Security in Distributed Systems Part One

Distributed Attack Vectors and Security Technology

Distributed Security is easy….

In a way one could say that distributed security is easy: find all places where TRUST is assumed and control/fix them.

But you must keep systems workable, stay within costs, account for user’s weaknesses etc.

At the core of distributed security there is always TRUST and the way it is established, verified and maintained through systems of independently interacting agents.

Overview

- Threats and Attacks due to Distribution
- Crypto Basics and Building Blocks
- Security Mantras: Authentication, Authorization, Integrity, Confidentiality, Non-Repudiation
- Security Mechanisms and Protocols, Channel- and Object-Security

Next session:

- Security Architectures, Middleware and Infrastructure
- Delegation concepts

Distributed Systems Security Threats

- **Interception (eavesdropping)**
- **Interruption (denial of service)**
- **Modification (illegal data change)**
- **Fabrication (replay attacks)**
- **Destruction (sabotage)**

- **Hostile Clients**
- **Attacks on Clients (indirect attacks)**

- **Internal Attacks**

The “remoteness” causes problems during transit but also difficulties in trust relationships and through the number of involved parties (abilities etc.)

Ways to attack

- Company internal attacks
- Password cracking
- Encryption cracking
- Denial of service
- Replay attacks
- Man in the middle
- Planting viruses (trojan horses)
- Masquerading
- Exploiting software bugs
- Social engineering
- Semantic Attacks

Where to attack (1) ?

“82 % of all identified frauds were committed by employees, almost of third of which were by management. Nearly half had been with the organization for more than five years and almost a quarter for more then ten years.”

Source: Ernst&Young Fraud Investigation Group, Report 2000.

It’s not “Only the Intranet” even though you hear this all the time within companies. For a security architecture of an Intranet: Frederick Thomas Martin, Top Secret Intranet – How US Intelligence built Intelink – The worlds largest most secure network. BUT: IS THIS REALLY TRUE? Where do the numbers come from?

Company internal attacks

Employed hackers

- Best knowledge of procedures and infrastructure
- Best knowledge of the value of items
- Legal access to information
- Easy mobile storage
- Wireless access

company

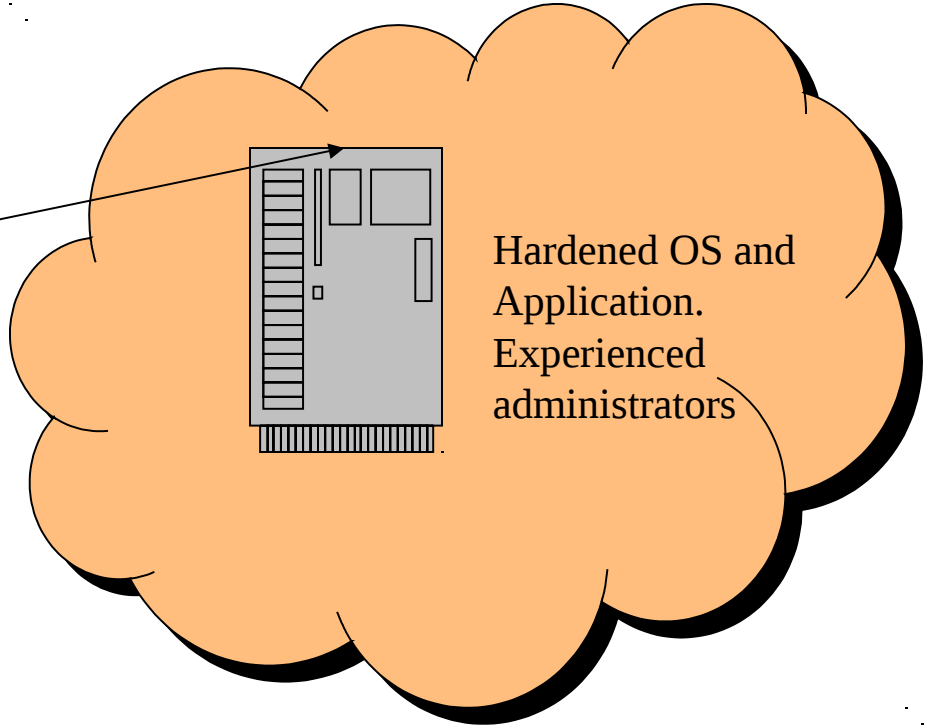
- Know what you need principle
- Role based access control
- Strong authentication
- No clear-text passwords over the wire
- Clear-desk policy
- 4-eyes principle
- Secret service
- Permanent encryption of important information

