PassLok and Public Key Infrastructure

PassLok, our tiny powerful privacy app, is revolutionary in more ways than its size. I'm going to tell you here how PassLok turns the whole paradigm of Public Key Infrastructure on its head. Because **Less is More** when it comes to privacy.

With due respect to Pretty Good Privacy (PGP) and other venerable public-key systems that have served the public well for decades, one aspect that has never worked well is the whole issue of public key authentication. PGP advanced the concept of "Web of Trust" where users sign each other's keys in order to certify their authenticity (that is, that the public key named "Bob's key" actually belongs to Bob, and not to Mallory, the malicious interloper). Not long afterward, the X.509 standard set up the concept of Certification Authority (CA), which is implicitly trusted by everyone and which issues authenticity certificates for individual keys, very much like the Web of Trust did.

All good, and it has helped immensely in setting up secure protocols between computers, such as SSL and its successor TLS, which is more and more an important part of the Internet (remember all those little locks on the address bar?). But by and large the scheme has not succeeded in securing communications between individuals. Sure, you can make a public key with PGP or similar software and you can post it in a keyserver website. You can print your public key and take it to a key-signing party along with two forms of government-issued identification, and have everyone in the party sign your key electronically. But 99% of emails are still sent unencrypted. Why?

In another article, I submit that the reasons for this failure are inherent in the mathematical structure of PGP and its derivatives. Because PGP was originally based on the RSA algorithm, which cannot produce arbitrarily chosen secret keys, it was necessary to use random keys which, given their length, were impossible to memorize by humans. This required the existence of "keyring" files that had to be kept secure, which worked against portability and required yet another piece of software to manage them. In the end, even users that were otherwise experienced found it very difficult to keep things straight. Using language with no direct analog outside of computer cryptography, such as "encrypting with a public key and decrypting with a private key" did not help matters.

But this article is about the Public Key Infrastructure, that is, the set of Webs of Trust, Certification Authorities, and whatnot that are supposed to assure users that their precious confidential information is not being phished out (to use the current term) by phony public keys. Let me start with a rather bold statement: for most practical uses, it is better to have **no special infrastructure at all**, so long as the keys are manageable.

The model I propose is that of phone numbers. Even adding three extra digits for "area code", which no longer is tied to a particular geographical location, in many cases, people

exchange those numbers without regard for authentication. They post it in their business cards, their stationery, their websites. No outside certification is given, and none is expected. Sure, there is an official phonebook issued by the phone carrier, but most useful numbers are not listed in order to deter marketers. The only phonebook that I have used in the last decade has been the yellow pages, which is largely a commercial listing that has been replicated manifold online.

Authentication of phone numbers works like this: If I got the number from a person whom I presume has authenticated the number, I take it as authentic and store it in my directory. If I got it from a website or a listing, I presume that the owner is watching so the posted number is good. If people start getting wrong numbers from those listings, they'll tell the owner right away so it is corrected. Same thing if I got a friend's number from a mutual friend.

Authentication of public keys does not have to be very different. Like a phone number, the main purpose of a public key is to establish a channel of private communication between two parties where none existed before. I take the recipient's public key and use it to encrypt a message. If the key was wrong, the recipient won't be able to read the message, and he/she will tell me that I've got a wrong or old key that he/she doesn't use anymore. If the message goes through and is successfully decrypted, I will soon get confirmation of the fact, which will raise my level of trust on the key I used. Like with a phone number, I don't need any authority to certify that the key is working.

Of course, Mallory could still be intercepting my emails, pretending to be Bob, but he could do the same with my phone calls if I'm not familiar with Bob's voice. And yet, I don't lie awake at night wondering if I really talked with the people I thought I was talking with on the phone. Could the problem be that we are still looking at encryption as some sort of cloak and dagger scheme for secret agents? Why can't we tone down a bit and live with email security as we live with private phone calls?

And yes, your knowledge of Bob's voice can also authenticate his public key. Just ask him to read aloud over the phone the fingerprint (or digest, or ID, or hash, or whatever is called in that encryption system) of his public key, and you'll know whether or not what you've got in your possession has been tampered with.

One of the key differences between PassLok and its parent URSA is that URSA has a (rudimentary) PKI by means of its official keyserver website ursakeys.com, and PassLok has none. That doesn't mean that people could not set up Internet-accessible listings of PassLok locks (that's the current name of a public key in PassLok), but why should they? The locks are short enough that I can email or even text them to a friend without any trouble. If my phone is out of coverage, the last version of PassLok will display my lock as a QR code, which others can scan. If I put that QR code on my business card, then everyone

who gets my card will get my Lock. I can append my Lock to my email signature, and it won't make it much longer. The only difference with a phone number is that I can, possibly, try to remember a phone number. Not that I've tried recently, though.

Anybody can keep a personal list of Locks, using the extra fields of the Contacts or phone listings app that they are already using. Starting with version 1.6, PassLok can keep its own database, which can be easily moved between devices. What's the need for a centralized key server? What's the need for an authority?

PGP could have gone this way, too, but along the way it got too complicated and its keys got too bloated (maybe because of all those certifications stuck to them?) to be able to do these things. When RSA set up Verisign as the first CA, a commercial interest began to oppose moving away from this model. But PassLok has a chance. You can get the whole thing at passlok.com. You can save it to your device, and then you won't have to download it anymore. Take it for a spin.