Information Security Issues

IBM Global Services - Security & Privacy Services

Paul Lebouille
19 February 2002

e-mail: Paul.Lebouille@nl.ibm.com

Objectives

- Basic Cryptography
- PKI concepts
- Wireless Security

Module Objectives

In this module, we will discuss

- symmetric and asymmetric cryptography
- encryption, digital signatures, non-repudiation

A very old problem is the transmission of messages which should only be legible by some specified person.

Consider the following attack scenarios:
Alice wants to communicate with Bob. But there may be an attacker, called Eve.

Scenario	Requirement	Attack	Remark
Alice wants to send a secret msg. to Bob	Confidentiality	Eve listens to the transmission	
Alice wants to make sure that the msg. is not modified.	Integrity	Eve intercepts the msg and sends a modified version to Bob	This attack is called "Man in the middle"
Bob wants to be sure that Alice is the author of the msg.	Identification/ Authentication	Eve sends a msg to Bob claiming to be Alice	Another option: Eve re-sends a msg from Alice, called "Replay attack"
Bob wants to make sure that Alice cannot repudiate her msg.	Non-repudiation	Alice claims not to be the author of the msg.	Also consider replay!
Bob wants to be sure that Alice is the author of the msg.	Identification/ Authentication	Eve intercepts a msg and delays the submission	This is called "Delay-Attack"

Trying to solve this problem can be seen as the beginning of cryptography.

Historical Ciphers

- Nonstandard hieroglyphics, 1900BC
- Atbash cipher (Old Testament, reversed Hebrew alphabet, 600BC)
- Caesar cipher:
  letter = letter + 3: "fish" -> "ilvk"
- rot13: Add 13/swap alphabet halves
  - Usenet convention used to hide possibly offensive jokes
  - Applying it twice restores original text
- Substitution Ciphers
  - Simple substitution cipher:
    a = p, b = m, c = f, ...
  - Break via letter frequency analysis
- Polyalphabetic substitution cipher
  - 1. a = p, b = m, c = f, ...
    2. a = l, b = t, c = a, ...
    3. a = f, b = x, c = p, ...
  - Break by decomposing into individual alphabets, then solve as simple substitution

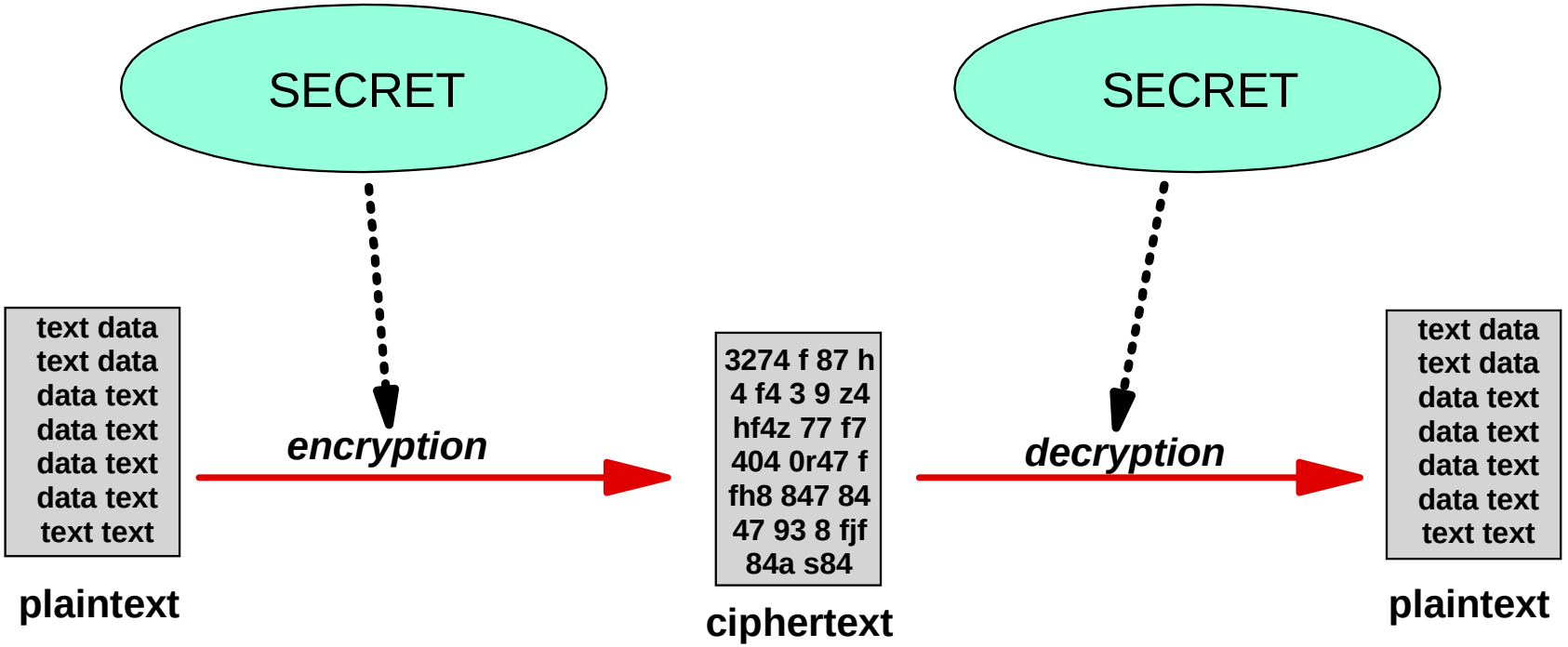
Over the time the people improved the concepts more and more.

One-time Pad (1917)

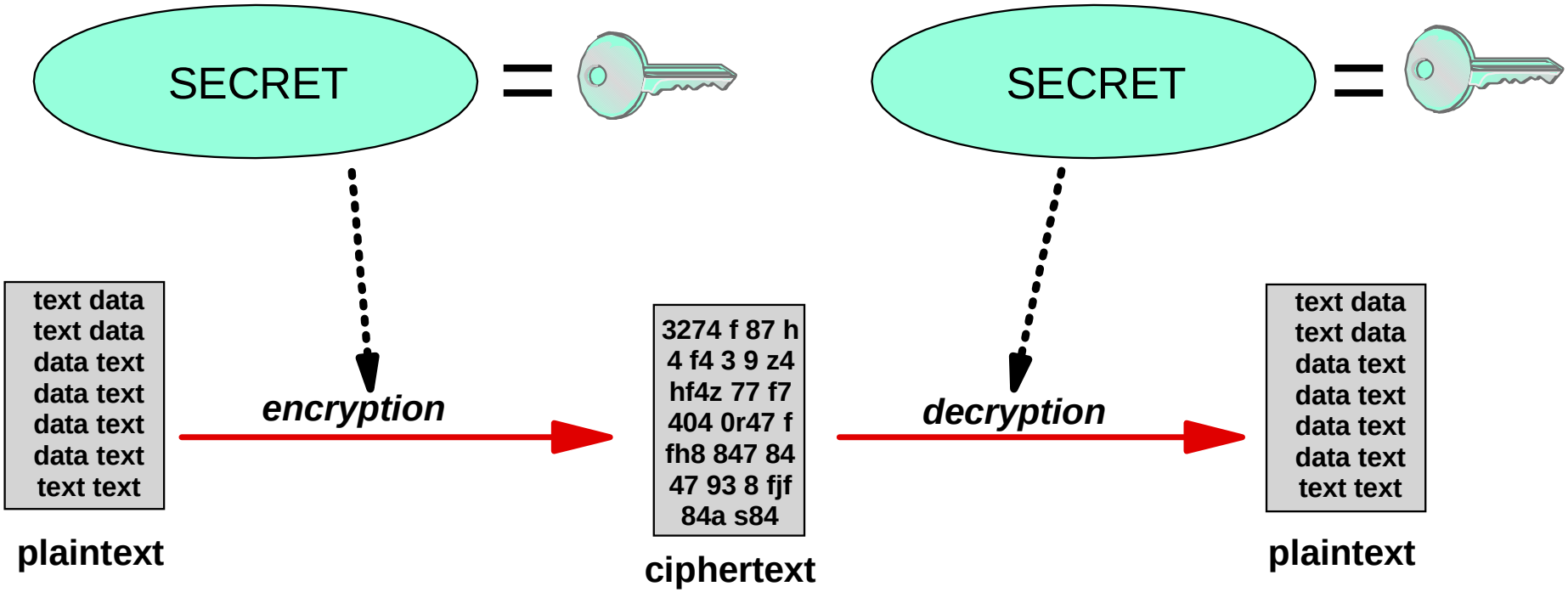
- OTP is unbreakable provided
  - Pad is never reused
  - Unpredictable random numbers are used (e.g. physical sources)
- Message s e c r e t = 18 5 3 17 5 19
  OTP + 15 8 1 12 19 5
  Encrypted message: 7 13 4 3 24 24
  OTP - 15 8 1 12 19 5
  Decrypted message: 18 5 3 17 5 19
- Many snake oil algorithms claim unbreakability by claiming to be a OTP
  - Pseudo-OTP 's give pseudo-security
- Cipher machines attempted to create approximations to OTP 's, first mechanically, then electronically

Think of the following thesis: every technical realisation of protecting information uses cryptography.

