1 X.509 certification path

Figure 1 presents two CA networks belonging to two distinct companies, `BigSofware corp.` and `DevTeam inc.`. Each arrow on the figure represents a certificate $\{xy\}$, issued by the source x of the arrow, for the target y of the arrow.

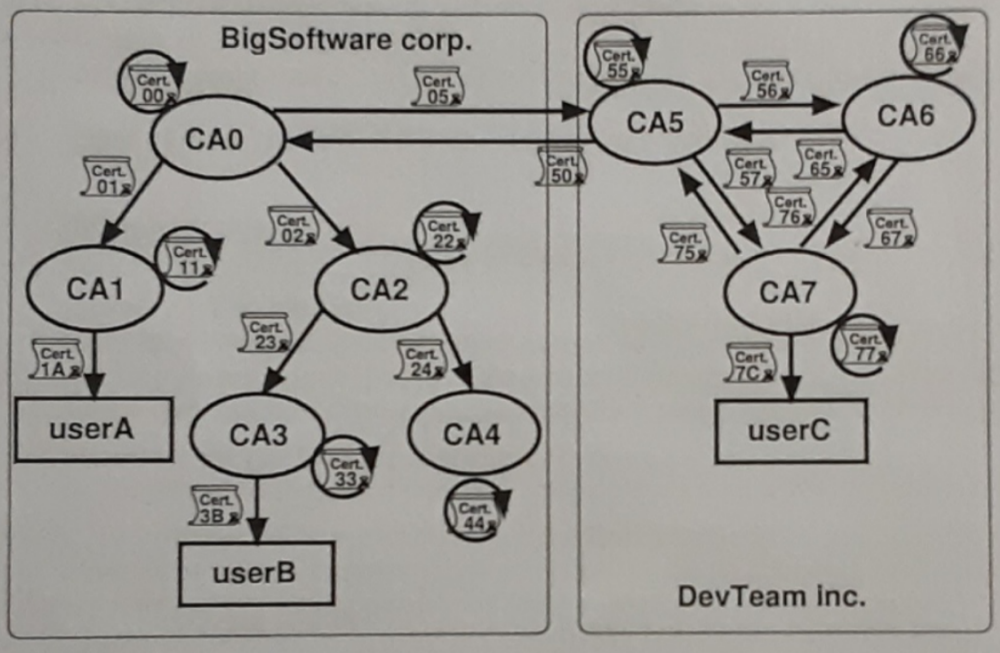

FIGURE 1 – A PKI architecture

1. What is the trust model used by `BigSoftware`? The model used by `DevTeam`?
2. What is a trust anchor and supposing that each user only has one, what could be those of `userA`, `userB` and `userC`?
3. What can you say about $\{xx\}$ certificates, with the same source and target?
4. What can you say about certificate pairs $\{xy\}$; $\{yx\}$, with complementary sources and targets?
5. `userB` sends a message in clear, but signed, to `userC`. Give a certification path used by `userC` to validate the signature of `userB`. Are there several possibilities?
6. `userC` sends a message in clear, but signed, to `userA`. Give a certification path used by `userA` to validate the signature of `userC`. Concludes on the different possibilities of trust anchors of Question 2.
7. What happens if `Alice` can use an OCSP responder, still if she has only one trust anchor?

(1)

The trust model adopted by BigSoftware is hierarchical while the trust model used by DevTeam is peer-to-peer

(2)

A trust anchor is a public key of a self signed certificate whose validity has been verified online. For example CA0 can be the trust Anchor for Alice and Bob, while user Charlie can have CA7.

(3)

They are self signed certificate that each one can produce. They are used, as an example, in the root level of the CAs

(4)

They are mutual cross certificate between two parties

(5)

The path to verify userB signature would require user C to go through:

CA7 → CA5 → CA0 → CA2 → CA3

Specifically we have: {7C} → {75} → {50} → {02} → {23} → {3B}

(6)

As before, we can follow the path from user A to user C.

Since we assumed ACO as trust anchor Alice can start from there

It would be: {7C} → {57} → {05} → {00}

(7)

For Alice to accept an OSPC response as a signature validation, it must be one of its trust anchors. The role of OSCP can be taken by ACO and it would make no difference to Alice

2 Forging X.509 Certificates (FLAME virus attack on Microsoft update certificates 2012)

We consider X.509 certificates signed by the `md5WithRSAEncryption`. We want to submit an RSA public key (N_1, e_1) to the certificate authority for certification such that we can infer a fake certificate for another RSA public key (N_2, e_2). RSA moduli are assumed to be 2048-bit long. We also assume that $e_1 = e_2 = 65537$ and that all fields except the moduli parts in both certificates are identical.

