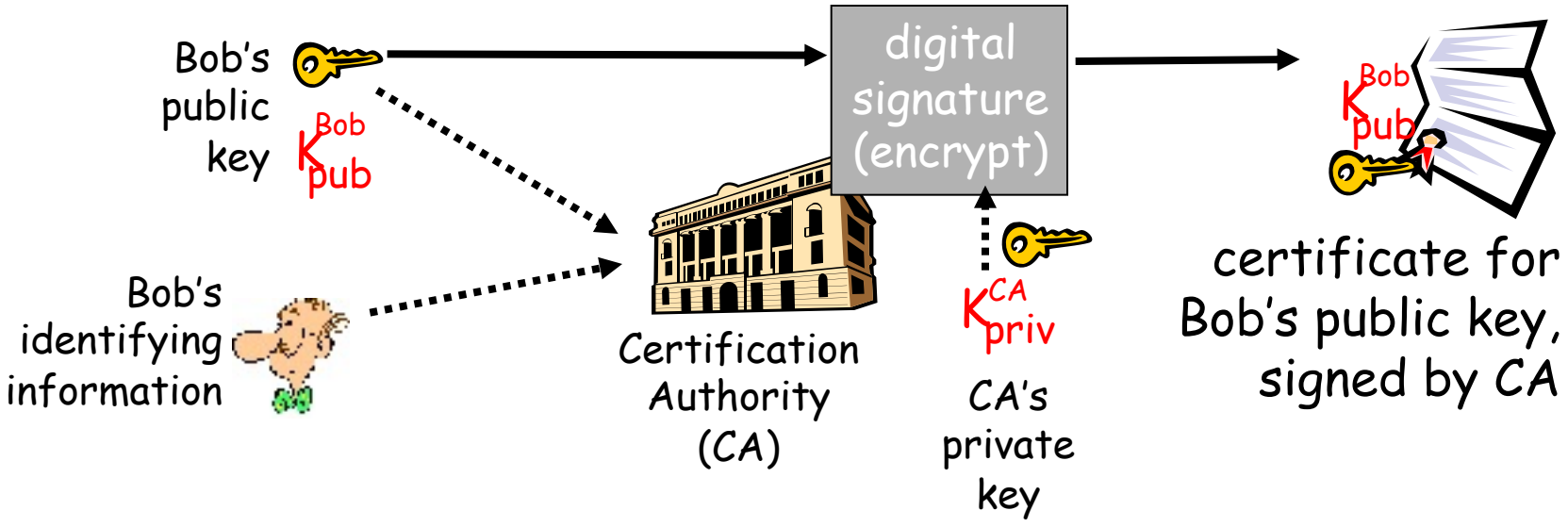
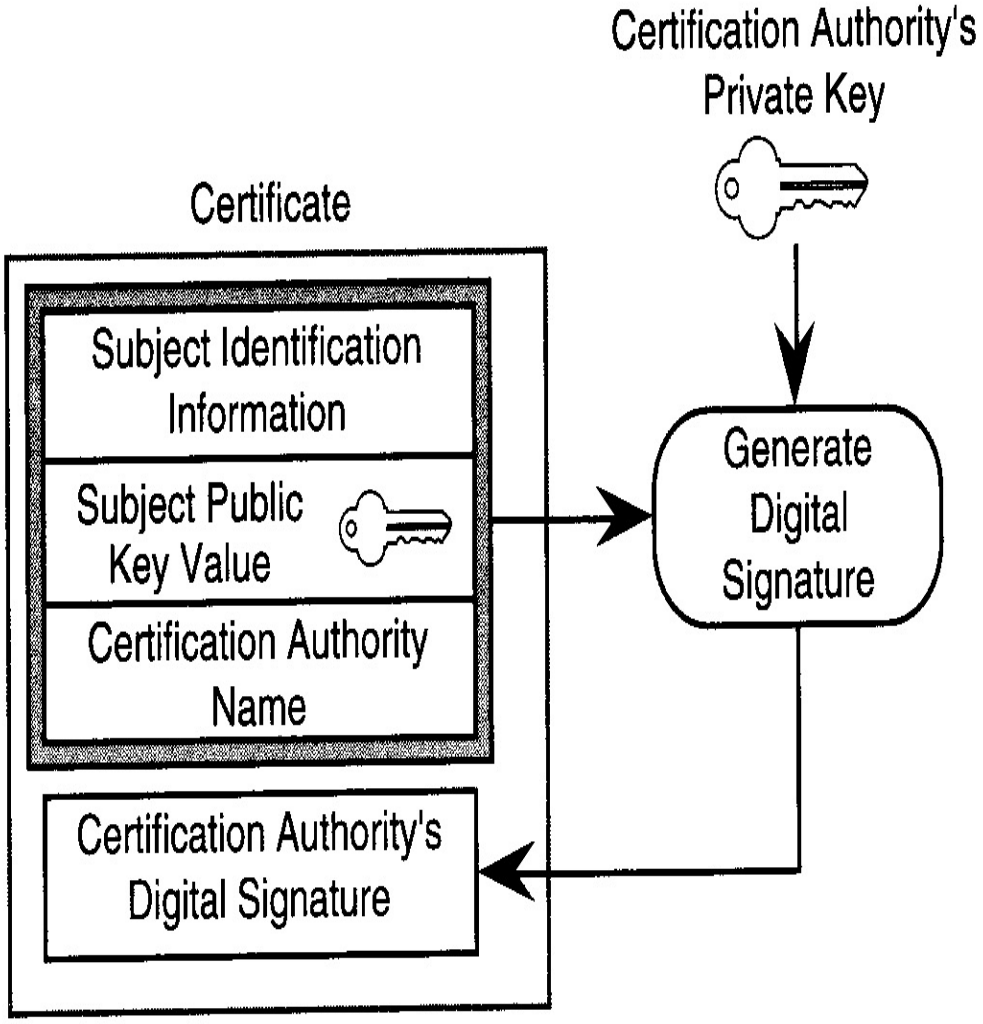
Digital Certificate and Public Key Infrastructure (PKI)

Certification Authorities

- Certification authority (CA): binds public key to particular entity, E.
- E (person, router) registers its public key with CA.
  - E provides "proof of identity" to CA.
  - CA creates certificate binding E to its public key.
  - certificate containing E's public key digitally signed by CA – CA says "this is E's public key"

