Authentication & Kerberos

Authentication Basics

- Clear Text Passwords
- Message Digests of Passwords
- Adding Randomness
- Password Encryption
- Authentication Token
- Certificate based authentication

Authentication Protocols

- used to convince parties of each other's identity and to exchange session keys
- may be one-way or mutual
- key issues are
  - confidentiality – to protect session keys
  - timeliness – to prevent replay attacks

Replay Attacks

- where a valid signed message is copied and later resent
  - simple replay: The opponent simply copies a message and replays it later
  - repetition that can be logged: An opponent can replay a timestamped message within the valid time window.
  - repetition that cannot be detected: This situation could arise because the original message could have been suppressed and thus did not arrive at its destination; only the replay message arrives.

  - backward replay without modification: This is a replay back to the message sender. This attack is possible if symmetric encryption is used and the sender cannot easily recognize the difference between messages sent and messages received on the basis of content

- countermeasures include
  - use of sequence numbers: it requires each party to keep track of the last sequence number for each claimant it has dealt with. It increases overhead, so impractical.

- challenge/response (using unique nonce): Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contains the correct nonce value

- timestamps (needs synchronized clocks): party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time. This approach requires that clocks among the various participants be synchronized.

Using Symmetric Encryption

- can use a two-level hierarchy of keys
- usually with a trusted Key Distribution Center (KDC)
  - each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - master keys used to distribute these to them

Needham-Schroeder Protocol

- original third-party key distribution protocol
- for session between A, B mediated by KDC
- protocol overview is:

  1. A→KDC: $ID_A \,||\, ID_B \,||\, N_1$

  2. KDC→A: $E_{Ka}[Ks \,||\, ID_B \,||\, N_1 \,||\, E_{Kb}[Ks||ID_A]\,]$

  3. A→B: $E_{Kb}[Ks||ID_A]$

  4. B→A: $E_{Ks}[N_2]$

  5. A→B: $E_{Ks}[f(N_2)]$

- Ka, Kb : Secret keys
- Ks : session key

Needham-Schroeder Protocol

- used to securely distribute a new session key for communications between A & B
- but is vulnerable to a replay attack if an old session key has been compromised
  - then message 3 can be resent convincing B that is communicating with A
- modifications to address this require:
  - timestamps
  - using an extra nonce

Using Public-Key Encryption

- have a range of approaches based on the use of public-key encryption
- need to ensure have correct public keys for other parties
- using a central Authentication Server (AS)
- various protocols exist using timestamps or nonce

Denning AS Protocol

- Denning 81 presented the following:
  1. A→AS: ID_A || ID_B || T
  2. AS→A: $E_{KRas}[ID_A||KU_a||T]$ || $E_{KRas}[ID_B||KU_b||T]$
  3. A→B: $E_{KRas}[ID_A||KU_a||T]$ || $E_{KRas}[ID_B||KU_b||T]$ || $E_{KUb}[K_s||T]$
- note session key is chosen by A, hence AS need not be trusted to protect it
- timestamps prevent replay but require synchronized clocks

Kerberos

- trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
  - allows users to access the services distributed through network
  - without needing to trust all workstations
  - rather all trust a central authentication server
  - Function of centralized authentication server is to authenticate users to servers and servers to users
  - It relies exclusively on symmetric encryption, no use of public key encryption.
- two versions in use: 4 & 5

Kerberos Requirements

- first published report identified its requirements as:
  - Security : A network eavesdropper should not be able to obtain the necessary information to impersonate a user.
  - Reliability : Kerberos should be highly reliable and should employ a distributed server architecture, with one system able to back up another
  - Transparency : Ideally, the user should not be aware that authentication is taking place, beyond the requirement to enter a password.
  - Scalability : Should support large number of clients and servers, this suggest a modular distributed architecture.
- implemented using an authentication protocol based on Needham-Schroeder

How does kerberos works?