While attacks from outside hackers always win a lot of attention, the real threats come from inside. Most companies do realize this fact but act differently (e.g. “it’s ONLY for the intRAnet”)

External attacks

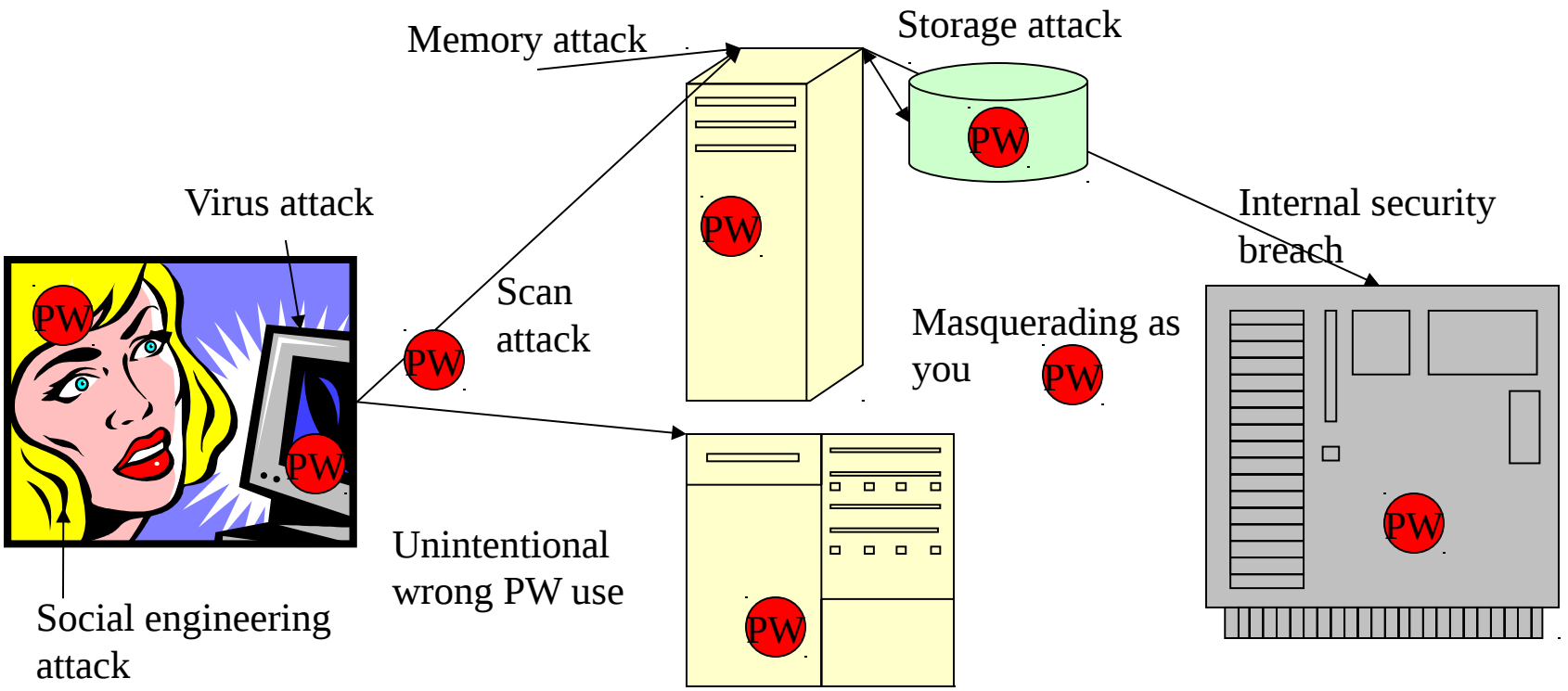
Companies clients

PC with
unreliable OS
and User



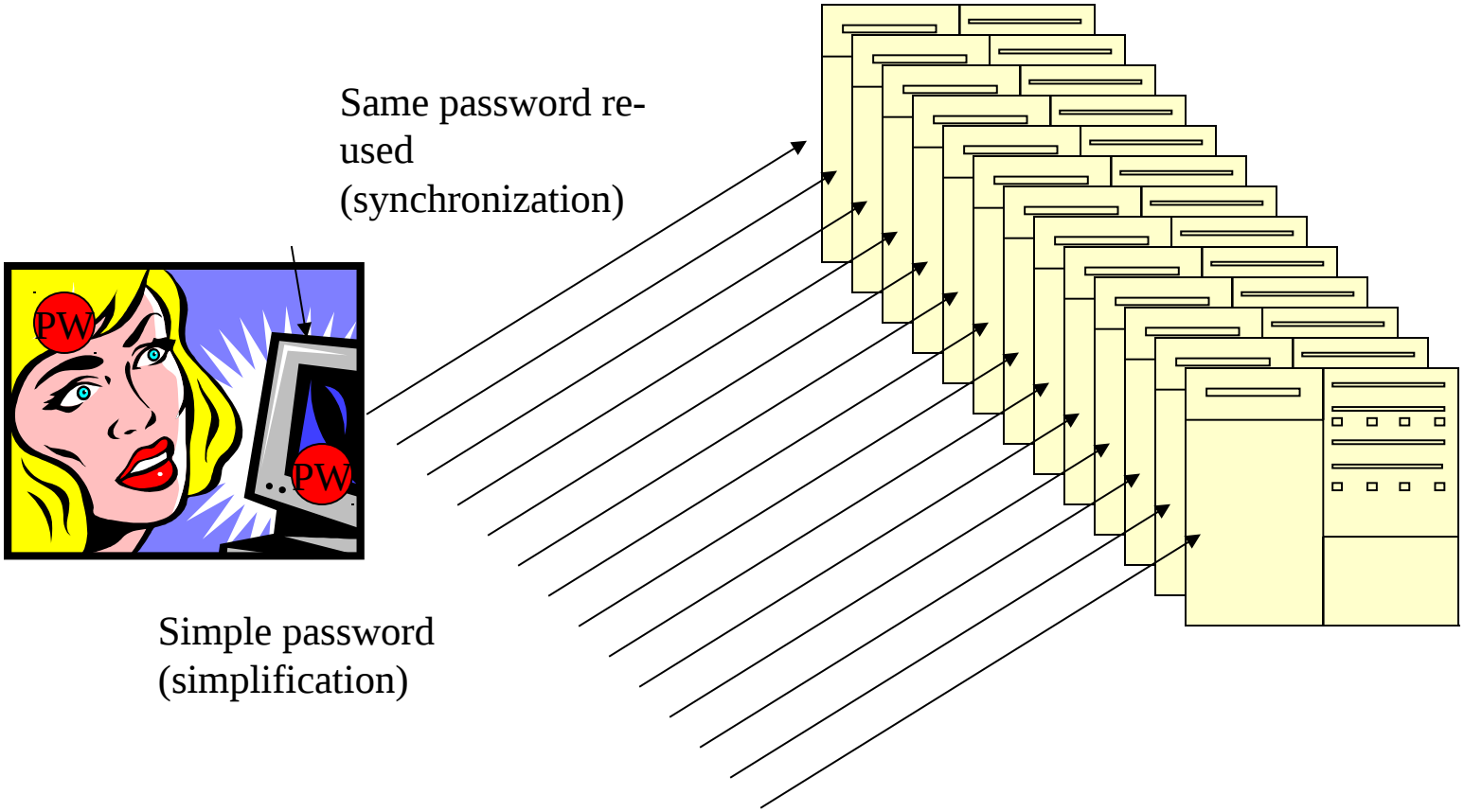
Going after the company host is much harder than attacking a companies clients at their weakly secured home computers. The advantage of hacking into a company host is that many clients could be affected all at once.

The problem of passwords (1)