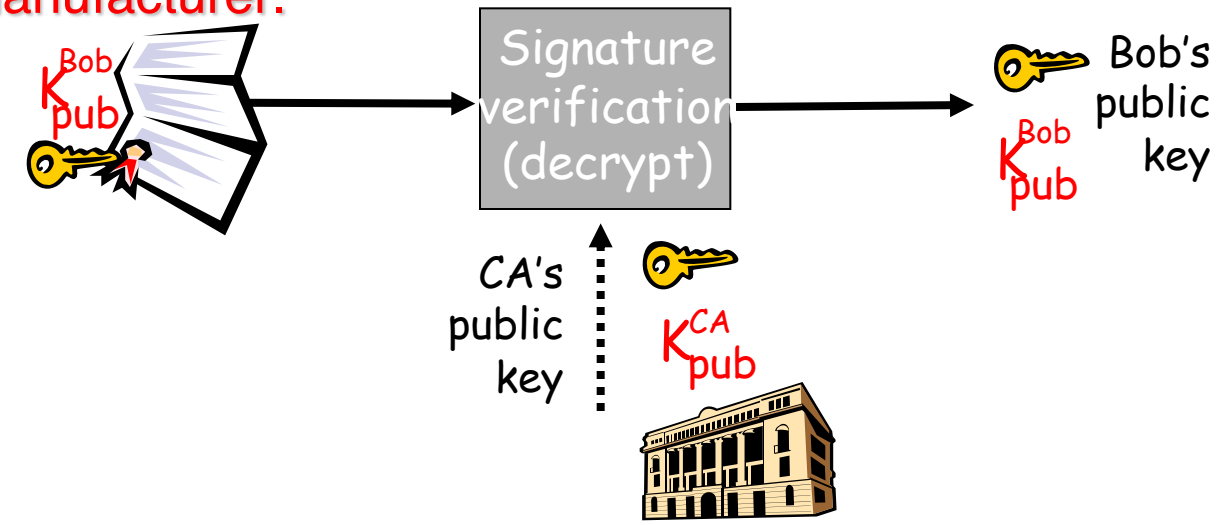


A Conceptual Digital Certificate



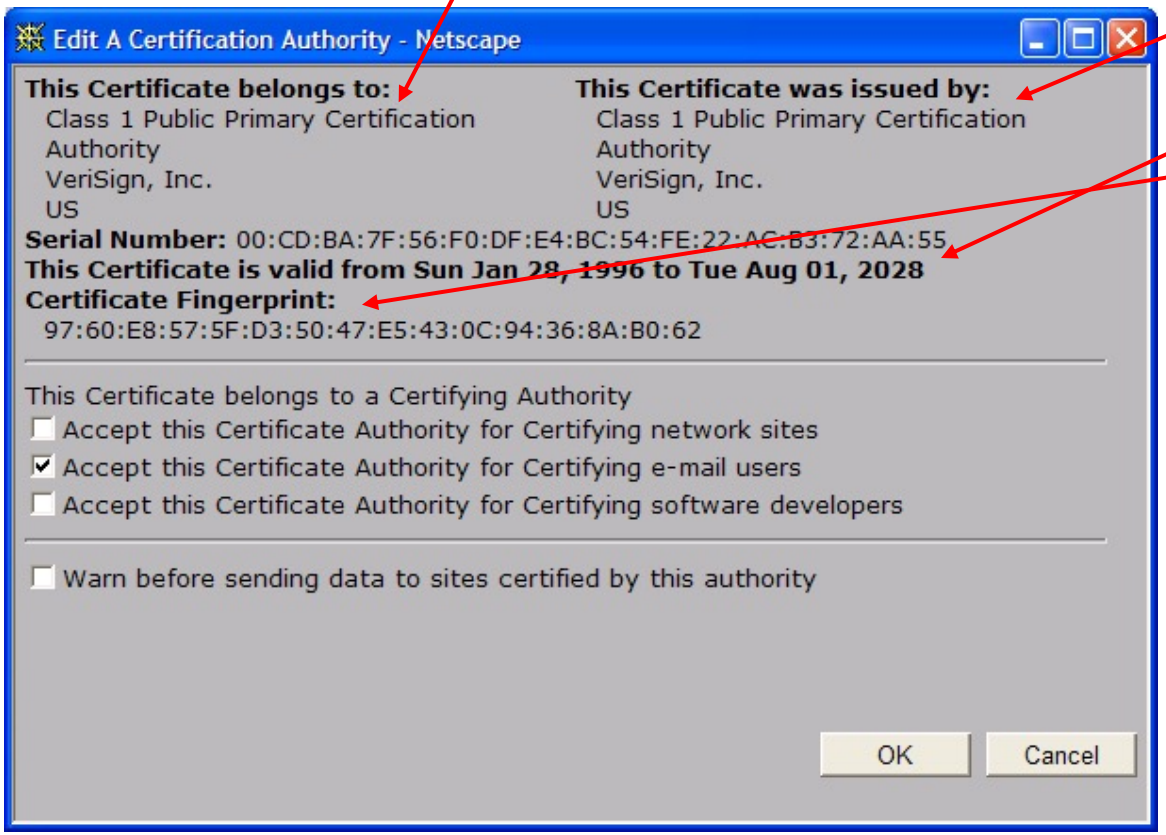
Certification Authorities

- When Alice wants Bob's public key:
  - gets Bob's certificate (from Bob or elsewhere).
  - apply CA's public key to verify Bob's certificate to confirm Bob's public key
  - Alice only needs to know the CA's public key in advance, e.g. preinstalled by computer/operating system manufacturer.



A Certificate contains:

- Serial number (unique to issuer)
- info about certificate owner, including algorithm and key value itself (not shown)

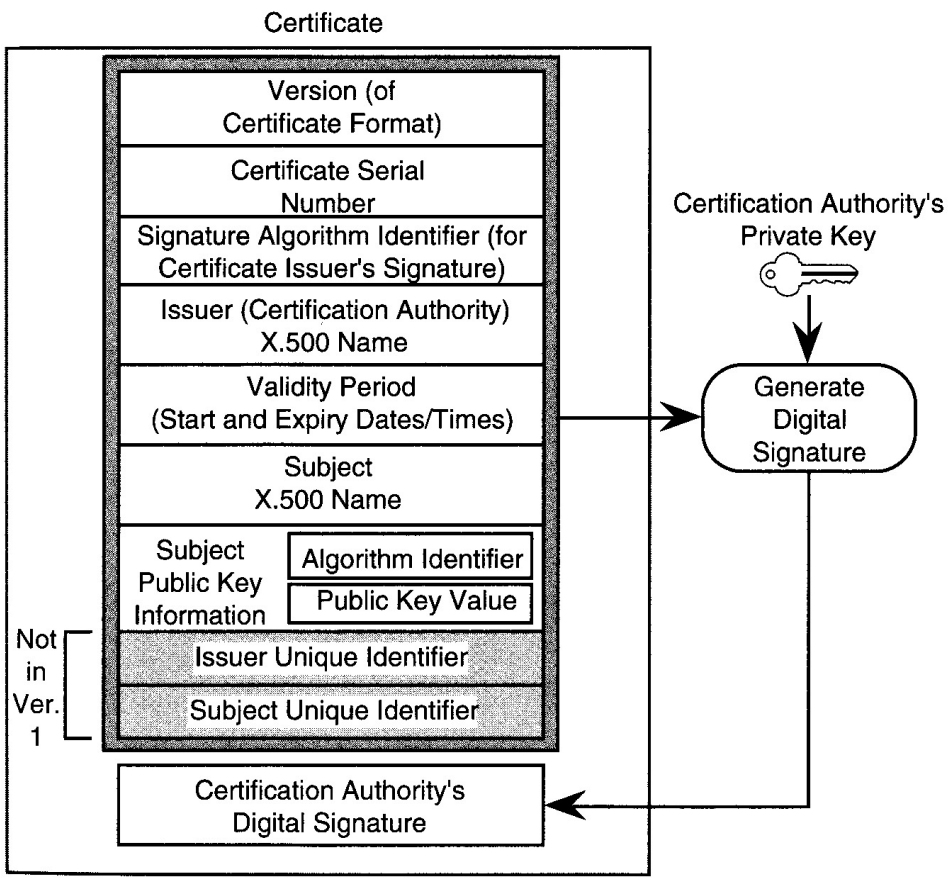


- info about certificate issuer
- valid dates
- digital signature by issuer

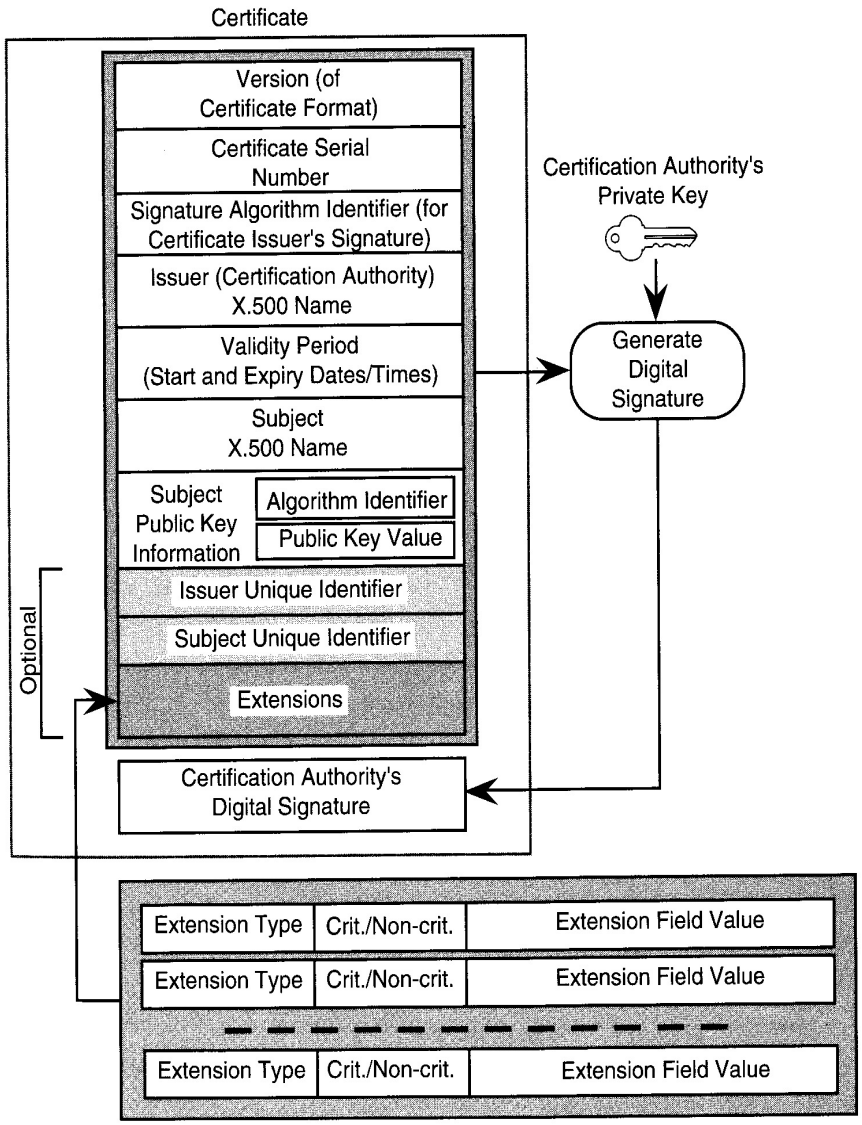
Certificate Distribution via Directory Services

