Naming and Certificates

Carl M. Ellison †

Introduction

> *"Without a [PKI] of trusted certificate authorities, users cannot know with whom they are dealing on the network...."*[1]
>
> *"How does Alice know she has my public key and not one from someone impersonating me? The answer is, she doesn't, unless I either physically deliver it in a secure fashion or she relies on someone else to authenticate me by digitally signing my public key. This third party is known as a Certificate Authority (CA)."*[2]

These beliefs have been published and discussed for so many years that people believe they are established truth. They presume that a CA is required to tell the world who you are. In fact, however, those beliefs are false and possibly dangerous. They are also extremely difficult to correct. They are based on human practices that date back to the invention of language and practices like those take centuries to evolve. The Internet is not allowing humans those centuries.

The major flaw is the belief that names are valid identifiers, without qualification. In our daily lives, in small communities, they are usually good identifiers. That's their purpose in life. However, they don't scale. The concept might, but the names we use don't.

PKI designers fall back on the theoretical potential of names to scale, when discussing ID certificates that would work over the whole world, while PKIs are deployed to people who use the names they were given at birth.

This situation is also dangerous from the point of view of privacy. Using a global name space to identify every important message traversing the Internet is a gold mine for intelligence activities, be they governmental, commercial or criminal. That situation turns out to be even worse because a public key is itself an identifier.

There are solutions to this problem that improve security for the end user and simultaneously improve the user's privacy, but those solutions have a deployment problem. There is no business model behind them while there appear to be many opportunities to make money under the established beliefs.

The Naming Problem

This problem dates back to the invention of language, among our prehistoric ancestors. Language needs names for things. We call those *nouns*. A proper noun – in particular a name for a person – is an identifier. We speak that name, when we want to communicate to a second person about a third. The entropy in that name depends on the size of the community. That is, if there were only two people in the world, the only name we would need would be "you" – for zero bits of entropy. If there were three people, say all women to simplify matters, we would need only two names: "you" and "she" – for one bit of entropy.

As we move beyond 3 people, we need to assign names just to talk about the situation. Let's call three people Alice, Bob and Carol. When Alice uses the name "Carol" in speaking to Bob, she intends for him to open up some memory location in his brain indexed by the name "Carol". If Bob knows Carol already, then Alice expects him to access the memories pertaining to the same person she means when she accesses her memories under the name "Carol". If he doesn't know Carol yet, she expects him to open up a fresh memory file and label it "Carol". This is standard human practice.

If names were assigned by some central authority, names would need $\log_2(N)$ bits of entropy for a community of N people. However, names are assigned by individual parents. If they were chosen at random, there would need to be $[2\log_2(N)+(K-1)]$ bits of entropy per name to have probability (2^{-K}) of two people having the same name. So, if names were assigned to the world's 6 billion people, you would need 32.5 bits of entropy in assigned names – e.g., decimal numbers of 10 digits. If you wanted to randomly generate names rather than use a central numbering authority, and wanted the chance of name collision to be one in a thousand, names would need 74 bits. If one used English spelling at 1.5 bits of entropy per character, that calls for names of 50 characters in length. Of course, parents don't choose names randomly. They borrow parts or even whole names from others in the family. They are also subject to fads. That is, name choices are not independent. Those entropy reductions call for even longer names[3] in compensation.

There is anecdotal evidence that as communities got larger, through improvements in transportation and communication, names got more complex – from "John", to "John, the Smith", to "John Andrew Smith". Apparently in Korea, where there are very few family names, it is customary for parents to "get poetic" in choosing individual names. In other words, if the entropy of one name is low, increase the entropy of the other name. It would be fascinating to compare the size of a community at a given time in history to the amount of entropy in names used in that community.

The habits for choosing names evolve slowly. Changes in those habits appear to require centuries. On the other hand, community size went through an explosion in the 1990's with the popularization of the Internet. There is more explosion to come. The vast majority of the world is not yet on the net. There are an estimated 200 million users on the Internet today, giving the net room for a factor of 30 in growth. By contrast, it appears that human name pairs carry about 20 bits of entropy. That is, they don't have enough bits to identify even the fraction of the world that is already on the Internet.

So, we find ourselves using names that have inadequate entropy for the global community size and a belief system, learned as infants, that tells us that names are valid identifiers – a falsehood.