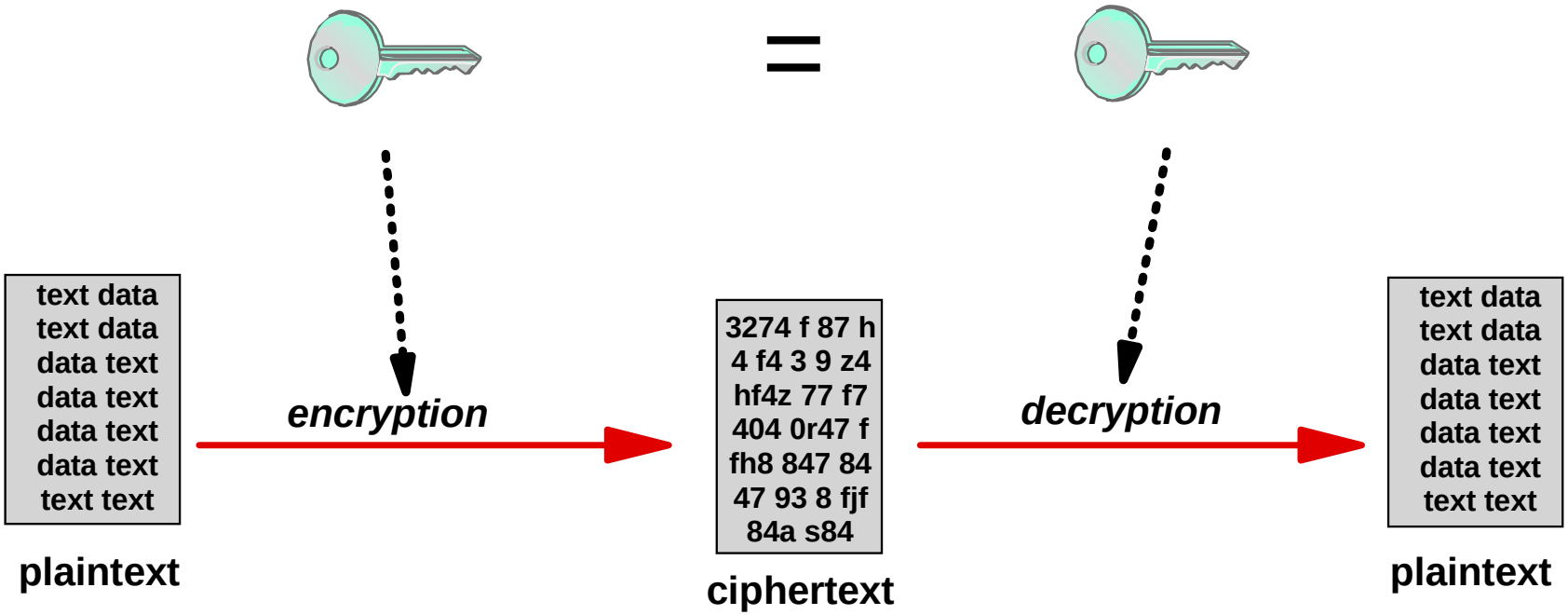
From a general point of view, cryptography is a mechanism using a defined distribution of secrets to transfer a message encrypted.



Usually, those secrets are called keys.



There are two main systems. The older one is called symmetric cryptography, where the secrets for encryption and decryption are equal.



Such a key is usually called a secret key. Main problem of this system is to agree on a shared secret key, i.e. to distribute keys.

Symmetric ciphers have a long history. There is a lot of experience about the ciphers and the methods attacking them (crytpo-analysis).

There is a big number of symmetric ciphers:

- RC2, RC4, RC5
- IDEA
- DES (Data Encryption Standard, developed by IBM and NSA)
- Blowfish
- Triple DES (3DES)
- The new Advanced Encryption Standard: Rijndael
  - In 1997 the National Institute of Standards and Technology announced a request for candidate algorithm nominations for a new Advanced Encryption Standard
  - Round by round, they kicked-of algorithms which didn't meet the requirements till the last round, where they had five algorithms. Rijndael was finally announced to be the winner.
  - See csrc.nist.gov/encryption/aes details like algorithm specifications, evaluations etc.
- MARS (AES finalist from IBM)
- ...

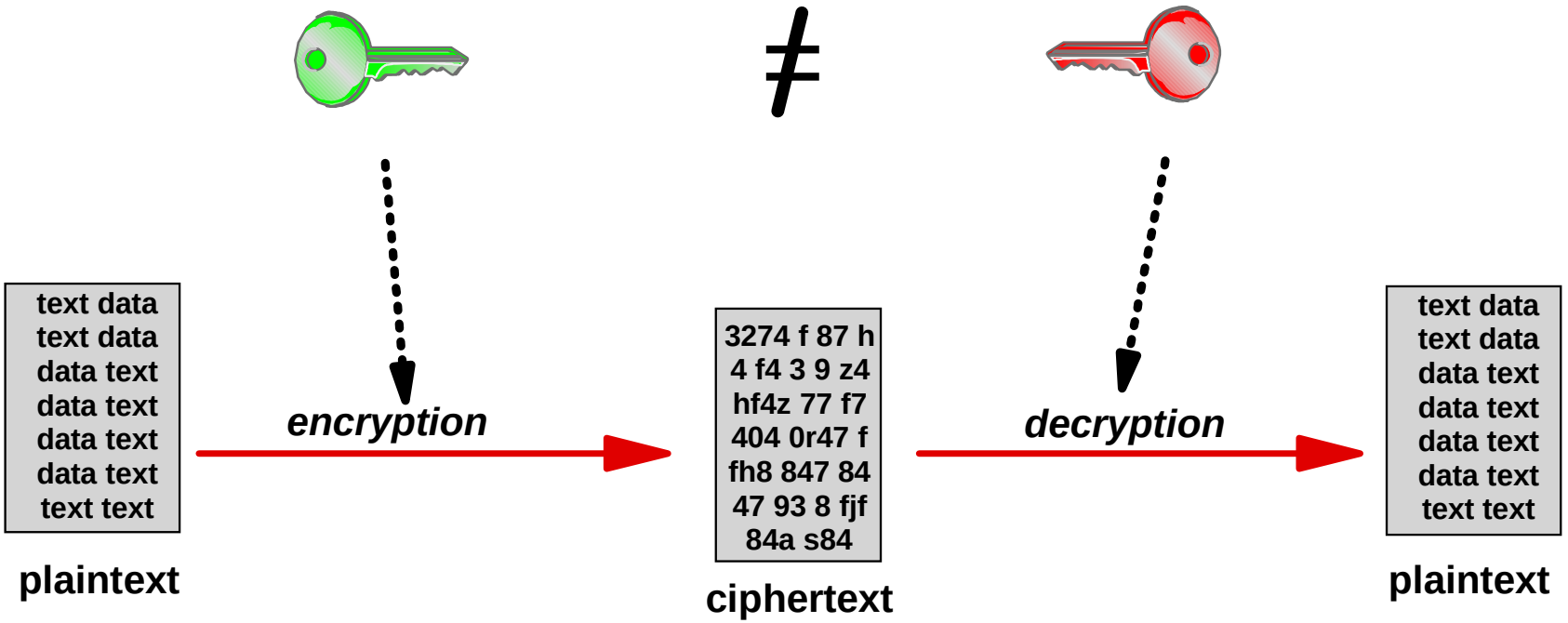
All of the above are block ciphers, which means that they treat messages by dividing them in blocks of a fixed length which then get encrypted.

How famous a cipher may be, how secure the authors claim it to be, be aware of the security flaws that can be contained!

Consider the history of DES:

- Developed by IBM based on an old algorithm called Lucifer.
- NSA made a minor change to it before being published in 1976 (which wasn't really intended by NSA). NSA made a statement about the strength of the algorithm, which led to its wide use (e.g. by financial institutions).
- In 1990 Shamir and Biham "discovered" a new crypto-analysis method called differential crypto-analysis. But DES showed to be incredibly resistant against this kind of attack. Why? Because the developers knew about this method - without revealing!
- In 1997 a financial institution lost a trial in court against somebody claiming that a theft has stolen his debit card (EC) and was able to pay with it though the PIN was not revealed.
  See www.jurpc.de/rechtspr/19980122.htm
  - **In the court decision, it is mentioned that DES can be broken with an equipment (at that time!) cheap enough to make this attractive for large criminal organisations.**
  - **As a reaction, german financial institution changed the PIN mechanisms, which urged them to replace around 40 mil. cards.**

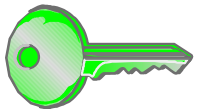
The newer systems work with different secrets, and are called asymmetric cryptography. Moreover, the secret for encryption is not secret!



In this case, the encryption key is called public key and the decryption key is called private key. The keys are strongly related by some simple conditions.

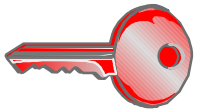
The main problem of asymmetric cryptography is to make sure, that the private key is held by the person one wants to send the message to.

public key



≠

private key

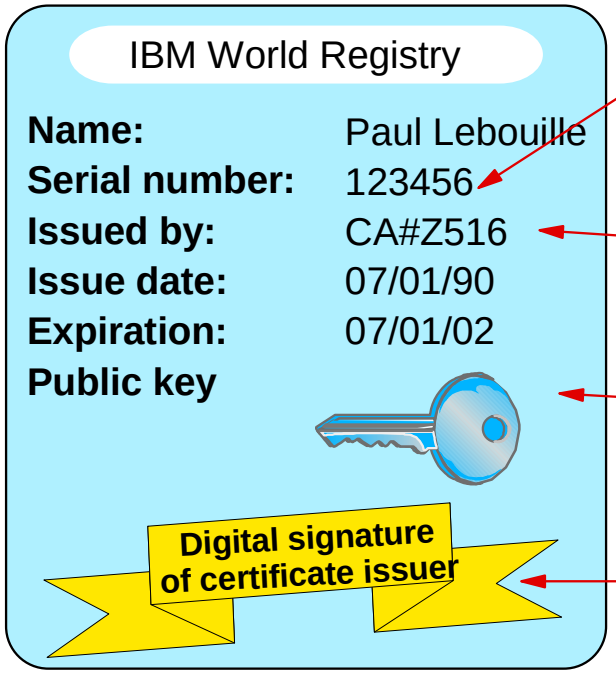


A message encrypted with a public key is legible by anybody holding the corresponding private key.

There is no inherent way of knowing the person who has the corresponding private key.

An (freaky) idea would be to have the person's public key right after its name in every public phone book! (How to prevent them from being modified?!)

This is where certificates enter. Certificates confirm that the public key given in the certificate belongs to a private key held by the person mentioned.