- ITU/ISO developed X.500 directory standards in mid-80's
  - X.500 directory intended to act as a source of information about people, network components etc
  - Designed to support multi-purpose (= heavy weight, complex) distributed directory services on a "potentially global scale", ranging from simple address lookup to attribute-keyed searching
  - It was also recognized then that X.500 directory standard can play a role in the distribution of digital certificates
    - ⇒ The X.509 certificate format were designed under the X.500 umbrella
- However, potential of X.500 never materializes.
  - To date, there is no widespread public deployment of X.500 directory services. (One exception: the Internet community has developed a light-weight version of the X.500 called Lightweight Directory Access Protocol (LDAP), but mostly used within an enterprise so far.)
- In practice, the dominant applications using Digital Certificates have been Internet (IP_ based applications, e.g. email, Web, IPSec.
  - The major mismatch between ITU/ISO telecom-based and the IP-based conventions, e.g. naming conventions, has complicated deployment and interoperability issues, at least initially.
  - Some people argued that it would have been better off using DNS to support digital certificate lookup/distribution instead

Versions 1 to 3 of an X.509 Digital Certificate

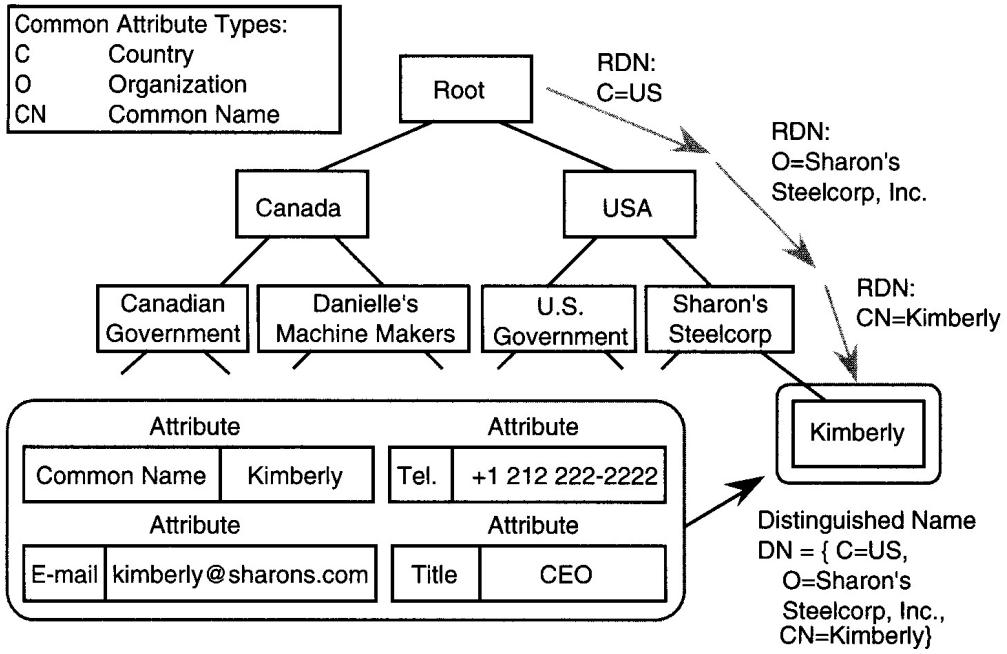


Version 1 and 2



Version 3

X.500 Naming Conventions



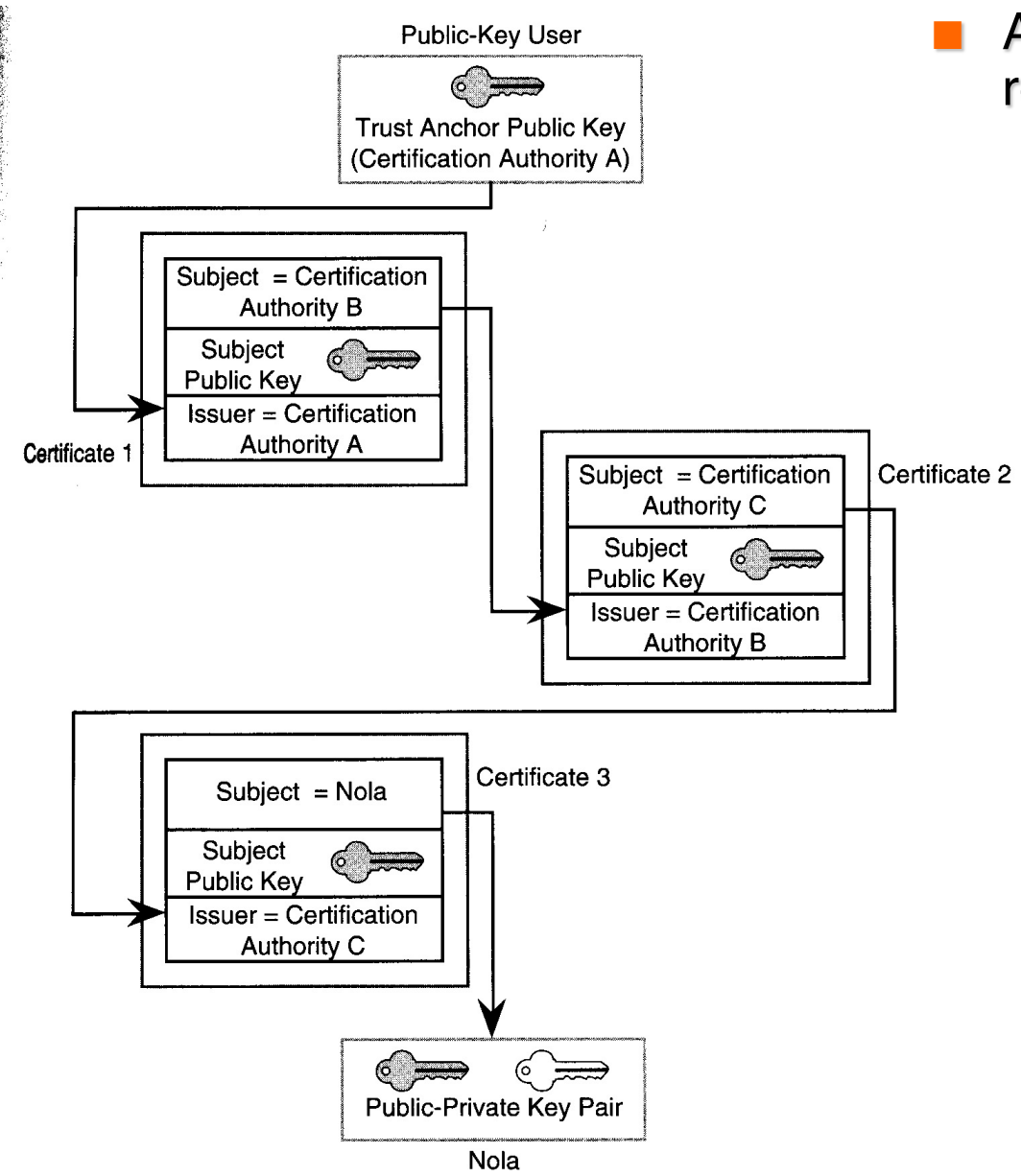
An Example of X.500 Name Construction

- How to represent wclau@ie.cuhk.edu.hk , or http://www.ie.cuhk.edu.hk in X.500 naming conventions ?
- Unlike DNS, the actual X.500 names are NOT encoded in ASCII. Instead, they are encoded in form of ASN.-based Object Identifier (OID)
  - harder to read/ process/ convert ;
  - also taking up more storage
  - Plus other ambiguities

