SSH, PKI

Secure Shell ,Public Key Infrastructure,
Certificate, X.509, CA, Repository, RA, CRL

Agenda

- **SSH**
- **PKI**

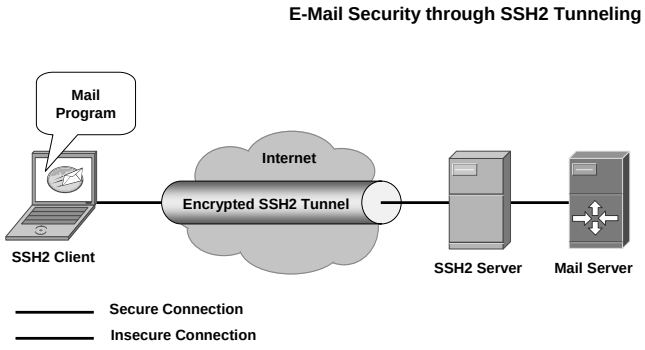
SSH Basics

- **Secures connections over the Internet**
  - Authentication (Client, Server)
  - Integrity, (Compression)
- **Encrypting all transmitted confidential data**
  - Passwords
  - Binary files
  - Administrative commands
- **Two versions of Secure Shell (not compatible)**
  - Secure Shell Version 1(SSH1 or SSH)
  - Secure Shell Version 2 (SSH2 or SecSH)
- **De-facto standard**
- **Client-server protocol**

SSH Basics

- **Solve two most acute problems in the Internet**
  - Secure remote terminal logins
    - ssh -l user-name machine-name
  - Secure remote command execution
    - ssh machine-name/path to exe-file
  - Secure file transfers
    - scp file user-name@machine-name
  - Port forwarding
    - ssh -L 3002:hostB:119 hostB
- **Tunnels TCP sessions over encrypted Secure Shell connection**
  - Secure the communication of other applications and protocols without modifying the applications

Principle

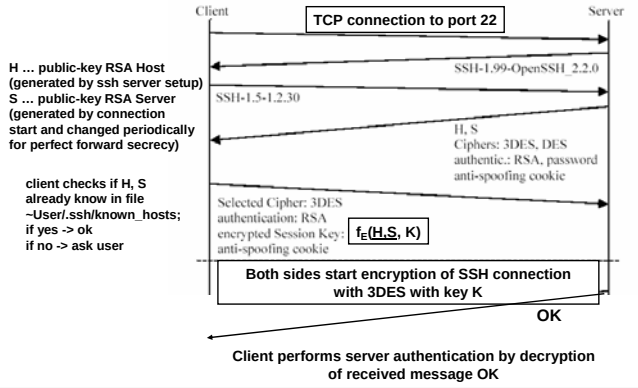


Encryption

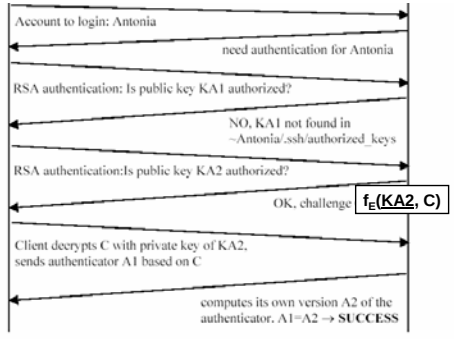
- **Support of the strongest available encryption algorithms**
  - 3DES
  - CAST-128
  - Twofish
  - AES
    - Advanced-Encryption-Standard (US)
    - 128-bit key!

Method	SSH1	SSH2
DES	X	-
3DES	X	X
IDEA	X	-
Blowfish	X	X
Twofish	-	X
Arcfour	-	X
AES	-	X
Cast128-cbc	-	X

SSH (v1) Procedures 1



SSH (v1) Procedures 2



SSH1 vs. SSH2

- **Two entirely different protocols**
- **SSH1 uses server and host keys to authenticate**
- **SSH2 only uses host keys and Diffie-Hellmann**
- **SSH2 encrypt different parts of the packet**
- **SSH2 is a complete rewrite of the protocol**
- **SSH2 is more secure**
- **Where to get:**
  - OpenSSH -> http://www.openssh.com/
    - ssh, scp, sftp, sshd, stfp-server
  - PuTTY -> http://www.chiark.greenend.org.uk/~sgtatham/putty/
    - Telnet and SSH client
  - SSH Tectia -> http://www.ssh.com/

Agenda

- **SSH**
- **PKI**

Public-Key Distribution

- **In order to verify a digital signature of a received message from Bob**
  - you compare the own computed hash of the message with the received signed hash
  - you need the public-key of Bob
- **In order to encrypt a message for Bob**
  - you will encrypt the message/session secret-key with Bob´s public-key
  - you need the public-key of Bob
- **Problem of secure public-key distribution**
  - man-in-the-middle

