6.857 Computer and Network Security October 25, 2001

Lecture Notes 14 : Public-Key Infrastructure

Lecturer: Ron Rivest *Scribe: Armour/Johann-Berkel/Owsley/Quealy*

1 Outline

- Public-Key Infrastructure
- X.509
- SPKI/SDSI

2 Public-Key Infrastructure (PKI)

2.1 Introduction

In cyberspace there is a need to verify the identities of individuals for a number of purposes. Some of these events include sending and receiving secure email, sending and receiving signed email, setting up a secure session (SSL), and accessing a protected resource. The way in which this goal of authentication is accomplished is by verifying that a public key belongs to an individual that you know and trust. Public-Key Infrastructure is designed to allow this kind of authentication.

2.2 Diffie Hellman Public-Key Encryption

"Public-Key Directory"

One way to associate public keys to individuals is by publishing a mapping of names to keys. This directory would act much like the WhitePages does for distributing phone numbers based on name. The directory must be trusted, therefore it must be authentic but need not be secret. Entries would be of the form:

$$\text{Alice} \longrightarrow (RSA, n = ..., e = 3)$$

$$\text{Bob} \longrightarrow (RSA, n = ..., e = 17)$$

Problem: Need to authenticate the issuer of the directory.
Solution: A possible solution would be for the issuer to sign the whole directory. (how do we get the issuer's PK? must be recursive)

Digital Certificates

Digital Certificates were proposed by Loren Kohnfelder here at MIT in a B.S. thesis in '78. They are an authenticated identifier pairing the public key to a significant name. This allows any user to identify themselves and establishes trust between themselves and a verifier who trusts the certificate authority. The CA is assumed to correctly identify the person who has requested the certificate.

This is the structure of a signed digital certificate.

$$\{\text{“}Alice'', (RSA, ...)\}_{CA}$$

Here is a representation of the exchange between a user with a certificate and a verifier.

$$(M)_{SK_A,cert} \longrightarrow Bob \text{ “relying party”}$$

Question: How does PKI deal with issues of dynamics in naming such as changing email addresses?
Answer: This does present a problem since information can change. The major issue becomes one of database update however.

Advantages

- Alice can include her certificate in an email or post it on the Web
- Bob only needs to know the Certificate Authority and its PK
- Alice may have more than one key (e.g., one for signing, one for encryption)
- Certificates can have a validity period (not before / not after a certain time)

Difficulty Issues

- Scalability
  - need multiple CAs
  - naming (unique? human-readable?)
- Robustness
  -compromised keys? (especially the root key!)
  -revoked certificate
- Certificate as Credential (Attribute Certificate instead of ID certificate)
- Trustworthiness of CA and procedures; liability?
- Privacy, Anonymity

Question: How do we deal with privacy issues in the CA?
Answer: Use certificate serial numbers instead of names.

2.3 Naming

PK infrastructure has a very intimate link with naming. We want a system that is easy to use for people, similar to that of file names. The naming relationship should be as follows:

- **Names** are for people to use.
- **Keys** are for machines to use.
- **PKI** can provide a binding between the two.

Naming is a large issue. Since the CA has the burden of properly identifying and labeling the parties with certificates, names must be made clear and accurate.

Naming provides an interface between people and cyberspace. People must then write security policy based on the name associated with a PK used to sign message. Writers of such policies need to know/understand the relationship between keys and names.

Desirable naming properties

- Descriptive
- Global uniqueness
- Dynamic

Examples

- Role (purchasing agent at IBM)
- Legal names
- Email
- Phone #'s ("enum")
- Mail address

Certificates can also be used for identifying much more about an individual than just identity. Attributes of a person can be given by certificates. For attributes, what exactly is the CA allowed to certify?

Example: John has brown hair.

How do they know?

Why should we believe them?

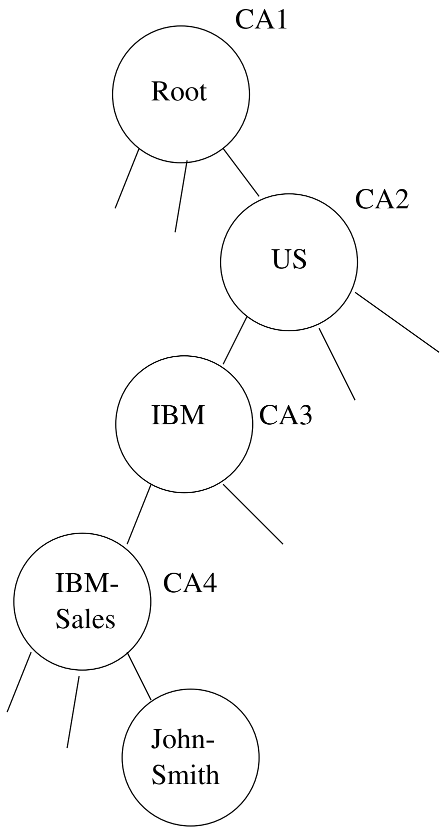
3 X.509

X.509 is one of the most popular standards specifying the contents of a digital certificate. One of the main goals of X.509 is global uniqueness of names.