PKIX Certificate Profile

- In 1994, IETF established a working group called Public Key Infrastructure (X.509) Working Group, known as PKIX to:
  - Refine X.509 to satisfy the needs of Internet Protocols and Applications, in particular,Web, email and IPSec
  - Define additional specifications needed to build interoperable implementations of the Internet protocols and applications that use X.509-based certificates
- PKIX produced a “profile” of the X.509 certificate format which means PKIX specifies which X.509 options/attributes should be supported (there were too many options in the original standard)
  - Conventions for specifying subject or issuer names using the name formats corresponding to IP addresses, Internet email addresses, DNS names and URLs.
  - Two Internet specific extensions:
    - The authority information access information provides a pointer in form of URL, to an address for accessing an online certificate status services
    - The subject information access extension conveys an address for contacting the subject

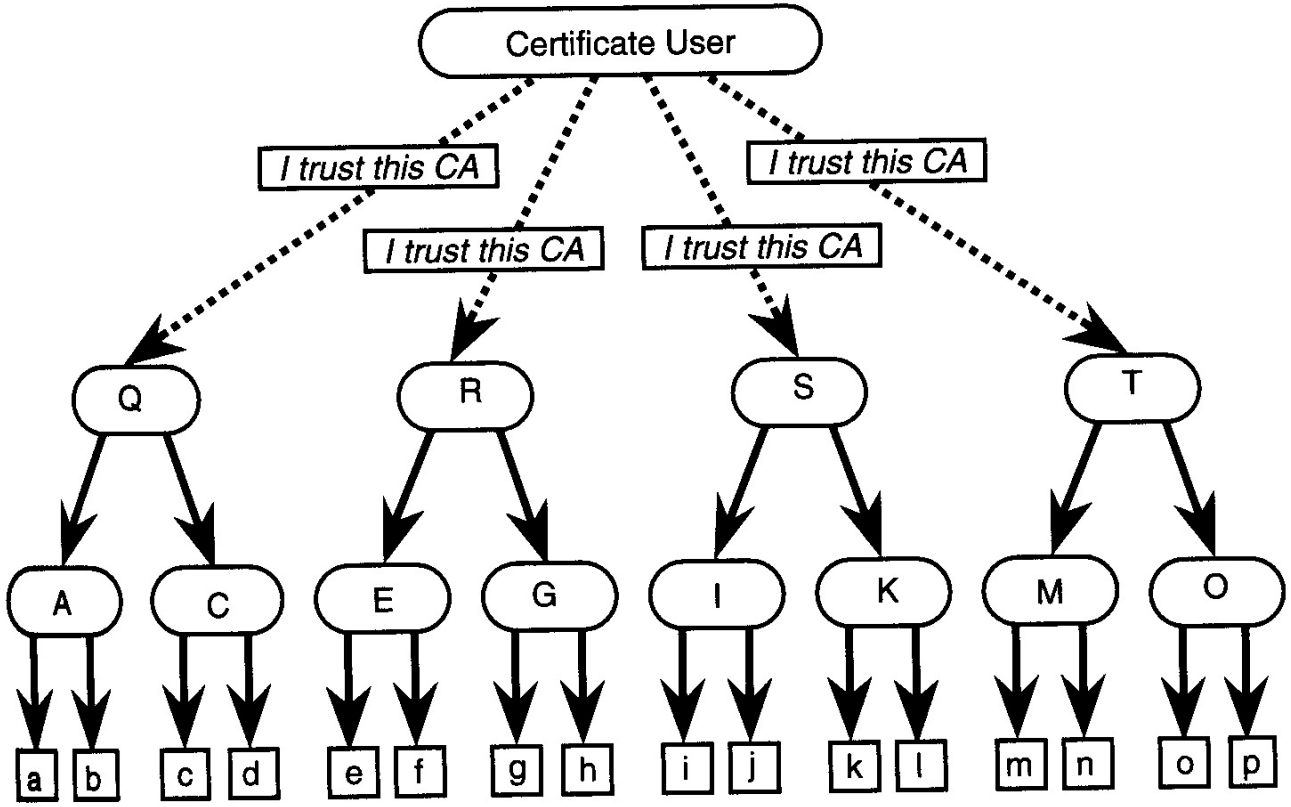
Certification Paths – chain of trust



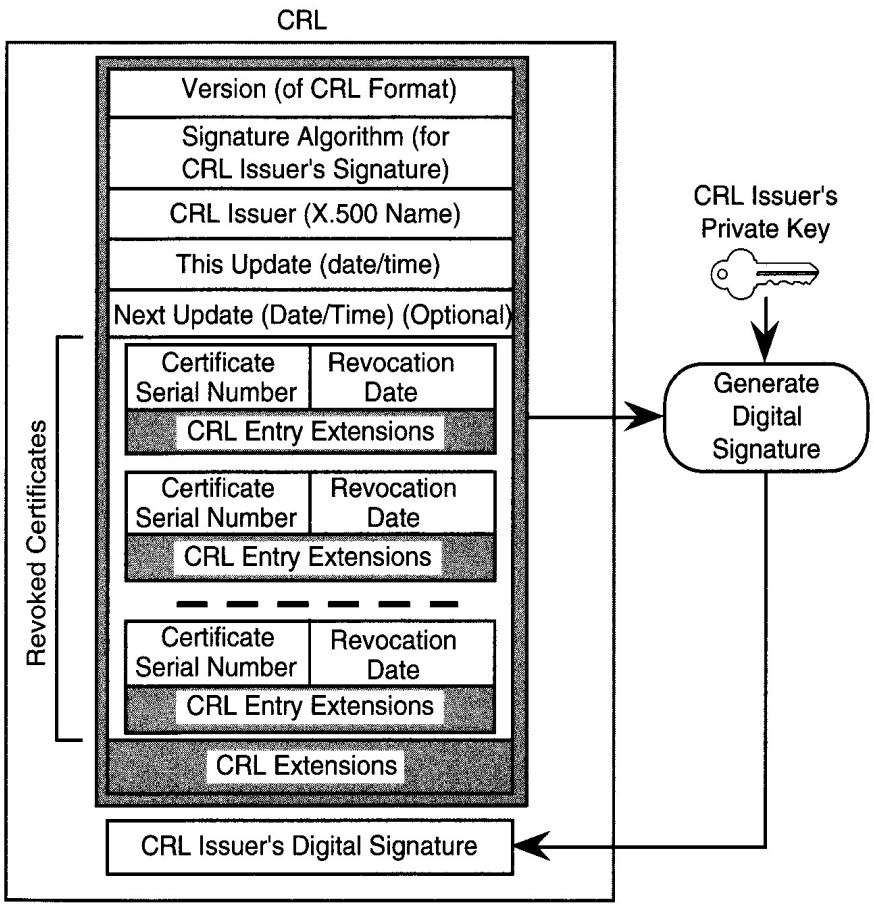
- Apply the certificate paradigm recursively
  - At the beginning, a public-key user acquires, with high assurance, the public-keys of one or more CAs called the "Trust anchors" or "Root Certification Authorities",
  - e.g. Public keys of those trust anchors may be preconfigured in your browser.
  - The public-key user can accept any public key of a key-pair holder provided that a trusted certification path exists from a trust anchor of the public-key user to that key-pair holder possibly via other intermediate certification authorities

PKI Trust Hierarchy

- In practice, it is unrealistic to have an “universial” Root CA world wide, e.g. due to monopolistic, political concerns. So the trust hierarchy typically begins at a non-root level of the tree with multiple “root” trust anchors



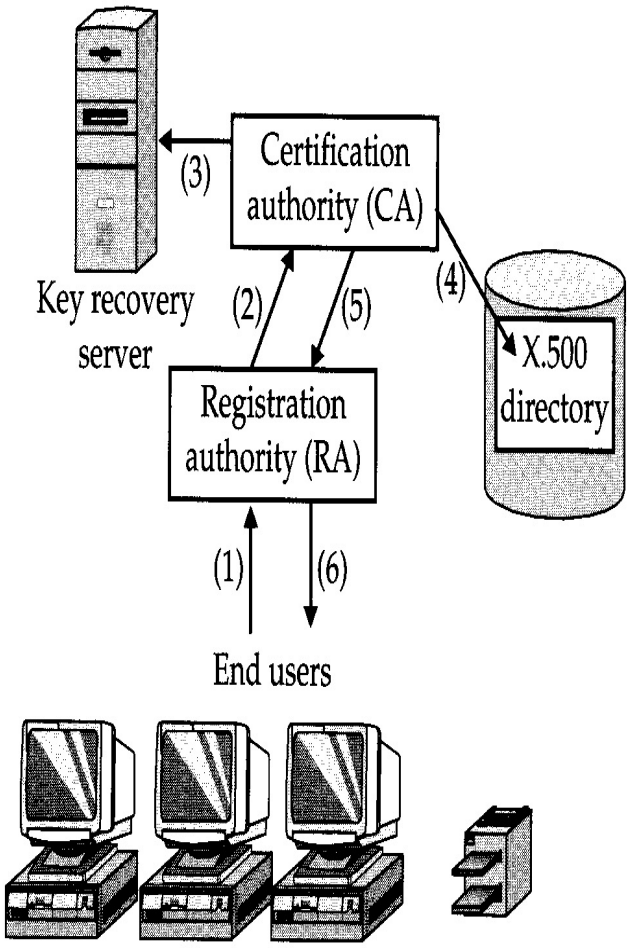
How to revoke an issued certificate ?



