From centralized certification systems to more internet-like ones

Alexandre Kervadec

Abstract—**This paper is a synthesis of a E.Gerck's paper :** *"Overview of Certification systems: X.509, CA, PGP and SKIP"***[1], but not only. First, it gives an overview of (the mains) certification systems which are X.509 and CAs, PGP, SKIP, DANE and the Certificate transparency by Google, with thoughts on pros and cons of each system, on a technical point of view and about the government's stranglehold on data exchanges.**

I. OVERVIEW OF CERTIFICATION SYSTEMS

First sections talk about existing (sections I-A and I-B) or extinguished (section I-C) certification systems. Then, further sections (I-E and I-G) deals with new, not implemented yet, technologies of digital certification which may solve current issues.

A. X.509 and CAs

X.509 and CAs[10] infrastructure is based on a directory method.
With this kind of certification system, there are three different entities, which are :

1) **CA : Certification Authority**, an entity that controls authentication services and management of digital certificates. It can be public (like banks with their clients), commercial (such as *Verisign* which sells its services) or private (like a compagny department, for an internal purposes).
2) **Subscriber :** the entity which sends informations to the CA to add it to his certificate. This entity is one that need to be trusted by the *user* entity.
3) **User :** ask informations about *subscribers* to CA(s), it's central to the process, since the user party is relying on the informations given by *CAs* and is thus at risk.

The way that certification system is the following:

- A subscriber gives informations to the CA, in order to create a digital certificate for itself in the CA directory.
- The CA receive those informations and put it in a certificate.
- A user (the public in general) receive a certificate from a subscriber. In order to authenticate this subscriber, the client ask to the CA if the transmitted certificate matches with the subscriber's identity.
- The CA answer to the user to give him the precious informations about the subscriber's identity.

There is a not outwardly perceived entity in this system, it is the Naming Authority (DN). It provides to the CA a unique distinguished name (DN) which matches to the subscriber
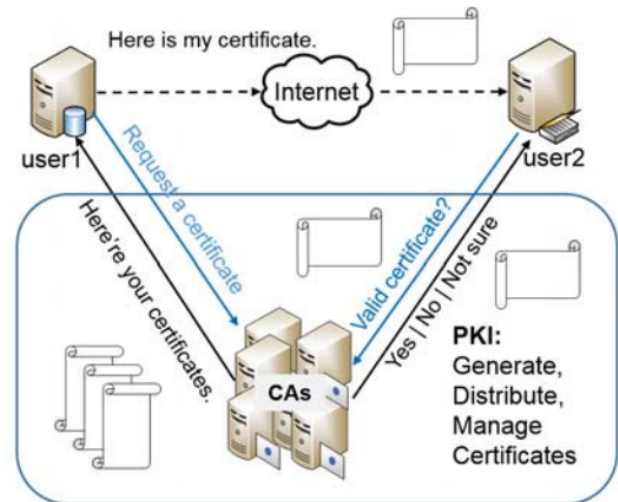
Figure 1. Diagram of the X.509 and CAs process (from [15])

identity in the certificate.
Note that the CA can double in NA, but they provide two different services: the CA certificate refers to a name but does not denote it, the NA does.

The main concerns about authentification services provided by CAs are :

- The content of a certificate needs to be discussed, as well as certificate revocation. For example, a subscriber can generate multiple DN for a same CA or the same DN to multiple CAs, so maybe, a better subscriber identificator need to be added.
- Are the validation procedures for the certified data that is included in a certificate solid enough? Because each CA has its own self-defined rules (Certification Practice Statement[1]), which can by completly different from one CA to another.

Going deeper with user validation : DN scheme based on X.500 Recomendation, but it is not completly defined, and will (in 1998) probably not be. Also, X509 certification depends on many others such as ISO, ANSI, ITU and IETF. Thus lead to a lack of harmonization. Plus, there is a big problem with CPS (Certification Practice Statements), that also can be seen such as flexibility (for pros), because each CA answer specific needs, so no harmonization again.

---

[1]CPS

Also, regarrding to the law, CAs deny all leak of information from themselves.

Some kind of conclusion about harmonization (lack of), in a world wide vision, if space there is.

B. PGP (Pretty Good Privacy)

PGP, created thanks to Phil Zimmermann researches. It has two parts: certification and encryption, following text only deals with certification.