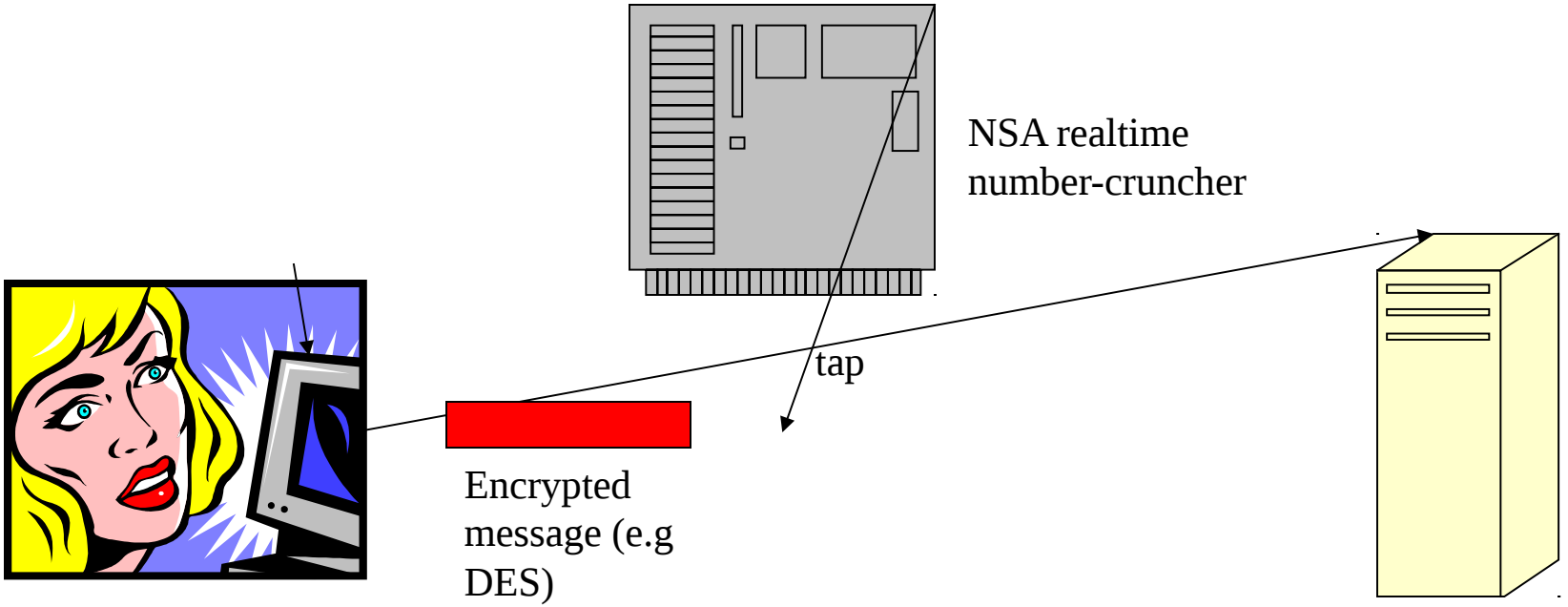
The core problem is that your secret is known by others. This make the safety of your secret (your IDENTITY) dependent on network, hosts, partners etc. To make things worse, because PW's should be long and complicated they are hard to remember. People tend to re-use them (PW synchronization) or to pick very simple ones (PW-simplification). Still, the basic problem is the fundamental exposure of your secret! That makes them fairly low risk for you actually – because everybody knows that a user-id/password combination is not safe and does NOT necessarily identify YOU.

The problem of passwords (2)