An X.509 Certificate Revocation List (CRL)

- One may want to invalidate an existing certificate before it expires due to, e.g.
  - Loss of the associated private key or
  - The person owning the certificate has left the company/organization and no longer authorize to do something entitled by the certificate
- The certificate issuer will maintain the list of such revoked certs in form of a CRL ; It's the responsibility of the end-user to check if a presented cert belongs to the CRL list or not.

Components of a Public Key Infrastructure



- Certification Authority (CA):
  - Issue, manage and revoke certificates for a community of users
- Registration Authority (RA):
  - Assist the CA in its day-to-day certificate processing functions:
    - accept and verify registration info about new registers
    - Generate keys on behalf of users
    - Accept and authorize requests for key backup and recovery
    - Accept and authorize requests for certificate revocation
    - Distribute/recover hardware tokens
- Certificate (X.500) Directory:
  - Provide a central location for storing and distributing user certificates
- Key Recovery Server
  - Back up private keys at time of creation and recover them later

Different Goals/ Services provided by Security

- **Confidentiality (privacy): Against Eavesdropping, Sniffing**
- **Integrity (has not been altered) Against Tampering**
- **Authentication (you are who you say you are) Against Impersonation, Masquerading, Spoofing**
- **Access control (only the intended can "use" the resources) Against unauthorized use/ abuse of resources**
- **Non-repudiation (the order is final) Against Denying One's Act, backing away from a deal**
- **Availability Against DoS Attacks**

Authentication

On the Internet, nobody knows you're a dog
- by Peter Steiner, New York, July 5, 1993

By 2006:
"On the Internet, EVERYBODY knows you're a dog, drinking Starbuck."

Authentication of People: To prove you are who you say you are

- By means of:
  - What you know ?

  e.g. password, your own HKID#
  - What you have ?

  e.g. A door-key, Secure token
  - Where you are ?

  e.g. caller-id, network (IP) address
  - Who you are ?

  e.g. fingerprint, other biometric: iris, retina patterns, voice
- Can combine more than one of the above, e.g. Automatic Teller Machine (ATM) use a 2-factor authentication scheme:
  - your ATM card + PIN

Password

- Proof by knowledge, sharing
- Password guessing
  - On-line attack: limit tries, alarm
  - Off-line dictionary attack ; need at least 2^{64} or more combinations (i.e. 64-bit key or longer) to be secure ;
  - Since each character on the keyboard can produce 6-bits of randomness if several punctuation marks, upper, lower cases, and digits are included,

  => 11 character-long password is needed

  - if we let users pick their own password freely, need 32-character long password because empirical data shows that, on average, only 2-bit per character of randomness is introduced by a typical user.

  => too long for human to remember

  - => password always vulnerable to offline dictionary attack

  - UNIX only considers the first 7 characters in a password
- Pick the 1st or last character of words in a song lyric etc to get a long but memorable password.

Humans and Computers

- Humans:
  - Short, memorable key (8 characters, 48 bits), directly or as key for longer key
  - Randomness of human-generated passwords ~= 2 bits per character
- Computers:
  - (Long) high-quality secret
  - Hidden key (encrypted by password), directly (e.g., hash of the password)

Address-based

- .rhosts
  - node, user name
- /etc/hosts.equiv
  - trusted hosts
- Threats:
  - break in one, break in all
  - address spoofing: MAC address, IP address, Caller-ID, SMS-sender-ID/Phone#

Trojan Horses

- A faked login prompt to capture passwords
- Counter measures:
  - Make it hard to have the appearance of login prompt
  - Use interrupts (CTRL-ALT-DEL to get login window in W2K)
  - Prevent login by user programs

Authentication Tokens

- What you have
- Smart cards:
  - Challenge/response
- Cryptographic calculator:
  - Interaction through a user (typing ...)

Biometrics

- Accuracy:
  - False acceptance rate.
  - False rejection rate.
- Socially acceptable user-interface
- Retinal scanner, fingerprint reader, handprint reader, voiceprint, keystroke timing, signature.

  - Can adversary select imposters?
    - Identical twins, family members, voice-recorder, copy of fingerprint on scotch-tape, warm finger…. etc.

Fingerprints

- Vulnerability:
  - Dummy fingers and dead fingers, fingerprints left on scotch-tape
- Suitability and stability:
  - Not for people with high probability of damaged fingerprints
  - Not for kids growing up
  - False +ve rate about 1 in 100,000

Voice Recognition

- Single phrase:
  - Can use tape recorder to fake
- Stability:
  - Background noise
  - Colds
  - Use with public phones
  - Less than US$100 per system

Keystroke Timing

- Each person has a distinct typing timing/style
  - Hand/finger movements
- Suitability:
  - Best done for “local” authentication
    - Avoid network traffic delay

Signatures

- Machines can't match human experts in recognizing shapes of signatures
- Add information of timing (dynamics) of movements
  - Signing or an electronic tablet

Security Handshake and Pitfalls

Threats of Concern

- Offline password cracking attacks
- Replay
- Security of Password Database at server Vs. sending password in clear across network
- Subsequent compromised of password to endanger previously encrypted (and recorded by the attacker) traffic
- Man in the Middle attack

Authentication Protocol

Goal: Bob wants Alice to "prove" her identity to him

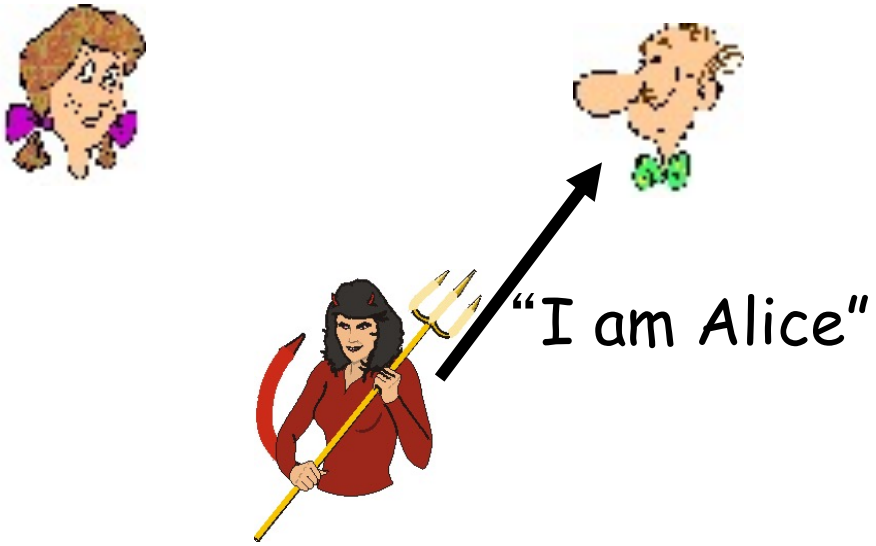
Protocol ap1.0: Alice says "I am Alice"

Failure scenario??