Identifies unique certificates

Certificate issuer identification

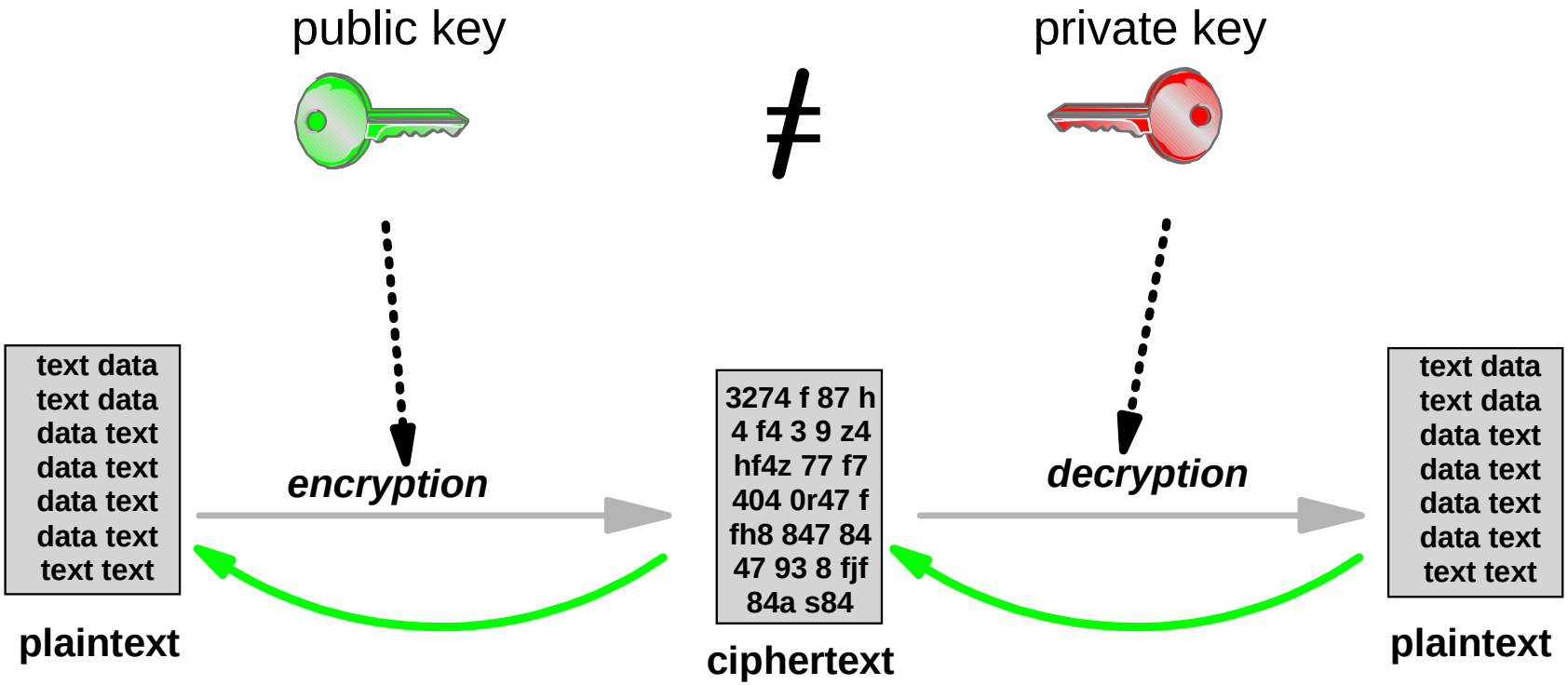
Consumer's public key

Digital Signature of the issuer

To trust a certificate means to trust the party who issued the certificate (and not the person for whom the certificate is issued).

To protect a certificate from being modified one uses digital signatures.

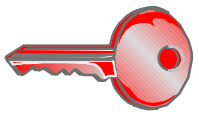
Using the system in the opposite direction provides that the message can only be created by the private key holder.



This *can* provide authorization and non-repudiation because the message must be created by the private key holder. That's the basis for (asymmetric) digital signatures.

Another concern with asymmetric cryptography is (similar to the symmetric case) to protect the private key from disclosure.

private key



Who has the private key, can read any message (e.g., mail) encrypted with the corresponding public key (like in the symmetric case) and can sign for the person the key-pair is attached to.

An idea to protect the private key is to store it not on a full system (i.e. a computer) but in a SmartCard, where all crypto-operations with it are performed. The SmartCard access gets restricted by the use of a PIN. The place where a user's private credentials are stored is called Personal Security Environment (PSE).

There are many realisations of asymmetric crypto-algorithms.

(Nearly) all realisations are based on mathematical problems.

- None of the systems is provably secure
- Experience in studying the underlying problem gives confidence

The most famous one is RSA, named after the inventors Rivest, Shamir, Adelman (1977).

There are many others

- Digital Signature Algorithm (DSA)
- Hidden monomial systems
- ElGamal
- ...

Currently, asymmetric algorithms based on elliptic curves are hot, due to the small key sizes and the performance benefits those systems offer.

Usually, symmetric and asymmetric cryptography are used in combination.

Usually messages get transferred like this:

- randomly choose a symmetric cipher and a corresponding secret key
- encrypt the message using this secret key
- append the secret key encrypted by asymmetric cryptography to the encrypted message

Advantage:

- performance benefits, due to the fact that symmetric systems are much quicker.
- higher security compared to using only symmetric ciphers (with fixed keys) due to the use of random keys
  - **make sure that the "random" function is appropriately realized - for some applications, random number generators build on hardware are required.**

Another important tool for practical use of cryptography are hash functions.

Hash function h take a message M (of variable) length, and calculate a value h(M) of fixed length (called the hash value), so that the following requirements are fulfilled:

- calculation h(M) for a given M is easy
- finding M to a given value h such that h=h(M) is hard
- finding M' to a given M such that h(M)=h(M') is hard

Use of such a function:

- storing passwords: instead of storing a password P in a file store h(P). If user then types in some Word P', the system only has to check whether h(P') equals the stored value h(P), but the particular password P has not to be stored somewhere.
- preventing tampering with some transferred document: before transferring the document D calculate h(D). Then transfer it, call the recipient and let him calculate the hash h(D') of the received document D'. If it equals h(D), the document is transfered unchanged.
  (Remark: the idea of a checksum is similar, but compare requirements above).

Here is a list of well known hash algorithms. Be aware of changing status!

Algorithms computing a hash function h:

- MD2: 128-bit output, deprecated
- MD4: 128-bit output, broken
- MD5: 128-bit output, weaknesses
- SHA-1: 160-bit output, NSA-designed US government secure hash algorithm, companion to DSA
- RIPEMD-160: 160-bit output
- ...

Hash functions are of particular importance for digitally signing (electronic) documents.

Use case: to protect a document from being tampered with (e.g. in the transmission through a public network).

- protection also prevents unintended modifications

Like ordinary signings, digital signatures also provide evidence of who created the message.

There are some other names for digitally signing documents

- Message Authentication Code (MAC) provided by hashing and encrypting with a secret (symmetric) key
- HMAC, which is hashing data plus a secret (symmetric) key: h(key, message)
- ...

The realisation of digital signatures is usually provided through a combination of hashing and encryption.

Digital signatures are made like this:

- For a given document D calculate h(D)
- Encrypt h(D) (symmetric or asymmetric) to some value S
- Transfer D and the encrypted hash value S
- Checking the signature:
  The recipient calculates h(D') for the received document D' and decrypts the received value S' to compare both values.
  equal => message transferred not modified
  unequal => message got modified

Some further remarks:

- If the encryption is done asymmetrically with the entity's private key, everybody having the public key can check the signature. Moreover it is shown, that only the private key holder can be the creator of the signature.
- If is it done symmetrically, only an entity knowing the secret key can check the signature (although everybody can read the document).

Another application is to provide time stamping services.

For a context, where an entitiy wants to have a proof of possession of a document without revealing to the public, it can use a time stamping service, which works like the following:

- Send h(D) of your document to the trusted time stamping service
- The service adds date+time and encrypts everything with its private key to get some value S
- Now if somebody wants to check the time of creation, he simply decrypts S with the public key of the service, checks whether the calculated h(D') equals the value in S and can then trust the given time (as long as he trusts the service!)
  REMARK: Time stamper doesn't know D !

In this module, we will discuss

- basic requirements for public key infrastructures
- trust models & components of a PKI
- PKI standards

The use of open and wide-spread networks offer a big number of new possibilities to the users.

"On the Internet, nobody knows you're a dog."

In today's e-business environment it is necessary to be able to establish a secure communication with an organisation in any part of the world.

Maybe

- one has never had contact before
- one is not able to visit the organisation personally

So exchanging keys is a difficult task, which directly leads to the need for a trusted third party (TTP).

- Small exercise: think about who is predestinated to be a trusted third party

As a result, symmetric cryptography is less useful, because the third party would have to protect confidentiality for the whole transmission of the key.

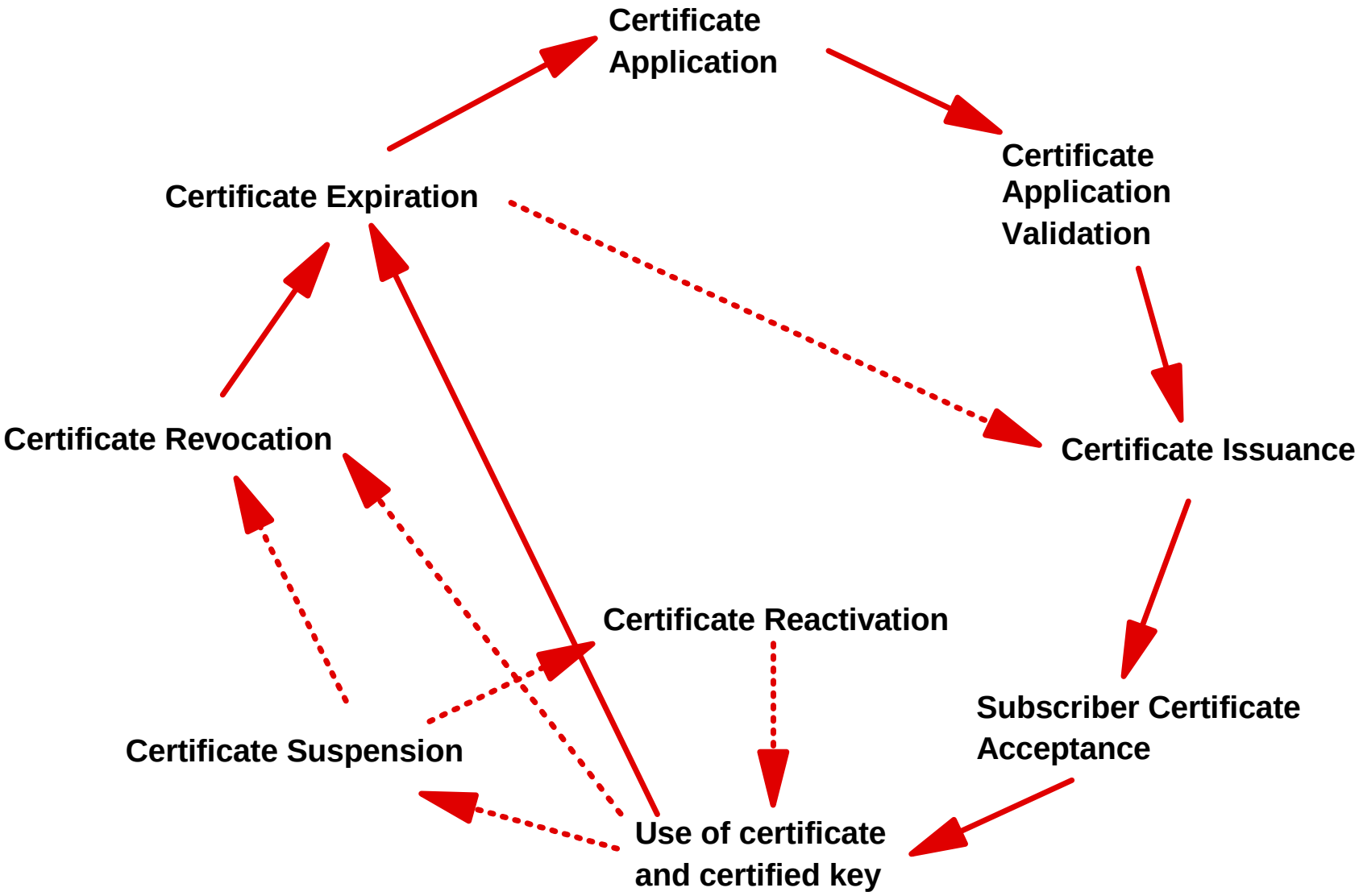
Transferring a public key only requires integrity, which can be provided easier.