We assume that we have filled all fields of the X.509 form, except the RSA modulus part (and the signature to be appended by the certificate authority). We also assume that the length of the form (represented as a string) from the beginning of the form to the beginning of the modulus field is a multiple of 512 bits. We finally assume that for an initial vector IV we already found two different 1024-bit blocks b_1 and b_2 such that $MD5_{IV}(b_1) = MD5_{IV}(b_2)$ (we actually can, very efficiently).

1. Show that for any 1024-bit string b, we have $MD5(b_1||b) = MD5(b_2||b)$.
2. We thus need to find b such that $N_1 = b_1||b$ and $N_2 = b_2||b$ are valid RSA moduli for which we know the factorization. Recall what it a valid RSA modulus.
3. Let p_1 and p_2 be two different arbitrary prime numbers of at most 512 bits. Show that we can compute an integer b_0 between 0 and p_1p_2 such that p_1 divides $b_1 2^{1024} + b_0$, and p_2 divides $b_2 2^{1024} + b_0$.
4. By taking $b = b_0 + kp_1p_2$ for $k = 0, 1, 2, \ldots$, (heuristically) show that we are likely to find k such that $(b_1 2^{1024} + b)/p_1$ and $(b_2 2^{1024} + b)/p_2$ are both primes. Conclude.
5. To what extent is the above attack devastating?
6. We now assume that given two vectors IV' and $IV"$ defining we can find two 1024-bit blocks b_1 and b_2 such that $MD5_{IV'}(b_1) = MD5_{IV"}(b_2)$. Can we now derive an even more dangerous attack?
7. We now assume that given a vector IV' and a 1024-bit block b_1 we can find another 1024-bit block b_2 such that $MD5_{IV'}(b_1) = MD5_{IV'}(b_2)$. Can we now derive an extremely dangerous attack?
8. What is the fix?

3 Serial number and forging certificates

1. What is the major use of an electronic certificate and what are thus the main informations contained in general in an X.509 electronic certificate?
2. Why are electronic certificates public?
3. What field(s) of X.509 certificates enable(s) a unique identification of certificates?

When a CA issues a certificate, it includes a signature on a hash of a payload denoted `tbsCertificate`. This payload includes a public key, a domain name, and a serial number. Supposing that the used hash function has known collision attacks, an attacker could choose two distinct values `tbsCertificate1` and `tbsCertificate2` that have the same hash with that function. In practice, the attacker uses the collision attacks on the hash function to generate two payloads with the following property : `tbsCertificate1` is a benign certificate payload that the CA would willingly sign (e.g., a certificate for some new domain that the attacker controls), but `tbsCertificate2` is an evil certificate payload that the CA would never sign (e.g., a certificate for microsoft.com). The attacker sends a certificate request corresponding to `tbsCertificate1` to the CA ; the CA signs it and returns the signed certificate.

4. In the above scenario, explain how the attacker could produce a forged certificate for the payload `tbsCertificate2`?
5. This attack requires the attacker to predict the value of the serial number that will be used when the CA signs the certificate for `tbsCertificate1`. In what kind of certification policy setting would this prediction be relatively easy to perform?
6. Discuss of solution(s) to prevent this prediction an their potential drawback(s).

(1)

The main use of an Electronic certificate is to create (and guarantee) a link between an identity and a public key.

The main components are: Identity, validity, public key, signature of CA

(2)

They are public as anyone may go and consult the content of it. This is particularly useful in the implementation of PKI

(3)

The unique identification of a certificate is given by: ISSUER + SERIAL NUMBER

(4)

The only action required is to grab the signature received by

the CA and apply it to the evil certificate. This is possible since there is a collision in the hash function and the signature on the tbsCertificate1 is the same that would be produced if it was applied by tbsCertificate2.

(5)

If we have a sequential number assignation policy then the next number would be easily predictable

(6)

The easiest way would be to create certificates with random numbers as identifier. However, it may become hard to "sort and search" and to avoid duplicates among CAs

4 The Denning-Sacco key transport protocol

Consider the following protocol, where $Cert_X$ is an X.509 certificate, T_i is a timestamp of time i, $(pubX/privX)$ is a pair of public/private asymetric keys, K_X is a symmetric session key and parts included in brackets [] are optional :

i Alice randomly generates K_{AB} ;
ii Alice fetches a certificate $Cert_B$ for Bob ;
iii Alice sends $[Cert_A]||E_{pubB}(Alice||K_{AB}||T_i||SIG_{privA}(K_{AB}||T_i))$ to Bob.