Authentication

Goal: Bob wants Alice to "prove" her identity to him

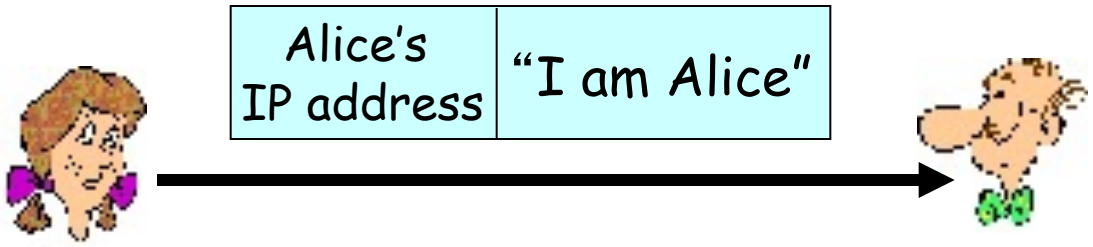
Protocol ap1.0: Alice says "I am Alice"



in a network,
Bob can not "see"
Alice, so Trudy simply
declares
herself to be Alice

Authentication: another try

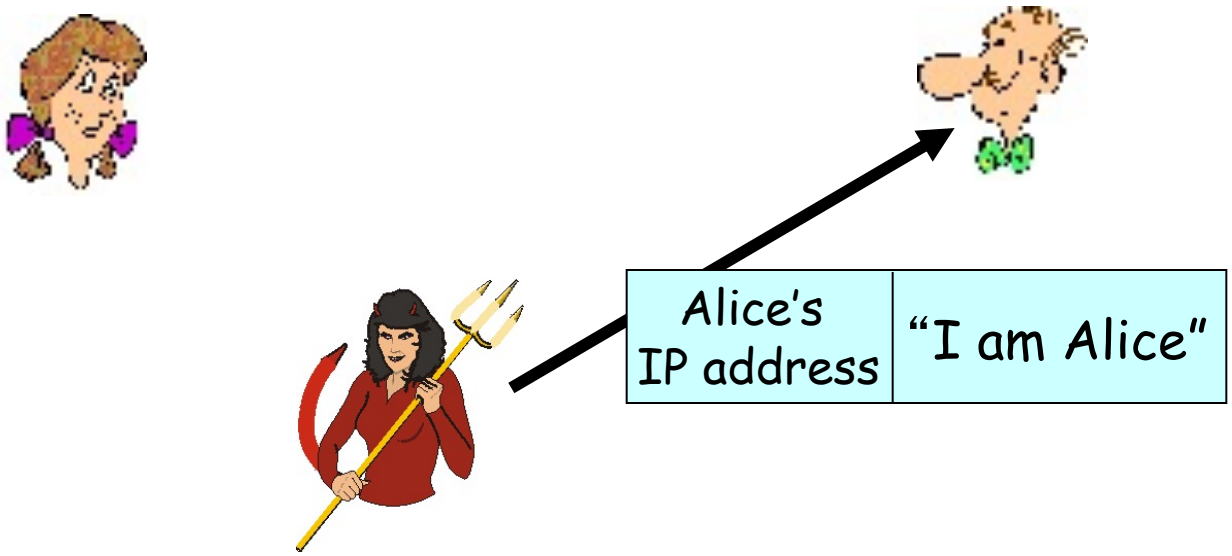
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



Failure scenario??

Authentication: another try

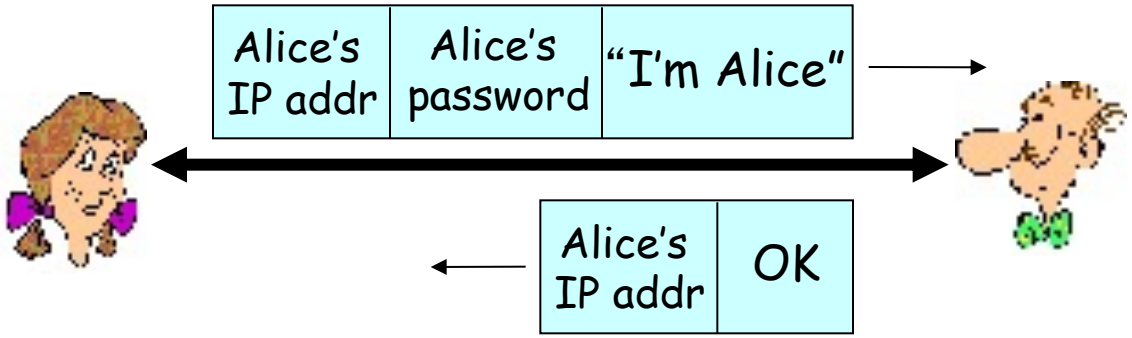
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



Trudy can create a packet "spoofing" Alice's address

Authentication: another try

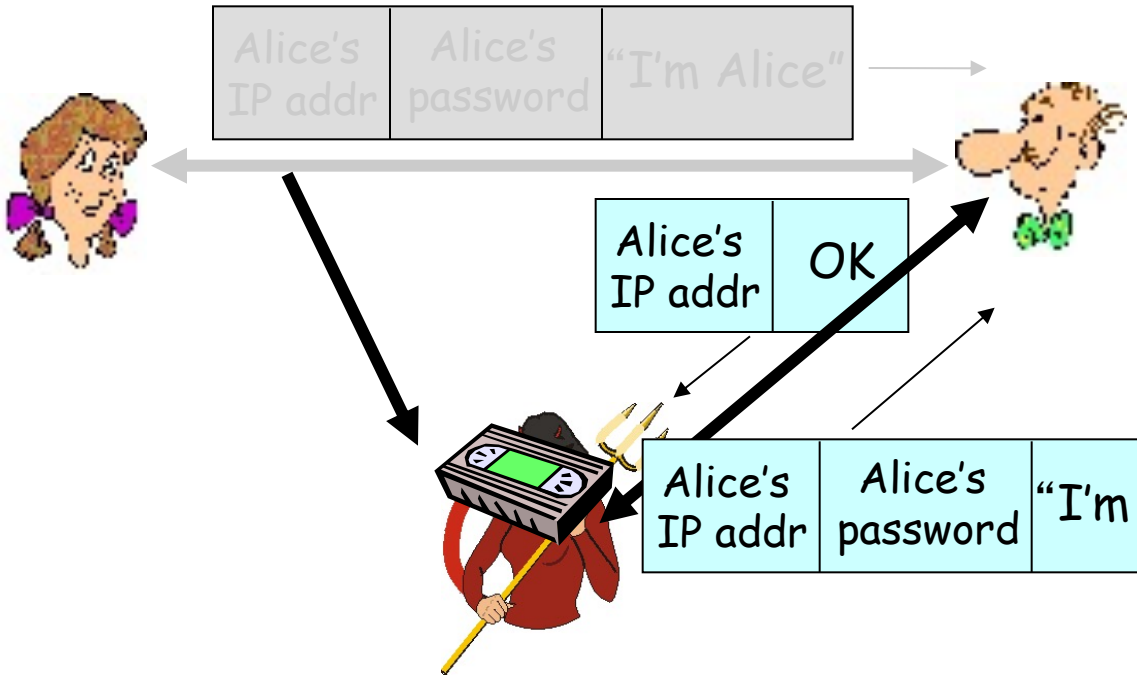
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario??

Authentication: another try

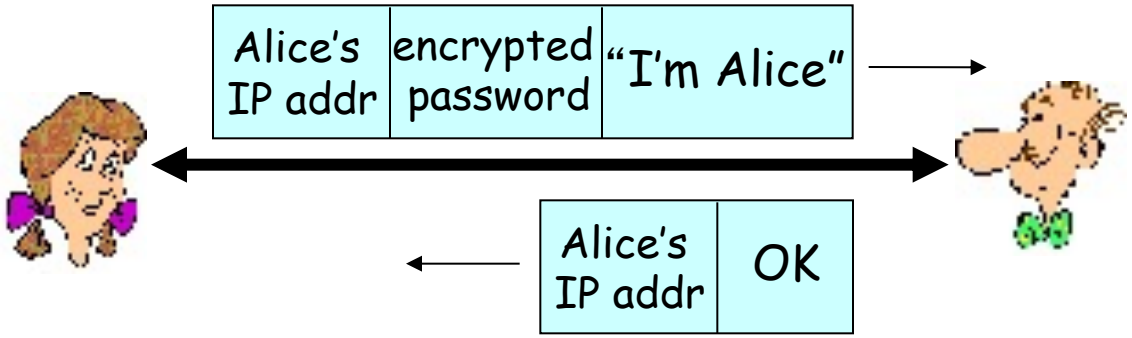
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