As discussed in the preceding module, the main problem of public key cryptography is to be sure about the person holding the corr. private key.

The mainly used practical solution is the use of certificates.

- A certificate confirms that the person having the private key corresponding to the public key in the certificate is the person provided.
- A certificate is usually issued by a third party
  - **you have to trust the third party, i.e. you have to trust in the way they issue certificates, i.e. their certification process**
  - **e.g. PostIdent (TeleSec) does the job for some financial institutions**
- A certificate is signed by the issuer of it. Everybody holding the corr. public key can verify it. Therefore it's important that every user has a correct public key, and that the corresponding private key is highly protected.
  - **freaky idea: public key is transmitted every day in the news on the radio on all stations**
  - **usually one can download the public key from web sites of the organisation**
    - **hopefully adequately secured**
  - **private key only on not-connected systems**
- On the other hand, certificates can be revoked due to expiration, or loss or disclosure of the private key
  - **this makes it necessary to have a component where one can check whether a certificate is revoked or check certificates online**

Moreover, one has to take into account, that a certificate has to be managed throughout its whole life cycle.



From this one can derive the basic components a PKI needs.

Components needed:

- issuing certificates: Certification Authority (CA)
- providing Certificates Revocation List (CRL) and validation of certificates: Directory Services (DIR)
- component handling certification request, revocation requests, storing certificates etc: Registration Authority (RA)

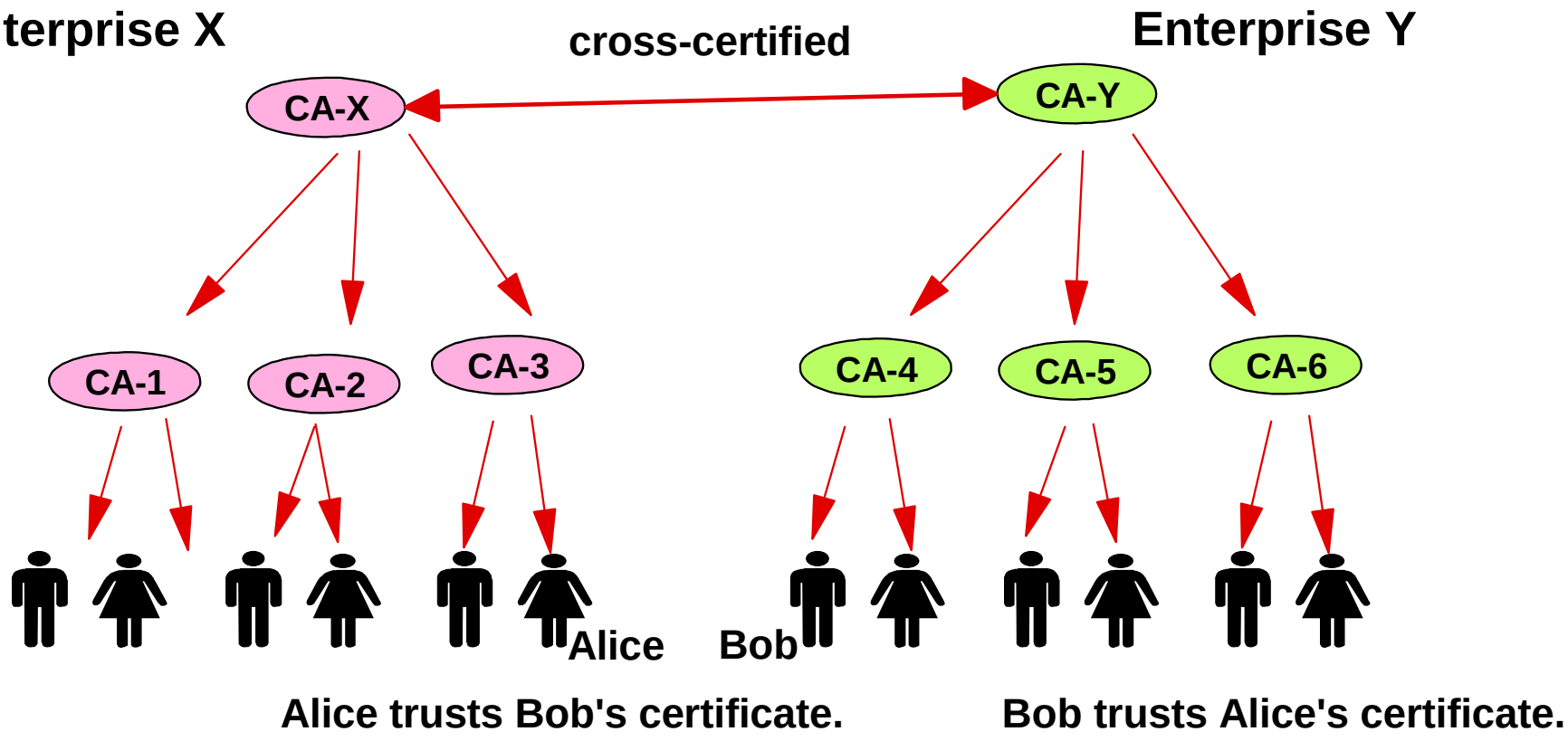
In the trust model used by PGP ("web of trust") this is all done by the individual user.

Another trustmodel is to have independent technical realisations of these components with defined information exchange.

More complex trust models allow layering different CAs hierarchically where each CA signs the certificates of all subsequent CAs.

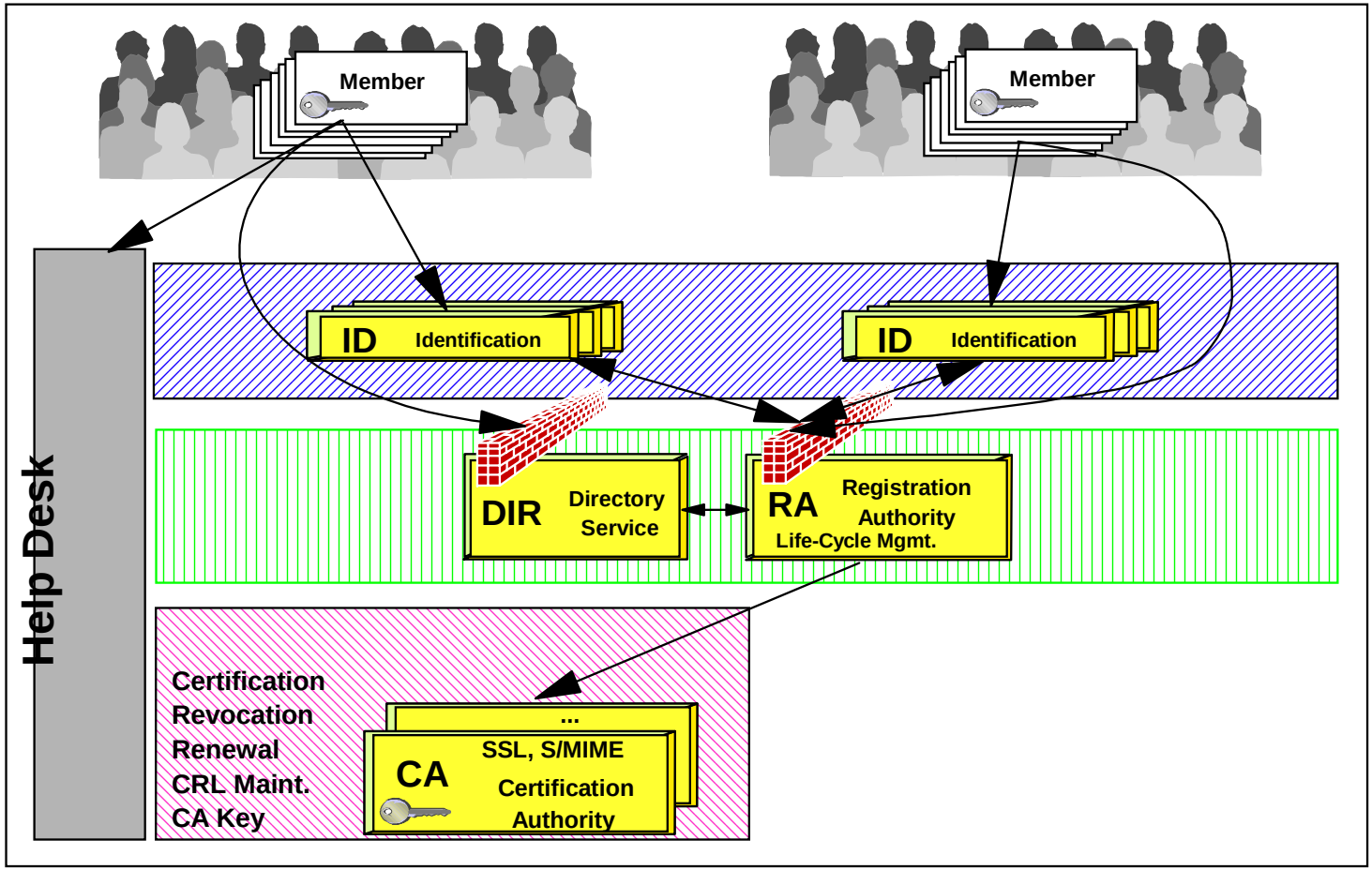
Another idea is to connect to CAs via cross-certification.

With this, two established CAs can extend the usability to the whole set of users.



