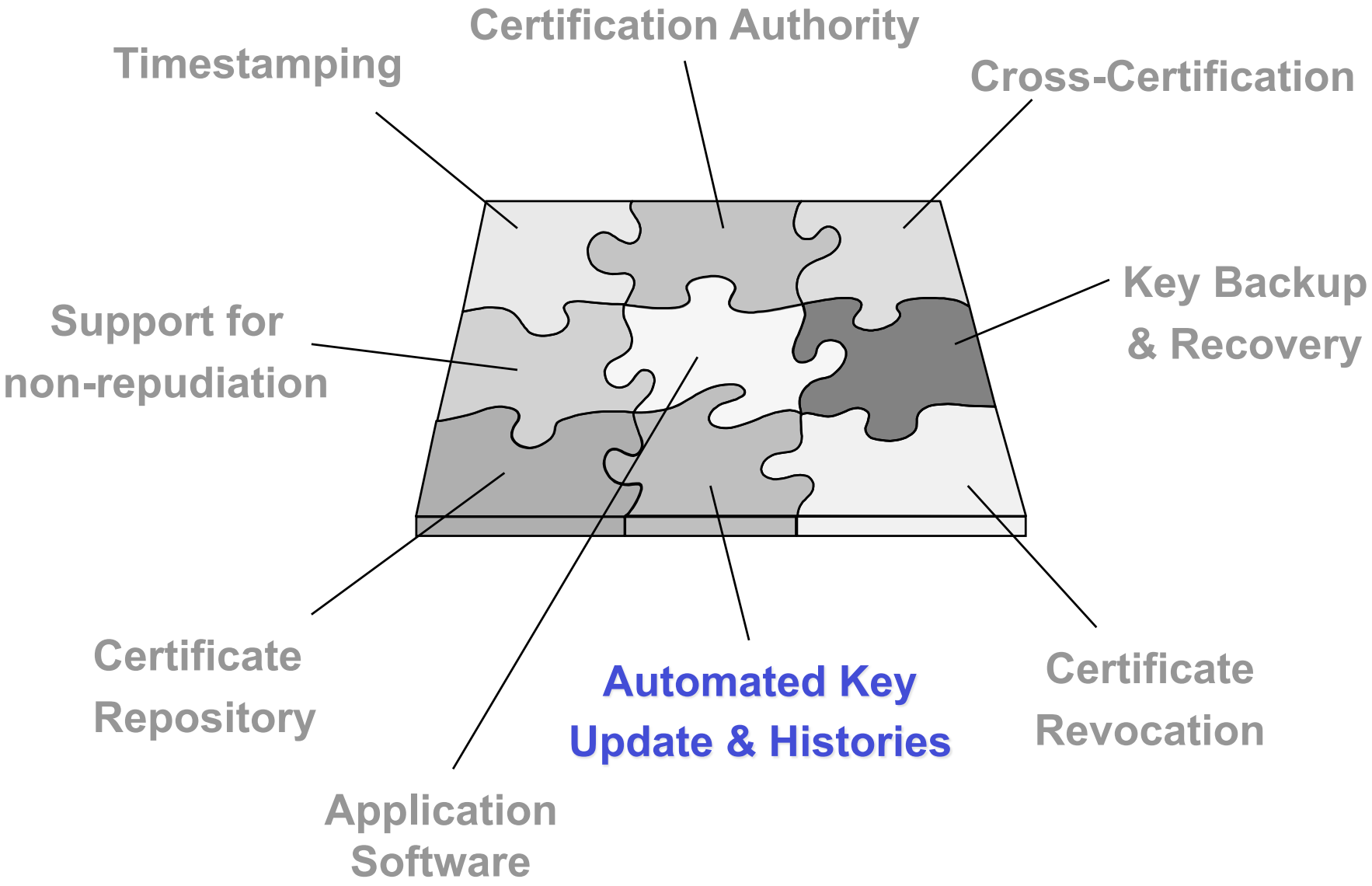
Certificate Revocation System



Certificate Revocation

- Automated CRL publishing
  - when certificate revoked, CRL can be automatically published to directory providing near-immediate availability
  - automated CRL checking by application
  - want to avoid applications which require manual end-user actions to check CRLs for each application or certificate usage

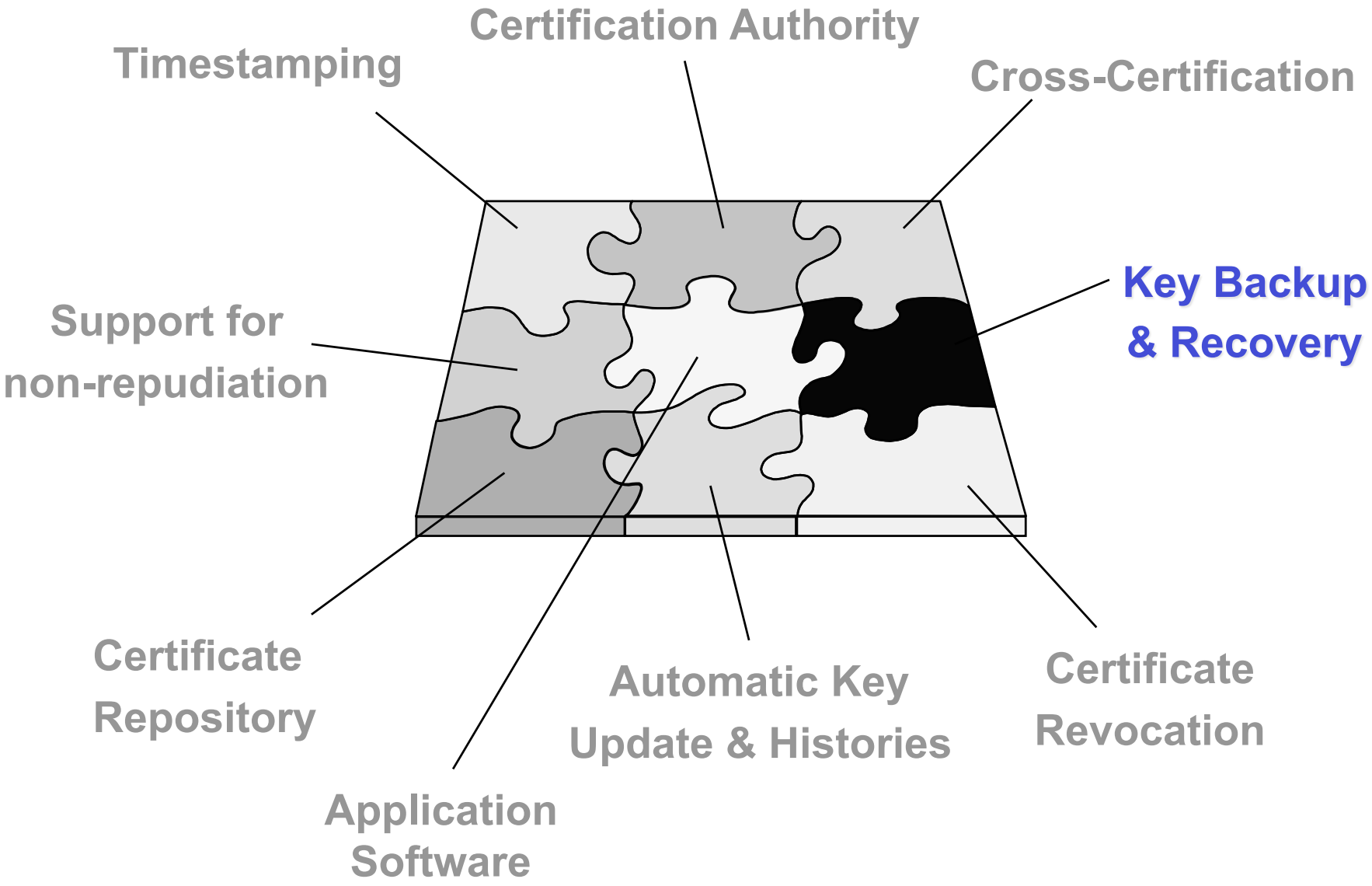
Automated Key Update & History



Automated Key Update & History

- Users should never even need to know they have their own certificates (password only)
- If key management is not automated or does not provide key history . . .
  - when certificate expires, lose access to all past encrypted data, e-mail, . . .
  - user must request new certificate and repeat entire registration process
- Should replace key, not just new expiry date
- Transparent triggering mechanism, ideally