playback attack: Trudy records Alice's packet and later plays it back to Bob

Authentication: yet another try

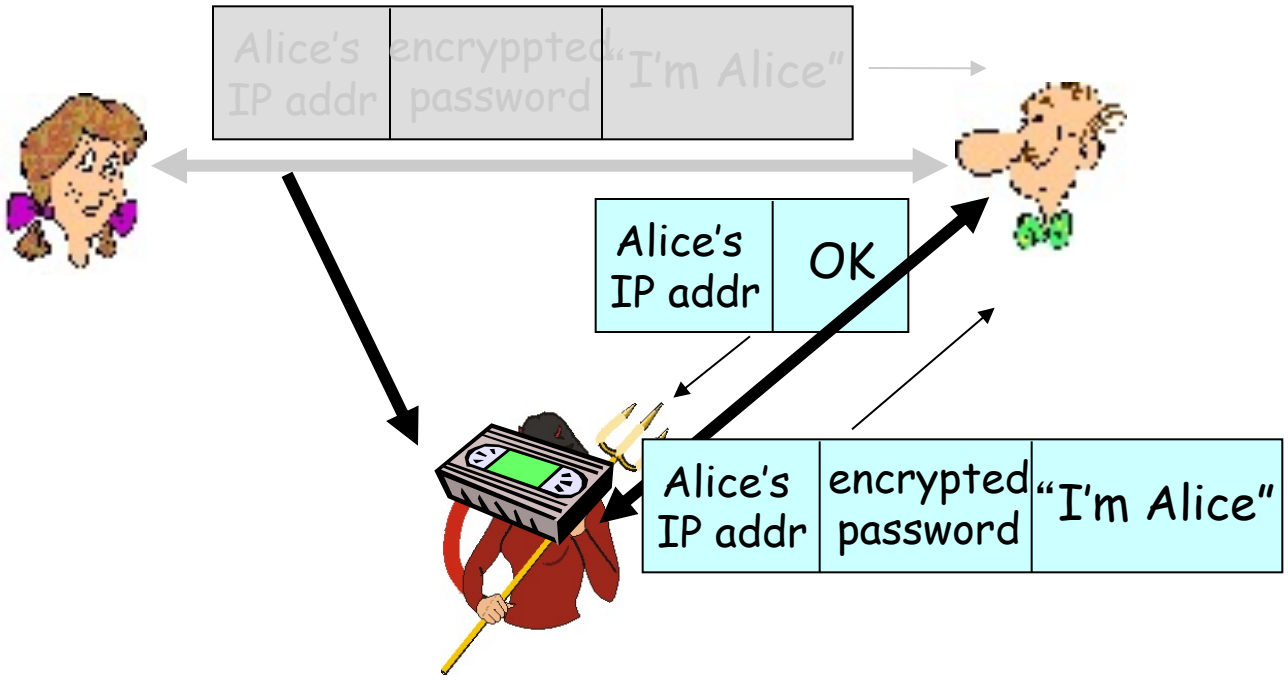
Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.



Failure scenario??

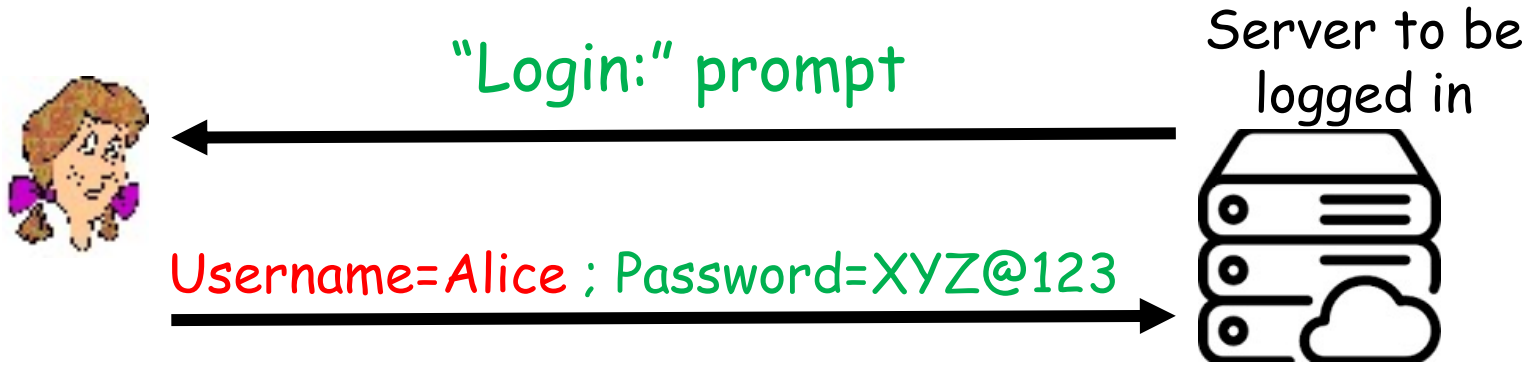
Authentication: another try

Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.



record and playback still works!

(OLD) Authentication Handshake in Unix/Linux



"Password Data" in the /etc/passwd file on the server

Username	Salt	Hash[Salt, Password of user]
Alice	x78a	23rj0fdsaklr2c
Bob	93bz	s930kfs0923js
Carol	142y	9823xwxteotpl
Ethan	wx99	zlfewferlkt3293

Drawbacks ?

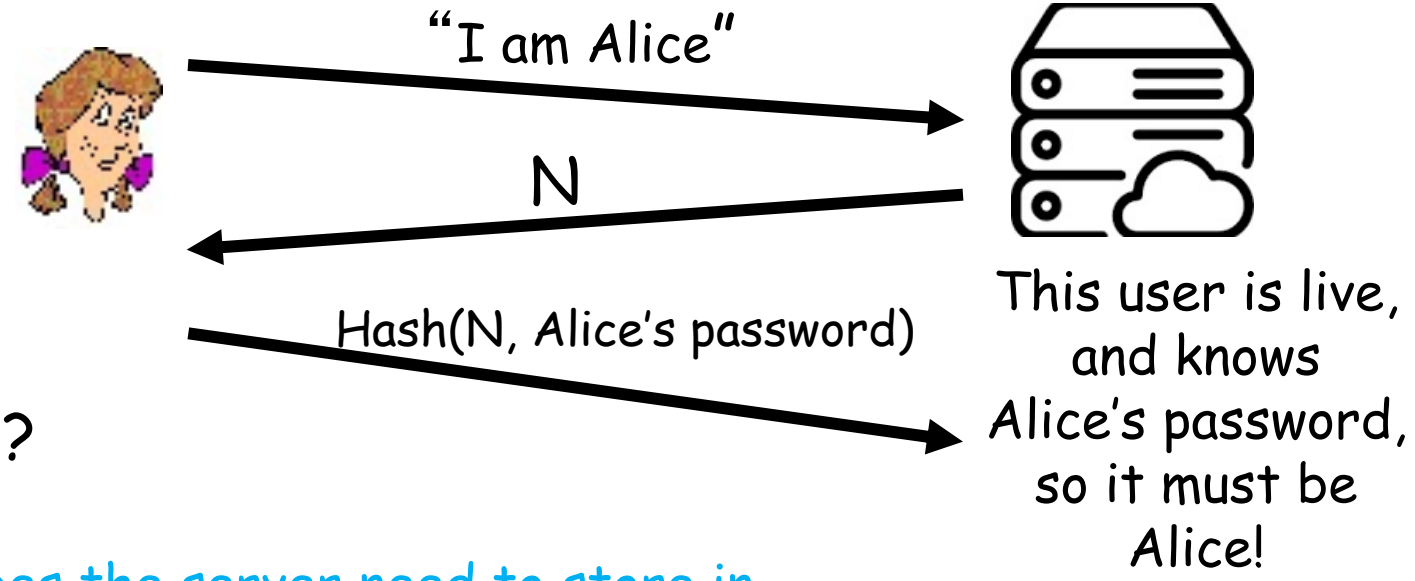
Advantages ?

Authentication: yet another try

Goal: avoid playback attack

Nonce: number (N) used only *once –in-a-lifetime*

ap4.0: To prove Alice's "live", Server sends Alice nonce, N.
Alice must return the Hash(N, Alice's password)
or in another variant, Alice encrypts N with her password



Drawbacks?

What info does the server need to store in its "Password" file in this case ?