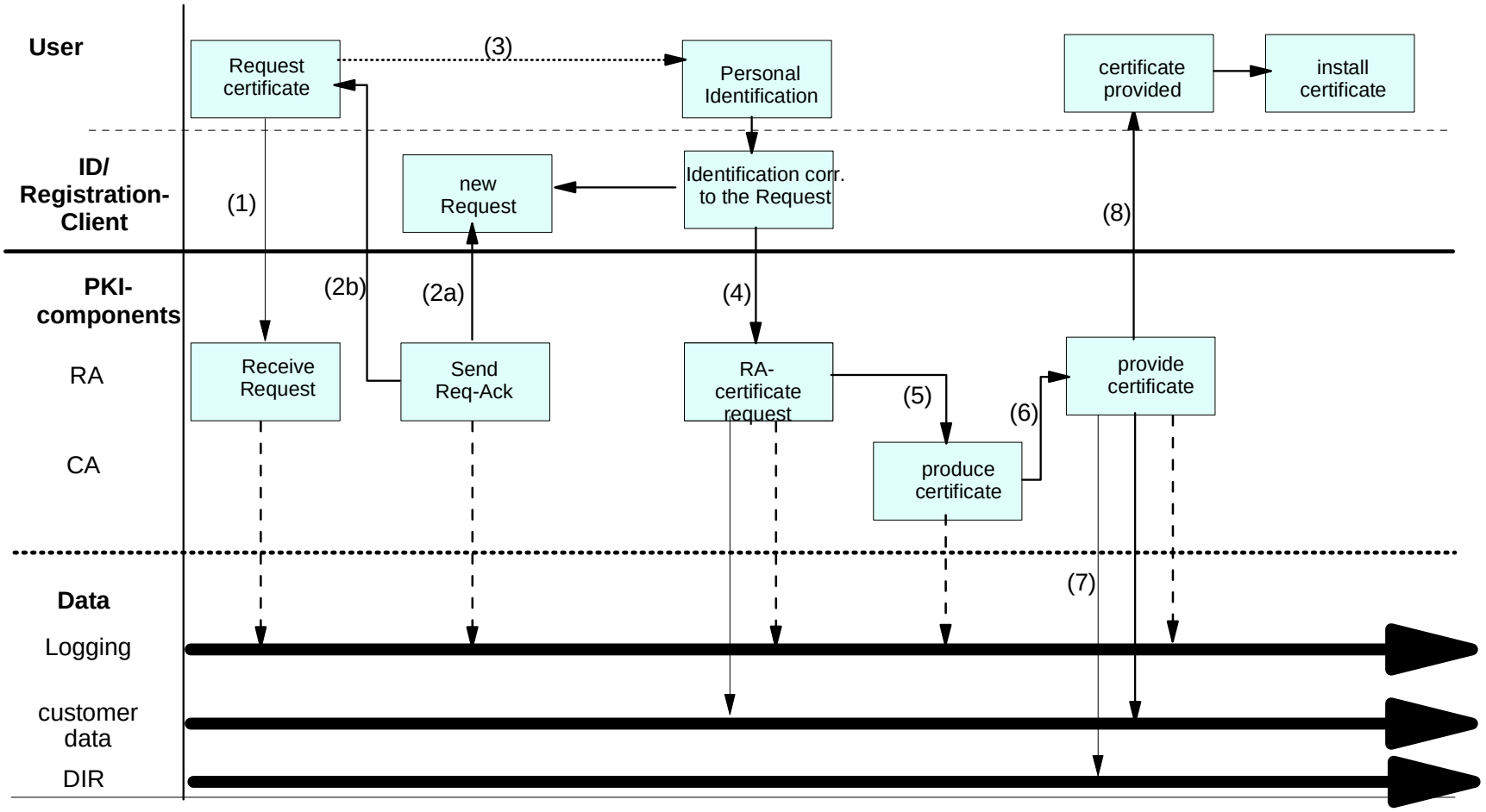
For building up a PKI one has a lot of options concerning the architecture of the system.

Remark: ID is the component the end user is talking to for requesting a certificate and identifying himself.



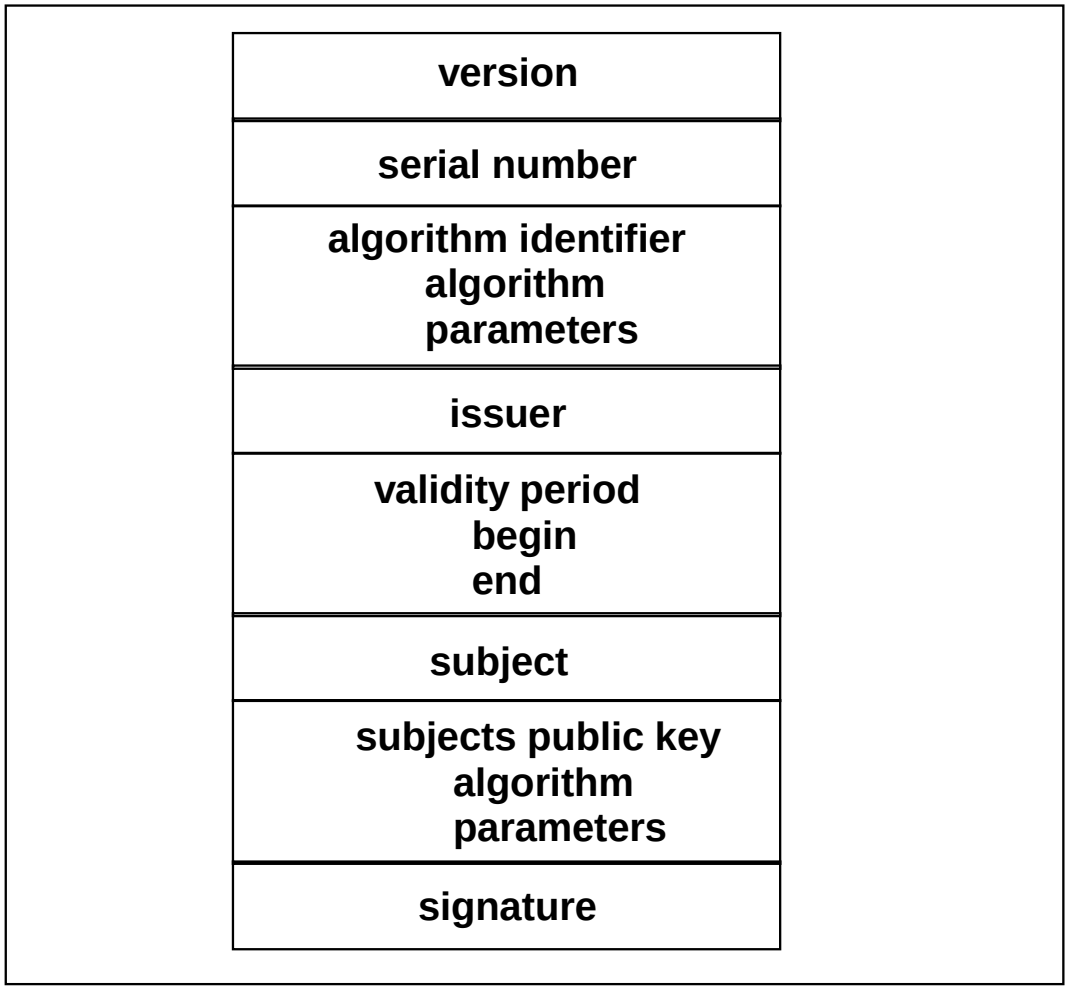
Also, policies and processes offer a lot of options, which must be taken according to the business requirements.

Example: Certificate request



The data structure for certificates has grown to a more an more general standard.

Certificates are mainly build according to X509.



Structure of
an X509 certificate

So far, only few other specifications are standard - mainly protocols connecting the different components and the certificates and data formats.

Standards:

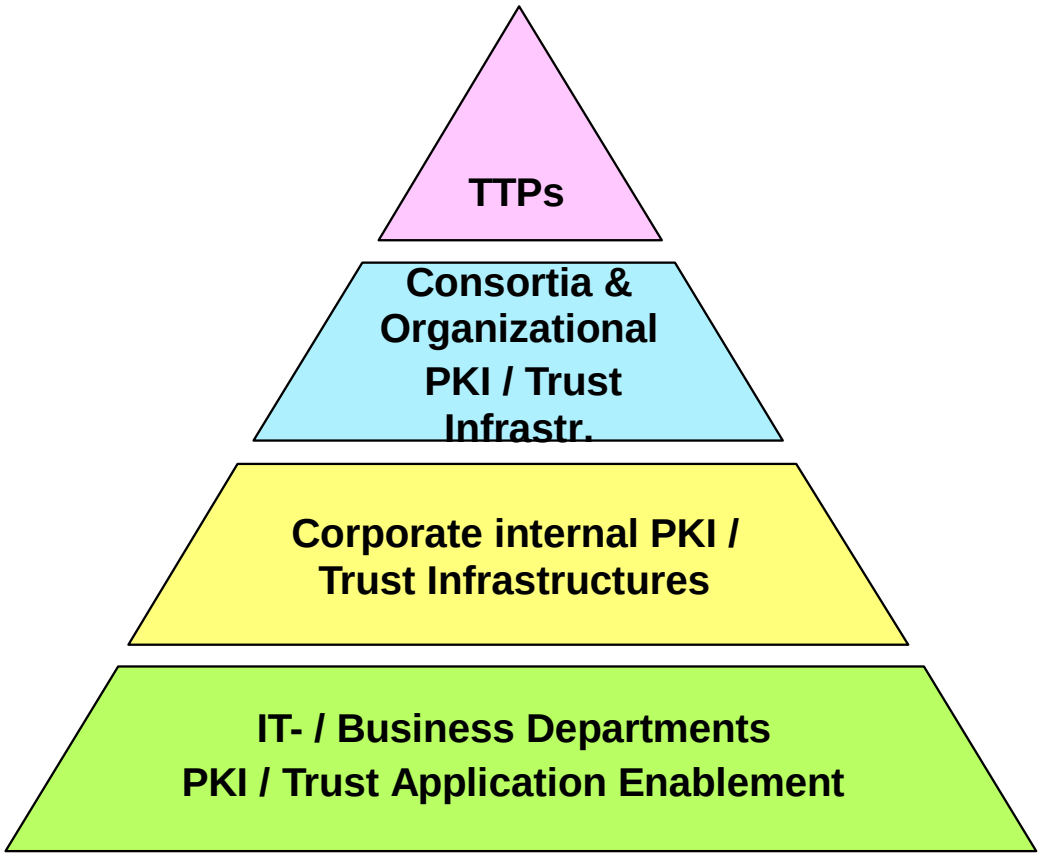
- PublicKeyCryptographicStandards (PKCS)
  - **Developed by RSA Security**
  - **Some reached RFC level, like**
    - **PKCS#1 - RFC 2437 defines RSA cryptography**
      The standard defines cryptographic primitives, encryption and signature schemes and a syntax for RSA keys
    - **PKCS#5 - RFC 2898**
      This standard defines key derivation functions and encryption schemes.
    - **PKCS#7 - RFC 2315 describes message syntax**
      This standard describes the syntax for encrypted or digitally signed data
    - **PKCS#9 - RFC 2985 defines auxiliary objects classes.**
    - **PKCS#10 - RFC 2986 defines a certificate request syntax.**
  - **other PKCS#**
    - **PKCS#11 defines cryptographic token interfaces called "cryptoki"**
- standard from the IETF-PKIX working group
  - **Certificate Managment Protocol (CMP)**
    - **specifies a protocol for communication of the system components of a PKI**
  - **Online Certificate Status Protocol (OCSP) - RFC 2560**
    - **protocol to determine the status of a given certificate, specifying request and response**

The only more precise specification of a complete system is given by the German digital signature law.