Key Backup & Recovery

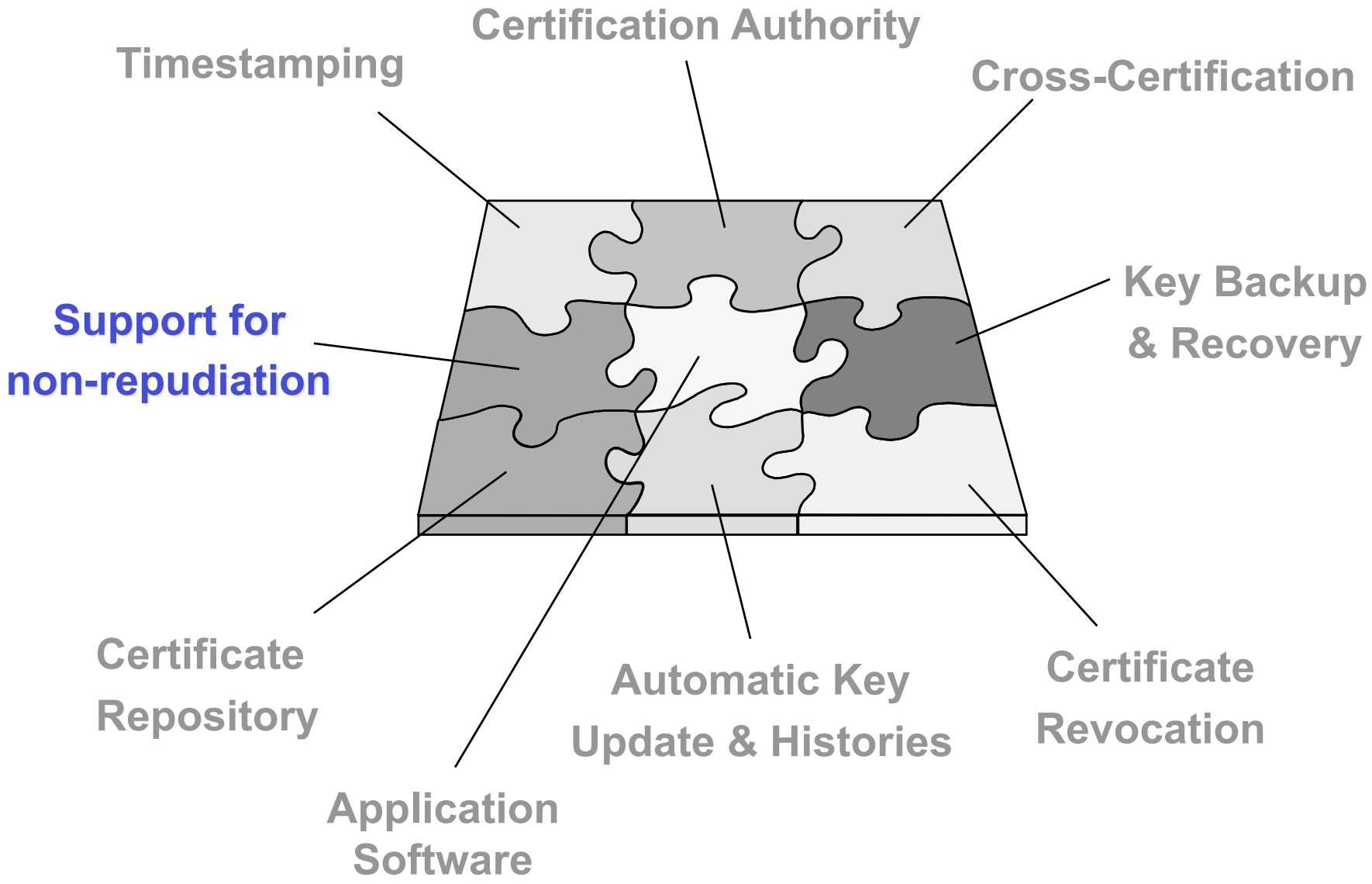


Key Backup & Recovery

- Enterprise will lose valuable data if keys used to encrypt data are not backed up
  - 20-40% of users forget passwords / year
  - employees leave the organization
- Allows the enterprise to control the backup
  - not reliant on 3rd parties
  - should be configurable to require multiple administrators to authorize access

Don’t confuse key backup for keys used on stored data with key escrow for government access to data communications

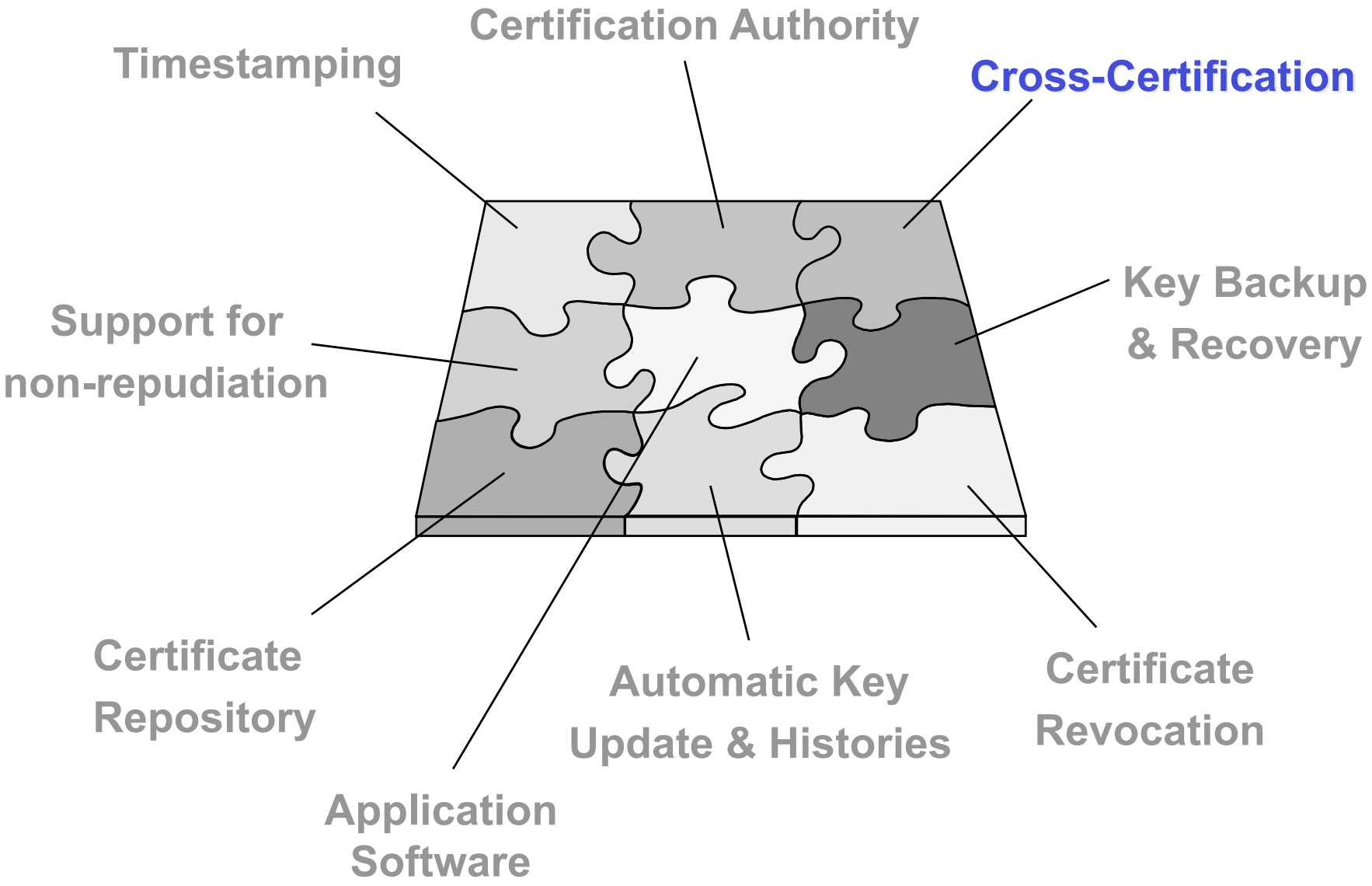
Support for Non-Repudiation



Support for Non-Repudiation

- Must use separate key pairs for digital signatures and encryption
  - want backup of encryption keys, **do not** want backup of signature private keys
- Separate key pairs allows lifecycles to be managed independently
- Different policy controls for each key pair
  - security requirements per pair may differ, e.g. valid lifetimes

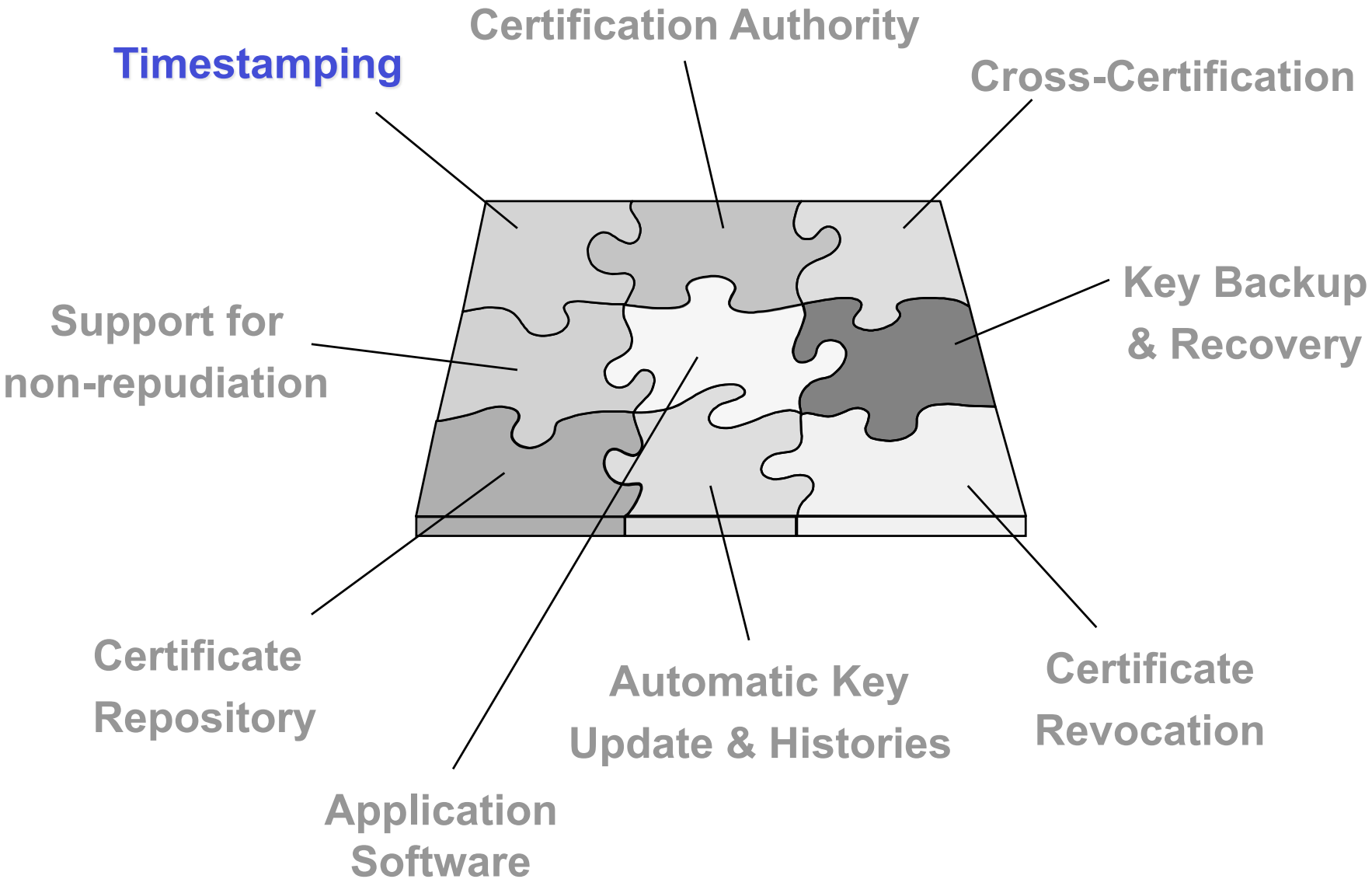
Cross-Certification



Cross-Certification

- Sufficiently flexible to model existing business relationships
  - includes 1-1 relationships and hierarchies
  - cross-certificate associated with an organization (vs. a service provider)
  - compare to web trust model: trust anyone signed by browser-embedded CAs
- Enterprise manages cross-certification policy & procedures, to reduce business risk
  - cross-certifcates created by authorized administrators, transparent to end-user