1. What is an X.509 certificate and what are the main informations it contains ?
2. What are a Certification Authority and a trust anchor ?

3. Give all the steps that Alice will perform in order to recover and verify Bob's public key.
4. Give all the steps that Bob will perform, with all the different used keys, in order to verify Alice's message, after having verified Alice's certificate.
5. Is it mandatory to cipher the signature ?
6. Why is $Cert_A$ optional in the exchange ?
7. Mount an identity spoofing attack on this protocol.
8. Propose a counter-measure.

(1)

X.509 certificates are a standard for electronic certificates that allows to create and guarantee a binding between an entity and a public key.
The main info contained are:
Identifier, Validity, Public Key, Signature of the CA
Then additionally we have: Country, Algorithm, Common Name, etc...

(2)

Certification authorities are organization that issue and guarantee certificates to users. Their certificate may be signed by another CA or self signed. Trust Anchors are public keys of a self signed certificate whose validity has been verified online.

(3)

To recover and verify the public key of Bob, Alice should recover the certificate. She can do it with OSCP for example or by asking to the autority that issued the certificate, then she would have to get the public key of the CA and use it to decipher the hash. Then she can verify it by hashing the document and comparing the result. She may want to check the signature of the CA too, at that point the process can go up to the root CA.

(4)

Bob has to verify Alice certificate (as explained above). Then he has to decrypt the message with its K_{privB}. Now he has access to the K_{AB} and to verify the signature by Alice he can VERIF (G', Kpub, M)

(5)

It is usefull to avoid tampering with it. It may cause DoS atlacks by messing with the orginal signature. In general it is not mandatory

(6)

Bob has the possibility to retrive it or may already have it stored locally.

(7)

If Mallory can interfere at step (ii) he may be able to sent his certificate to Alice. From that moment on Alice will perform all the

operations with CertM instead of B and a MITM can occur

(8)

Alice should verify the certificate that she has received and be sure that it is the correct one. Moreover, there could be the possibility of collition so she should be aware of that.

5 Security Architecture for the "no-key" Massey-Omura cipher

Alice whises to send a message to Bob, encoded vy an element m of a cyclic group G of order n. They use the following (key-free) protocol :

i Alice secretly chooses an integer x_A, such that $1 < x_A < n$ and $gcd(x_A, n) = 1$. Then she sends to Bob the element $a = m^{x_A}$.

ii Bob secretly chooses an integer x_B, such that$1 < x_B < n$ and $gcd(x_B, n) = 1$. Then he sends to Alice the element $b = a^{x_B}$.

iii Alice computes the integer y_A such that $1 < y_A < n$ and $x_A y_A \equiv 1 \mod n$. Then she sends to Bob the element $c = b^{y_A}$.

iv Bob computes the integer y_B such that $1 < y_B < n$ and $x_B y_B \equiv 1 \mod n$. Then he computes c^{y_B}.

1. Show that we have $m = c^{y_B}$.
2. Take $G = (F_{19}^*, \times)$. Suppose that Alice chooses the integer $x_A = 5$ and that she sends to Bob $a = 2$, ... Find the corresponding element m, without exhaustive search.
3. Describe a Man-in-the-middle attack on this protocol.
4. What general technique could prevent such an attack ?
5. Propose some modifications of this protocol and outline the main components of a security architecture that could a priori ensure a protection against Man-in-the-middle attacks.

6 Expiration and revocation

1. What are the main X.509 certificate revocation mechanisms? Briefly describe their characteristics.
2. Why associate a revocation mechanism to certificates that comprise a validity period (and thus an expiration date)?
3. If the use of one or several revocation mechanism is systematically enforced, why use also an expiry date?

(1)

The main revocation mechanism are:

CRL → Certificate Revocation List

- A list of certificates that have been revoked by the CA
- Posted regularly by CA and may leverage delta CRL to avoid growing and growing lists.

OCSP → Online Certificate Status Protocol

- A server to which we can query and have real time response about certificates that are valid NOW
- This avoids the distribution of huge CRLs but we cannot say anything about prior validity of a certificate

(2)

Because we may have certificates that have been compromised befor the expiration date (private key has been leaked)

(3)

It is useful to avoid everlasting certificates that may have to be verified again