The first version of the law came in 1997. With this, Germany was the first country to establish such a law.

- Unfortunately, there was not much use of it, because the restriction were much too strong to be useful.
  - **certificates only for natural persons**
  - **offline CA**
  - **time-stamping service required**
  - **certificates only in SmartCards**
  - **...**
- The new version of this law (in the context of the EU directive) lowered the requirements, but still sees to be too restrictive.
  - **Qualified and non-qualified certificates**
  - **Issuer of qualified certificates subject to supervision through RegTP**
  - **Certificates only for natural persons**
  - **Digital signature equated with usual signatures**
  - **Liability provision of 500.000,- DM per harm through failure postulated**
- Additionally the new law comes with a modification of the supplementing by-laws.

From a business point of view there are many different drivers to set up a PKI.



- **Trusted 3rd Party Service Provider (TTPs) who want to sell Security and Trust Services to other Customers**
- **Organizations who want to participate in Cross-Organizational and/or Consortium driven PKI / Trust Infrastructures**
- **IT- / Business Departments of Organizations who want to build an internal PKI / Trust Infrastructure on a Department or Corporate wide Level**
- **IT- / Business Departments with specific Requirements for Secure & Trusted e-business Applications**

Issuing qualified certificates has been discovered to be a business, but so far none of the competitors seems to be extensively successful with it.

- European TTPs ...
  - DE: TC-TrustCenter (www.trustcenter.de)
  - DE: Dt. Telekom (www.telesec.de)
  - DE: Deutsche Post (www.signtrust.de)
  - AT: Datakom (www.a-sign.at) ***done by IBM***
  - SP: SiCer (www.siscer.com)
  - NL: NLSign (www.nlsign.nl)
  - GlobalSign (www.globalsign.net)
  - IT: Telecom Italia (security.tin.it)
  - IT: SSB (ca.ssb.net)
  - IR: Baltimore/CyberTrust (www.baltimore.com)
  - DK: TeleDanmark (www.certifikat.dk)
  - UK: BT TrustWise (www.bttrustwise.com)

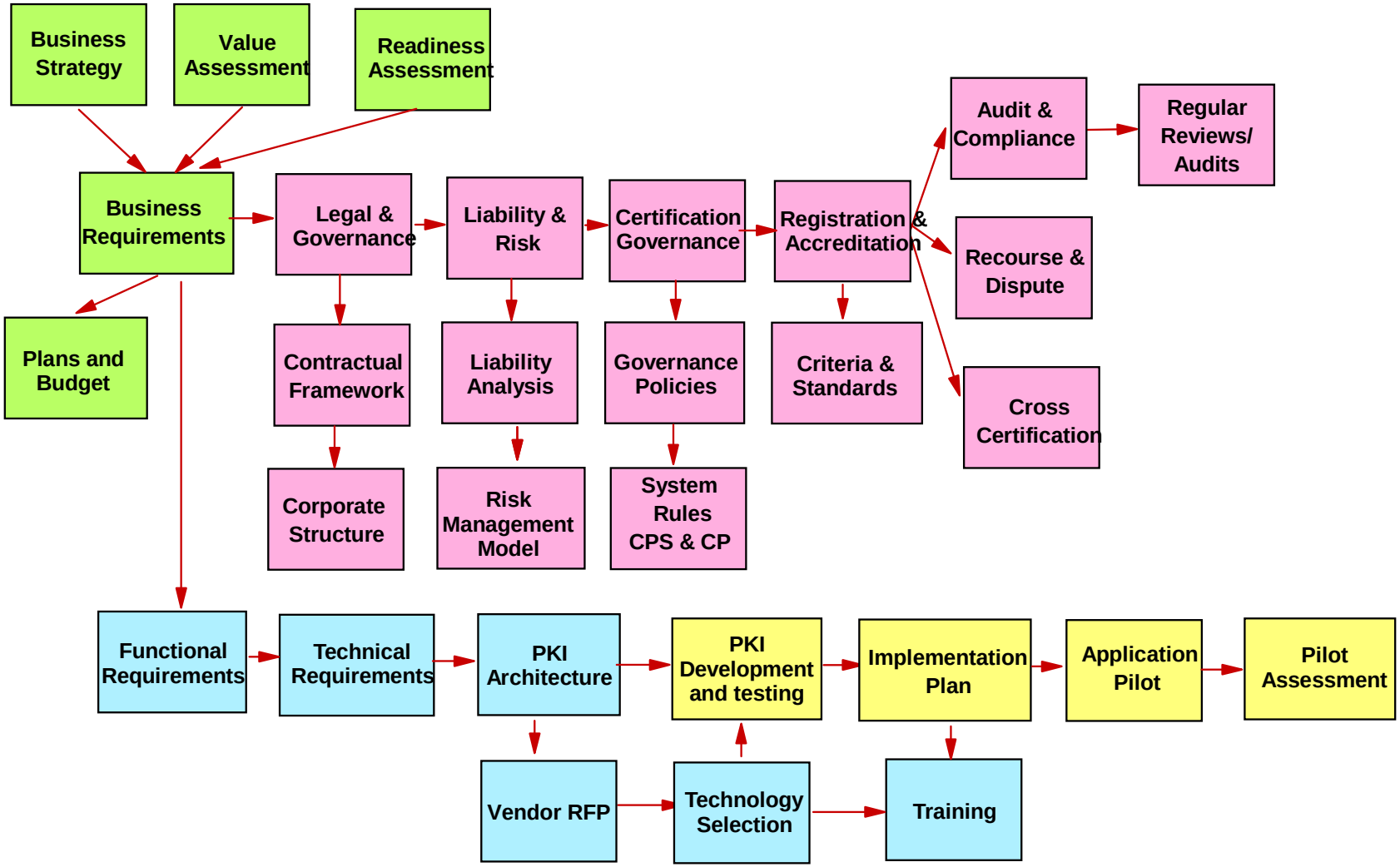
Certificaat

Also, a lot of companies offer PKI products. But establishing a PKI is not done via simply buying a product and installing it.

Once again, there are many difficult questions to answer

- certificate policies
- integration of the PKI into the applications
- processes around the PKI
- outsourcing the CA or the RA or the helpdesk or...
- scalable architecture
- rollout for 100, 1000, 10000, 100000+ customers
- software certificates, SmardCards, ServerCertificates, HSMs..
- costs & benefits
- ...

Building up a PKI is a complex task, for which a good understanding is needed of what the business requirements are.



End of module.

"On the Internet, nobody knows you're a dog."

... with PKI you know who is the dog ...

Wireless Security

In this module, we will discuss

- where to use mobile and wireless techniques
- some basic mobile and wireless protocols
- in how far security is different in contrast to traditional connections

Mobile Commerce - some facts, visions and definitions.

Any Transaction with monetary value conducted via mobile telco network (Durlacher)

”…a billion people interacting with a million e-businesses with a trillion intelligent devices interconnected ..."
(L.V. Gerstner)

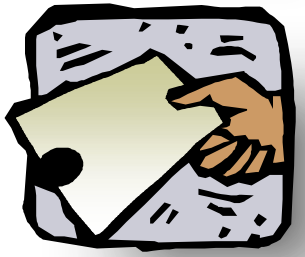
estimated mCommerce market europe ´03:
23 Mrd. € (Durlacher)

More handsets than PCs connected to the Internet by the end of 2003! (MeT)

Some examples for mobile applications.

"Ericsson estimates that by 2004 there will be around one billion users of mobile telephony and some 600 million mobile Internet subscribers worldwide. The most important thing that is needed to get all these consumers to start using mobile e-commerce is a standard, which makes it safe and easy to use."

- **Mobile Messaging**
- **Mobile Banking**
- **Mobile Trading**
- **Mobile Ticketing**
- **Mobile Betting**
- **Mobile Shopping**



Below you find security issues which have to be mapped to mobile business.

Origin: existing Internet Security issues, tools and standards

• **Authentication**	"Is this person who he says he is ?"	➡ **Digital Certificates,Validation Services**
• **Confidentiality**	"Is any personal information I give out being compromised ?"	➡ **Symmetric & Asymmetric Encryption**
• **Integrity**	"Am I confident that the data I receive and send is not being tampered with ?"	➡ **Hash-Algorythms, Digital Signatures**
• **Non-Repudiation**	"How can I ensure that the data was received, signed for and time stamped ? Will it stand up in court ?"	➡ **Digital Signatures, Timestamping**
• **Authorization**	"Is this person authorized to access a specific Application ?"	➡ **Authorization Services based on the PKI Authentication**
• **Auditability**	"Who was involved in a Transaction and when did it take place ? What was the nature and the result of the Transaction ?"	➡ **Logging and Archiving based on PKI Authentication and Non-Repudiation**

➡ why not simply adopt these solutions to the mobile world ?

Mobile security has special requirements because of physical limitations and incomplete standards.