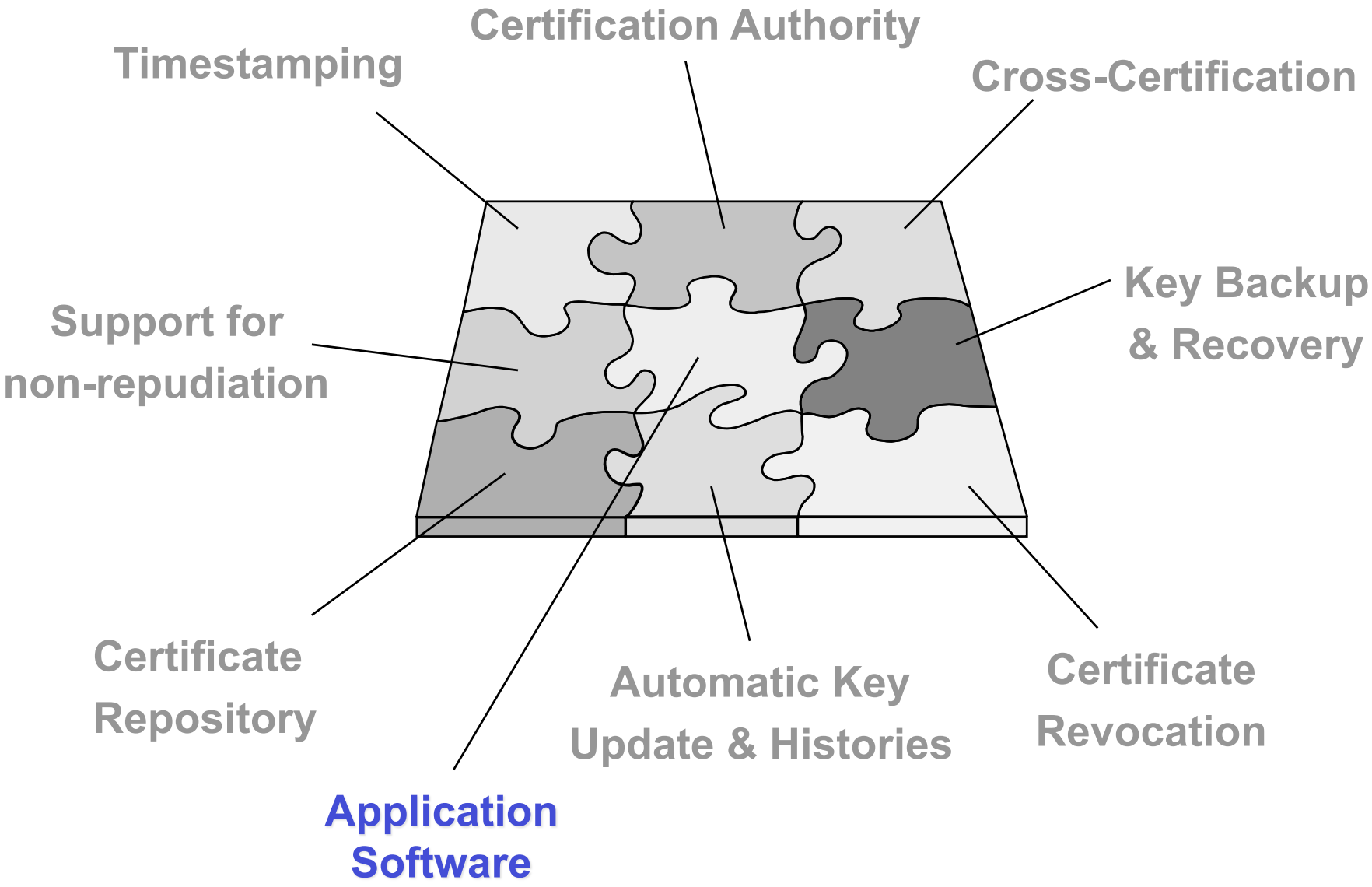
Timestamping



Timestamping

- Legal requirements
- Business requirements related to fixing transactions in time
- Technical requirements related to certificate revocation (non-repudiation)

Application Software



Application Software

- Designed to be enabled to use the PKI ("PKI-ready")

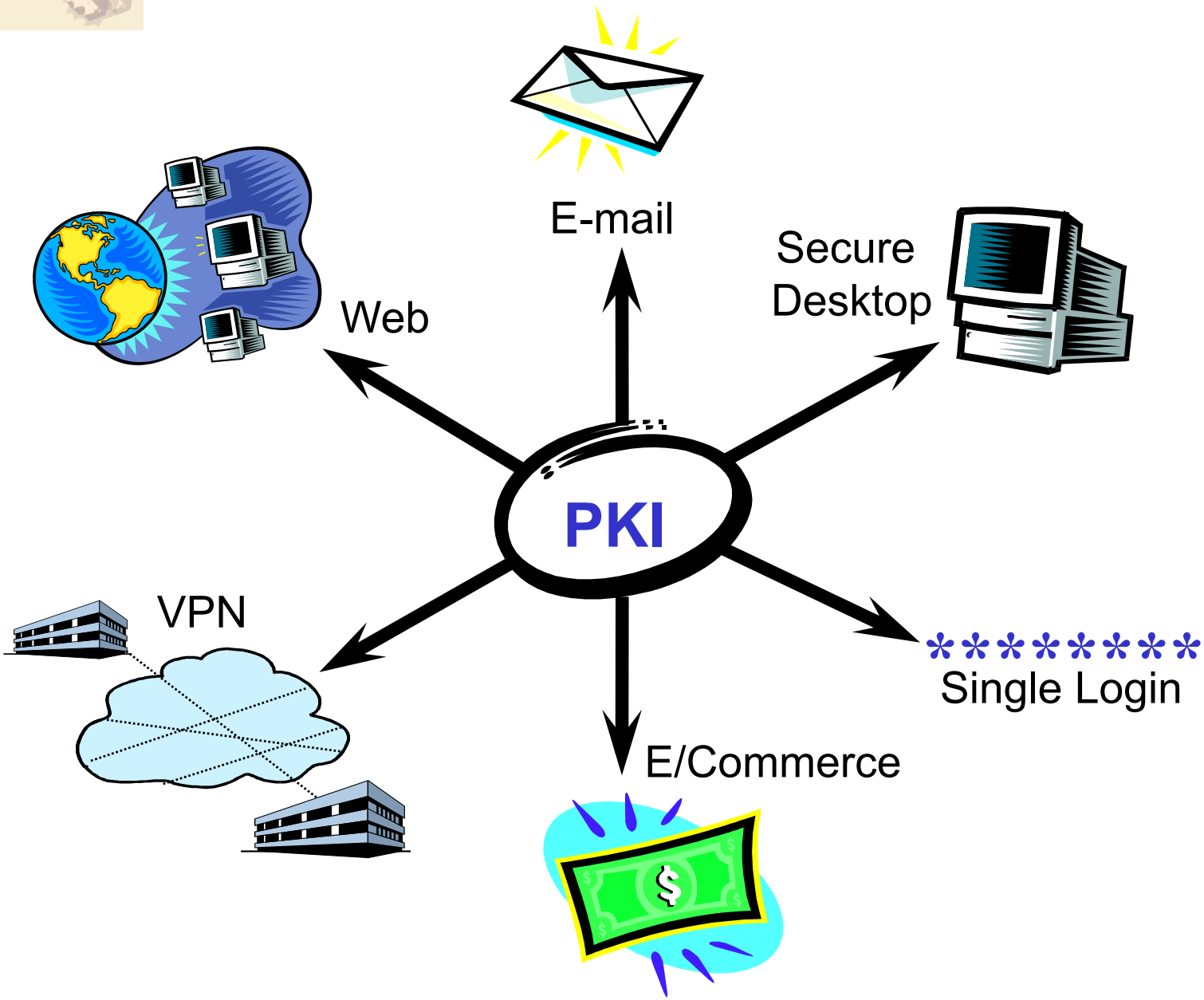
application software
(email, file encryption, VPN, web security/SSL, ...)

PKI

key & certificate lifecycle mgmt
(certificate validation, key update, ...)

crypto algorithms (symmetric encryption, signature, hash, MAC, key establishment, …)

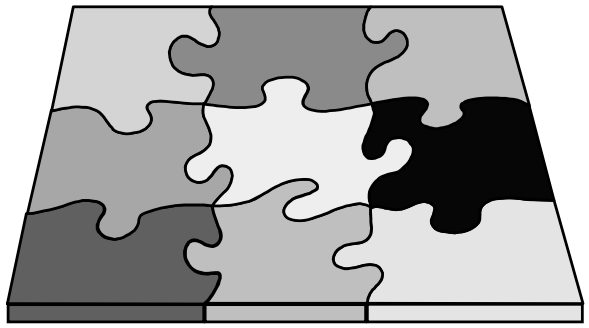
PKI-ready application software completes the picture



Summary - Essential PKI Components

Much more than a “certificate server” or set of toolkit calls

- Certification Authority
- Revocation system
- Certificate repository (“directory”)
- Key backup and recovery system
- Support for non-repudiation
- Automatic key update
- Management of key histories
- Cross-certification
- PKI-ready application software



More info: IETF PKIX Working Group

www.ietf.org

- de facto standards for Internet PKI, X.509-based
- Certificate & CRL Profile [PKIX-1]:
  RFC 2459
- Certificate Mgmt Protocols [PKIX-CMP, PKIX-3]:
  RFC 2510
- PKIX roadmap: www.ietf.org/internet-drafts/draft-ietf-pkix-roadmap-01.txt

PKI vs. Privilege Management

- Public key certificate binds a public key to an entity
- Establishes who owns a key vs. what privileges that key / owner is granted
- Certificate-processing software (relying party) may implicitly grant privileges
- Privilege Management Infrastructure (PMI) makes privileges explicit
- PMI may utilize PKI as base infrastructure