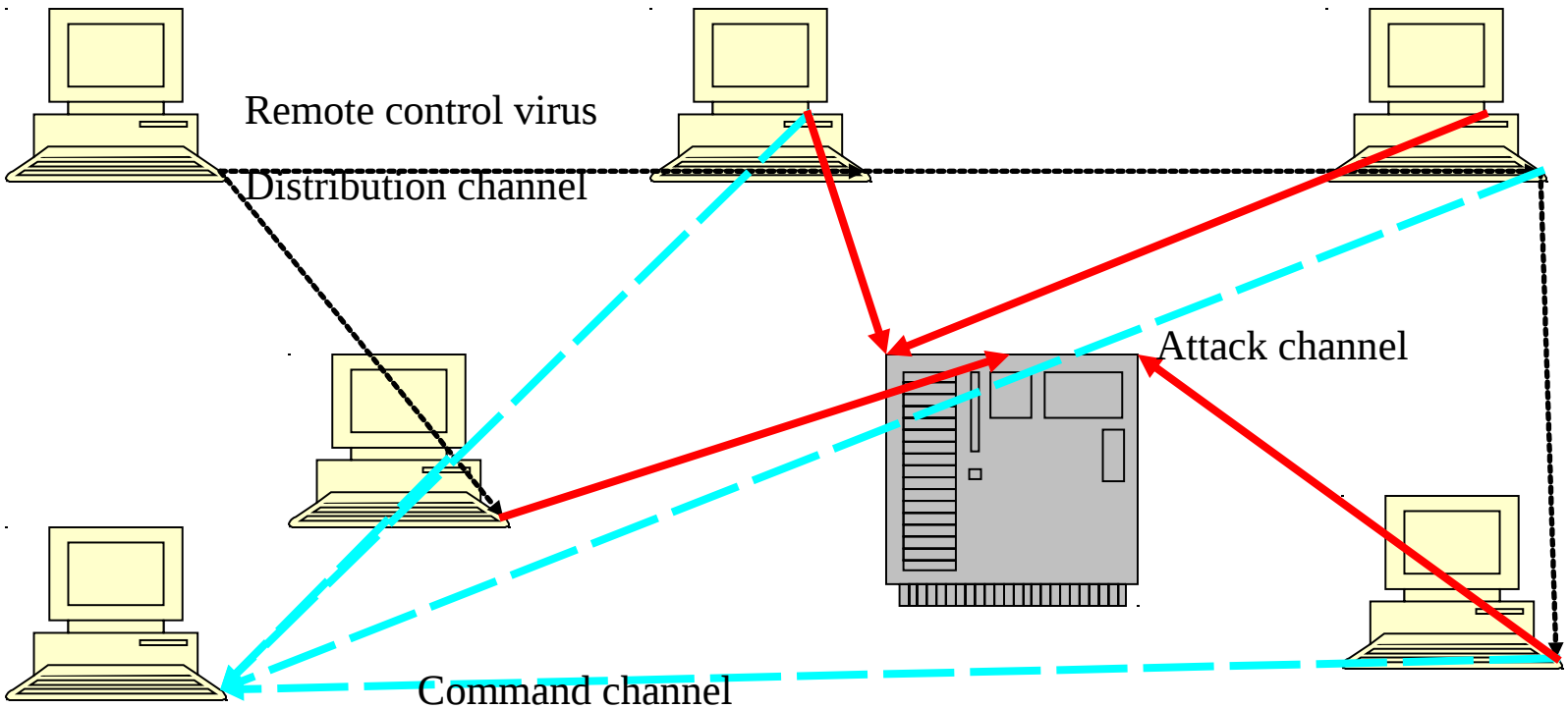
Besides shipping and exposing your password on every use the whole mechanism does not scale as well! With every new participant in a distributed system ALL participants would have to invent a new password (no re-use or synchronization) of sufficient quality (no simplification). This is a system administration nightmare (is there a sys admin at all?)

Cracking ciphers



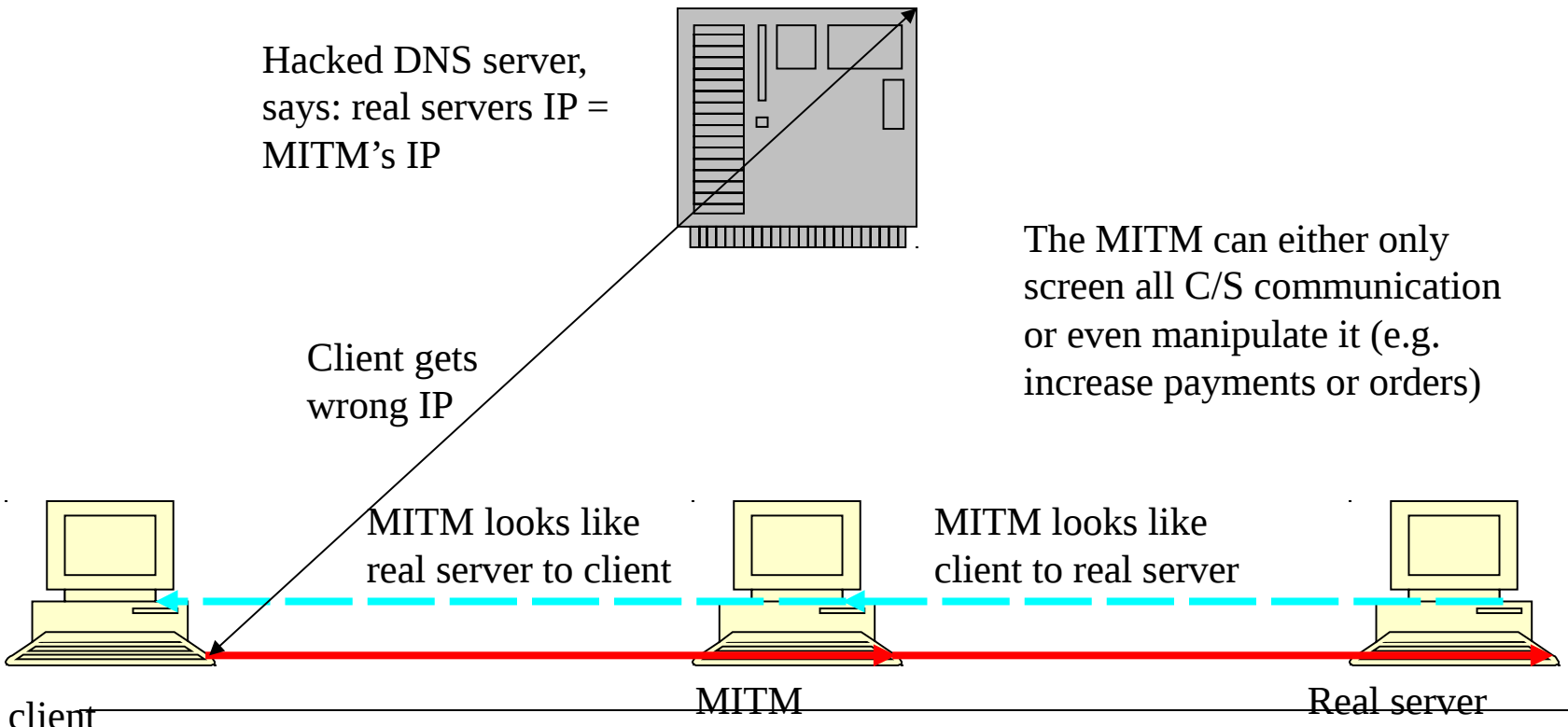
Key length and encryption technology determine the ease of cracking your cipher. Given enough time every encryption can be cracked. To determine the appropriate technique you need to consider: If one message is cracked, does it mean ALL previous messages can now be read too? How bad is that for you? If you find you (or guess) that you key has been broken – can you revoke it quickly? How will you partners find out about the broken key? Your distributed system has a HISTORY!