Physical limitations

- Transmission bandwidth
- Session frequently interrupted
- High network latency
- Low processor power
- Limited power consumption
- Limited memory
- Thin operating systems
- User interfaces
- Insecure persistent storage
- Lack of secure crypto adapter

Increased complexity

- Geography differences
- Devices
- Protocols
- Customer requirements
- Scale

Incomplete standards

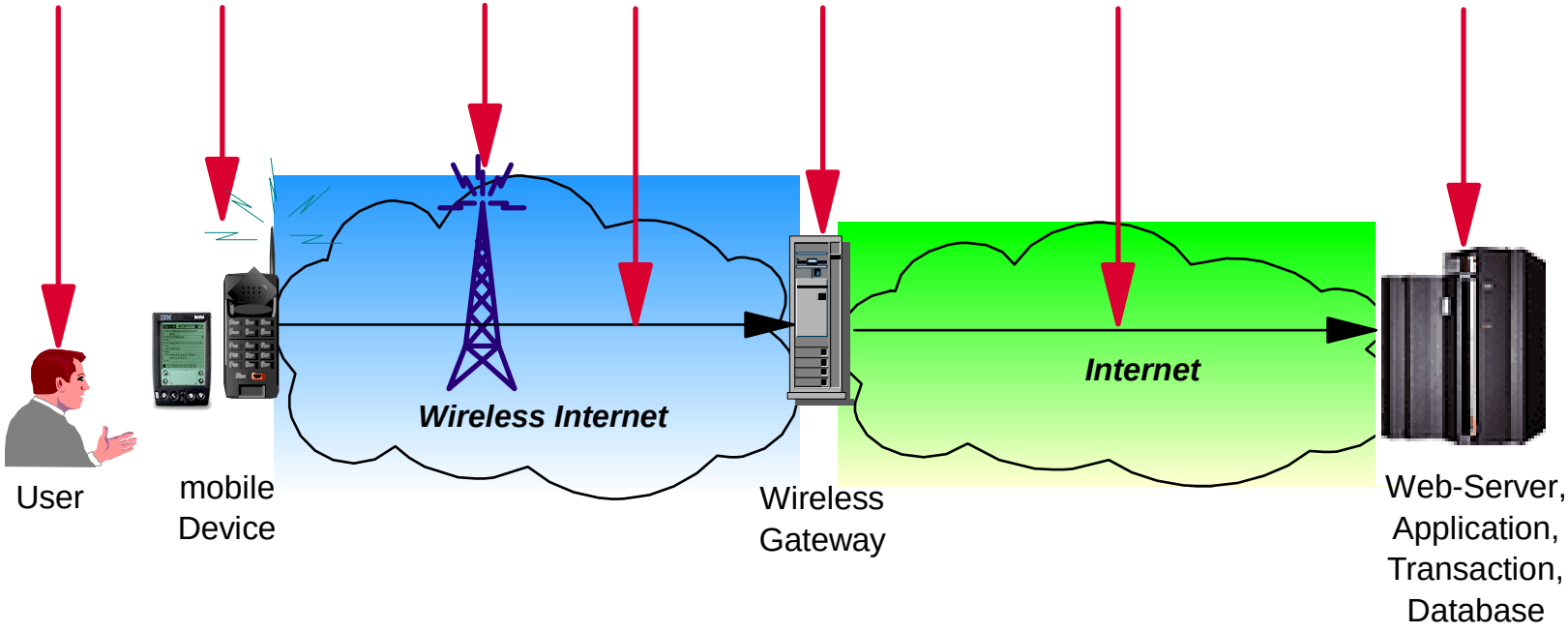
- "End to End" security
- Public Key Infrastructure (PKI)

➡ special requirements to mobile security

There are many different transfer mode systems and protocols.

- Wireless keyboards and mouse
- Bluetooth
- Wireless LAN (IEEE802.11b) as basic for TCP/IP
- Cell phone protocols like:
  - GSM, HSCSD (Highspeed Circuit Speed Data), GPRS (General Packet Radio Service), UMTS
  - are used to transfer e.g. SMS, WAP, i-mode, ...
  - WAP is seen as a key-technologie for mobile commerce
  - GSM, GPRS, Bluetooth, ... could be used as carrier for WAP-Applications

Mobile and wireless systems contain new risks and weaknesses.



Security for wireless applications poses new challenges - securing data and transactions from the wireless device, through the air, through the carrier links, over the Internet, and onto the protected server involves bringing together security issues at multiple levels. The issue arise from the users and devices themselves, at the network level and application level, and across the enterprise.

For example: mobile devices are small and highly portable - that makes them easy to carry but also easy to misuse. Because of their mobility and portability, the devices could easily be theft, break ore used for access by unauthorised people.

Some technical informations on Bluetooth and IEEE 802b.

Bluetooth

- range ~ 10m
- unique ID for each device (48bit), private key (authentication, 128Bit), private encryption key (4-128bit)
- trusted / untrusted device

IEEE 802.11b

- WEP - Wired Equivalent Privacy
- WEP has unfortunately been compromised (August 2001)
- MAC-address filtering
- Encryption

IEEE 802.11b has unfortunately been compromised.

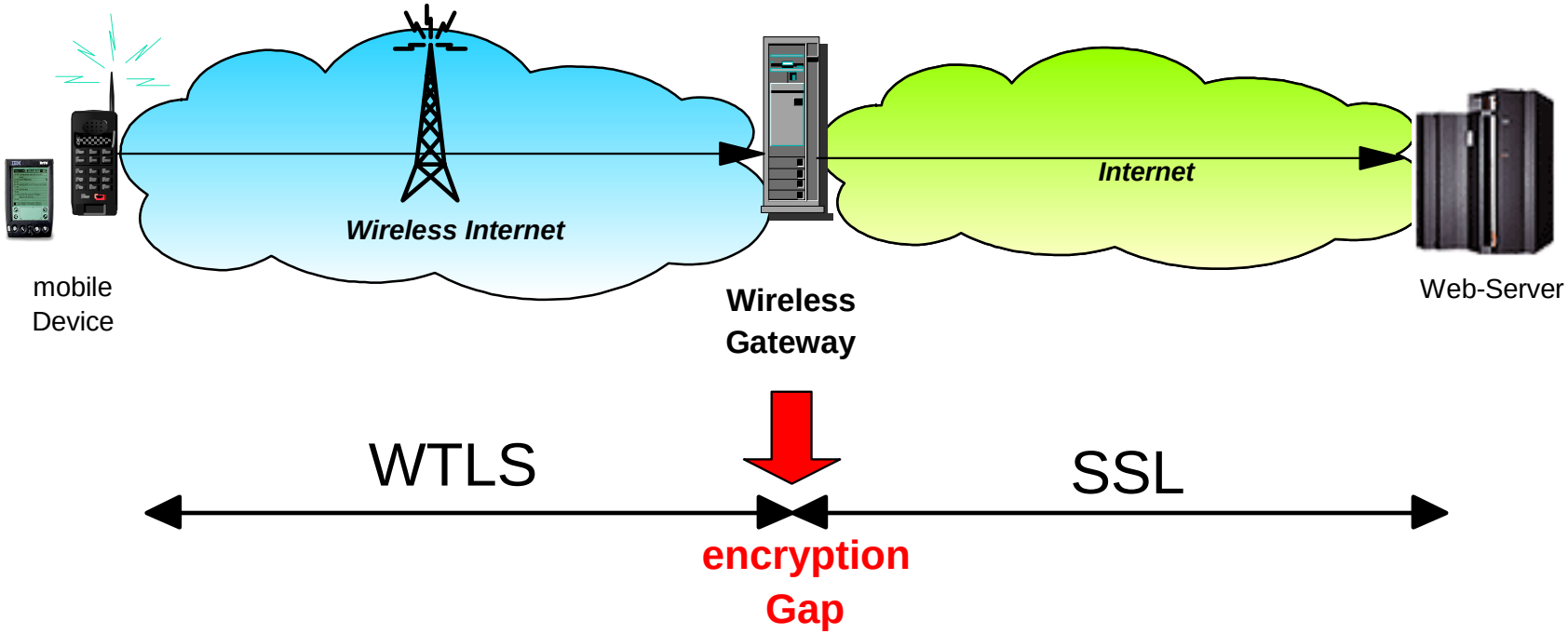
- IEEE 802.11b Hacking: Wireless LANs can easily be accessed because security is often turned off. Just pass an office building with a laptop and 802.11 card and you are in ("war driving").

- IEEE 802.11b has been compromised. A software called AirSnort (running with Linux kernel 2.4) can decrypt the communication easily due to a problem with the RC4 algorithm. See also: http://airsnort.sourceforge.net/

There are a variety of wireless security technologies for different needs.

- SIM *Subscriber Identity Module*
- WIM *Wireless Identity Module*
- WMLScript Crypto API *for digital signatures*
- WTLS *the equivalence to SSL*
  - *3 Classes of Authentication in WTLS*
  - *Class I - Anonymous - No Authentication*
  - *Class II - Server Authentication only*
  - *Class III - Client & Server Authentication*
- WPKI *Wireless Public Key Infrastructure*

The famous "GAP" in WAP.



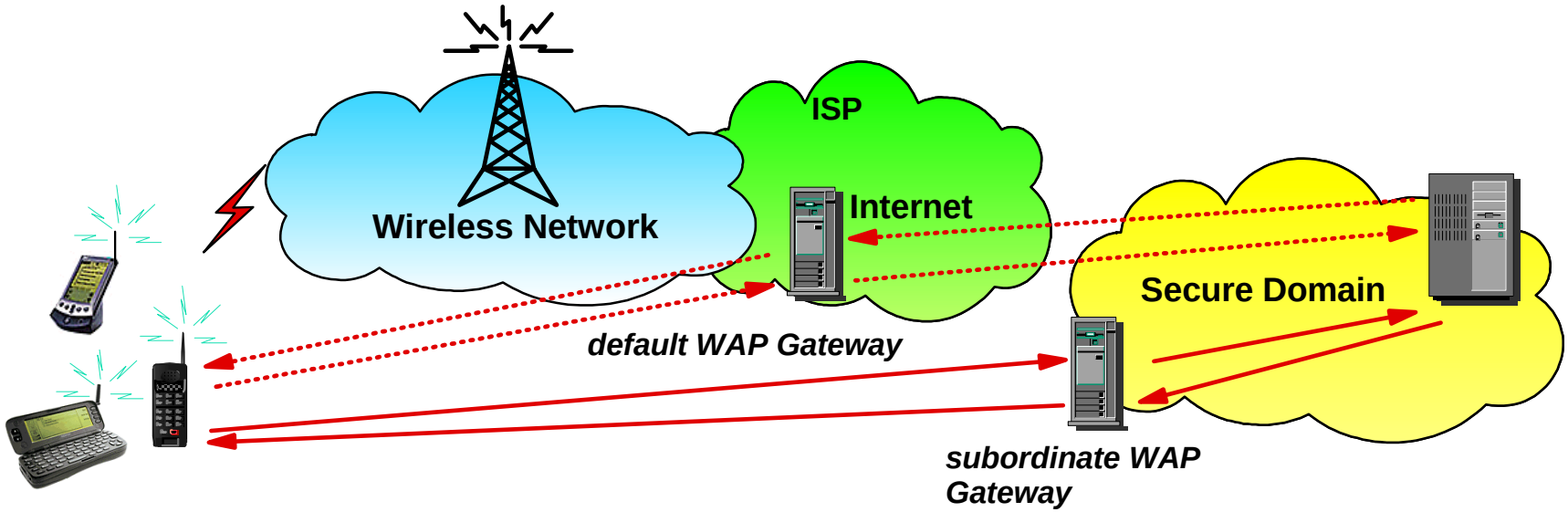
Due to the difference between WTLS and SSL, encryption in wireless security is solved in two steps:

- first, from the mobile device to the Gateway the data is encrypted with WTLS
- then the transmission briefly becomes decrypted inside the gateway
- and afterwards reencrypted with SSL

➡ This is not a full End 2 End Security - It's "End 2 Gateway 2 End"

One solution can be "Gateway Redirection".