- example: attribute certificates

PKI vs. Privilege Management

- Public key certificate binds a public key to an entity
- Establishes who owns a key vs. what privileges that key / owner is granted
- Certificate-processing software (relying party) may implicitly grant privileges
- Privilege Management Infrastructure (PMI) makes privileges explicit
- PMI may utilize PKI as base infrastructure

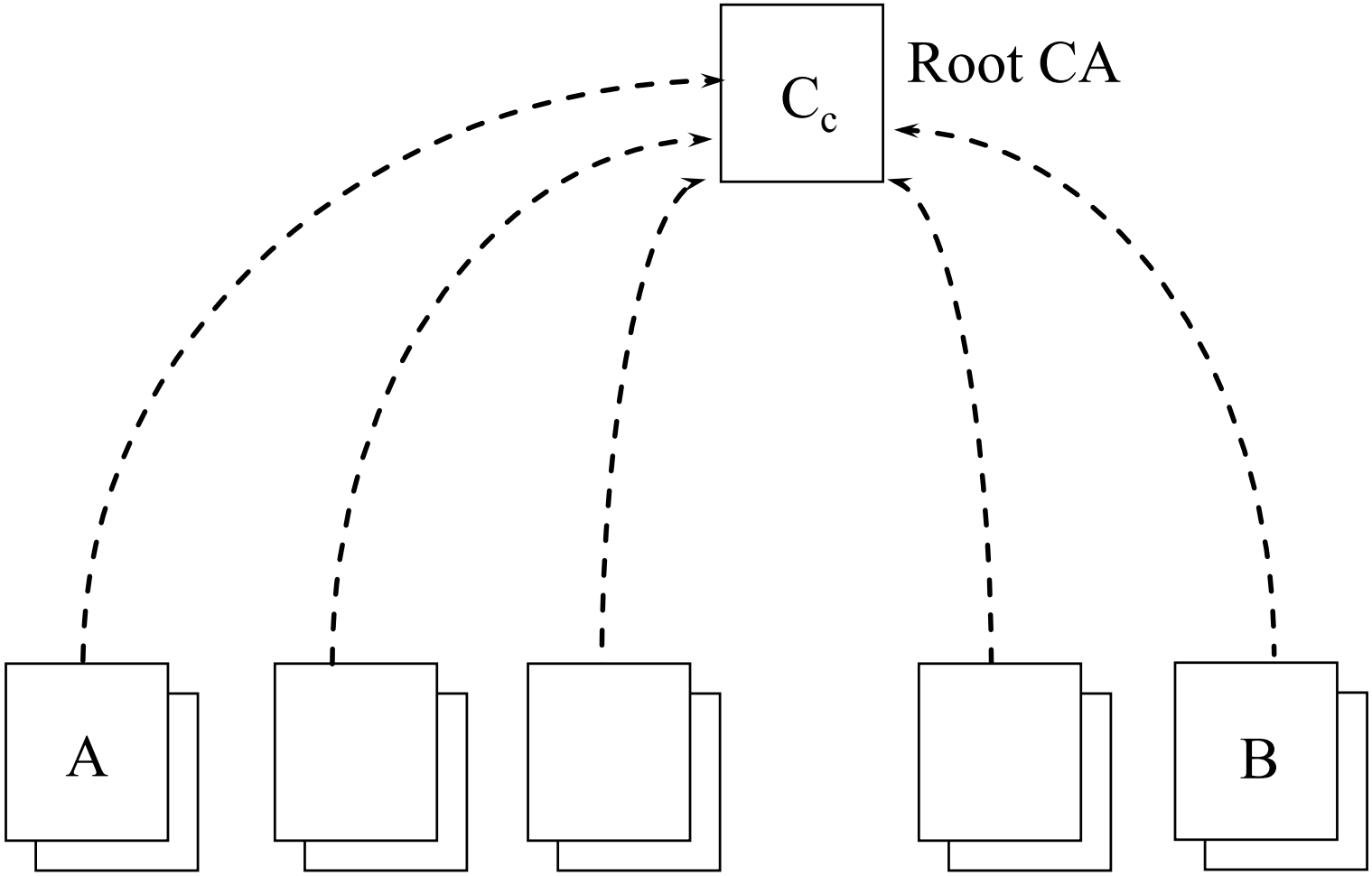
- example: attribute certificates

Key generation: where?

- CA generates key for user
  - absolute trust
  - need transport of private keys
  - easier management for backup/recovery
- user generates his/her key
  - does user have the expertise? (ok if smart card)
  - need to transport of public keys (integrity channel)
- specialised third party generates keys

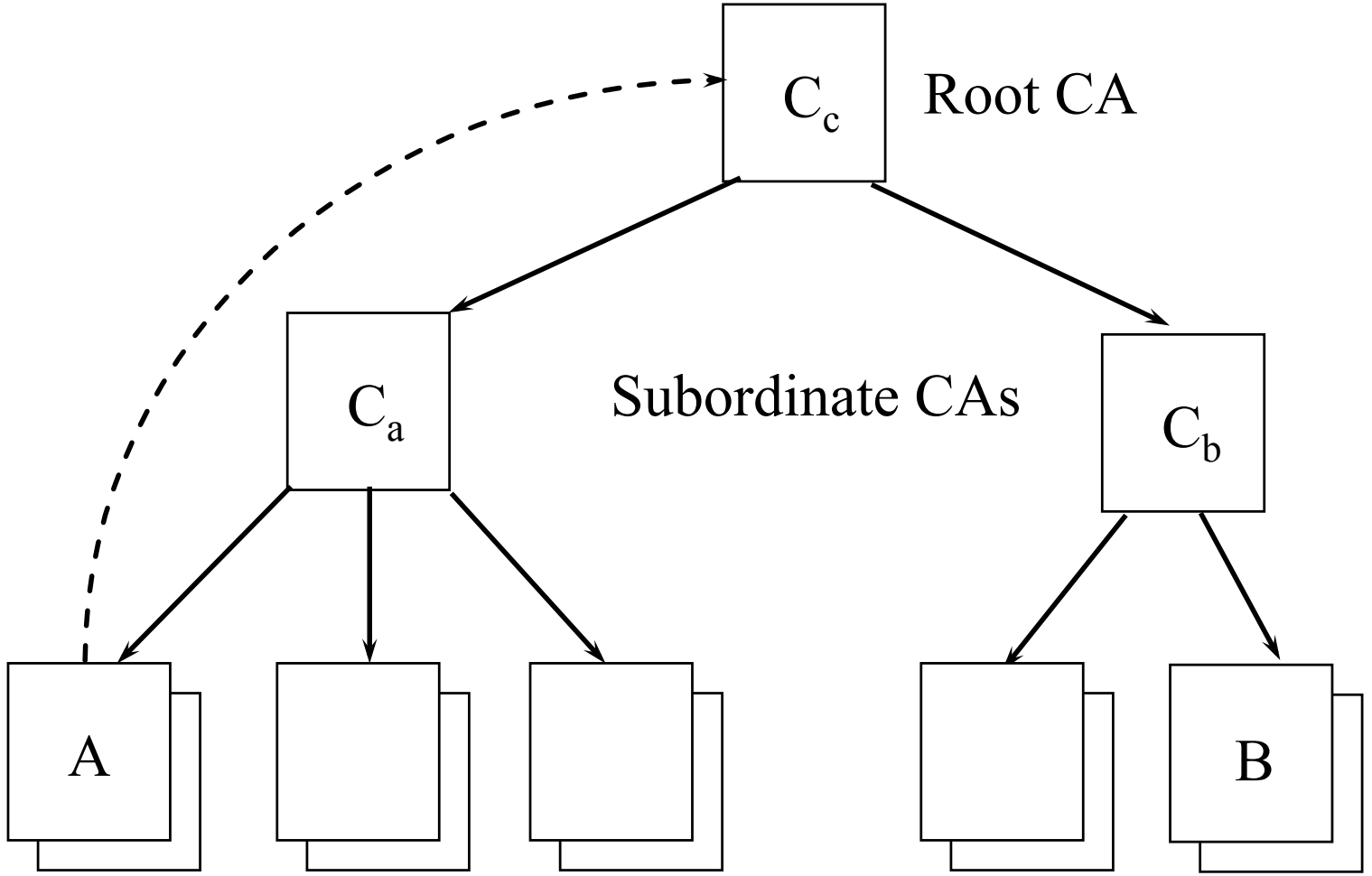
Trust Models

Hierarchical trust model



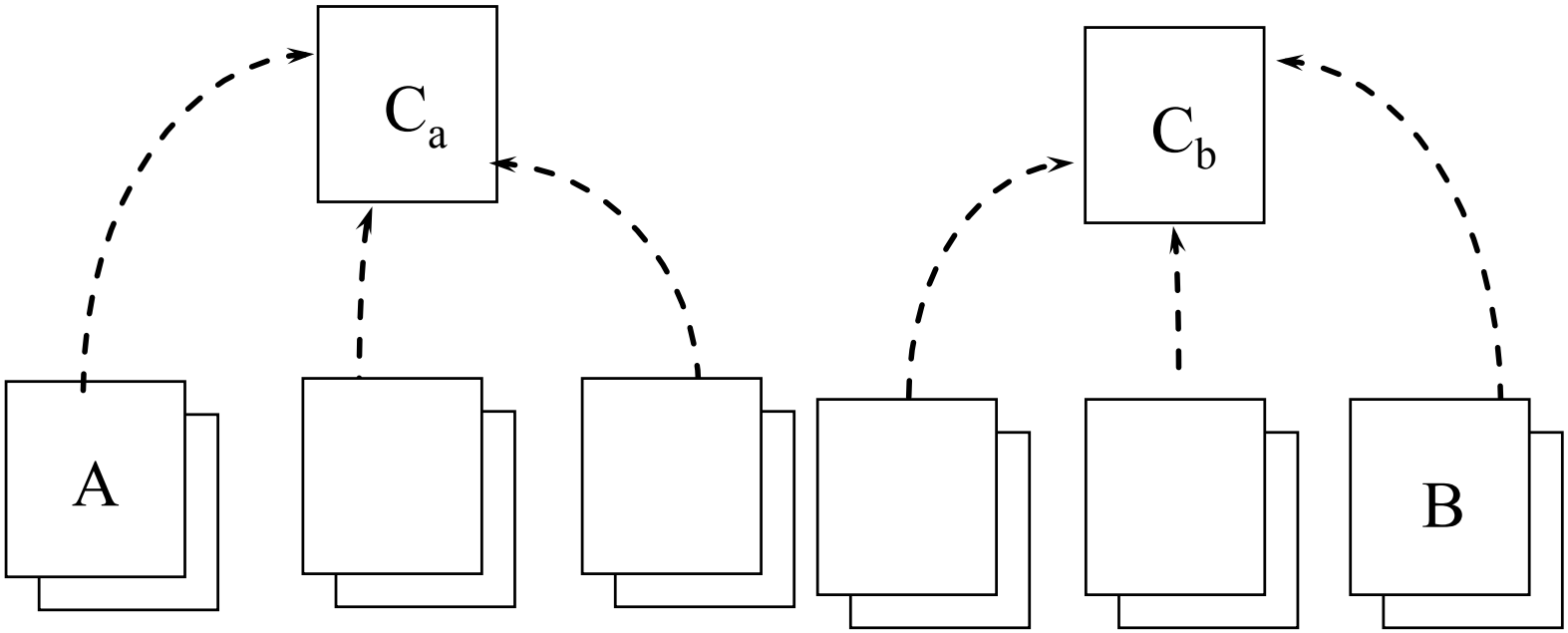
Relying parties transfer risk to the Root CA

