Exercise 8 Solutions

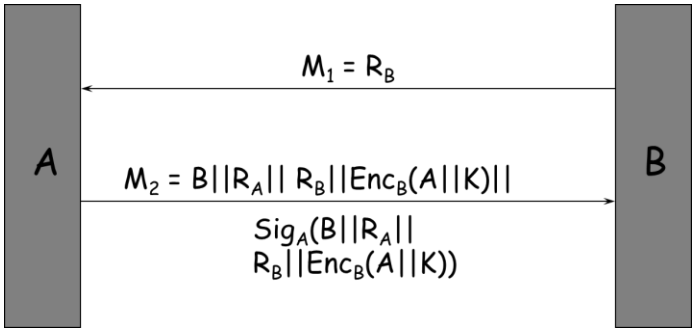
CS4286

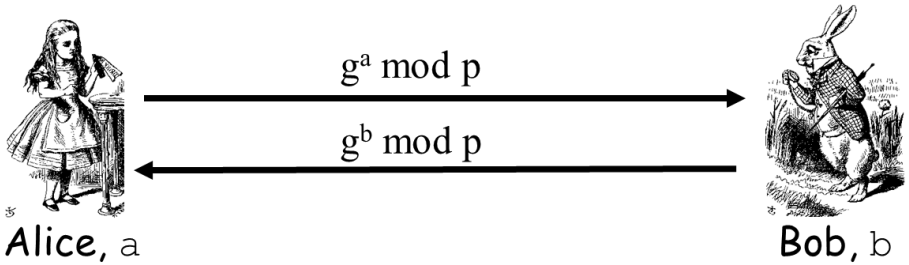
Gerhard Hancke

Question 1

i. How would you use asymmetric cryptographic to do key transport between only two parties.
   - Use public key encryption for one party to send key to the other.

ii. How would you use asymmetric cryptographic to do key agreement between only two parties.
   - Use Diffie-Hellman, both parties contribute towards the key value.





Question 2

Assume that we require *X*=2 trusted signatures or *Y*=3 marginally trusted signatures, consider whether in the following cases Alice will trust Bob’s key:

I. Alice has 2 *trusted* friends who have signed Bob's key.
II. Alice has 2 *marginally trusted* friends who have signed Bob's key.
III. Alice has 1 *trusted* friend and 1 *marginally trusted* friend who have signed Bob's key.
IV. Alice has 1 *trusted* friend and 2 *marginally trusted* friends who have signed Bob's key.

Question 2

I. 2 signatures of *trusted* friends are sufficient, Alice trusts Bob's key.
II. 2 signatures of *marginally trusted* friends are not sufficient, Alice does not trust Bob's key.
III. 1 signature of a *trusted* friend and 1 signature of a *marginally trusted* friend can be expressed as $1 \cdot \frac{1}{2} + 1 \cdot \frac{1}{3} = \frac{5}{6} < 1$, and therefore Alice does not trust Bob's key.
IV. 1 signature of a *trusted* friend and 2 signatures of *marginally trusted* friends can be expressed as $1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{3} = \frac{7}{6} \geq 1$, and therefore Alice trusts Bob's key.

Question 3

- (a) A certificate does not necessarily have to be signed directly by a CA, there is something called a $certificate\ chain$. Can you imagine what a certificate chain is?

Question 3

- Solution to (a)

For organizational reasons it is sometimes desirable that not all certificates are signed using the master key of a CA. Instead, there exists intermediate certificates which are used to certify keys.

For example, imagine a company named $ACME\ CA$ possesses a master key and the corresponding certificate. It then creates two organizational units $ACME\ Business$ and $ACME\ Personal$. Both of these two units have their own private/public key-pair. The parent $ACME\ CA$ now creates a certificate for each of the two units, certifying that they own their respective public key.

Question 3

- Solution to (a) cont

If a company $Company\ Corp.$ wants to certify their public key, they contact $ACME\ Business$, pay lots of money (digital certificates are sometimes called the "most expensive bits in the internet") and then get a certificate from $ACME\ Business$ certifying that they own the public key.

If John then wants to verify the public key of $Company\ Corp.$, he does the following:

(1) Using the certificate of $ACME\ Business$ he verifies that $ACME\ Business$ did indeed sign the certificate of $Company\ Corp.$.

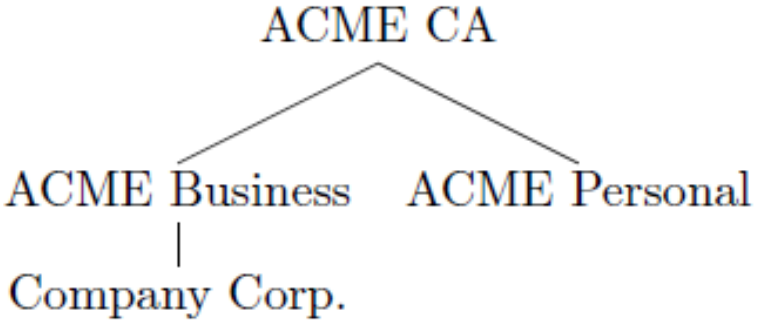
Question 3

- Solution to (a) cont

(2) Using the certificate of $ACME\ CA$ John verifies that the certificate of $ACME\ Business$ was indeed signed by $ACME\ CA$.

