Cryptography and Network Security

Third Edition

by William Stallings

Lecture slides by Lawrie Brown

Chapter 14 – Authentication Applications

We cannot enter into alliance with neighbouring princes until we are acquainted with their designs.

—*The Art of War*, Sun Tzu

Authentication Applications

- will consider authentication functions
- developed to support application-level authentication & digital signatures
- will consider Kerberos – a private-key authentication service
- then X.509 directory authentication service

Threats in a distributed environment

- Distributed computing model, client/server
- A user gains access to a WS, and pretend to be another
- A user alters the network address of a WS to impersonate another WS
- A user eavesdrops and uses a replay to gain entrance or disrupt operations

Kerberos

- trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
  - allows users access to services distributed through network
  - without needing to trust all workstations
  - rather all trust a central authentication server
  - Efficiency
- two versions in use: 4 & 5

Kerberos Requirements

- first published report identified its requirements as:
  - security
  - reliability
  - transparency
  - scalability
- implemented using an authentication protocol based on Needham-Schroeder
- A pure private-key scheme

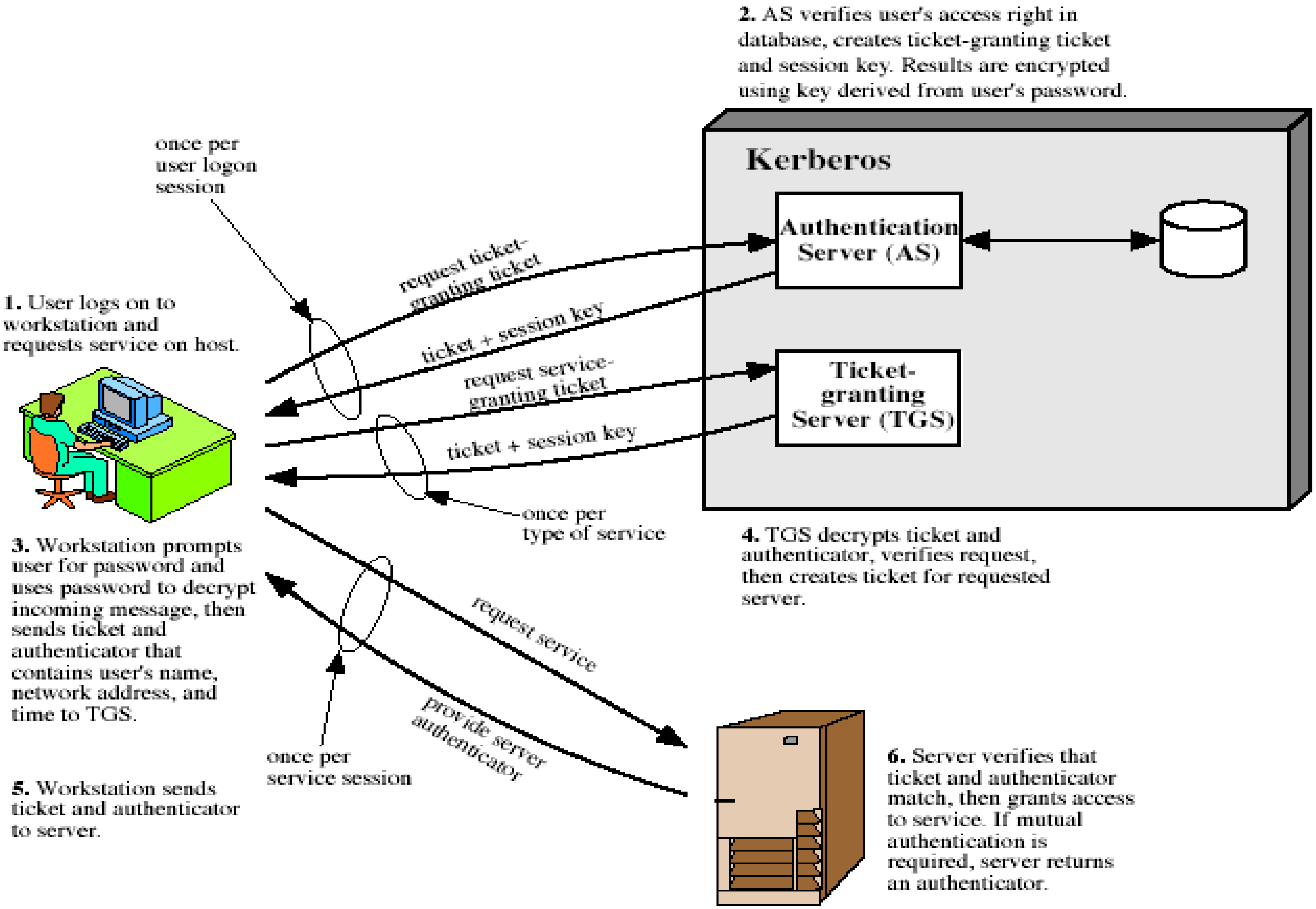
A 3-step improvements leading to Kerberos V4