Hierarchical trust model



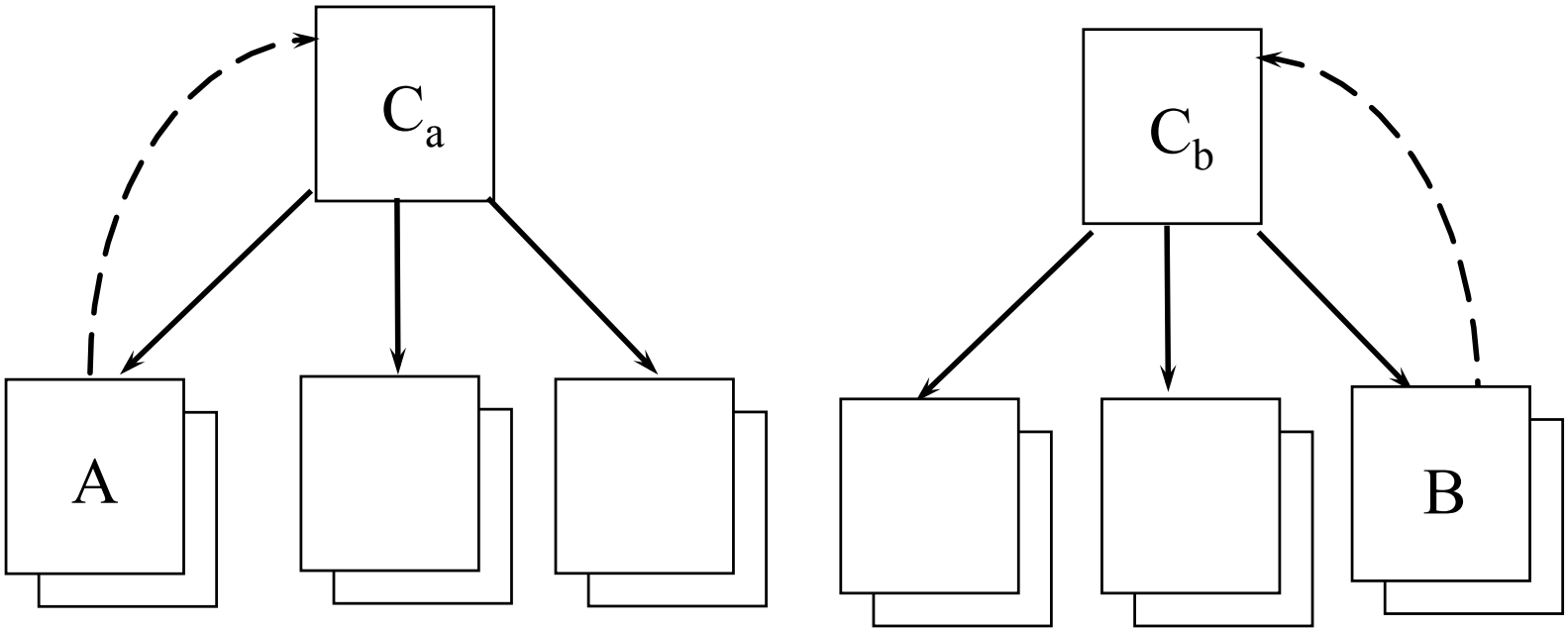
Root CA "deputizes" subordinate CAs, which issue certificates

Enterprise trust model



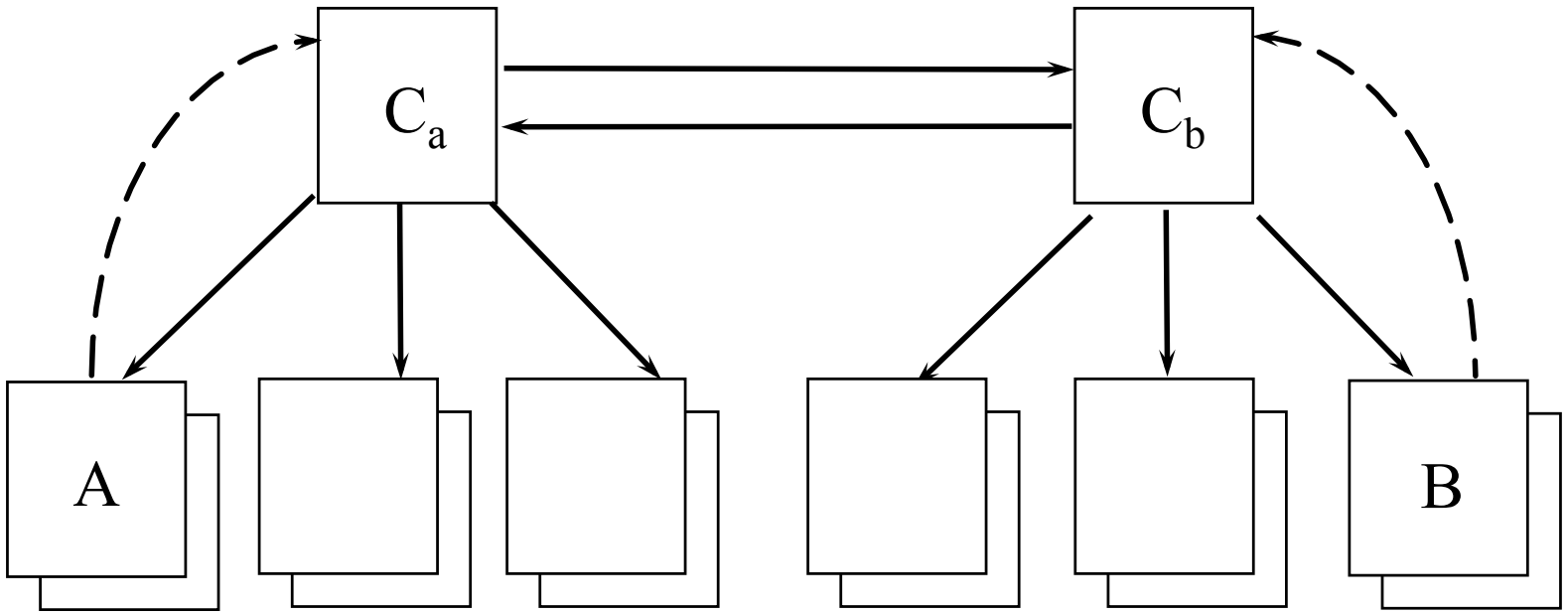
Relying parties transfer risk to their local CA

Enterprise trust model



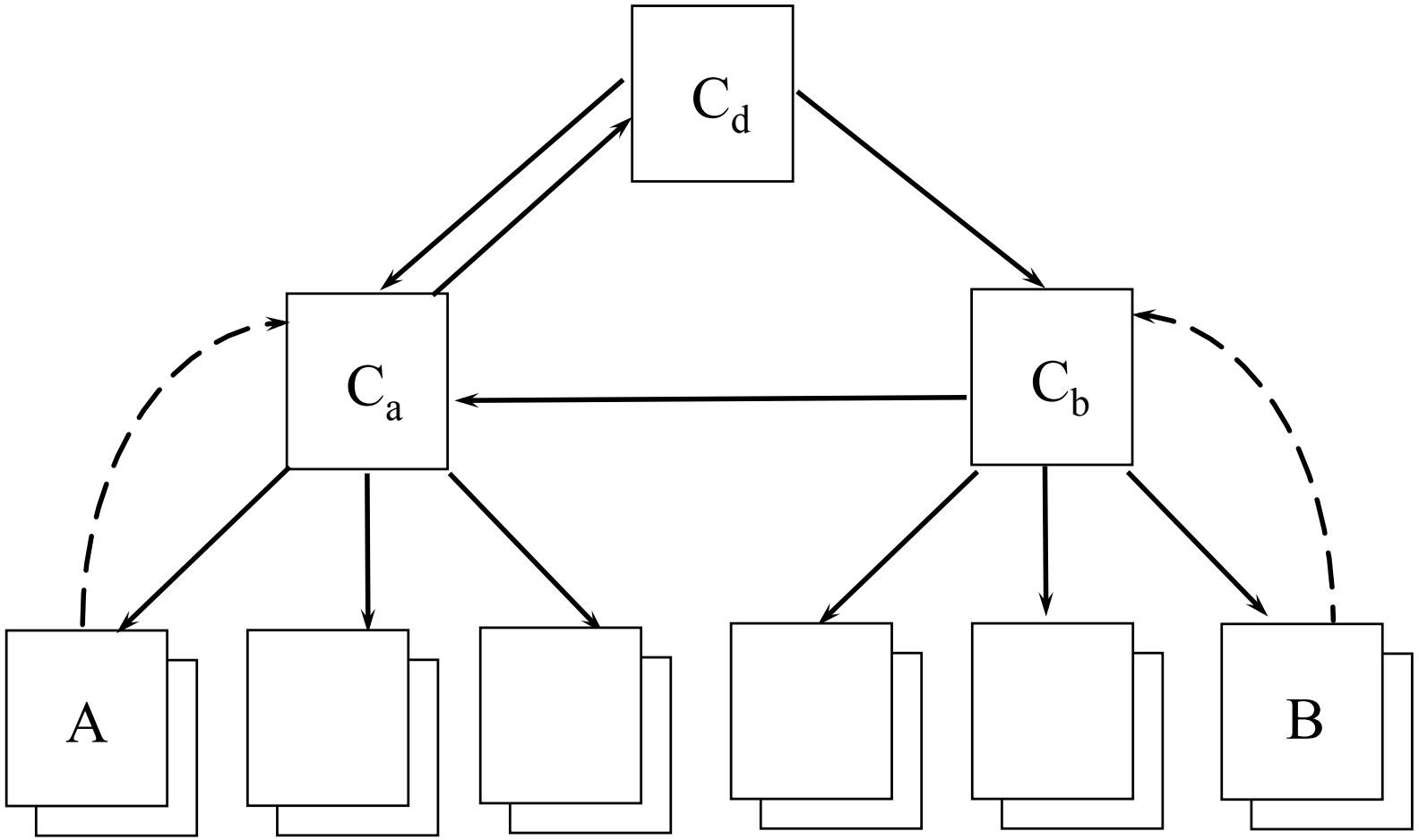
The same local CA issues certificates to these parties