Solutions for Public-Key Distribution 1

- **Web of Trust**
  - public-keys are exchanged personally between persons
    - out-of-band transport
  - public-keys are exchanged over an insecure network between end-entities
    - in-band transport
    - verification of fingerprints over out-of-band network
  - public-keys are signed by trusted persons
  - end-entity to end-entity key exchange
  - does not scale
  - e.g. PGP

Solutions for Public-Key Distribution 2

- **Certification Authority (CA)**
  - "Trusted Third Party"
  - confirm that a public-key really belongs to a given person (end-entity)
  - done by usage of certificates
- **Certificate**
  - document which bind a name of an end-entity to a public-key
  - signed by an CA (using CA´s private-key) and verified using the public-key of the CA
- **Problem of key distribution reduced**
  - to get the public-key of the CA in a secure way

Handling of Certificates

- **Strength of the binding name to public-key**
  - depends on the policy of a CA
  - more critical usage of a given public-key means a stronger policy e.g. for identity control of public-key holder
  - Certification Practice Statements (CPS´s)
- **Storage of certificates**
  - on directory server, so called "Repository"
- **If Alice wants to get Bob´s public-key**
  - either get it from the repository (public-key + certificate) or Bob directly provides the public-key and the certificate
  - in both cases Alice verifies the validity of the CA´s signature using CA´s public-key

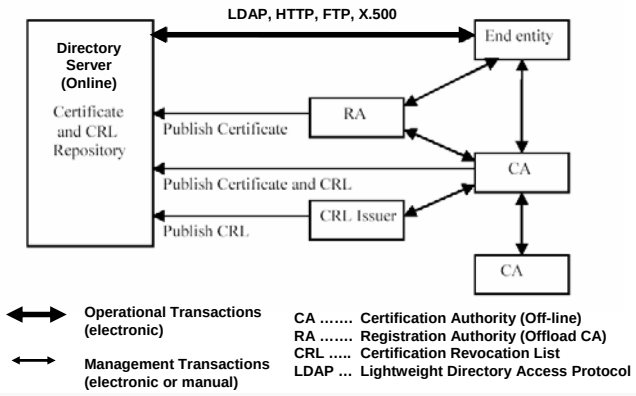
Standard Format for Certificates

- **ITU recommendation X.509**
  - X.500 standard series -> OSI directory systems
  - newest version is 3
  - certificates are encoded using ASN.1 (Abstract Syntax Notation 1)
- **RFC 3280**
  - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
  - defines a profile of X.509 for usage in the Internet
    - note: a so called profile defines actual chosen parameters/subset of a general standard specification
  - revocation methods
    - deals with how to stop validity of a signed public-key before end of lifetime of the issued certificate -> CRL lists

X.509 Basic Fields

Version:	**Which version of X.509**
Serial Number:	**Together with CA´s name uniquely identifies the certificate, also used in CRL´s**
Signature Algorithm:	**The algorithm used to sign the certificate**
Issuer Name:	**The name of the CA, usually a X.500 distinguished name**
Validity Period:	**The starting and ending times of a validity period**
Subject Name:	**The entity whose keys is being certified in the same format as the name of the CA**
Subject Public Key Info:	**The ID of the algorithm used and the subject´s public-key**
Issuer ID:	**An optional ID uniquely identifying the issuer**
Subject ID:	**An optional ID uniquely identifying the subject**
Signature Value:	**The certificate´s hash signed by the CA´s private-key (fingerprint)**

X.509 - PKI Infrastructure (RFC 3280)



 SSH, PKI, v4.3 17

PKI Transactions

- **Operational**
  - allow end user access to certificates and CRL lists
    - CRL is used to revoke a certificate before end of lifetime
    - lifetime of a certificate maybe some years
- **Management**
  - register user with a CA or RA
  - initializing end-entity with public-key of CA
  - certifying a public-key of an end-entity and publishing in repository
  - key-pair recovery (backup at the CA)
  - key-pair update (refresh the certificate)
  - request for key revocation from the end-entity
  - cross-certificates (certifies public-key of other CA)

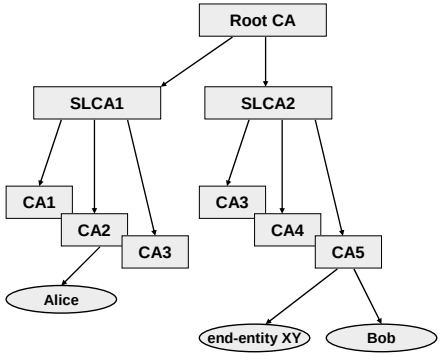
 SSH, PKI, v4.3 18

Simple Form of PKI: Hierarchy of CA´s 1

- **Root CA**
  - top-level CA
  - signs public-keys of second-level (SL) CA´s
  - = issuing certificates for second-level (SL) CA´s
- **SL CA**
  - signs public-keys of real CA´s
  - = issuing certificates for real CA´s
- **Real CA´s**
  - signs public-keys of end users
  - = issuing certificates for end users
- **Delegation/distribution of trust**

