Simple Public Key Infrastructure
(SPKI Certificate Theory, RFC 2693)

Varun Pandey
Department of Computer Science
Virginia Tech
X.509 certificates

Aim

authentication

• <public keys, X.500 distinguished names (globally unique)> signed by a CA

  X.500 DN: {CN=John Duke, OU=Java Software Division, O=Sun Microsystems Inc, C=US}

• Alternative Names (allows other identities to be associated with the public key, for e.g. email addresses, DNS names, IP addresses, URIs)

  (X.509 version 3, March 1999, rfc2459)

access control

• which X.500 directory nodes can be modified by which key-holder
  (X.509 version 2, 1993, never widely used)

• Key Usage (limits the usage of keys for purpose such as “signing only”)

  (X.509 version 3, March 1999, rfc2459)
X.509 basic key usage (only in version 3)

Defines the purpose of the key

id-ce-keyUsage OBJECT IDENTIFIER ::= { id-ce 15 }

KeyUsage ::= BIT STRING {
    digitalSignature (0),
    nonRepudiation (1),
    keyEncipherment (2),
    dataEncipherment (3),
    keyAgreement (4),
    keyCertSign (5),
    cRLSign (6),
    encipherOnly (7),
    decipherOnly (8)
}
X.509 extended key usage (only in version 3)

- **id-kp-serverAuth** OBJECT IDENTIFIER ::= {id-kp 1}
  -- TLS Web server authentication
  -- Key usage bits that may be consistent: digitalSignature, keyEncipherment or keyAgreement

- **id-kp-clientAuth** OBJECT IDENTIFIER ::= {id-kp 2}
  -- TLS Web client authentication
  -- Key usage bits that may be consistent: digitalSignature and/or keyAgreement

- **id-kp-codeSigning** OBJECT IDENTIFIER ::= {id-kp 3}
  -- Signing of downloadable executable code
  -- Key usage bits that may be consistent: digitalSignature

- **id-kp-emailProtection** OBJECT IDENTIFIER ::= {id-kp 4}
  -- E-mail protection
  -- Key usage bits that may be consistent: digitalSignature, nonRepudiation, and/or (keyEncipherment or keyAgreement)

- **id-kp-timeStamping** OBJECT IDENTIFIER ::= {id-kp 8}
  -- Binding the hash of an object to a time from an agreed-upon time
  -- Source. Key usage bits that may be consistent: digitalSignature, nonRepudiation
To handle the possibility of reuse of subject and/or issuer names over time. (v3 recommends not to use these extensions.)
X.509 certificates

Problems

1. collection of directory entries are considered valuable (e.g. CIA’s list of agents)

2. requires a single, global naming discipline: already too many entities in the business of defining names, not under a single discipline. So uniqueness is a problem.

3. certificate revocation lists: number of CRLs to be issued and frequency of issuing them (for e.g. with change of names, institution, organization etc. on a global scale)
Local Names

SPKI uses SDSI 2.0 (Simple Distributed Security Infrastructure) local naming scheme.
- Operational simplicity
- Security advantage

**Basic SDSI name:** namespace: (name “chosen name”)

Example: george: (name fred)
- defines a name “fred” in the namespace “george” (anchor namespace)

**Compound SDSI name:**

Example: fred: (name sam)
- a name “sam” in the namespace “fred”
  george: (name fred sam)
- name “sam” in the namespace “fred”
  which is in the namespace “george” (anchor)
Fully qualified SDSI names using Public Key

In SPKI, the SDSI name is made globally unique by anchoring it using Public Key for e.g. \((\text{name (public-key encryption } K_{JOHN} \text{) MOM HUSBAND})\)

\[
\text{name (public-key (rsa-pkcs1-sha1 (e #23#) (n [AlmuUeR3N//ZTUducfR8M8gxMpkkhiqcZJDBnEnPqFdo1bjgf1Dx2mZBEvZ+AllanBmnk55dJsB5sCh1hR4KRwEJuVQHCW2AO04eDKsgGlgel2U9L4bncwfiEj0tbpSp5bDuzq6vcUHacXIL4xL63Mmc65RUF8rnPupHsdlwLBgt|) ) ) MOM HUSBAND )}
\]

\(\text{MOM} \equiv \text{another SDSI name } K_{JOHN} \text{ MOM in the namespace } K_{JOHN} \)

\(<K_{JOHN} \text{ MOM, } K_{MOM}> \rightarrow \text{binding of SDSI name to a key } K_{MOM} \)

\(\text{HUSBAND} \equiv \text{another SDSI name } K_{JOHN} \text{ MOM HUSBAND in the namespace bound to the key } K_{MOM} \)