Compared to X.509, PGP is more *internet-like*, this because of its *introducer-model* base.

PGP depends on a chain of authentificators: the users themselves. This chain forms a ring, but not in a closed way but in a mathematical way like a list or a *web-of-trust*. You may not know everyone in this ring, but you can assume that you will know somebody who know this user. You can also assume that different rings can have some contact points to garantee the referrals.
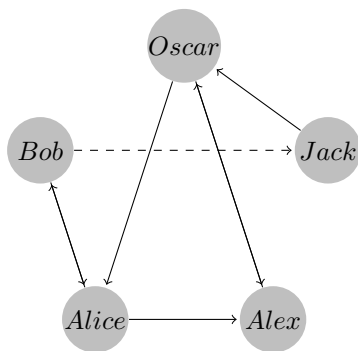

Figure 2. Communities of trust: Jack introduces Oscar to Bob's public-key certificate before Bob receives it.

Thus, we can figure out that PGP breaks the traditional hierarchical trust architecture with this *web-of-trust* approach.

Let's get an example[2]: Bob want to exchange datas with Oscar, which he didn't knew before. But Jack, a colleague of Bob's, signs Oscar's public-key certificate whch he knows is authentic.

Oscar then forwards his signed certificate to Bob who wishes to communicate with Oscar privately.

Bob, who knows and truts Jack as an *introducer*, find out, after verification, that Jack is among Oscar's certificate signer. Therefor, Bob can be confident that Oscar's public key is authentic.

However, had Bob not known or trusted Oscar's signers, including Jack, he would have been skeptical about the authenticity of Oscar's public-key. Oscar would have to find another introducer whom Bob trusts to sign Oscar's public-key certificate. This is illustrated in Figure 2.

This is illustrated in Figure 1, where the dotted arrow indicates "Bob trusts Jack as introducer" and the solid arrow indicates "Jack trusts Oscar's public key validity".

[2]Example from "The PGP trust model"[4]

PGP also includes a public-key certificate and an introducer trustwhorthiness. They have different levels, for a public-key certificate they are:

- *Undefined:* we cannot say wether this public-key is valid or not.
- *Marginal:* this public-key *may* be valid but we cannot be sure.
- *Complete:* we can be fully confident that this public-key is valid.

For an introducer, those levels of trustworthiness are the following:

- *Full:* this public-key is fully trusted to introduce another public-key.
- *Marginal:* this public-key can be trusted to introduce another public-key, but, it is uncertain whether it is fully competent to do that.
- *Untrustworthy:* this public-key should not be trustedto introduce another, therefore any occurence of this key as a signature on another public-key should be ignored.

Only the trustworthiness of public-key's validity is automatically evaluated by PGP. Introducer's worthiness is manually assigned by each user of to the public key, and exists only within each individual user's public-key ring[3].

The big point is the use of such a tool in a world wide range: that is raisonnably not possible. We can assume to use it in a little group, where trustworthiness is effective, but in a more wide area, it is problematic to trust such a amount of people, or find trusted introducers. Furthermore, in a commercial use, there is no entity responsible when something goes wrong.

C. SKIP (Simple Key-Management for Internet Protocol)

SKIP implements a linked-chain of two-sided node authenticators, where each node derives its informations from a sort of *directory service*.

The problem is that SKIP happens at a low level, so it is transparent to the user; also assuming that the user has no control over security or other certification choices. So it has to be complemented by a second higher level protocol. Furthermore, SKIP does not support IP translation (such as NAT behind a firewall), that makes it quite useless.

So that protocol imediatly looks unapropriate for our study. In other way, SKIP does not exist anymore, so it is pointless to go further in it study.

D. Thoughts about X.509, CAs and PGP

To be used in a commercial way (for responsabilities purpose), X.509, CAs and PGP need a centralized certification control system. However, there is a struggle between this need of centralization and the internet architecture (that is decentralized). This show us the uselessness of centralized governement controls, since we have no central government on the internet which give a world governance.

So the question emerging in there is: how can we (and do we need to) have a main control on an internet environement, to

[3]PGP allows users to have files representing multipke *key rings* to store public or secret keys[4].

assure the goodness of datas and legitimacy of interlocutors? Some answers have been made since then (thus since the publication of the original article), answers that we are going to explain further.

E. DANE (DNS-Based Authentication of Named Entities)

DANE is not implemented yet, but a IETF team is working on the standard.