- Four parties involved in kerberos protocol:
  - **Alice:** Client workstation
  - **Authentication Server (AS):** Verifies users during login
  - **Ticket Granting Server (TGS) :** Issues **tickets** to certify the proof of identity
  - **Bob:** Server offering various services

Kerberos 4 Overview

- a basic third-party authentication scheme
- have an Authentication Server (AS)
  - It knows the passwords of all users and stores these in a centralized database
  - It shares a unique secret key with each server
  - Keys distributed physically or in some other secure manner
  - users initially negotiate with AS to identify self
  - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)

- Alice -> AS; IDa ||Pa||IDb
- AS -> Alice Ticket
- Alice -> Bob IDa ||Ticket

–Ticket = Ekb ([IDa] || ADa || IDb])

- Alice - Client
- AS - Authentication server
- Bob - Server
- IDa - Identifier of user, Alice
- IDb - Identifier of server, Bob
- Pa - password of user, Alice
- ADa - network address of Alice
- Kb - secret encryption key shared by AS and Bob

- Problems
  - User would need a new ticket for every different service
  - A plaintext transmission of the password, an eavesdropper could capture the password and use any service accessible to the victim.
  - Ticket Granting server (TGS) can solve this problem

- users subsequently request access to other services from TGS on basis of users TGT
- Once per user logon session:

  Alice ->AS IDa || ID_{tgs}

  AS -> Alice Eka [$Ticket_{tgs}$]

  Once per type of service:

  Alice -> TGS IDa || IDb ||$Ticket_{tgs}$

  TGS -> Alice $Ticket_B$

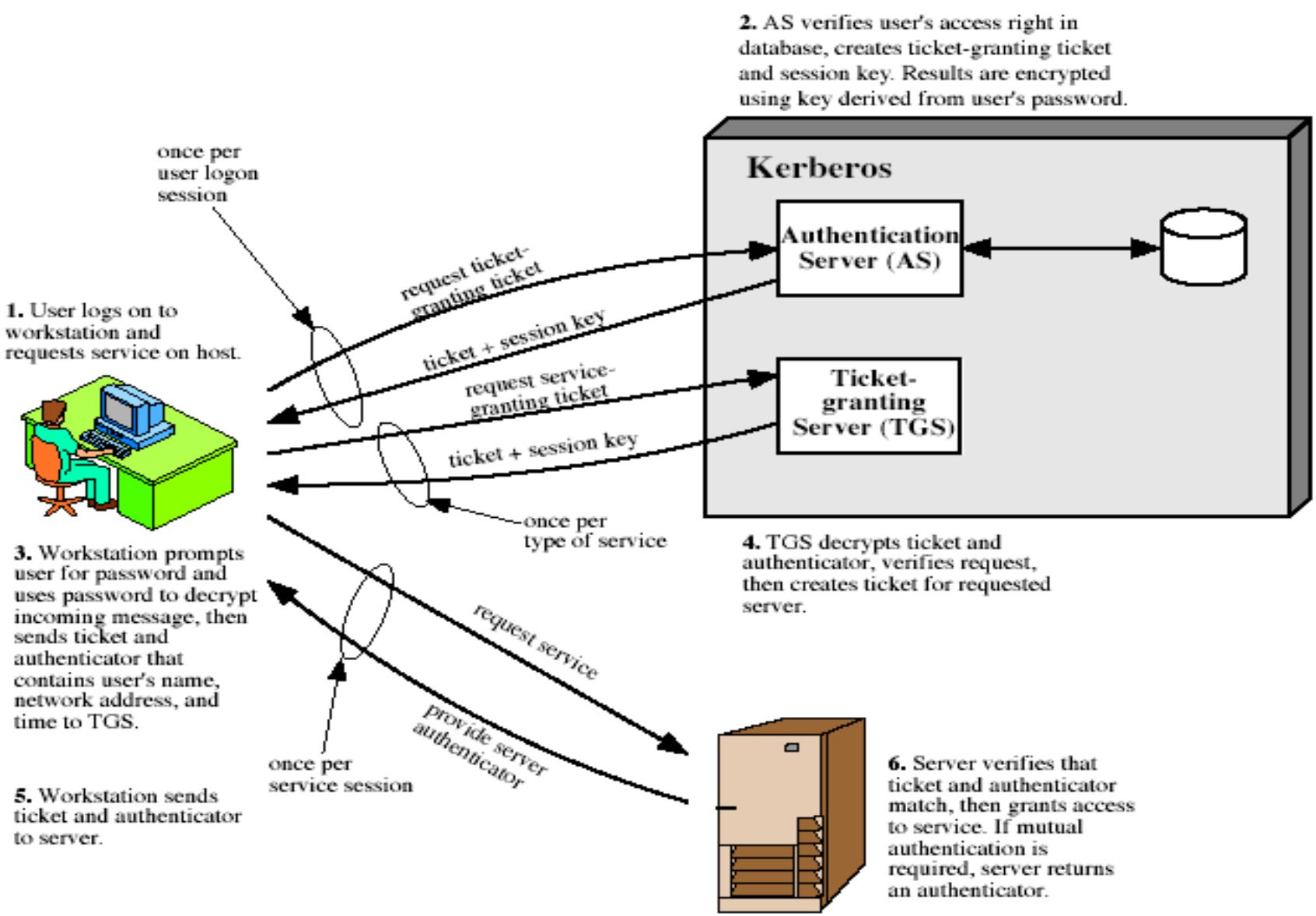
  Once per service session:

