From centralized certification systems to more internet-like ones

Alexandre Kervadec
alexandre.kervadec@etu.u-bordeaux.fr

***Abstract*—The current document is a study of a paper written by 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 main) certification systems which are X.509 and CAs, PGP, SKIP, DANE and the Certificate transparency by Google, with thought on the 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

X.509, CAs and PGP are the only technologies used to certify the authenticity of a user since the beginning of the 90's. But those technologies appear to have issues, for example, Google "became aware of unauthorized digital certificates for several Google domains."[1] This kind of threat led some researchers to find new certification systems.

First sections show existing (sections I-A and I-B) or extinguished (section I-C) certification systems. Then, further sections (I-E and I-G) introduce 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:

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 company department, for an internal purposes).
2) **Subscriber :** the entity which sends data to the CA to add it to his certificate. This entity is one that needs to be trusted by the *user* entity.
3) **User :** ask information about *subscribers* to CA(s), it is 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 works 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 receives the information and puts it in a certificate.
- A user (the public in general) receives a certificate from a subscriber. In order to authenticate this subscriber, the client asks to the CA if the transmitted certificate matches with the subscriber's identity.
- The CA answers to the user to give him the precious informations about the subscriber's identity.

Let us get an example: *user1* wants to prove his identity to *Bob* with a CA (Jack) (Figure 1).
First, Oscar needs to get a certificate signed by Jack, so he requests it from Jack with some informations about himself. Jack makes the certificate and signs it, then he sends it back to Oscar.
Now Bob wants to communicate with Oscar, so to prove his identity, Oscar gives to Bob his certificate (forged by Jack). In order to verify Oscar's identity, Bob asks to Jack if the certificate is valid. Then Jack examines this certificate and answers to Bob, the answer can be:

- Yes, it is valid
- No, it is not valid
- Not sure about its validity

With that information, Bob can choose to trust or not Oscar.

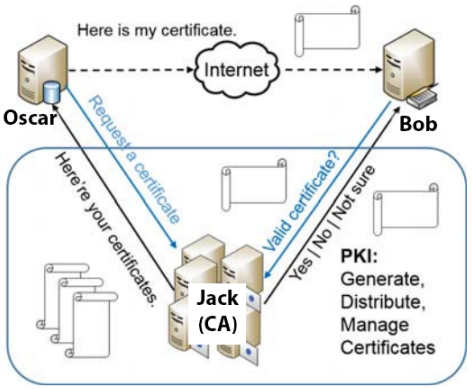

Figure 1. Diagram of the X.509 and CAs process (from [15])

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 with the subscriber's 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 :

Tutor : A.Guermouche

[1]https://googleonlinesecurity.blogspot.fr/2013/12/further-improving-digital-certificate.html

- 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 needs 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[2]), which can be completly different from one CA to another.

There is a big problem with CPS (Certification Practice Statements), that can be noted such as flexibility (for pros), because each CA answer has specific needs, so that is a sort of lack of harmonization.

Also, regarding the law, CAs deny any leak of information from themselves [1].

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 research. It has two parts: certification and encryption, the 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 authenticators: 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 knows 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 guarantee the referrals.

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

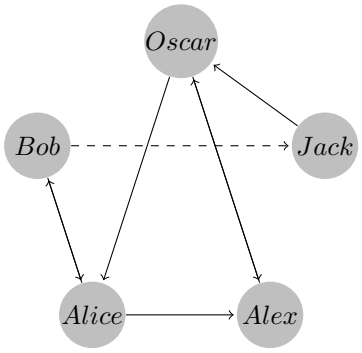

Figure 2. Communities of trust: Jack introduces Oscar to Bob's public-key certificate before Bob receives it.

Let us get an example[3] (Figure 2): Bob wants to exchange data with Oscar, whom he didn't know before. But Jack, a colleague of Bob's, signs Oscar's public-key certificate which he knows is authentic.

[2]CPS

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

Oscar then forwards his signed certificate to Bob who wishes to communicate with Oscar privately.
Bob, who knows and trusts Jack as an *introducer*, finds out, after verification, that Jack is among Oscar's certificate signer. Therefore, 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 then have to find another introducer whom Bob trusts to sign Oscar's public-key certificate. This is illustrated in Figure 2.
The dotted arrow indicates "Bob trusts Jack as introducer" and the solid arrow indicates "Jack trusts Oscar's public key validity".

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. The 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[4].

The big point is the use of such a tool in a world wide range: that is reasonably not possible. We can assume it can be used in a little group, where trustworthiness is effective, but in a wider area, it is problematic to trust such a number 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.

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

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

D. Thoughts about X.509, CAs and PGP

To be used in a commercial way, X.509, CAs and PGP need a centralized certification control system (a matter of responsibility). However, there is a struggle between this need of centralization and the decentralized internet architecture. It shows us the futility of certification authority for the whole world, since we have no centralized world governance or law. The main question is:
How can we have a main control over an internet environment, to assure the security of the data and legitimacy of users/servers?
Some answers have been provided since then (thus since the publication of the original article).

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

DANE is not implemented yet, but an IETF team is working on the standard.
It has begun with a simple assertion: 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 allows 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) might forge a false certificate for his 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 fears that Charlie might 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 emit it via DANE on DNS, that gives the entire responsibility on DNS servers.

