Carl M. Ellison
Intel

28 May 1999

SPKI Requirements

<draft-ietf-spki-cert-req-03.txt>

Status of This Document

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

This document supersedes draft-ietf-spki-cert-req-02.txt.

It is a compilation of the requirements for SPKI [Simple Public Key Infrastructure] certificates gathered from the discussion on the SPKI Working Group mailing list. It is provided for information only.

Distribution of this document is unlimited. Comments should be sent to the SPKI (Simple Public Key Infrastructure) Working Group mailing list <spki@c2.net> or to the author.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months. Internet-Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as a ``working draft'' or ``work in progress.''

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf/org/shadow.html

Abstract

The IETF Simple Public Key Infrastructure [SPKI] Working Group is tasked with producing a certificate structure and operating procedure to meet the needs of the Internet community for trust management in as easy, simple and extensible a way as possible.

The SPKI Working Group first established a list of things one might want to do with certificates (attached at the end of this document), and then summarized that list of desires into requirements. This document presents that summary of requirements.

Charter of the SPKI working group

Many Internet protocols and applications which use the Internet employ public key technology for security purposes and require a public key infrastructure to manage public keys.

The task of the working group will be to develop Internet standards for an IETF sponsored public key certificate format, associated signature and other formats, and key acquisition protocols. The key certificate format and associated protocols are to be simple to understand, implement, and use. For purposes of the working group, the resulting formats and protocols are to be known as the Simple Public Key Infrastructure, or SPKI.

The SPKI is intended to provide mechanisms to support security in a wide range of Internet applications, including IPSEC protocols, encrypted electronic mail and WWW documents, payment protocols, and any other application which will require the use of public key certificates and the ability to access them. It is intended that the Simple Public Key Infrastructure will support a range of trust models.

Table Of Contents

Background

The term certificate traces back to the MIT bachelor's thesis of Loren M. Kohnfelder [KOHN]. Kohnfelder, in turn, was responding to a suggestion by Diffie and Hellman in their seminal paper [DH]. Diffie and Hellman noted that with public key cryptography, one no longer needs a secure channel over which to transmit secret keys between communicants. Instead, they suggested, one can publish a modified telephone book -- one with public keys in place of telephone numbers. One could then look up his or her desired communication partner in the directory, find that person's public key and open a secure channel to that person. Kohnfelder took that suggestion and noted that an on-line service has the disadvantage of being a performance bottleneck. To replace it, he proposed creation of digitally signed directory entries which he called certificates. In the time since 1978, the term certificate has frequently been assumed to mean a binding between name and key.

The SPKI team directly addressed the issue of <name,key> bindings and realized that such certificates are of extremely limited use for trust management. A keyholder's name is one attribute of the keyholder, but as can be seen in the list of needs in this document, a person's name is rarely of security interest. A user of a certificate needs to know whether a given keyholder has been granted some specific authorization.

General Requirements

We define the term KEYHOLDER of a public key to refer to the person or other entity that controls the corresponding private key.

The main purpose of an SPKI certificate is to authorize some action, give permission, grant a capability, etc. to or for a keyholder.

The keyholder is most directly identified by the public key itself, although for convenience or other purposes some indirection (delayed binding) may be employed. That indirection can be via a collision-free hash of the public key or via a name, later to be resolved into a key.

The definition of attributes or authorizations in a certificate is up to the author of code which uses the certificate. The creation of new authorizations should not require interaction with any other person or organization but rather be under the total control of the author of the code using the certificate.

Because SPKI certificates might carry information that the keyholder might not want to publish, we assume that certificates will be distributed directly by the keyholder to the verifier. If the keyholder wishes to use a global repository, such as LDAP, the global PGP key server or the DNS database, that is up to the keyholder and not for the SPKI WG to specify.

Because SPKI certificates will carry information that, taken together over all certificates, might constitute a dossier and therefore a privacy violation, each SPKI certificate should carry the minimum information necessary to get a job done. The SPKI certificate is then to be like a single key rather than a key ring or a single credit card rather than a whole wallet. The keyholder should be able to release a minimum of information in order to prove his or her permission to act.

It is necessary for at least some certificates to be anonymous.

Because one use of SPKI certificates is in secret balloting and similar applications, an SPKI certificate must be able to assign an attribute to a blinded signature key.