  Alice -> Bob IDa || $Ticket_B$

  $Ticket_{tgs} = E_{ktgs}[IDa || ADa || ID_{tgs} || TS_1 || Lifetime_1]$

  $Ticket_B = E_{kb}[IDa || ADa || IDb || TS_2 || Lifetime_2]$

Kerberos 4 Overview



Kerberos Realms

- a Kerberos environment consists of:
  - a Kerberos server
  - a number of clients, all registered with server
  - application servers, sharing keys with server
- this is termed a **Kerberos realm**
  - typically a single administrative domain
- if have multiple realms, their Kerberos servers must share keys and trust

Kerberos Version 5

- developed in mid 1990's
- provides improvements over v4

It addresses

  - environmental shortcomings
    - encryption algo, network protocol, byte order, ticket lifetime, authentication forwarding, interrealm auth
  - and technical deficiencies
    - double encryption, non-std mode of use, session keys, password attacks
- specified as Internet standard RFC 1510

Comparison V4 *Vs* V5

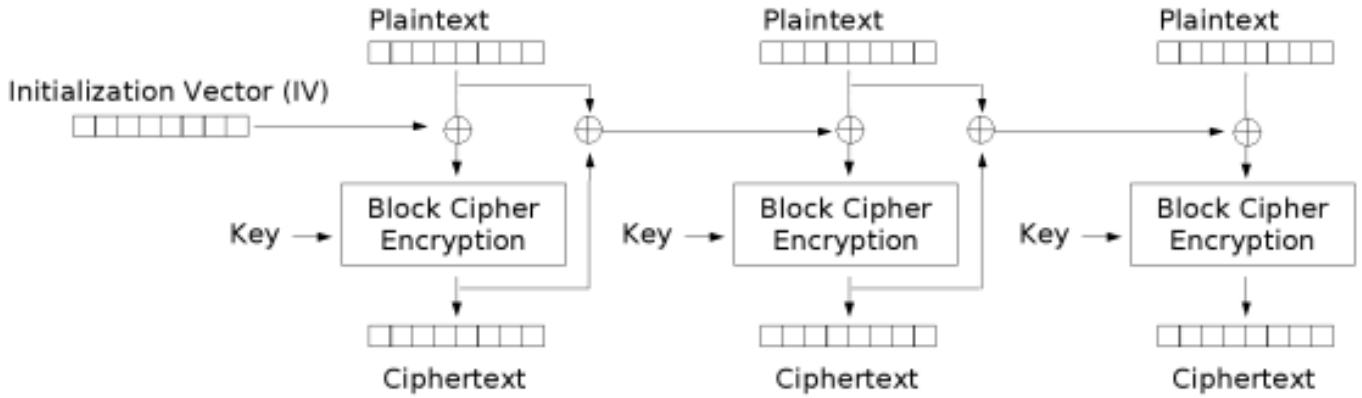
Points	Version 4	Version 5
Encryption system dependence	requires the use of DES	Cypertext is tagged with an encryption type identifier so any encryption techniques can be used.
Internet protocol dependence	Requires the use of IP addresses. Other addresses such as ISO network address are not accommodated.	Network addresses are tagged with type and length, so network address can be used.
Message byte ordering	Sender employs a byte ordering of its own choosing and tags the message to indicate LSB in lowest address or MSB in lowest address.	Message structures are defined using Abstract Syntax Notation One and basic Encoding Rules (BER), which provide an unambiguous byte ordering