This type is in a new register called TLSA[5] in DNS. Her is an example of a TLSA register:

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

```
_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.

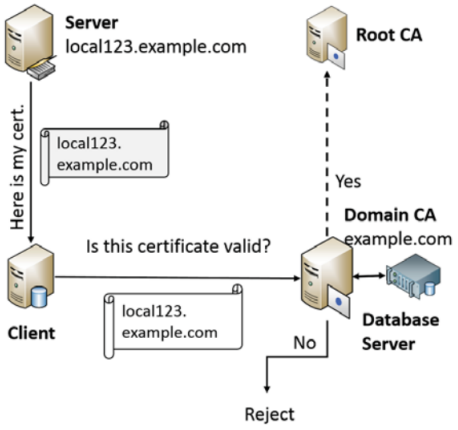

Figure 3. Process of certificate verification (from [15])

As we said before, DNS is not secured, so DNSEC has been deployed. DNSSEC is a protocol that allows to the DNS server (master) to encrypt and sign its answer with a public key cryptographic system. It also permits to return a `NXDOMAIN` type ("No Such Domain").
Nervertheless DNSSEC has some vulnaribilities such as:

- Replay attacks
- Attacks against intermediaries: because of the hierarchic and centralized approach of DNS (there is only one root), the responsibility of intermediaries is really high and a remote control by a thief should be damaging (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[6].
- The end to end validation necessity end to end is quite disturbing knowing that there is no standard and deployable solution for wide use about the last kilometer securisation.
- DNSSEC is not possible behind a DNS proxy (which filters/breaks DNS answers to the point of creating a DoS for all TLS services.

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

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

F. Sovereign keys

Sovereign keys are an Electric Frontier Foundation (EFF[7]) proposition.

A sovereign key is a cryptographic pair of keys where the public key is associated to a domain name (or a subdomain) to create a trust anchor[8] for itself.
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 associated to.
During a handshake with a TLS server, the certificate is sent with its protected domain key signature, by the same mechanism than as 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's validity, that the domain is the same as the one in the current transaction and that it signs the receive certificate.

A sovereign key has a cycle of life that begins with the generation of a cryptographic pair of keys by the domain administrator (who wants to use TLS server on his domain). This administrator must at first, prove that he really is controlling his domain before seeing it in public data-structures.
To do so, he must give his certificate (of which the CN covers his domain name declared protected by the sovereign key) signed by an AC recognized by the *timeline servers*[9] or through DANE (once the standard has been published and implemented).
During the public key registering by the timeline servers, proofs of control on the domain are examined and added to the sovereign key metadata. Other data are added such as:

- Expiration date.
- Names (and ports) of TLS services uses and protected by the sovereign key.
- A list (that may be empty) of domain names whose sovereign keys are granted to revoke (in case of loss of control of the private part of the sovereign key) the entry being entered.

A data-structure of sovereign keys being a append-only list, once the sovereign key is entered, it cannot be removed. The domain administrator can only modify informations by pushing data in that data-structure. To remove a sovereign key (in the case the private key has been lost), the domain adminstrator needs to use a domain name with a sovereign key granted to modify its own sovereign key (thus present in the list of his own sovereign keys, see in the item list below) or wait for his sovereign key's expiration date.

This structure is useful because it protects from DNS or AC compromise of sovereign key by smurf attacks[10]. It also needs to be used with care, because an unadvised administrator may lose control of its own domain.

The drawbacks of this systems are:

- A service declared protected in a data-structure of a timeline server needs to use a secure protocol (HTTPS instead of HTTP for example) to avoid downgrade attacks. It may be seen as an intrusion into the freedom of a service governance.
- Loosing a private key may result in a long term denial of service if no domain sovereign key is granted to modify the private key's sovereign key.
- A compromized private key may lead to a total loss of control of the entire domain.
- The use of ACs to certify the domain owner is the big flaw of this system.

G. CATA: Certificate Authority Transparency and Auditability

This protocol has been developed by a Google team a few years ago [8]. It is still under development [9] so it is not implemented and deployed yet. It also has to be improve, like everything in computing science, 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 validated (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 should the revokation work?
- What should happen if a certificate is no longer available (path unreachable)?

II. Conclusion

Despite the great need to replace the PKIX security model[11], none of those presented above may be valid, or well tested to be implemented on a large scale (world wide).

Therefore, the best model may be the sovereign keys system associated with the DANE control of a domain control certification, as soon as security problems are fixed in DANE with DNSSEC.

Otherwise, another point of view that needs to be discussed, is the economic impact of such a revolution: companies like VeriSign[12] or GeoTrust[13] may face bankruptcy with the removal of all CAs. Knowing that they are big deciders in the internet activity, they detain a great lever on governments (because money leads capitalism), to stop or at least slow down the deployment process of a new certification system.

In general, to fit the Internet, experience has shown that we need to develop systems that are the most distributed possible, like the internet paradigm.

[7] www.eff.org

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

[9] The list of AC's recognized by the timeline servers is public and must be up to date, by conception.

[10] Attacker able to give proof of domain control.

[11] X.509 and CAs

[12] www.verisign.com

[13] www.geotrust.com

ACKNOWLEDGMENT

I would like to thank Pr. Abdou Guermouche for his feedback and thought on this paper, and also Pr. Jean-Jacques Bernaulte for his grammatical review.

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.
[16] Maury, Florian. "La fin annoncée des autorités de certification? Alternatives: TOFU, Convergence, CATA, Clés souveraines, DANE." Convergence 10 (2011): 509.
[17] Correia, Miguel Medeiros, and Mustafa Tok. DNS-based Authentication of Named Entities (DANE). Technical report, Universidade do Porto, 2011–2012, 2015.