One attribute of a keyholder is a name. There are names the keyholder prefers to be called and there are names by which the keyholder is known to various other keyholders. An SPKI certificate must be able to bind a key to such names. The SDSI work of Rivest and Lampson has done an especially good job of defining and using local name spaces, therefore if possible SPKI should support the SDSI name construct. [Note: SPKI and SDSI have merged.]

Validity and CRLs

An SPKI certificate, like any other, should be able to carry a validity period: dates within which it is valid. It may also be necessary to have on-line refinement of validity. This is frequently achieved via a Certificate Revocation List (CRL) in previous certificate designs.

A minimal CRL contains a list of revoked certificates, identified uniquely, a sequence number and a signature. Its method of transmission is not specified. If it encounters some certificate that it lists, then it annihilates that certificate. If it encounters a previous CRL, as indicated by sequence number, then it annihilates that previous CRL. Such a CRL leads to non-deterministic program behavior. Therefore, we take as a requirement that if SPKI uses CRLs, then the certificate that uses it must explicitly tell the verifier where to find the CRL, the CRL must carry explicit validity dates and the dates of a sequence of CRLs must not overlap. Under this set of requirements, behavior of certificate validation is deterministic (aside from the question of clock skew).

A CRL is a negative statement. It is the digital equivalent of the little paper books of bad checks or bad credit cards that were distributed to cashiers in the 1970's and before. These have been replaced in the retail world by positive statements -- on-line validation of a single check, ATM card or credit card.

SPKI should support both positive and negative on-line validations.

Any CRL or revalidation instrument must have its own lifetime. A lifetime of 0 is not possible because of communication delays and clock skews, although one can consider an instrument whose lifetime is "one use" and which is delivered only as part of a challenge/response protocol.

Implementation of Certificates

The authorization certificates that are envisioned for SPKI (and needed to meet the demands of the list given at the end of this document) should be generated by any keyholder empowered to grant or delegate the authorization in question. The code to generate certificates should be written by many different developers, frequently persons acting alone, operating out of garages or dorm rooms. This leads to a number of constraints on the structure and encoding of certificates. In addition, SPKI certificates should be usable in very constrained environments, such as smart cards or small embedded systems. The code to process them and the memory to store them should both be as small as possible.

An SPKI certificate should be as simple as possible. There should be a bare minimum of fields necessary to get the job done and there should be an absolute minimum of optional fields. In particular, the structure should be specific enough that the creator of a certificate is constrained by the structure definition, not by complaints (or error messages) from the reader of a certificate.

An SPKI certificate should be described in as simple a method as possible, relating directly to the kind of structures a C or PASCAL programmer would normally write.

No library code should be required for the packing or parsing of SPKI certificates. In particular, ASN.1 is not to be used.

A certificate should be signed exactly as it is transmitted. There should be no reformatting called for in the process of checking a certificate's signature (although one might canonicalize white space during certificate input, for example, if the format is text).

For efficiency, if possible, an SPKI certificate should be encoded in an LR(0) grammar. That is, neither packing nor parsing of the structure should require a scan of the data. Data should be read into the kind of structure a programmer would want to use without touching the incoming bytes more than once.

For efficiency, if possible, an SPKI certificate should be packed and parsed without any recursion.

List of Certificate Uses

The list below is a brainstorming list, accumulated on the SPKI mailing list, of uses of such certificates.

- I need a certificate to give me permission to write electronic checks.

- My bank would need a certificate, proving to others that it is a bank capable of cashing electronic checks and permitted to give permission to people to write electronic checks.

- My bank would issue a certificate signing the key of a master bank certifier -- perhaps NACHA -- so that I could follow a certificate chain from a key I know (my bank's) to the key of any other bank in the US and, similarly, to any other bank in the world.

- I might generate a certificate (a ``reputation voucher'') for a friend to introduce him to another friend -- in which certificate I could testify to my friend's political opinion (e.g., libertarian cypherpunk) or physical characteristics or anything else of interest.

- I might have a certificate giving my security clearance, signed by a governmental issuing authority.

- I want a certificate for some software I have downloaded and am considering running on my computer -- to make sure it hasn't changed and that some reputable company or person stands behind it.

- I need certificates to bind names to public keys:

    - [traditional certificate] binding a key to a name, implying "all the attributes of the real person having this name are transferred to this key by this certificate". This requires unique identification of a person (which is difficult in non-digital space, as it is) and someone trustworthy binding that unique name to the key in question. In this model, a key starts out naked and acquires attributes, permissions and authority from the person bound to it.