IN SPKI the principal (identity) is associated with the public key and not with the person i.e. the public key defines the namespace unlike the X.509 distinguished names such as \(O=\text{MIT}\).
Group memberships using SPKI

\(<K_{kafura}, K_{CS \; 6204} >\)
\(<K_{CS \; 6204}, K_{glenn} >\)
\(<K_{CS \; 6204}, K_{varun} >\)
\(<K_{CS \; 6204}, K_{craig} >\)
\(<K_{CS \; 6204}, K_{ranjit} >\)

Advantage of local namespace: any change in the link above for e.g.
\(<K_{kafura}, K_{CS \; 6204} >\), doesn’t require a change in the other certificates.
In X.509 distinguished naming scheme, a change requiring a change in any link
required a change in many certificates.
Authorization and Naming

New feature: SPKI separates authorization from naming (unlike X.509 certificates)

SPKI authorization mapping:
  authorization → key
  for e.g. <K_{JOHN}, authorization attributes>
  An authorization certificate will have only PUBLIC KEY as the issuer

SPKI naming certificate:
  key → name
  for e.g. <K_{JOHN} MOM, K_{MOM}>
  A naming certificate will have only
  (PUBLIC_KEY ONLY_ONE_IDENTIFIER)
  as the issuer to ensure one has control over ONLY his/her namespace
  Naming certificates don’t carry any authority
Delegation

SPKI provides Boolean control (TRUE/FALSE) for delegation.
So a subject of a certificate may him/herself exercise permission granted
Also if delegation=TRUE may also delegate all or part of the permissions

<table>
<thead>
<tr>
<th>DELEGATION</th>
<th>Access Control Lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only one ACL entry at PEP</td>
<td>List of &lt;name, access attributes&gt;</td>
</tr>
<tr>
<td>Management simple</td>
<td>required at PEP</td>
</tr>
<tr>
<td>to gain access a chain of certificates</td>
<td>List would become very huge</td>
</tr>
<tr>
<td>up to the one ACL entry required</td>
<td>Management a problem</td>
</tr>
<tr>
<td>the chain of certificates are issued</td>
<td></td>
</tr>
<tr>
<td>by users with the power to delegate</td>
<td></td>
</tr>
</tbody>
</table>
Validity Conditions

SPKI permits use of

1. **Timed CRLs**
   1. Certificate must list the key that would sign the CRL and also mention the location to fetch the CRL from
   2. CRL must have validity dates
   3. CRL validity dates must not intersect with previous CRLs

2. **Timed Revalidations**
   1. Instead of CRL, fetch certificate valid list under the same rules as above
   3. **Setting the Validity Interval in the certificate**
      1. Set the valid interval in certificates based on risk management and expected monetary loss. If the certificate has been compromised just after the issue of a CRL the false certificate can be included only in the next CRL.

4. **One-time revalidation**
   1. Equivalent to validity interval equal to 0. A nonce is used to prevent replay attack. Useful for applications requiring one-time authorization.

5. **Short-lived certificates**
Authorization and Naming Certificates

Authorization certificate ≡ 5-tuple

5-tuple: \(<\text{Issuer}, \text{Subject}, \text{Delegation}, \text{Authorization}, \text{Validity}>\)

Name certificate ≡ 4-tuple

4-tuple: \(<\text{Issuer}, \text{Name}, \text{Subject}, \text{Validity}>\)

Issuer: public key (or hash) or the work “Self”. Identifies the entity speaking the intermediate result in reduction or defining the name in the private namespace.

Name: a byte string

Subject: public key / name

Delegation: a boolean. If TRUE, then subject can delegate permission in this intermediate result.

Authorization: an S-expression

Validity: a not-before-date or not-after-date. If not-before-date missing then minus infinity is assumed. If not-after-date is missing then plus infinity is assumed.
Typical SPKI/SDSI Naming Certificate

SPKI/SDSI certificates are encoded as S-expressions (Lisp-like expressions)
e.g. a typical naming certificate

(cert
  (issuer (name
    (public-key (rsa-pkcs1-sha1
      (e #23#)
      (n
        |AImuUeR3N//ZTUducfR8M8gxMpkkh
        iqcZJDBnEnPqFd01bjgf1Dx2mZBEvZ+
        AllanBmnk55dJsB5sCh1hR4KRwEJuVQ
        HCW2AO04eDKsgGlgel2U9L4bncwfiE
        j0tbpSp5bDuzq6vcUHacXIL4xL63Mmc
        65RUF8rnPupHsdlwLBgt|))
    Masters-Alumnus))
  (subject (name Jean-Emile_Elien)))
  (not-before `1998-06-05_15:00:00")
)