Enterprise trust model



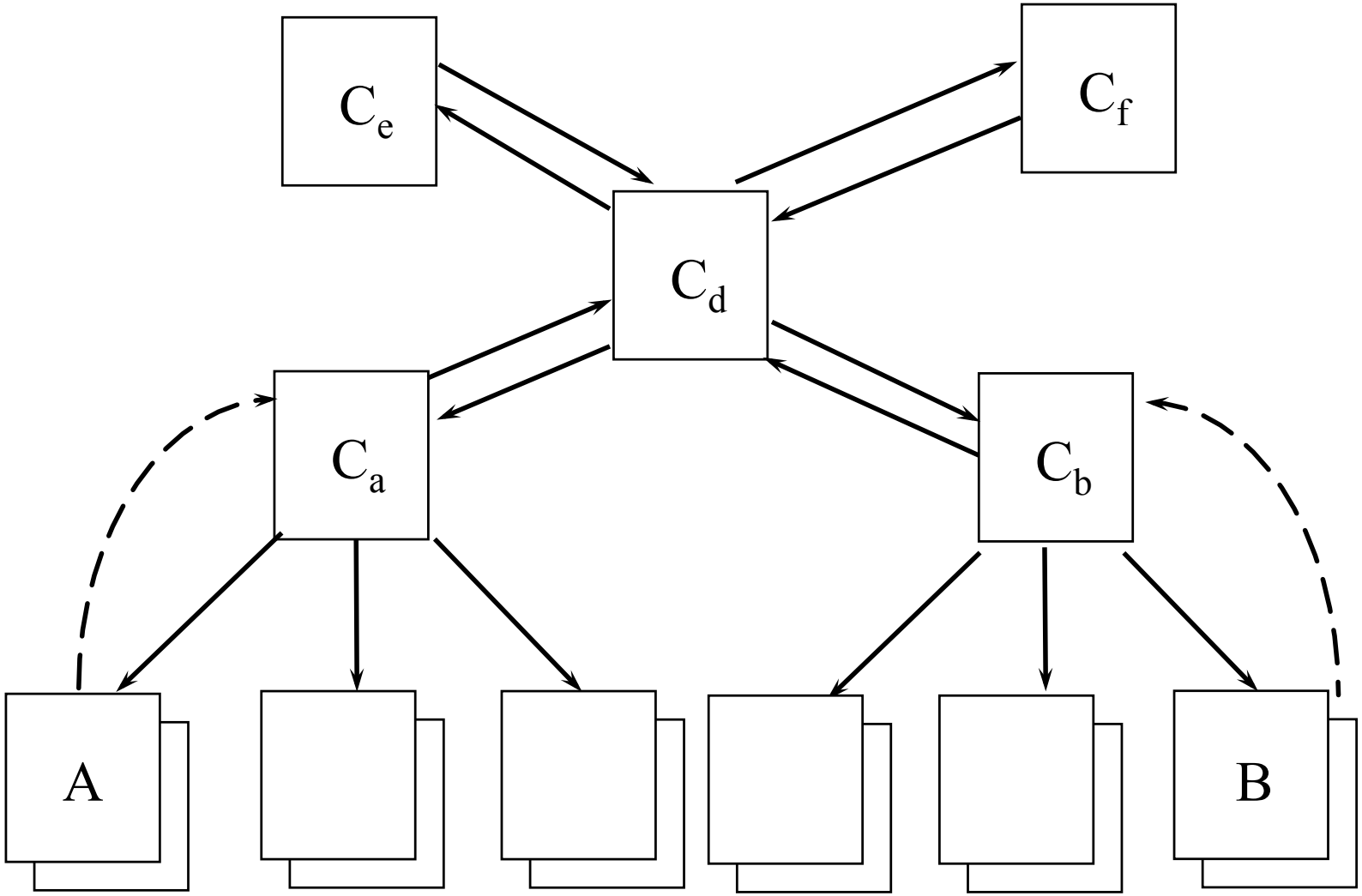
Qualified relationships between CAs are established

Enterprise trust model



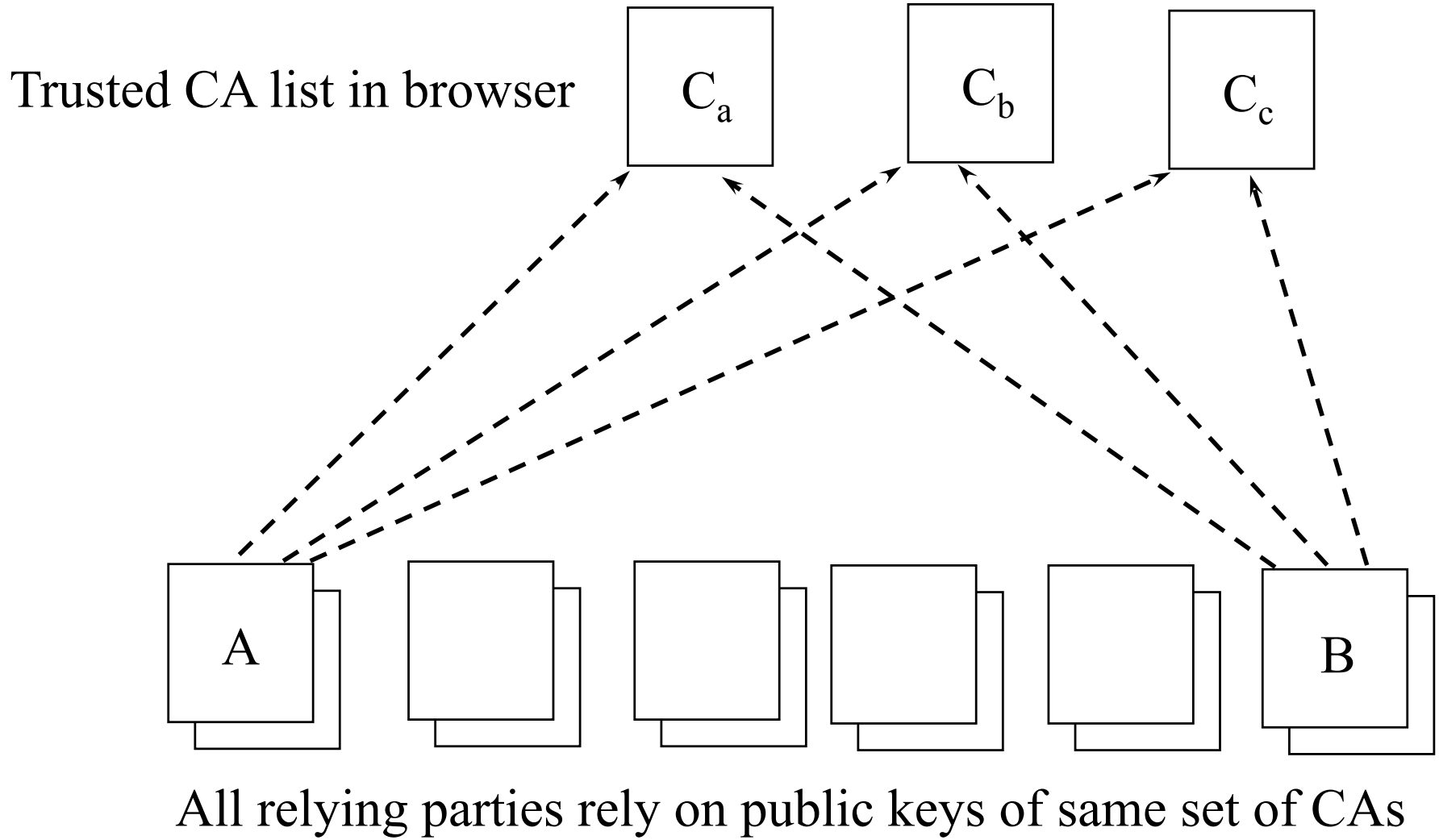
Hierarchical relationships are a special case