It has begin with a simple assert: why use a new trust infrastructure if we already have to use DNS to resolve a domain name?

From here, DNS must be securised, knowing that it is easily to set a MITM with DNS cache poisoning [6] using the Kaminsky attack [5]. That is the main purpose of DNSSEC: secure DNS. DANE allow to store keys in DNS. Thereby, `www.example.org` would be able to secure `https://www.example.org` without using a suspicious third part, just by putting its keys in the domain name that it controls. By its arborescent nature, `example.org` shall not be able to cheat on `example.com`.

Lets give us an example:

From the classic schem, we have *Alice*, the subscriber (`alice.example.com`), *Bob* its client, and *Charlie*, the serious AC. There are three different scenarios:

- *Type 0:* Alice has got a certificate delivered by Charlie and is really satisfied by it. But Alice is afraid that *Oscar* (a bad AC) forge a false certificate for its website. So Alice uses DANE to tell in DNS that Charlie and only Charlie is its AC.
- *Type 1:* Alice has got a certificate delivered by Charlie, but fear that Charlie give its certificate to bad people. So Alice uses DANE to publish in DNS the origin of its certificate.
- *Type 2:* that is the *max scenario* in DANE. Alice doesn't trust ACs anymore, so it decides to emite it via DANE on DNS, that gives the entire responsability on DNS servers.

This type is in a new register called TLSA[4] in DNS. Her is an example of a TLSA register:

```
_443._tcp.www.example.com. IN TLSA (
        0 0 1 d2abde240d7cd3ee6b4b28c54df0&34b9
        7983a1d16e8a410e4561cb106618e971 )
```

With:

- `443` : the port used.
- `tcp` : the protocole used.
- `www.example.com` the domain name.
- `0 x x` : the type used, here it is a type 0 (`1 x x` for type 1 and `2 x x` for type 2).
- The rest is the certificate itself.

As we said before, DNS is not secured, so DNSEC has been deployed. DNSSEC is a protocol that allow to the DNS server (master) ti encrypt and sign its answer with a public key cryptographic system. It also permit to return a `NXDOMAIN`
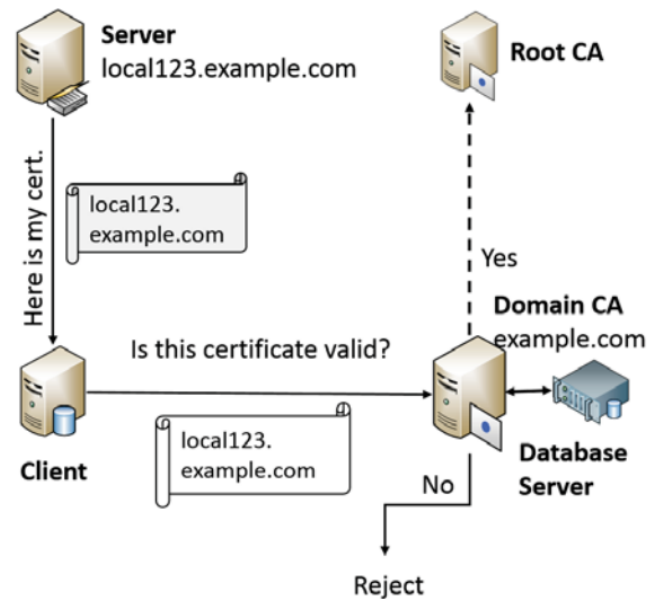

Figure 3.   Process of certificate verification (from [15])

type ("No Such Domain").

Nervertheless DNSSEC has some vulnaribilities such as:

- Replay attacks
- Attacks against intermediaries: because of the hierarchic and centralized approch of DNS (there is only one root), the responsability of intermediaries is really high and a remote control by a thief should be bad (the thief controls the domain thus). History has proved that some of those intermediaries are not that trustworthy [11] [12].

To make a conclusion about DANE, we have multiple points to point out:

- Introducing a new register TLSA shall be very complicated, knowing that things are not moving that fast in DNS[5].
- The validation necessity end to end is quite disturbing knowing that there is no standart and deployable solution for wide use about the last kilometer securisation.
- DNSSEC is not possible behind a DNS proxy (which filter/break DNS answers to the point of creating a DoS for all TLS services.

Therefor, DANE is on a good way, particulary for communications between servers, where there is no (not much) constraints of security are (more) easily achievable.