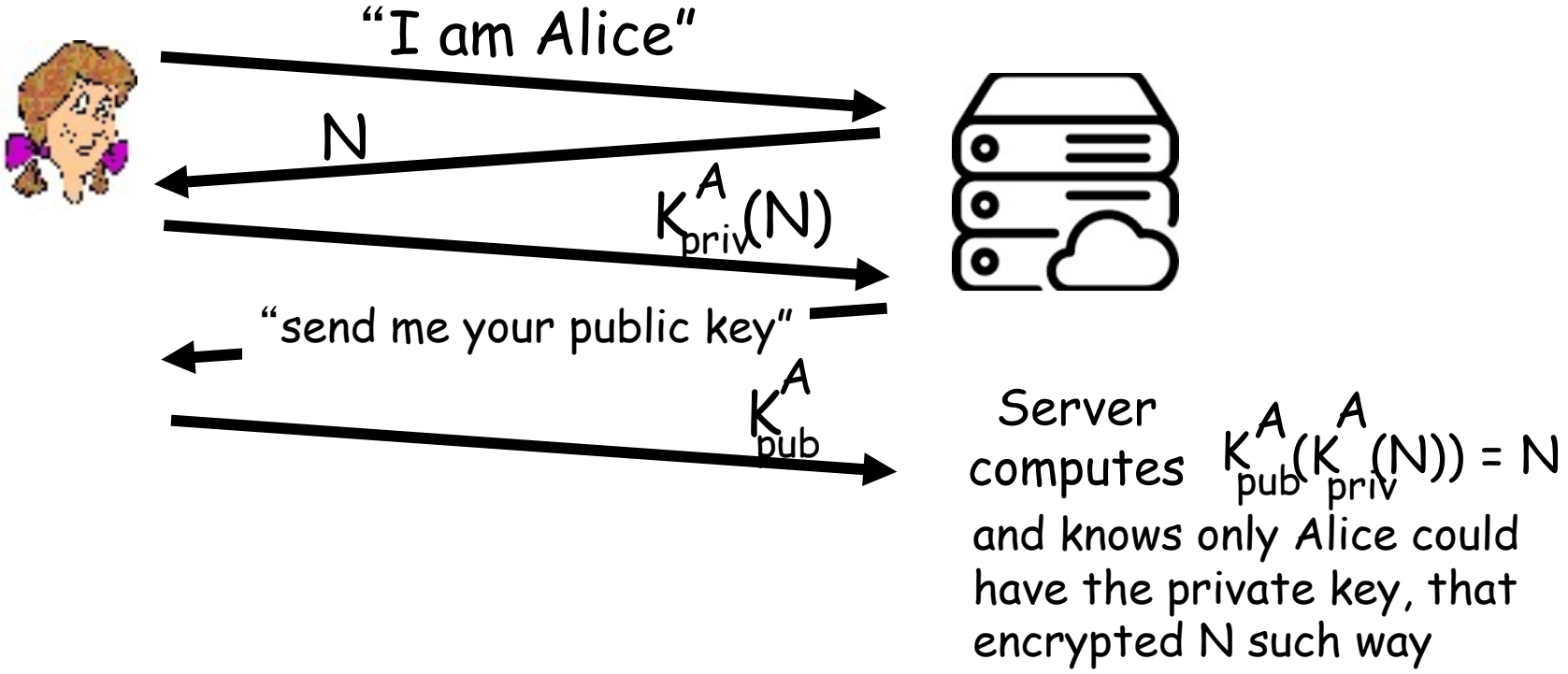
Authentication: ap5.0

ap4.0 requires pre-shared secret,

i.e. the password of each user pre-configured and stored at server

- Can we authenticate using public key techniques?

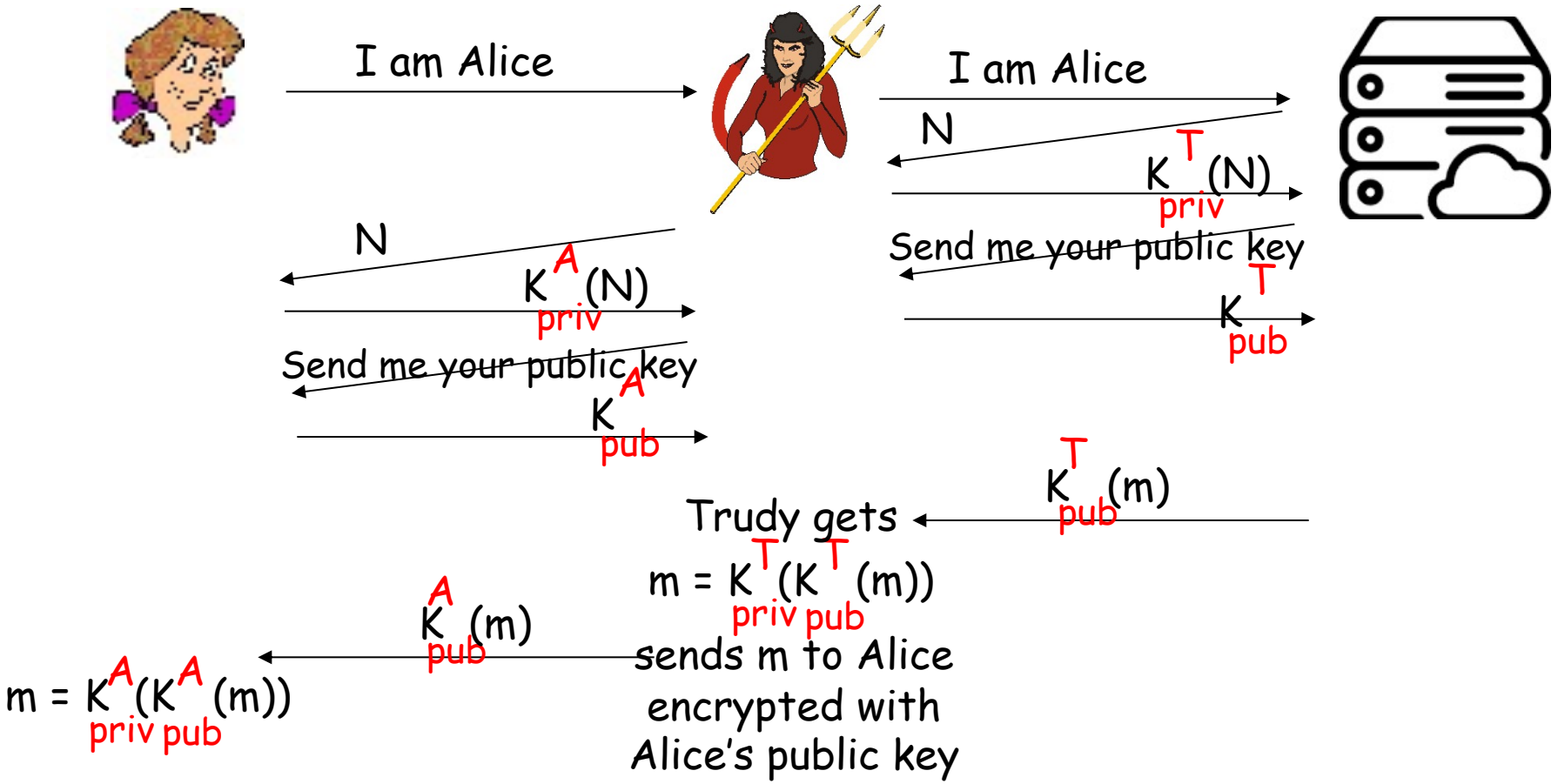
ap5.0: use nonce, public key cryptography



Also, what if "N" is some message (digest) that Alice does not want to sign
=> Each person uses multiple pairs of public/private keys ; one pair for encryption/decryption ; the other pair for authentication/signing

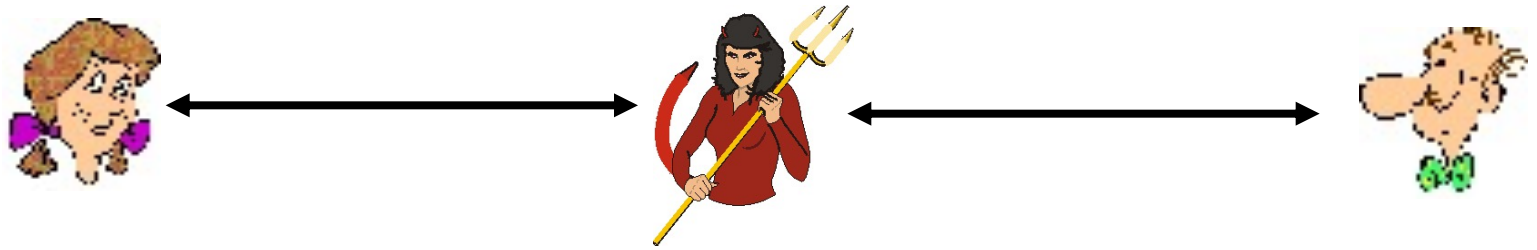
ap5.0: security hole

Man in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



ap5.0: security hole

Man in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



Difficult to detect:

- Bob receives everything that Alice sends, and vice versa. (e.g., so Bob, Alice can meet one week later and recall conversation)
- problem is that Trudy receives all messages as well!