Diffie and Hellman fell into this trap in their 1976 paper, "New Directions in Cryptography". They suggested the notion of a modified telephone directory as the solution to the key management problem, now that the world has public key cryptography. In 1978, Loren Kohnfelder extended that notion by defining the first digital **certificate**. Both of these concepts bound names to public keys.[4]

In the late 1980's, the X.500 effort set out to name the entire world – all of its people and all objects with which one might want to communicate. The Internet, at about the same time, was naming

the whole world with e-mail names and soon thereafter with URLs. All of these names are unique by assignment, with a hierarchy of naming authorities. Their uniqueness is derived from the fact that there is a single root for the hierarchy. The X.500 single root never came to pass. The Internet single root did.

That didn't stop the X.500 community and its spin-off, X.509. Those efforts have set up local name spaces, with uniqueness only within the local name space. In the X.509 case people are using those names globally even though the names they use are valid only under one of many roots. As a result there are parts of the X.509 design that don't work and have to be kludged around.[5]

So, some of us have URLs; most people reading this paper have e-mail names under the Internet Domain Name System (DNS); a few people have X.509 certificates (e.g., for S/MIME).

X.509 Names

One feature of X.500/X.400/X.509 *Distinguished Names* (DN) is that they include a field called the Common Name. This is the name people use in daily life. This is a normal name like "Carl Ellison".

If there were a single naming root, as envisioned by X.500, the DN would be a globally unique name and suitable for global use. This does not mean that the common name is a valid identifier.

There are tools that use this DN, even though there is no single naming root. Perhaps the most popular of these is Microsoft Outlook/Exchange, using them for e-mail.[6] Outlook, however, is designed to be friendly to the user – to present a pleasing and functional user interface – and as a result what is displayed to the user is only the Common Name portion of the DN – the user-friendly portion. As a result, any decision made by a user viewing the provided interface will be made based on a name that has inadequate entropy for the Internet community. These applications will work well in small deployments (e.g., companies under 1000 in size) but show errors due to name confusion as the deployments get larger. These errors can be humorous or embarrassing, but as we start to make security decisions based on these names, they can have very serious consequences. A digitally signed message, displayed in Outlook, still shows only the common name of the signer. Even if it showed the whole DN, the user is likely to look only at the common name. If the message requests the recipient to perform some action (e.g., transmit secret information, sell assets, launch an attack[7]), the consequences of name confusion can be very serious.

So, why would we use X.509 names? X.509 is a global standard. There are people who demand standards-based designs. The first products using certificates were designed to use X.509 and those are deployed. There are companies basing their business on the X.509 model and those companies have sales forces, busy gaining mind share in the world.

Public Key as Name

For those using public key cryptography the solution is remarkably simple. As a side effect of having a key pair, an individual has a public key. That is a functional quantity. One can use it to verify signatures or to encrypt messages. However, it is also a byte string. It is globally unique. It need not be kept secret. It is generated at random and even if you generated 2^{42} keys (just under a thousand keys for each human on Earth) and used very small (insecure) 512 bit RSA keys, the probability that any two of these keys would be the same is roughly 2^{-429}. If you want to save space and use a 160-bit hash of a key instead of the key as the names of the 2^{42} keys, the probability that any two of them would have the same hash is roughly 2^{-77} or one part in 151 thousand million million million. Therefore, for all practical purposes, even the relatively short hash is globally unique while the full key certainly is. The key is bound to a particular person, ***the keyholder***, because it is bound math-

ematically to the corresponding private key and the private key is bound to the keyholder by security safeguards (or else the entire notion of public key cryptography has fallen apart[8]). Therefore, the public key is one global name for the keyholder.[9] As a name, it has a strong advantage – it doesn't need a certificate in order to bind the key to the keyholder's name[10].

SDSI Names[11]

Ron Rivest and Butler Lampson addressed the naming problem in their Simple Distributed Security Infrastructure (SDSI). They abandoned the notion of a global name space and replaced it by defining a local name space for each participant, defined by that participant and naming people of concern to that participant. They then defined the machinery for linking those name spaces together. The result is that one can use names (which humans demand to do anyway), and do so securely. Their work built upon the notion that the primary identifier of a keyholder is the public key. The name is an aid for the human user.

Privacy Implications

Use of a public key as a name seems at first blush to provide anonymity, but that is a false impression. A name is still a name, even if it is not pronounceable by humans.