Enterprise trust model

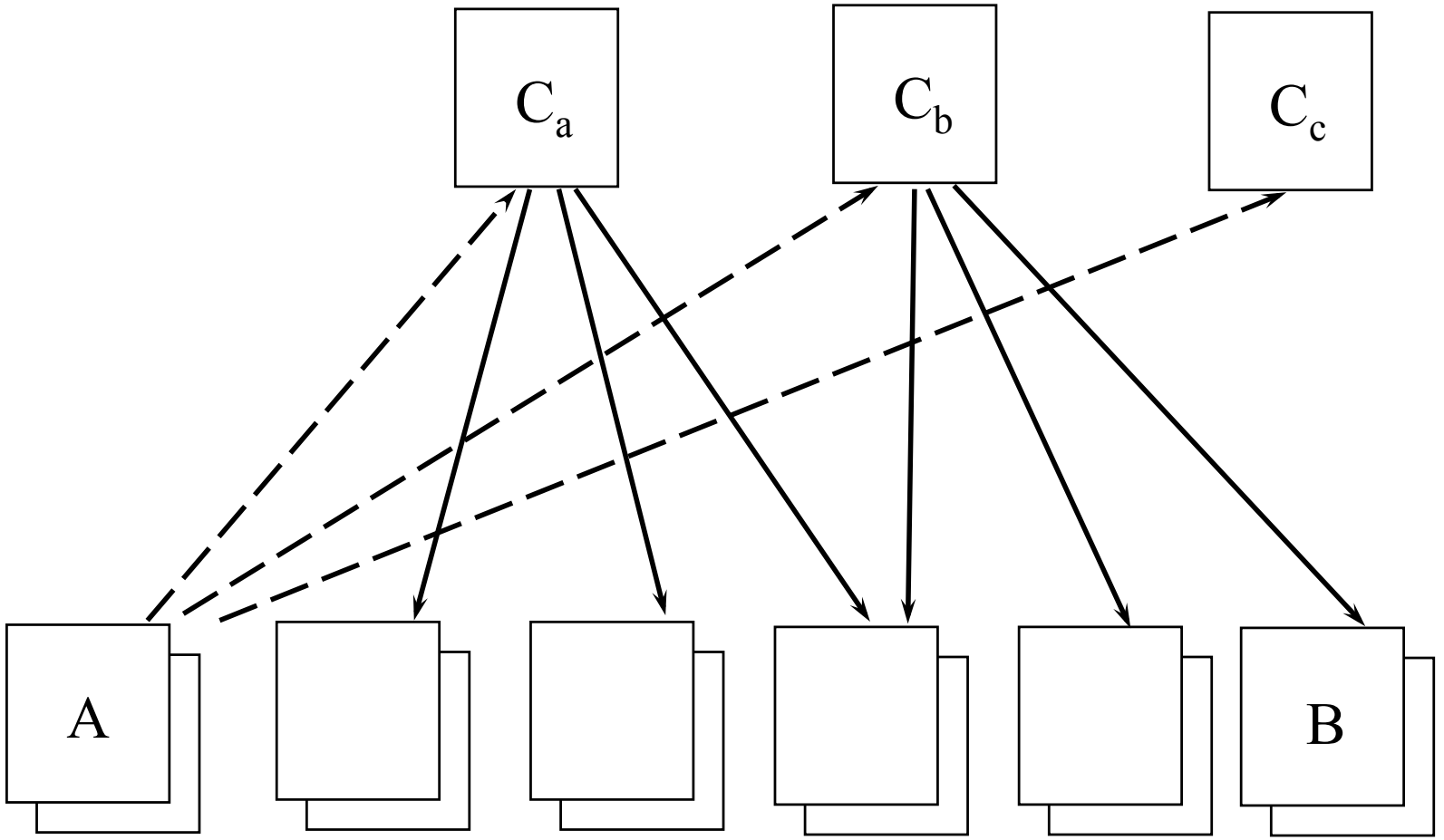


Spoke-and-hub model is another special case

Browser trust model

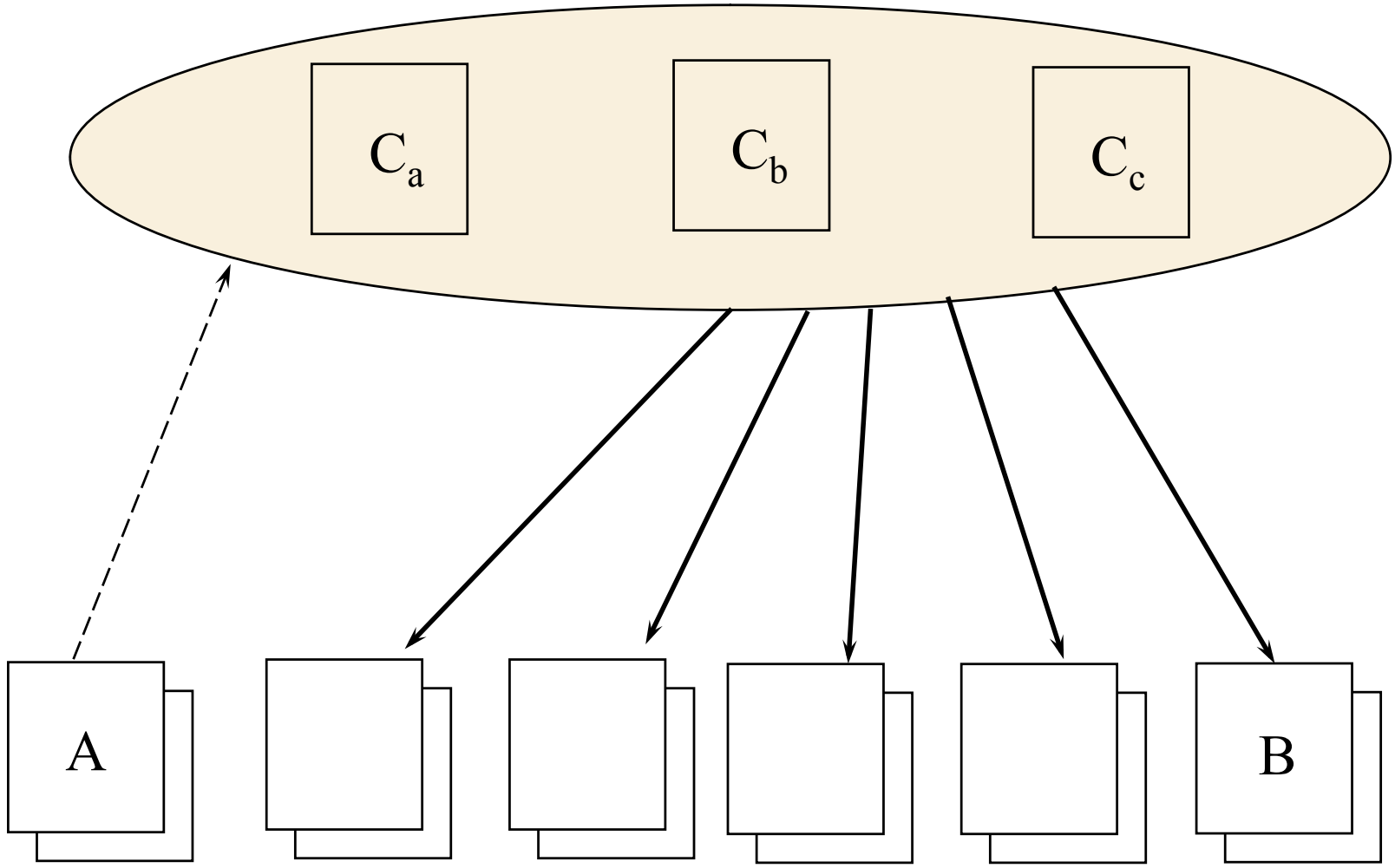


Browser trust model



Each of these CAs defines its own community of trust

Browser trust model



A relying party trusts the union of these communities

The CA Mess on the web

[Eckersley10] “An observatory for the SSLiverse”

- 10.8M servers start SSL handshake
- 4.3M use valid certificate chains
- 1482 CA certs trustable by Windows or Firefox
- 1.4M unique valid leaf certs
  - 300K signed by one GoDaddy cert
- 80 distinct keys used in multiple CA certs
- several CAs sign the IP address 192.168.1.2 (reserved by RFC 1918)
- 2 leaf certs have 508-bit keys
- Debian OpenSSL bug (2006-2008)
  - resulted in 28K vulnerable certs
  - fortunately only 530 validate
  - only 73 revoked

How can we fix this mess?

Personal trust model (and related: “web-of-trust”)

- all entities are end-users (CAs do not exist)
- keys are essentially self-guaranteed
- some end-users may also be *introducers*
- end-user imports public keys of others

CHARACTERISTICS

- suits individuals, not enterprise/corporations
- user-centric
- requires security-aware end-users
- poor scalability

Trust models & Revocation

- public-key systems are commonly engineered with long-life certificates
- certificates bind a key-pair to identity (and potentially privilege information)
- circumstances change over certificate life
  - keys may become compromised
  - identifying information may change
  - privilege may be withdrawn
- need ability to terminate the binding expressed in the certificate
- revocation: most difficult issue in practice

Revocation options

mechanisms indicating valid certificates

- short-lifetime certificates

mechanisms indicating invalid certificates

- certificate revocation lists - CRLs (v1 X.509)
- CRL fragments (v2 X.509), including ...
  - segmented CRLs (CRL distribution points)
  - delta CRLs
  - indirect CRLs

mechanisms providing a proof of status

- status-checking protocols (OCSP, ValiCert)
- iterated hash schemes (Micali)
- certificate revocation trees

CRL: properties

- basic CRL
  - simplicity
  - high communication cost from directory to user
- improved CRL
  - very flexible
  - more complex
  - reduced communication and storage

Online Certificate Status Protocol (OCSP) [RFC 2560]

- on-line query to
  - CA
  - or Trusted Responder
  - or CA designated responder
- containing
  - hash of public key CA
  - hash of public key in certificate
  - certificate serial number

OCSP: signed answer

- status
  - good: not revoked
  - revoked
  - unknown
- time
  - thisUpdate
  - nextUpdate
  - producedAt

OCSP: evaluation

- [+] positive and negative information
- [-] need to be on-line
  - risk for denial of service
  - not always possible
- ! OCSP may send you freshly signed but old information