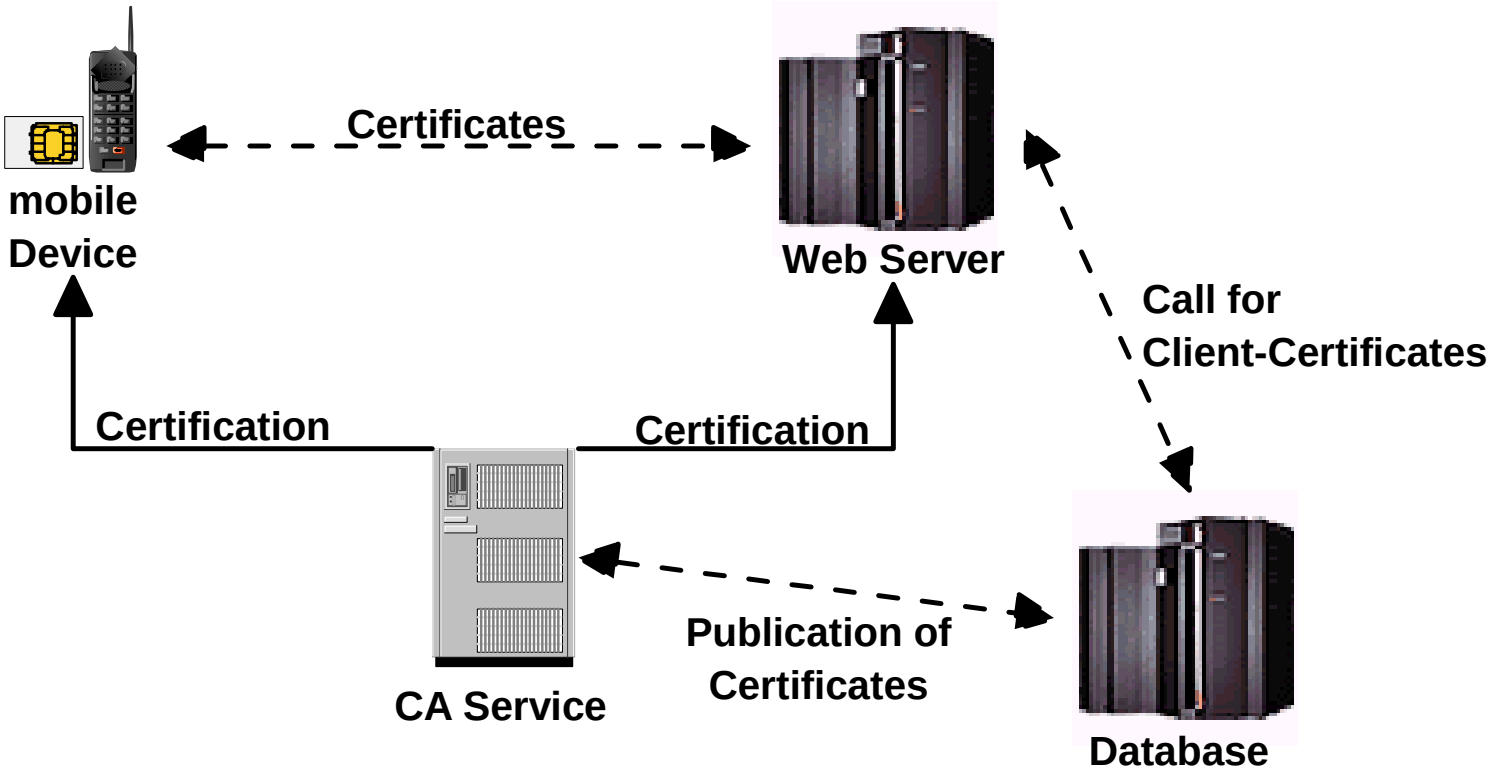
WAP End 2 End security today requires some extensions to the standards and a second Gateway within the secured and trusted domain. The new specification does not change the general problem, but allows the sensitive operation to take place within a customer's secure zone as opposed to at the ISP.



- The newly approved WAP specification for "end-to end" security makes use of redirection to a gateway which resides within a secure domain
- The WAP client attempts to send a transaction through its default gateway to a secure domain
- The secure content server determines that the transaction must arrive through the WAP gateway in its domain and returns a HTTP redirect message
- The default gateway validates the redirect and transmits it to the client
- The client then caches the new connection route and transmits transactions destined for the secure domain to the subordinate WAP gateway
- After the connection is terminated the default gateway is reselected

A Wireless PKI consists of the same elements as a common PKI system with some differences.

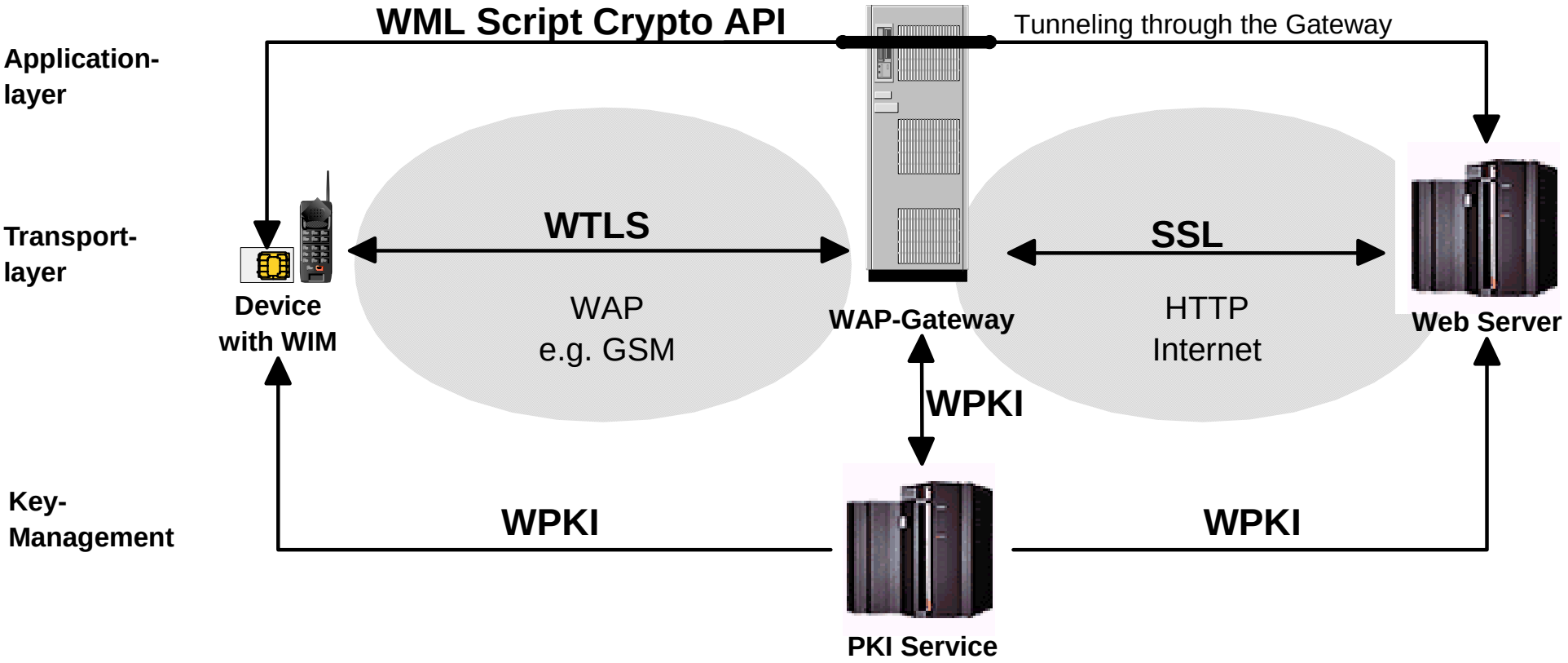
There is especially one difference concerning the client´s certificate. Due to the physical limitations of a mobile device, the client sends a URL instead of the certificate itself. Then the Server has to call for the certificate on a database server.



The goal of WAP PKI is to reuse existing PKI standards.

The goal of the WAP PKI is to reuse existing PKI standards where available, and only develop new standards where necessary to support the specific requirements of WAP (WAP Forum).

In WAP a WPKI could be built with the standardized elements WTLS, WML Script Sign, WIM and a PKI Service.



Some Wireless / Mobile Security Organizations and Initiatives

MeT - Mobile Electronic Transactions

Initiative between the three leading phone manufacturers Nokia, Ericcson and Motorola will support development in this crucial area of m-commerce. The aim is to develop an open specification for encryption and authentication over mobile phones. MeT will cover all core mobile technologies including current standards such as the Wireless Application Protocol's security layer, and the Wireless Identification Modules (WIMs) that will start to replace SIM cards from WAP Version 1.2 on.

Mobey (www.mobey.org)

Worlds´s leading on-line financial institutions and the leading companies in mobile Internet technologies such as WAP. The leading mobile phone manufacturers Ericsson, Motorola and Nokia acknowledge that the Mobey Forum will play a valuable part in the development of online wireless financial services.

Raddichio (www.raddicchio.org)

Global initiative to define a standard security platform for mobile e-commerce using Wireless PKI (Public Key Infrastructure). The initiative was formed by SONERA, GEMPLUS and EDS and is joined by more than 50 companies worldwide.

MoSign and mSign

Are both independent initiatives of Vendors and Organizations that develop products, applications, and solutions for the mobile phone market as well as for the Internet. Among these are Mobile Phone Producers, Smart Card Producers, PKI Provider, and Mobile Operators. Both have the aim to establish a mobile electronic signature API to put digital signature capabilities on every new SIM-Card in the future.

Outlook

Mobile Security today:

- technical limitations
- the "Gateway - Problem"
- no common standards
- no real End-to-End Security => "Best-Practice-Solutions"

Mobile Security tomorrow:

- WTLS/SSL -> TLS as common standard (wired and wireless)
- "One big PKI" ?
- End-to-End Security