    - [direct certificate] binding a name to a key, implying "I (the person who is able to use the associated private key to make this signature) declare that I go by the name of XXXXXXX." The unique identification of the key is

automatic -- from the key itself or a cryptographic hash of the key. The binding is done by the key itself -- in a self-signed certificate. In this model, a key is loaded with attributes, permissions and authority directly by other certificates, not indirectly through some person's name, and this certificate declares only a name or nickname by which the key's owner likes to be addressed.

- [personal binding] binding a key to a nickname. This kind of certificate is signed by me, singing someone else's key and binding it to a nickname by which I know that person. It is for my use only -- never given out -- and is a signed certificate to prevent tampering with my own private directory of keys. It says nothing about how I certified the binding to my own satisfaction between the key and my friend.

- I might be doing genealogy and be collecting what amounts to 3x5 cards with facts to be linked together. Some of these links would be from one content to another reference [e.g., indexing and cross-referencing]. Others might be links to the researcher who collected the fact. By rights, the fact should be signed by that researcher. Viewing only the signature on the fact and the link to the researcher, this electronic 3x5 card becomes a certificate.

- I want to sign a contract to buy a house. What kind of certificate do I need?

- I have found someone on the net and she sounds really nice. Things are leading up to cybersex. How do I make sure she's not really some 80-year-old man in a nursing home?

- I have met someone on the net and would like a picture of her and her height, weight and other measurements from a trustworthy source.

- Can I have a digital marriage license?

- Can I have a digital divorce decree?

- ..a digital Voter Registration Card?

- There are a number of cards one carries in a typical wallet which could become certificates attached to a public key:

- health insurance card

- prescription drug card

- driver's license (for permission to drive)

- driver's license (for permission to buy alcohol)

- supermarket discount card

- supermarket check-cashing card [I know -- anachronism]

- Blockbuster Video rental card

- ATM card

- Credit card

- membership card in the ACLU, NRA, Republican party, Operation Rescue, NARAL, ACM, IEEE, ICAR....

- Red Cross blood donor card

- Starbuck's Coffee buy-10-get-1-free card

- DC Metro fare card

- Phone calling card

- Alumni Association card

- REI Membership card

- Car insurance card

- claim check for a suitcase

- claim check for a pawned radio

- authorization for followup visits to a doctor, after surgery

- Better Business Bureau [BBB] style reputation certificates [testimonies from satisfied customers]

- BBB-style certificate that no complaints exist against a business or doctor or dentist, etc.

- LDS Temple Recommend

- Stock certificate

- Stock option

- Car title

- deed to land

- proof of ownership of electronic equipment with an ID number

- time card certificate [activating a digital time clock]

- proof of degree earned [PhD, LLD, MD, ...]

- permission to write digitally signed prescriptions for drugs

- permission to spend up to $X of a company's money

- permission to issue nuclear launch codes

- I'm a sysadmin, I want to carry a certificate, signed by SAGE, that says I'm good at the things sysadmins are good at.

- I'm that same sysadmin, I want an ephemeral certificate that grants me root access to certain systems for the day, or the week, or...

  Certain applications *will* want some form of auditing, but the audit identity should be in the domain of the particular application... For instance an "is a system administrator of this host" certificate would probably want to include an audit identity, so you can figure out which of your multiple admins screwed something up.

- I'm an amateur radio operator. I want a signed certificate that says I'm allowed to engage in amateur radio, issued by the DOC. [I currently have a paper version of one]. This would be useful in enforcing access policies to the amateur spectrum; and in tracking abuse of that same spectrum. Heck! extend this concept to all licensed spectrum users.

- I'm the a purchasing agent for a large corporation. I want to posses a certificate that tells our suppliers that I'm authorized to make purchases up to $15,000. I don't want the suppliers to know my name, lest their sales people bug me too much. I don't want to have to share a single "Megacorp Purchasing Department Certificate" with others doing the same job [the private key would need to be shared--yuck!].

- "This signed-key should be considered equivalent to the certifying-key until this certificate expires for the following purposes ..."
  [This is desirable when you wish to reduce the exposure of long-term keys. One way to do this is to use smart cards, but those typically have slow processors and are connected through low-bandwidth links; however, if you only use the

smart card at "login" time to certify a short-term key pair, you get high performance and low exposure of the long term key.