Comparison V4 Vs V5 cont…

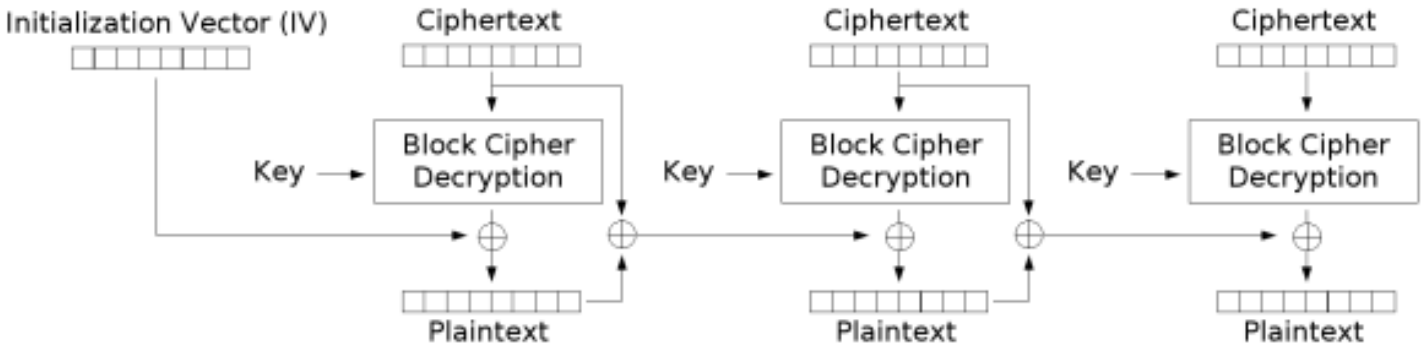
	Version 4	Version 5
Ticket Lifetime	Encoded in an 8-bit quantity in units of five minutes thus max. lifetime is 1280 minutes	Tickets include an explicit start time and end time
Authentication forwarding	Not allow	Allow
Interrealm authentication	Interoperability among N realms requires N^2 Kerberos to Kerberos relationships	Requires fewer relationships

V4 -Technical deficiencies

- Double encryption
  - Once with the secret key of the target server
  - Again with a secret key known to the client
- PCBC encryption
  - making use of non standard mode of DES known as propagating block chaining (PCBC)
  - Vulnerable to an attack involving the interchange of ciphertext blocks



Propagating Cipher Block Chaining (PCBC) mode encryption



Propagating Cipher Block Chaining (PCBC) mode decryption

- Session keys
  - Same ticket may be used repeatedly to gain service from the server, there is the risk that an opponent will replay messages from an old session to the client or the server
  - In V5, it is possible to negotiate a subsession key

  Password attack

  - Both versions are vulnerable to a password attack.

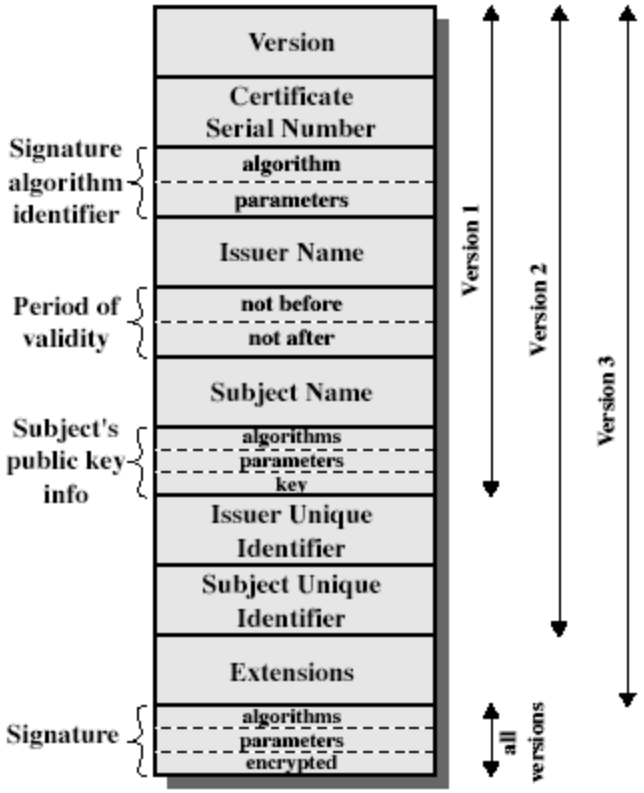
X.509 Authentication Service

- CCITT : International Telegraph and Telephone Consultative Committee
- part of CCITT X.500 directory service standards
  - distributed servers maintaining some info database
- defines framework for authentication services
  - directory may store public-key certificates
  - with public key of user
  - signed by certification authority
- also defines authentication protocols
- uses public-key crypto & digital signatures
  - algorithms not standardised, but RSA recommended