(3) Because John trusts $ACME\ CA$, he therefore also trusts $ACME\ Business$ and by extension now also $Company\ Corp$..

John in fact verified the certificate chain up to a $trust\ anchor$ (root CA) (i.e., $ACME\ CA$). Graphically, the chain would look like this:



Question 3

- (b) Who signs the certificate of a CA?

Question 3

- Solution to (b)

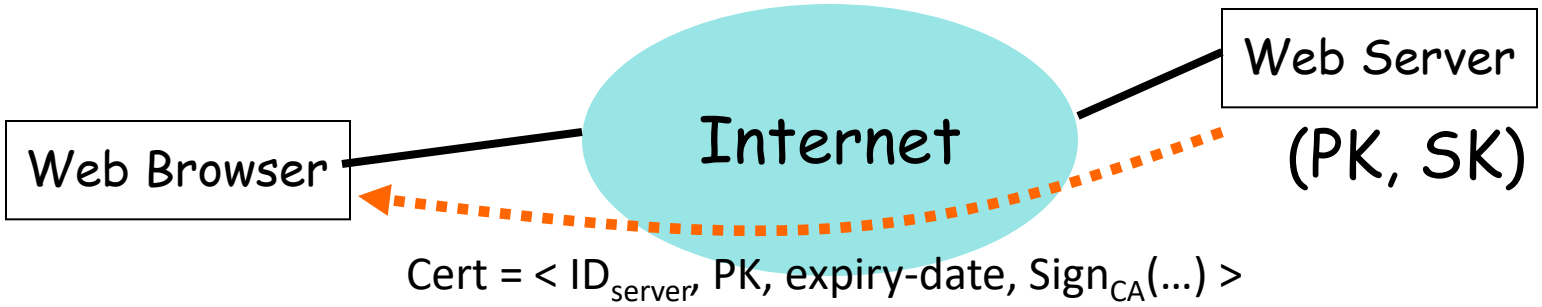
A CA's certificate is signed by another CA (cross-certification)or by a more powerful CA (the root CA).

Alternatively the CA itself signs the key. These are so-called *self-signed* certificates, because they use their own private key to sign their certificates.

Question 3

- (c) Considering the default list of trusted certificates built into every browser, explain how one malicious CA could compromise *every* secure connection made by a particular user.

How to Use PKI – Secure Web Browsing



- The web browser has the CA's public key built in.
  - **The legitimacy of the web browser software becomes crucial for ensuring the security of digital certificates**
  - A certificate is **NO** more secure than the security of the web browser
- In practice, each browser trusts multiple CAs rather than just one
- Exercise: find out the number of CAs that your browser trusts

Question 3

- Solution to (c)

The special property of CA certificates included in a web browser is that the web browser implicitly trusts any certificate signed by one of these CAs.

If a CA is malicious (and has the possibility to intercept the data traffic of a user) it can do the following (note that when a user connects to a server over SSL, the server will always first send its certificate):

Question 3

- Solution to (c) cont

① John decides to visit https://mail.google.com and types the address into the browser.

② The malicious CA intercepts the attempt, and realises that John tries to access Google Mail. It then (on-the-fly) generates a new key and a corresponding certificate, signs it with its own key, and sends the certificate to John.

③ John's browser receives the certificate and checks the issuer. Because the issuer of the certificate (i.e., the malicious CA) is in the list of trusted CAs, the browser will typically accept the certificate without any other formality.

Question 3

- Solution to (c) cont

④ The malicious CA then simply forwards all following traffic to the real Google Mail, being able to read all the traffic in cleartext.

⑤ John is now browsing Google Mail, but instead of his connection being secure, it is in fact compromised and the malicious CA can read everything.

Modern web browser may have some limited protection against this, for example by alerting the user if a certificate for a website suddenly changes. But ask yourself, if you get a notification about some problem with a certificate when connecting to a website, do actually read the text carefully, or do you just click whatever button will make the notification go away?

Question 4

In the lecture we discussed how good key management solve practical problems in open and closed payment systems. What would such a protocol look like?

Design a protocol for allowing a point-of-sale and a payment card to conduct on offline payment transaction (without help of a online third party).

Sometimes a card also might want online verification before approving transaction, how would a card talk to its issuer?

Question 4

- Use public key cryptography and certificates.
- Merchant POS stores certificates for issuers (banks or credit card company)
- Card stores its private key, and public key cert
  - Cert is signed by his issuer
- Card also stores symmetric key (K_I)
  - Key is shared with issuer
  - Key is unique to card

Question 4

POS >> Card: R_M, Verify transaction data M

Card >> POS: Cert_Card,$(R_M, M)_{signed\_Card}$

- POS: Verify certificate of card
  - Verify signature of issuer using stored issuer cert
- Use card public key from cert to verify signing
- Card is thus authenticated, transaction approved
- If the card can be online (with help of POS) then the issuer/card use symmetric crypto (shared secret keys)
  - Card >> POS (passed on to Issuer) $E\_K_I(R_I$, Data for issuer)