Authorization certificate reduction rule

\[ \langle I_1, S_1, D_1, A_1, V_1 \rangle + \langle I_2, S_2, D_2, A_2, V_2 \rangle = \langle I_1, S_2, D_2, A_{\text{Intersect}(A_1,A_2)}, V_{\text{Intersect}(V_1,V_2)} \rangle \]

provided \( S_1 = I_2 \) and \( D_1 = \text{true} \)
AIntersect

Actual authorization string definitions are application dependent
Special semantics would require special reduction software
AIntersect provides rules for automatic intersection of the authorization
strings for application developers to know the semantics
Match is element by element. If the shorter list is a subset of the longer list
then the longer list is the result.

For e.g.
A1 = (ftp (host ftp.clark.net))  =>  ftp access to all directories and all
access i.e. read, write, delete etc.
A2 = (ftp (host ftp.clark.net) (dir /pub/cme))
    => ftp access to only directory /pub/cme
    with full access permissions
AIntersect(A1,A2) = (ftp (host ftp.clark.net) (dir /pub/cme))
\textbf{AIntersect (contd.)}

\begin{align*}
\text{AIntersect} & \left( \tag{ftp \ ftp.clark.net cme (* set read write)) , (tag (*)) \right) \\
& = (\tag{ftp ftp.clark.net cme (* set read write)) )
\end{align*}

\begin{align*}
\text{AIntersect} & \left( \tag{* prefix http://www.clark.net/pub/) , (tag (* prefix http://www.clark.net/pub/cme/html/) ) \right) \\
& = (\tag{* prefix http://www.clark.net/pub/cme/html/) )
\end{align*}
VIntersect

Straight forward intersection:

V = VIntersect( (Xmin, Xmax), (Ymin, Ymax) )
Vmin = max(Xmin, Ymin)
Vmax = min(Xmax, Ymax)
if ( Vmin > Vmax )
    no intersection;
else
    intersection;
Naming certificate reduction rules

Two kinds of name certificates:

1. \([ \text{name } K_1 N \rightarrow K_2 ]\) (defines the name as a key)

2. \([ \text{name } K_1 N \rightarrow (\text{name } K_2 N_1 N_2 \ldots N_k ) ]\) (defines the name as another name)

Rule for reduction: replace the name just defined by its definition

- e.g. for case 1
  \(\text{name } K_1 N N_1 N_2 N_3 + [\text{name } K_1 N \rightarrow K_2] \rightarrow (\text{name } K_2 N_1 N_2 N_3)\)

- e.g. for case 2
  \(\text{name } K_1 N Na Nb Nc + [\text{name } K_1 N \rightarrow (\text{name } K_2 N_1 N_2 \ldots N_k ) ] \rightarrow (\text{name } K_2 N_1 N_2 \ldots N_k Na Nb Nc)\)

Users are expected to deliver certificates in order as needed by the prover. Validity intersection is the same as discussed before.
Certificate Discovery

John wants to read the online course catalog on VT’s server. Vincent is the VT Registrar’s server. Vincent has an acl entry for the SDSI name \(<K_{VT \text{ STUDENT}}\) such that any VT student can read the course catalog.

A sample protocol for certificate discovery:
1. John sends Vincent a request for access
2. Vincent sends back the SDSI names allowed access
3. John sends back a valid certificate chain to Vincent.

An example of a valid chain would be:
\(<K_{Vincent}, K_{VT \text{ STUDENT}}\> \quad \text{(delegation of authority to SDSI name } K_{VT \text{ STUDENT}})\)
\(<K_{VT \text{ STUDENT}}, K_{VT \text{ CS STUDENT}}\> \quad \text{(name certificate)}\)
\(<K_{VT \text{ CS}}, K_{CS}\> \quad \text{(name certificate)}\)
\(<K_{CS \text{ STUDENT}}, K_{CS \text{ CS-STUDENT}}\> \quad \text{(name certificate)}\)
\(<K_{CS \text{ CS-STUDENT}}, K_{John}\> \quad \text{(name certificate)}\)

John is supposed to provide the above mentioned certificates in sequence to gain access to the VT’s server.
Threshold Subjects

Defined by two parameters: N and K
Usage: K of N subjects are required to sign a particular request or transaction.

K of the N subjects are reduced to the same subject individually through a sequence of certificates.

An intermediate form for a threshold subject would be copy of a tuple with only that particular threshold subject. The tuple is reduced independently to a particular common subject.

Then the K tuples with the same final subject are processed for validity-intersection, authorization-intersection and delegation-intersection to yield a single tuple.
References