F. Sovereign keys

Sovereign keys are an Electric Frontier Foundation (EFF) proposition.

A sovereign key is a cryptographic pair of key where the public key is associated to a domain name (or a subdomain) to create a trust anchor[6] for itself.

---

[4]TLSA doesn't stand for anything: it is just the name of the RRtype (Record Resource type).

[5]Like it use to be with EDNS introduction few years ago.

[6]A trust anchor is an authoritative entity for which trust is assumed and not derived [13].

This association *public key/domain name* is registered in a append-only data structure, kept on servers named "*timeline servers*". There is a small number, around 10-20, of these [14]. The private part of the sovereign key (private key) allows to sign certificates of which the CN is the domain (or subdomain) name which it is associate with.

During a handshake with a TLS server, the certificate is sent with its protected domain key signature, by the same mecanism than a TLS transaction based on PKIX. The client has then the possibility to recover the public key associated with that sovereign key from the timeline servers to check the key validity, that the domain is the same that the one in the current transaction and that it signs the receive certificate.

G. CATA: Certificate Authority Transparency and Auditability

This protocol has been developed by a Google team few years ago [8]. It is still in developpment [9] so it is not implemented and deployed yet. It also has to be improve, like everythign in computing sciences, in the early years.

CATA's purpose is a public list of all certificates that are emitted by CAs. Thus, clients are allowed to search in that public list, and verify that the received certificate is in that list, to be validate (if it is not in the list, it shall be rejected). In that way, a(conscientious) domain administrator can have access to that list and monitor all fraudulent certificates for his domain.

For the moment, there are plenty of questions to answer like :

- Who should hold those lists?
- How many lists should be created?
- How does the revokation should work?
- What should happen if a certificate is no longer available (path unreachable)?

REFERENCES

[1] E.Gerck, Overview of Certification Systems: X. 509, CA, PGP and SKIP, 1998.

[2] Ashar Aziz, Tom Markson, Hemma Prafullchandra, Sun Microsystems, Inc., v.06, 1995. https://tools.ietf.org/html/draft-ietf-ipsec-skip-06

[3] Oracle, SunScreen SKIP User's Guide, Release 1.1, 2010.

[4] Caronni, G. (2000). Walking the web of trust. In Enabling Technologies: Infrastructure for Collaborative Enterprises, 2000.(WET ICE 2000). Proeedings. IEEE 9th International Workshops on (pp. 153-158). IEEE.

[5] Stéphane Bortzmeyer. Comment fonctionne la faille kaminsky ? http://www.bortzmeyer.org/comment-fonctionne-la-faille-kaminsky.html, 2008.

[6] D. Atkins and R. Austein. Rfc 3833 : Threat analysis of the domain name system (dns). http://www. ietf.org/rfc/rfc3833.txt, 2004.

[7] P Hoffman, J Schlyter, The DNS-based authentication of named entities (DANE) transport layer security (TLS) protocol: TLSA, 2012. https://www.rfc-editor.org/rfc/pdfrfc/rfc6698.txt.pdf

[8] Ben Laurie and Adam Langley. Certificate authority transparency and auditability. http://www.links. org/files/CertificateAuthorityTransparencyandAuditability.pdf, 2011.

[9] Laurie, B., Langley, A., & Kasper, E. (2013). Certificate transparency (No. RFC 6962).

[10] Solo, D., Housley, R., & Ford, W. (1999). Internet X. 509 public key infrastructure certificate and CRL profile.

[11] Dancho Danchev. Hackers hijack dns records of high profile new zealand sites. http://m.zdnet.com/blog/security/hackers-hijack-dns-records-of-high-profile-new-zealand-sites/3185, 2008.

[12] Utkarsh. Domaining.com hacked. http://dotsutra.com/domain-names/industry-news/domaining-com-hacked/, 2011.

[13] Wallace, C. Trust Anchor Format. No. RFC 5914. 2010.

[14] P. Eckersley, "Sovereign key cryptography for internet domains", online, 2012, https://git.eff.org/?p=sovereign-keys.git;a=blob_plain;f=sovereign-key-design.txt;hb=master, last retrieved on December 21, 2014.

[15] Wang, Xinli, Yan Bai, and Lihui Hu. "Domain based certification and revocation." Proceedings of the 2015 International Conference on Security and Management, SAM. Vol. 15. 2015.