I'll note here that this flies in the face of attempts to prevent delegation of certain rights.. Maybe we need a "delegation-allowed" bit -- but there's nothing to stop someone who wishes to delegate against the rules from also loaning out their private key..].

- "I am the current legitimate owner of a particular chunk of Internet address space."
  [I'd like to see IPSEC eventually become usable, at least for privacy, without need for prior arrangement between sites, but I think there's a need for a "I own this address"/"I own this address range" certificate in order for IPSEC to coexist with existing ip-address-based firewalls]

- "I am the current legitimate owner of a this DNS name or subtree."

- "I am the legitimate receiver of mail sent to this rfc822 email address. [this might need to be signed by a key which itself had been certified by the appropriate "DNS name owner" certificate]."
  [This is in case I know someone owns a particular e-mail address but I don't know their key.]

- Encryption keys for E-mail and file encryption

- Authentication of people or other entities

- Digital signatures (unforgeability)

- Timestamping / notary services

- Host authentication

- Service authentication

Other requirements:

- Trust model must be a web (people want to choose whom they trust). People must be able to choose whom they trust or consider reliable roots (maybe with varying reliabilities).

- Some applications (e.g., notary services) require highly trusted keys; generation complexity is not an issue here

- Some applications (e.g., host authentication) require

extremely light (or no) bureaucracy. Even communication with the central administrator may be a problem.

- Especially in lower-end applications (e.g. host authentication) the people generating the keys (e.g., administrators) will change, and you will no longer want them to be able to certify. On the other hand, you will usually also not want all keys they have generated to expire. This may imply a "certification right expiration" certificate requirement, probably to be implemented together with notary services.

- Keys will need to be cached locally to avoid long delays fetching frequently used keys. Cf. current name servers. The key infrastructure may in future get used almost as often as the name server. The caching and performance requirements are similar.

- Reliable distribution of key revocations and other certificates (e.g., the ceasing of the right to make new certificates). May involve goals like "will have spread everywhere in 24 hours" or something like that. This interacts with caching.

Open Questions

Given such certificates, there remain some questions, most to do with proofs of the opposite of what a certificate is designed to do. These do not have answers provided by certificate definition or issuing alone.

- Someone digitally signs a threatening e-mail message with my private key and sends it to president@whitehouse.gov. How do I prove that I didn't compose and send the message? What kind of certificate characteristic might help me in this?

  This is an issue of (non-)repudiation and therefore a matter of private key protection. Although this is of interest to the user of certificates, certificate format, contents or issuing machinery can not ensure the protection of a user's private key or prove whether or not a private key has been stolen or misused.

- Can certificates help do a title scan for purchase of a house?

  Certificates might be employed to carry information in a

tamper-proof way, but building the database necessary to record all house titles and all liens is a project not related to certificate structure.

- Can a certificate be issued to guarantee that I am not already married, so that I can then get a digital marriage license?

  The absence of attributes can be determined only if all relevant records are digitized and all parties have inescapable IDs. The former is not likely to happen in our lifetimes and the latter receives political resistance.

  A certificate can communicate the 'positive' attribute "not already married" or "not registered as a voter in any other district". That assumes that some organization is capable of determining that fact for a given keyholder. The method of determining such a negative fact is not part of the certificate definition.

- The assumption in most certificates is that the proper user will protect his private key very well, to prevent anyone else from accessing his funds. However, in some cases the certificate itself might have monetary value [permission to prescribe drugs, permission to buy alcohol, ...]. What is to prevent the holder of such a certificate from loaning out his private key?

  This is a potential flaw in any system providing authorization and an interesting topic for study. What prevents a doctor or dentist from selling prescriptions for controlled substances to drug abusers?

References

[DH] Diffie and Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory IT-22, 6 (Nov. 1976), 644-654

[KOHN] Loren Kohnfelder, "Towards a Practical Public-key Cryptosystem", Bachelor's thesis, MIT, May, 1978

Author's Address

Carl M. Ellison
Intel Corporation
2111 NE 25th Ave M/S JF3-373
Hillsboro OR 97124-5961 USA

Telephone: +1-503-264-2900
Fax: +1-503-264-6225
EMail: carl.m.ellison@intel.com (work, Outlook)
cme@jf.intel.com (work, Eudora)
cme@alum.mit.edu, cme@acm.org (home, Eudora)
Web: http://www.pobox.com/~cme

Expiration and File Name

This draft expires 3 December 1999.

Its file name is draft-ietf-spki-cert-req-03.txt