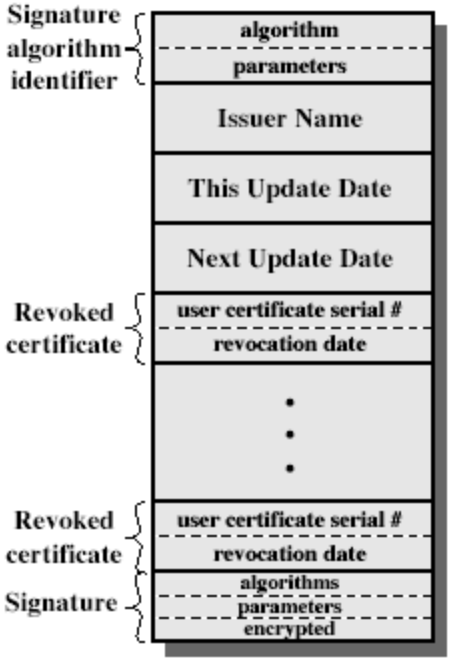
X.509 Certificates

- issued by a Certification Authority (CA), containing:
  - version (1, 2, or 3)
  - serial number (unique within CA) identifying certificate
  - signature algorithm identifier
  - issuer X.500 name (CA)
  - period of validity (from - to dates)
  - subject X.500 name (name of owner)
  - subject public-key info (algorithm, parameters, key)
  - issuer unique identifier (v2+)
  - subject unique identifier (v2+)
  - extension fields (v3)
  - signature (of hash of all fields in certificate)
- notation `CA<<A>>` denotes certificate for A signed by CA