- A simple authentication dialogue
  - Has to enter password for each server
  - Plaintext transmission of password
- AS+TGS model
  - Enter the password once for multiple services
  - Difficulty in choosing lifetime
- V4 model
  - Use private session keys
  - Can also verify server
  - AS is the KDC for (C, TGS)
  - TGS is the KDC for (C, V)

Kerberos 4 Overview

- a basic third-party authentication scheme
- have an Authentication Server (AS)
  - users initially negotiate with AS to identify self
  - AS provides a authentication credential (ticket granting ticket TGT)
- have a Ticket Granting server (TGS)
  - users subsequently request access to other services from TGS on basis of users TGT

Kerberos 4 Overview



1. User logs on to workstation and requests service on host.

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.

3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.

4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.

5. Workstation sends ticket and authenticator to server.

6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Kerberos
Authentication Server (AS)
Ticket-granting Server (TGS)
once per user logon session
request ticket-granting ticket
ticket + session key
request service-granting ticket
ticket + session key
once per type of service
request service
provide server authenticator
once per service session

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 realm
  - typically a single administrative domain
- Inter-realm authentication possible
  - Mutual trust required

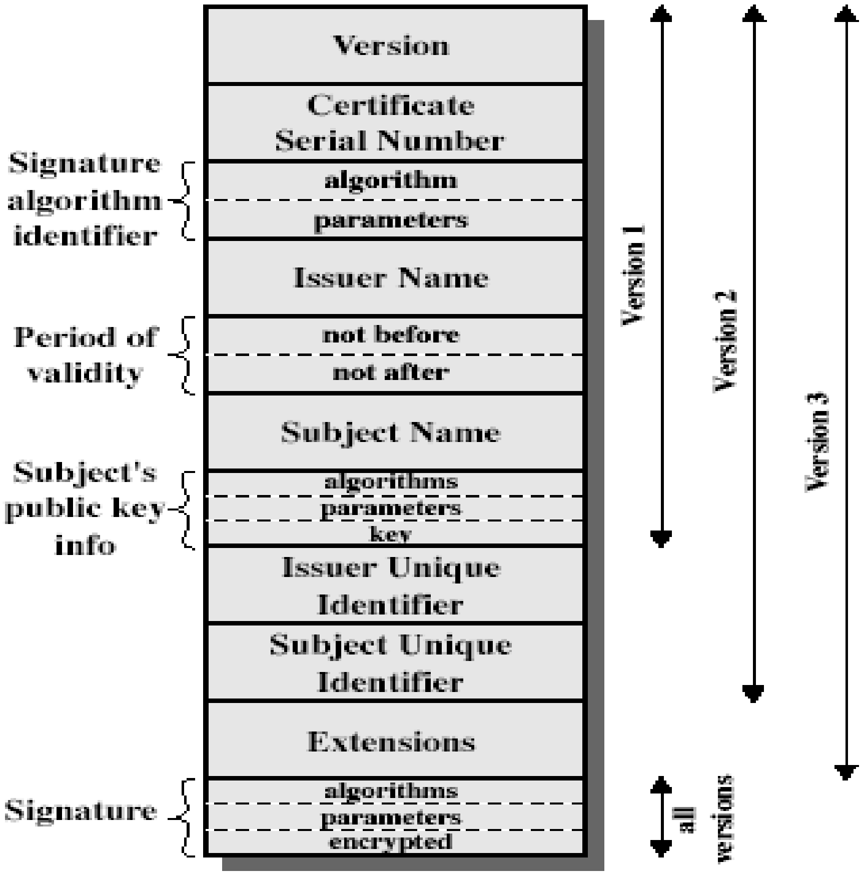
Kerberos Version 5

- developed in mid 1990’s
- provides improvements over v4
  - addresses environmental shortcomings
    - encryption alg, network protocol, byte order, ticket lifetime, authentication forwarding, interrealm auth
  - and technical deficiencies
    - double encryption, non-std mode of use, subsession keys
- specified as Internet standard RFC 1510

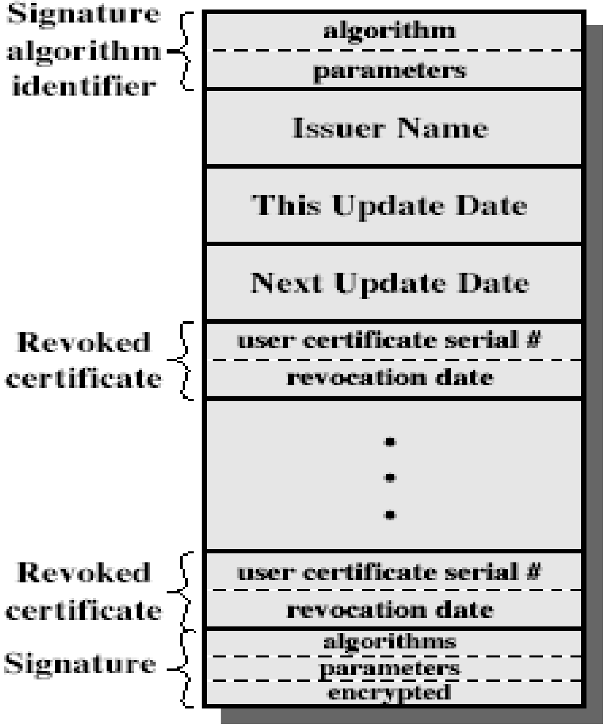
X.509 Authentication Service

- 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
  - Used in various contexts, e.g email security, IP security, web security

X.509 Certificates



(a) X.509 Certificate



(b) Certificate Revocation List

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+) , in case of name reuse
  - subject unique identifier (v2+) , in case of name reuse
  - extension fields (v3)
  - signature (of hash of all fields in certificate, encrypted by the private key of the CA)
- notation `CA<<A>>` denotes certificate for A signed by CA