3.1 X.509 Hierarchical Structure

X.509 maintains properties of distinguished names (DNs) and is organized in a hierarchical structure. This naming scheme for certificates traverses through local CAs until arriving at a specific name. Each local CA is responsible for only certificates in its specific domain.

Below is the graphical representation of the path between the root CA and John Smith. /root/us/ibm/ibm-sales/John-Smith



Some major problems with DN here is that single points of failure disrupt the system. The structure itself is also awkward.

Question: What happens if the root CA is compromised?
Answer: Instead of having one root CA we can implement a threshold system. This would help

eliminate single points of failure.

3.2 What's included in X.509 version 3 certificates?

- Version #
- Certificate Serial #
- Signature Algorithm Identifier
- Issuer Distinguished Name (DN)
- Validity Period
- Subject DN
- Subject PK Information
  - algorithm identifier
  - associated key parameters
- Issuer Unique #
- Subject Unique #
- Extensions
  - key usage
  - certificate policies
  - subject/issuer alternate names
  - path constraints
  - criticality bits

3.3 Revocation or Compromised Key

Is the assertion of the certificate ("Public Key is Alice's PK") valid any more? Who's decision is it to revoke a certificate? Revocation is hard to do. It is much easier to have certificates expire. A good method is to use short validity periods.

Certificate Revocation Lists (CRLs)

CRLs provide a listing of serial numbers of revoked certificates. For example, if Alice's Laptop were stolen and her secret key compromised, we would want her certificate to be revoked. CRLs could also include the reasons why certificates were revoked. A consideration with CRLs is how frequently they should be updated (daily?, weekly?, monthly?). Another option for updating CRLs is to post $\triangle$ CRLs which only update CRLs whenever a change occurs.

CRLs present a problem since they place the burden of certificate validity upon the CA and not the verifier. For example, if the CA only updated CRLs monthly but Bob wants daily verification, a conflict arises. One possible solution for Bob would be to refuse certificates that had not been signed within the last 24 hours. In this way Bob can set his own security policies.

For maximum verification of certificates there should be short validity periods and then the relying party can determine what is fresh enough.

Question: What if certificates were stored on a piece of hardware such as a smart card?
Answer: This is not a primary concern because we are currently focused upon the theory behind certificate freshness.

Question: Would X.509 have to redo the US certificate every day?
Answer: Yes. This shows the limitations of X.509. At each level new certificates are reissued for desired freshness. Another possible solution is to reveal a new hash value for the certificate each day. Assuming that we use a OW hash, the values guarantee the certificate is still valid.

$$\overset{x_{31}}{\bullet} \longrightarrow \ldots \longrightarrow \overset{x_4}{\bullet} \longrightarrow \overset{x_3}{\bullet} \longrightarrow \overset{x_2}{\bullet} \longrightarrow \overset{x_1}{\bullet} \longrightarrow \overset{x_0}{\bullet}$$

In this diagram the certificate originally has the value x_0. On day one value x_1 is sent to the certificate holder such that the hash of $x_1 = x_0$. This is repeated everyday the certificate is still valid. This hashing technique is due to Silvio Micali.

4 SPKI/SDSI

SPKI/SDSI is a novel PKI emphasizing naming, groups, ease-of-use, and flexible authorization. To access a protected resource, a client must present to the server a proof that the client is authorized. This proof takes the form of a "certificate chain" proving that the clients public key is in one of the groups on the resource's ACL, or that the clients public key has been delegated authority from a key in one of the groups on the resource's ACL.

How it works

- No global names
- Each key has its own local name space
- Two types of certificates: Name and Authority

Name Cert:

$$K\ Alice \text{ (local name)} \longrightarrow value \text{ (PK, or another name)}$$

K = your key

$K_0 = \text{MIT}$

$K_1 = \text{EECS}$

Example 1

$$K\ Alice \longrightarrow K_0\ AliceSmith$$

$$K_0\ AliceSmith \longrightarrow K_s$$

$$K\ Alice \longrightarrow K_s$$

Example 2

$$K\ Alice \longrightarrow K_0\ EECS\ AliceSmith$$

$$K_0\ EECS \longrightarrow K_1$$

$$K_1\ AliceSmith \longrightarrow K_s$$

Example 3, Groups

$$K\ Friends \longrightarrow K_1\ Alice$$

$$K\ Friends \longrightarrow K_1\ Bob$$

$$K\ Friends \longrightarrow K_0\ EECS\ student$$

This a bottom-up approach to authentication. As opposed to the top-down approach outlined in X.509.