X.509 Certificates



(a) X.509 Certificate



(b) Certificate Revocation List

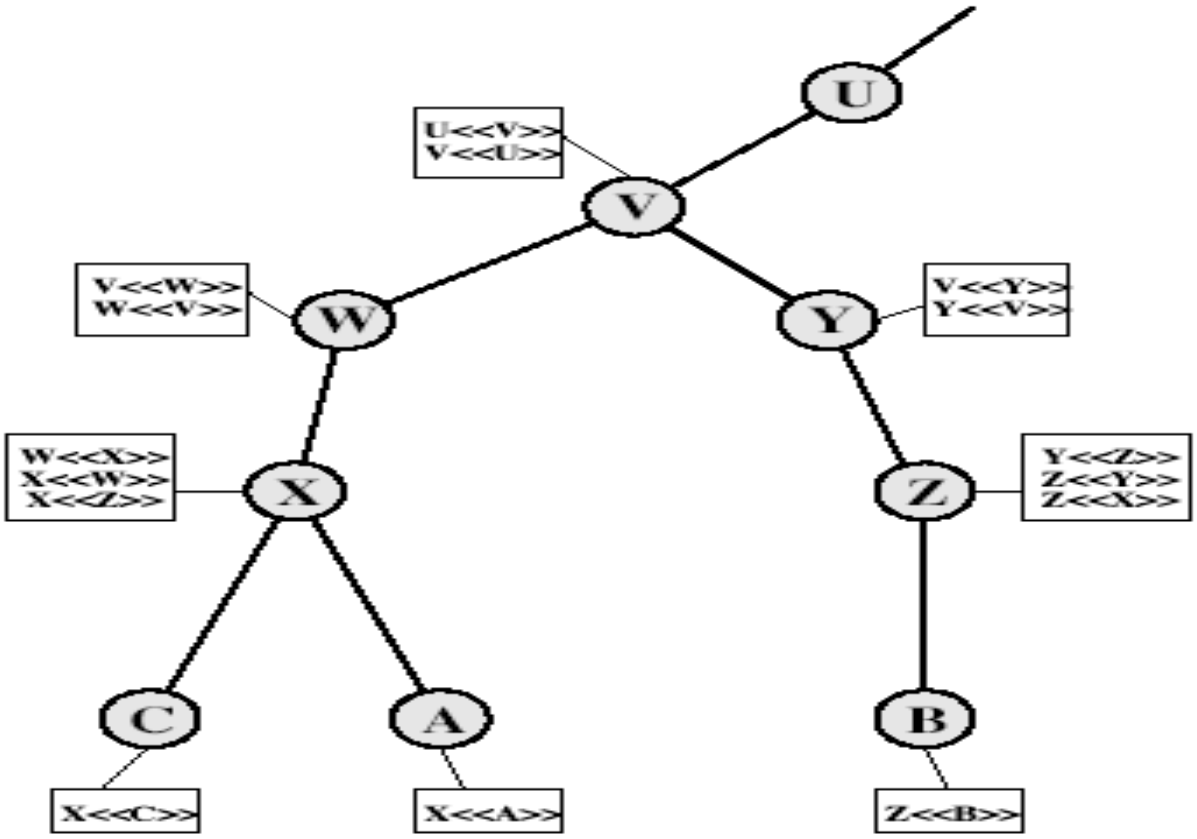
Obtaining a Certificate

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
  - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy

CA Hierarchy Use



Certificate Revocation

- certificates have a period of validity
- may need to revoke before expiry, eg:
  1. user's private key is compromised
  2. user is no longer certified by this CA
  3. CA's certificate is compromised
- CA’s maintain list of revoked certificates
  - the Certificate Revocation List (CRL)
- users should check certs with CA’s CRL

Authentication Procedures

- X.509 includes three alternative authentication procedures:
  - One-Way Authentication
  - Two-Way Authentication
  - Three-Way Authentication
  - all use public-key signatures

X.509 Version 3

- has been recognised that additional information is needed in a certificate
  - email/URL, policy details, usage constraints
- rather than explicitly naming new fields defined a general extension method
- extensions consist of:
  - extension identifier
  - criticality indicator
  - extension value

Certificate Extensions

- key and policy information
  - convey info about subject & issuer keys, plus indicators of certificate policy
- certificate subject and issuer attributes
  - support alternative names, in alternative formats for certificate subject and/or issuer
- certificate path constraints
  - allow constraints on use of certificates by other CA's

Certificate filename extensions

Common filename extensions for

X.509 certificates are:

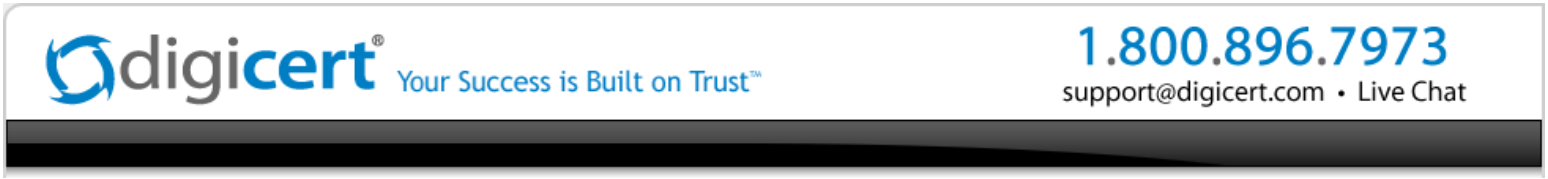
- .pem - (Privacy Enhanced Mail)
- .cer, .crt, .der
- .p7b, .p7c - PKCS#7 SignedData structure without data, just certificate(s) or CRL(s)
- .p12 - PKCS#12, may contain certificate(s) (public) and private keys (password protected)
- .pfx - e.g., with PFX files generated in IIS

Indian Licensed CAs

Controller of Certifying Authority CCA

- National Informatics Center (for govt. depts/undertakings only)
- (n)Code Solutions CA
- Safescrypt
- IDRBT(Institute for Development & Research in Banking Technology)
- TCS
- MTNL
- e Mudra CA

Well Known CAs



Well Known Global CAs

- Comodo
- DigiCert
- Symantec
- Entrust
- GeoTrust
- GlobalSign
- GoDaddy
- Network Solutions
- StartCom
- SwissSign
- Thawte
- Trustwave

Summary

- have considered:
  - Kerberos trusted key server system
  - X.509 authentication and certificates