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

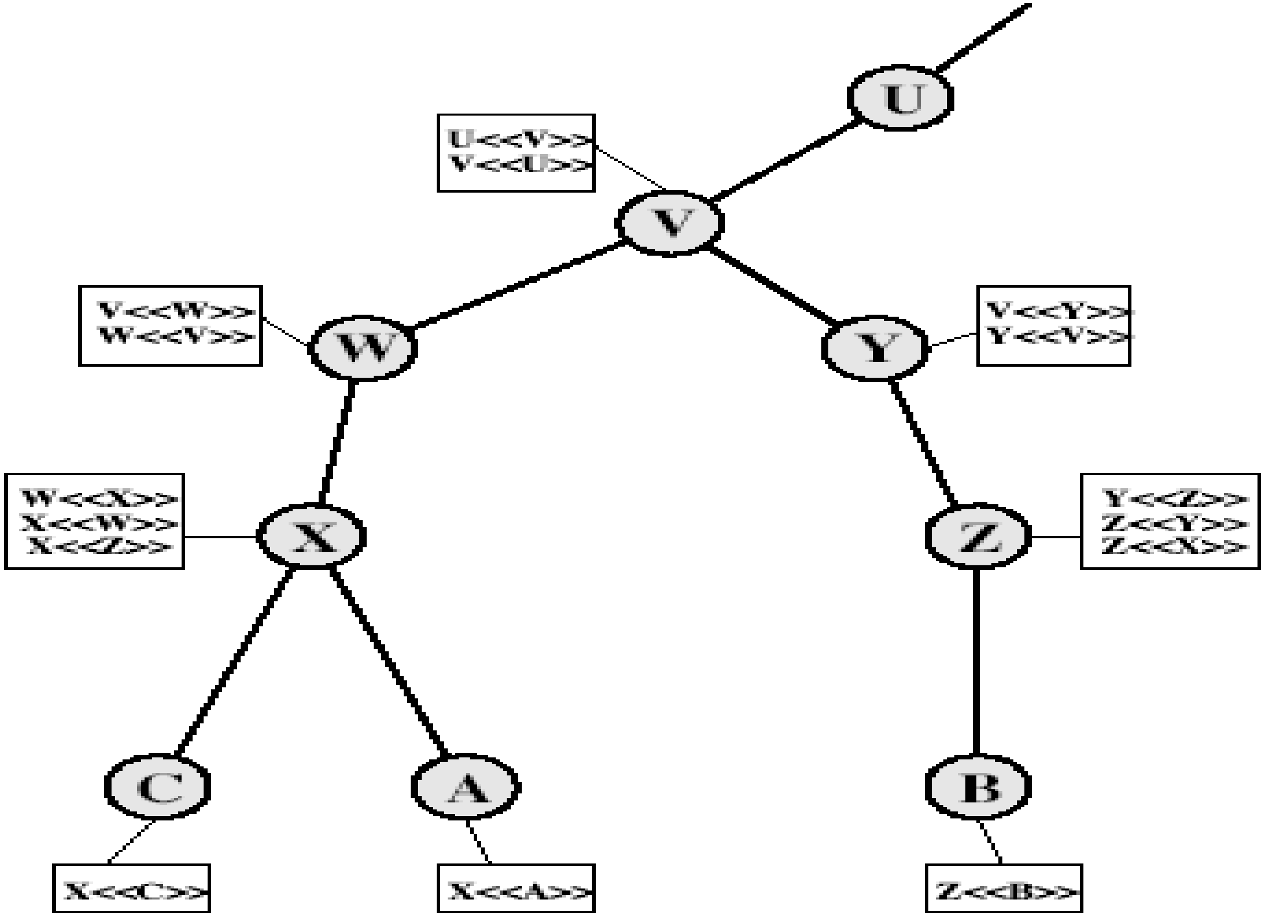
Multiple CAs

- Users in one CA are OK
- What if users from different CAs
  - A from X1
  - B from X2
  - B’s certificate is useless to A w/o knowing X2’s public key
  - Can work if two CAs exchanged public keys
  - A can use X1<<X2>> , X2<<B>>
- Chain: X1<<X2>> X2<<X3>> … XN<<B>>

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:
  - Assumes each already knows the certified public key of the other
- One-Way Authentication
- Two-Way Authentication
- Three-Way Authentication
- all use public-key signatures

One-Way Authentication

- 1 message (A->B) used to establish
  - the identity of A and that message is from A
  - message was intended for B
  - integrity & originality of message
- message must include timestamp, nonce, B's identity and is signed by A

Two-Way Authentication

- 2 messages (A->B, B->A) which also establishes in addition:
  - the identity of B and that reply is from B
  - that reply is intended for A
  - integrity & originality of reply
- reply includes original nonce from A, also timestamp and nonce from B

Three-Way Authentication

- 3 messages (A->B, B->A, A->B) which enables above authentication without synchronized clocks
- has reply from A back to B containing signed copy of nonce from B
- means that timestamps need not be checked or relied upon

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

Summary

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