Revocation summary

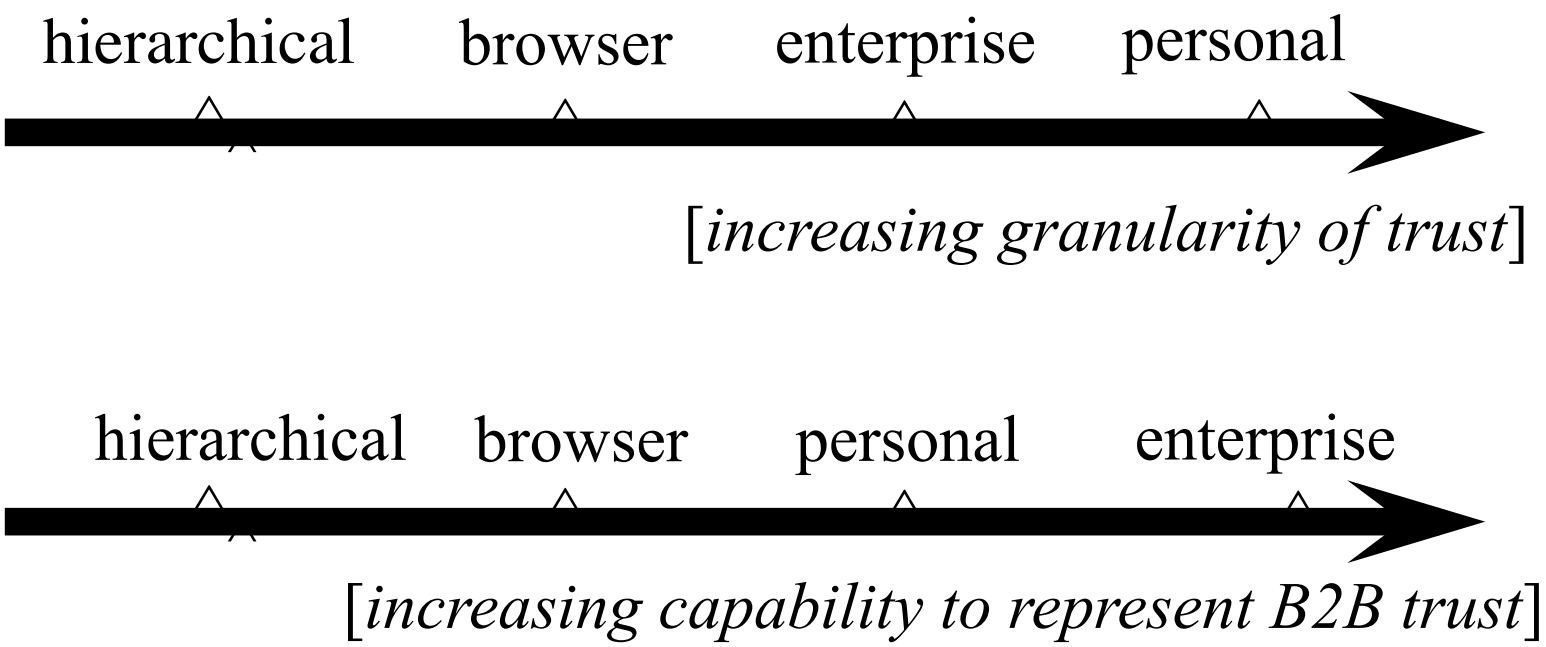
- established standard meets needs of major application categories
  - ITU-T X.509: 1997, ISO/IEC 9594-8: 1997
  - v2 CRLs
- continued industry discussion of further options for certificate revocation and validation
  - other standard solutions may emerge
  - vendors will support mainstream alternatives

Characterizing questions for trust models

- what are the types/roles of entities involved
- who certifies public keys
- are trust relationships easily
  created, maintained, updated
- granularity of trust relationships
- ability of particular technology to support existing business models of trust
- how is revocation handled?

  . . . of end-users . . . of certification authorities

Trust model continuums



Many other continuums can be formulated

Trust model summary

Key idea: manageability of trust relationships

Each model has its place --

- personal trust model: okay for security-aware individuals working in small communities
- browser model: simple, large communities, everyone trusts all CAs defined by s/w vendor
- hierarchical model: best given an *obvious* global root and a *grand design* methodology
- enterprise trust model: best between peer organizations, where trust flexibility is required
- global PKI will include variety of trust models

Identity based encryption

- Extra material for information

Identity-Based Encryption (IBE)

- IBE is an old idea
  - Originally proposed by Adi Shamir, S in RSA, in 1984
  - Not possible to build an IBE system based on RSA
- First practical implementation
  - Cocks IMA 2001 and Boneh-Franklin Algorithm Crypto 2001
  - Bilinear Maps (Pairings) on Elliptic Curves
    - Based on well-tested mathematical building blocks
  - Public Key Algorithm used for Key Transport
- The IBE breakthrough is having major impact
  - Now over 400 scientific publications on IBE and Pairing Based Cryptography
  - Major deployments in industry
- Standardization Efforts
  - IBE mathematics is being standardized in IEEE 1363.3
  - IETF S/MIME Informational RFC

IBE Public Keys … Introduce This Elegance

Public-key Encryption where Identities are used as Public Keys

- **IBE Public Key:**

  alice@gmail.com

- **RSA Public Key:**

  Public exponent=0x10001

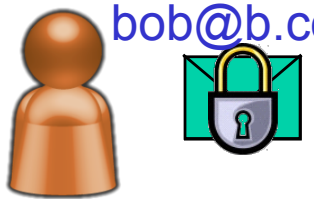
  Modulus=135066410865995223349603216278805969938881475605667027524485143851526510604859533833940287150571909441798207282164471551373680419703964191743046496589274256239341020864383202110372958725762358509643110564073501508187510676594629205563685529475213500852879416377328533906109750544334999811150056977236890927563

How IBE works in practice
Alice sends a Message to Bob

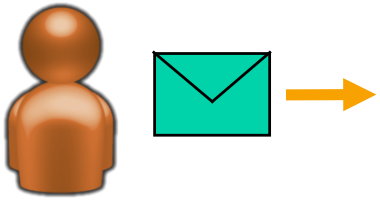
Key Server
- Master Secret
- Public Parameters

2 Requests private key, authenticates

3 Receives Private Key for bob@b.com



bob@b.com



alice@a.com

bob@b.com

1 Alice encrypts with bob@b.com

4 Bob decrypts with Private Key

How IBE works in practice
Alice sends a Message to Bob

Key Server

Fully off-line - no connection to server required

1

Charlie encrypts with bob@b.com

2

Bob decrypts with Private Key

IBE Public Key Composition

v2 ||

public key definition version

ibe-server.acme.com#1234 ||

server location and public parameter version

week = 252 ||

key validity period

bob@acme.com

e-mail address

IBE Benefits

Dynamic "As Needed" Public and Private Key Generation

- No pre-generation or distribution of certificates
- Built-in Key Recovery – No ADKs
- Allows content, SPAM, and virus scanning at enterprise boundary
- Facilitates archiving in the clear per SEC regulations

Policy in the Public Key

- e.g. Key Validity Period
- No CRLs

Dynamic Groups

- Identities can be groups and roles; no re-issuing keys when group or role changes

Minimal System State

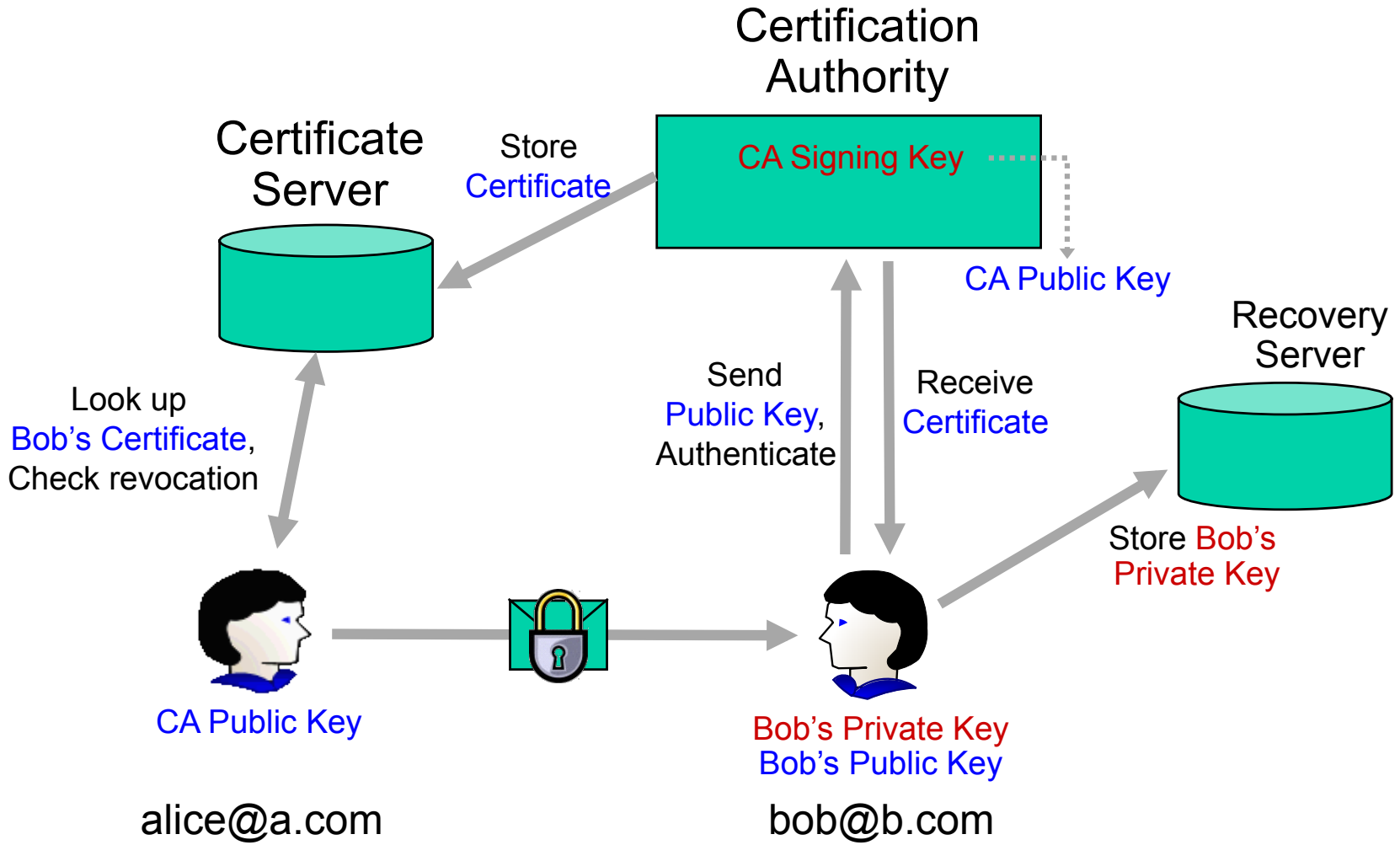
- Master Secret / Public Parameters (~50KB) all you need for disaster recovery
- End user keys and message not stored on server
- Server scalability not limited by number of messages

Benefits lead to:

High system usability
Highly scalable architecture
Low operational impact
Fully stateless operation

Public Key Infrastructure

Certificate Server binds Identity to Public Key



Identity Based Encryption

Binding of Identity to Key is implicit