 SSH, PKI, v4.3 19

Simple Form of PKI: Hierarchy of CA´s 2

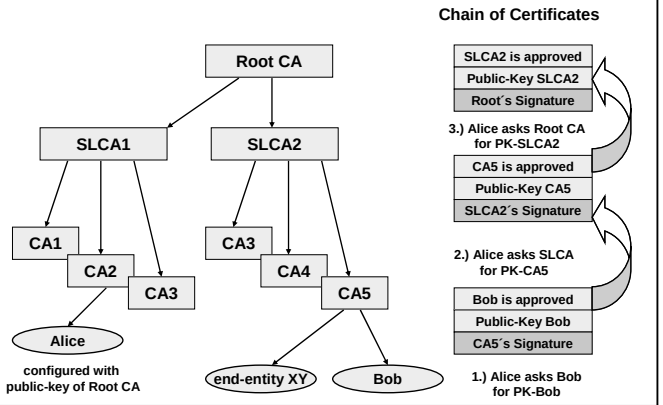


 SSH, PKI, v4.3 20

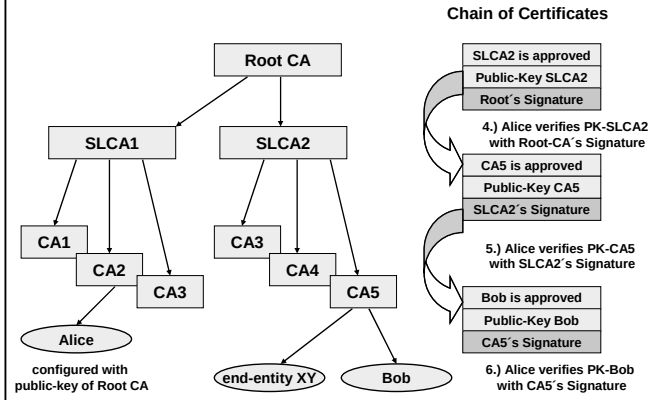
Simple Form of PKI: Hierarchy of CA´s 3

- **If an end user of one CA wants to verify a public-key of a end user registered by a foreign CA**
  - follow a chain of certificates towards root by getting in contact with every intermediate CA
    - chain of trust
    - certification path
  - verify each public-key starting from the root
    - note: end users must be configured with the public-key of the Root CA in a secure way
- **Instead of asking every intermediate CA**
  - the requested end user will provide his public-key with all necessary certificates for following the path towards the root

Simple Form of PKI: Hierarchy of CA´s 4



Simple Form of PKI: Hierarchy of CA´s 5



Hiearchy PKI versus Real Today's PKI

- **A single worldwide Root CA would be a single point of failure**
- **Nobody want do perform the Root CA**
  - political problem: some want a government organisation others want no government organisation
  - PEM suffered from this problem
- **Solution:**
  - have many roots, each with its own hierarchy of SLCA´s and real CA´s
  - modern browsers come preloaded with the public-keys for over 100 roots -> trust anchors
  - trust anchors need not to be at the root level

Miscellaneous Terms 1

- **Self-signed certificate**
  - issuer name = subject name
  - used for distributing CA´s public-key in a certificate
- **Cross-certified CA´s**
  - alternative to strict CA hierarchy of trust model
  - CA´s form a flat hierarchy (= CA´s are single root for themselves) and certifies (signs) each other
  - black hole in PKI

Miscellaneous Terms 2

- **Enrolment**
  - is the procedure of adding a PKI user (certificate holder) to the PKI in a secure way
    - on the CA side: Have we received the correct client´s PK?
    - on the client side: Have we received the correct CA certificate?
  - mutual authentication is necessary
  - integrity must be checked
    - via out-band fingerprint verification if performed over non-secure network connections
  - e.g. SCEP (Simple Certificate Enrolment Protocol) in the VPN arena
    - enables VPN devices (router/firewall with IPsec) to enrol to the PKI server (e.g. VeriSign OnSite Service, MS Windows 2000 Certification Services, ….)

Institute of Computer Technology - Vienna University of Technology

L98 - SSH, Public Key Infrastructure

Today's PKI and Future PKI

- **Many products for PKI on the market**
- **But standardization of PKI and worldwide-organisation of PKI**
  - is an ongoing story
  - many problems need to be solved
  - therefore worldwide PKI and worldwide PKI certification paths are still far away