The task of the privacy violator is to associate data together. We usually think of this as associating observed facts with a person (with the name of that person), but any association of facts known to apply to the same person starts to build a dossier and naming that dossier can come much later in the process.

If we use a totally anonymous public key as an ID, but use it in everything we do, then we make the job of the privacy violator much easier. That attacker can now link together actions and words from the same keyholder with a degree of certainty that even non-anonymous e-mail doesn't provide.

One solution to this problem might be to hide the signature within an encrypted block, as is done in PGP. However, that solution is not adequate. A major danger of privacy violation is among colluding signature verifiers[12] and all of those need to see inside any encrypted block.

The real solution is for the user to generate multiple key pairs and use them for carefully walled-off purposes. That way the eavesdropping attacker would not be handed the association between those different actions for free. He would have to do work to establish it. The trouble is, the user would also have to do work.

Likely Scenario

With enough effort and with the proper use of public key technology, the user is in a position to take charge of her own privacy. However, tool development will probably be driven at least in part by the marketing and standards efforts of those who have a profit motive – the commercial CAs. Those products and standards will likely mandate use of the output of those CAs – an ID certificate. Those certificates might start costing money, so that users are discouraged financially from having more than one. Meanwhile, most users are not inclined even to ask for anonymous credentials. Most users claim not to consider their own privacy to be very important. On top of that, we're faced with an observation by one veteran of the security business: "the potential user community for a security product is cut in half with each key stroke or button push required to use the product". That is, security is seen as a nuisance rather than a benefit and users almost always want to just swat it away, or turn it off completely.

Ray of Hope

There is, however, a ray of hope. People really do need to know whom they're dealing with on the network, but it's not names they need. They need to know reputations. E-bay has demonstrated this very effectively. People report trusting a given seller on e-bay based on that seller's customer feedback. Such reputation credentials need not be associated with true names or anything else that could be used to identify the seller, except of course his or her previous selling history. In a sense, that is the price for getting on-line trust – not registration with some CA but registration with some auction house that tracks results of sales. This form of reputation service will doubtless spread to other fields of human activity, and in that there is hope, provided it is handled with privacy designed in from the start.

† cme@jf.intel.com or cme@acm.org

[1] McConnell and Appel, "Enabling Privacy, Commerce, Security and Public Safety in the Global Information Infrastructure", report of the Interagency Working Group on Cryptography Policy, May 12, 1996; [quote from paragraph 5 of the Introduction]

[2] PC Magazine, January 18, 2000, p. 140, col. 1, bottom

[3] First and last name, the parts that other people are likely to remember. Names should also be lengthened to compensate for homonym mapping, since people often learn names orally. They should also be lengthened for nickname mapping. That is, John = Jon = Jonathan or Betty = Elizabeth.

[4] See RFC2693: SPKI Certificate Theory, ftp://ftp.isi.edu/in-notes/rfc2693.txt, section 2 for a more detailed examination of this history.

[5] Cross-certification is such an effort to modify the design to respond to this conflict.

[6] Any application using X.509 certificates handles a DN by definition, but the most popular of these (the web browsers) do not use the DN at all in a security decision and present parts of the DN to the user only if the user specifically requests it for a given session. As a result, the actual display of even portions of the DN is extremely infrequent in such applications.

[7] The PKI being set up for the US Department of Defense is expected to have 5 million certified keyholders, all carrying X.509 ID certificates.

[8] ibid. We observe there that current computer system architecture is inadequate to bind the private key to the keyholder and therefore the notion of "non-repudiation", central to so many digital signature laws, is not viable. This does not mean that public key cryptography has no use. It means that if one is to use it for a high security application, one needs to take special precautions with the entire computer – restricting the code loaded onto it and the people allowed to get within reach of it, physically or electronically.

[9] See RFC2693 for more details about how to use keys as the only names for keyholders.

[10] That is, to bind it to itself.

[11] See http://theory.lcs.mit.edu/~cis/sdsi.html

[12] For example, merchant sites can be infiltrated by network attack or corruption of an employee. According to credit card companies, merchants are the main source of theft or misuse of consumers' credit cards. Our experience on the Internet shows the same. Hackers steal credit card numbers by breaking into merchant computers. This is far more productive than stealing a number one at a time from individual sessions.