Distributed Denial-Of-Service Attacks



Little can be done against distributed DOS. Especially easy are attacks on session holding services. Why? Because of the peer-to-peer structure of the internet? The problem of service abuse in P2P settings is a hard one to solve. Read at www.grc.com about DDOS.

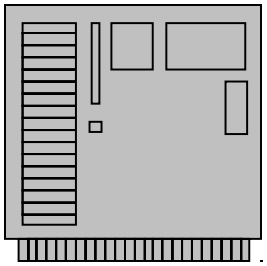
Man-In-The-Middle (Proxy?) Attacks



MITM attacks are not so uncommon. Even large banks were already attacked. How do you find out that somebody is playing a MITM attack? Strong authentication and signed requests (with nonces) fight those attacks efficiently. Does multi-factor authentication work?

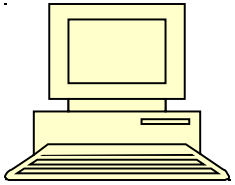
Planting Viruses and exploiting SW bugs

A top target: a big server, hard to plant the virus but THEN a lot of clients can be infected! And the modern virus is a remote-control device!



Another top target:

The typical small windows box. Open and unprotected but NOW connected via DSL or CABLE modem – high bandwidth, permanent connectivity!



The platform threat model has become a core part of distributed system security. We can no longer assume uncompromised or non-malicious partners.

Masquerading/Impersonating?

Mail from:
walter.kriha.de

Subject:xxxxx

How do you know the mail is really from me?

Nslookup foo.bar.de

192.68.100.10

How do you know the IP address is correct?

Virtually every form of identity on the internet can be spoofed and used for masquerading attacks. Somebody else is impersonating you or one of your partners. IP addresses can be faked easily and should not be used for security purposes. Use a program like PrettyGoodPrivacy (PGP) to get mail which is authentic, private and integer

Social Engineering

Hi, this is John from IT-Security. Your last password change did not work, you have a percent character in there now!

So you say you did not? According to my files it is wrong.

So you say you definitely changed it to FOO without a percent character? OK, I will check this again and let you know

Social engineering is an extremely easy and powerful way to break security. It works best in a REMOTE environment which reduces the chances of a victim to VERIFY statements and identities! Surprise factors further decrease verification options. Trust chains lead to skipped verification of identities.

Crypto-Basics and Building Blocks

- The crypto community: Open Source forever!
- Symmetric keys
- Asymmetric (Public) Keys
- One-way hash codes
- Digital signatures
- Steganography

Principles of the crypto community

- An algorithm must be available in source (Implementation)
- An algorithm must be secure even if attackers know how it works
- An algorithm must be secure even if the input text and the associated cypher text are known (brute force attack) and can be chosen at will.
- The strength of an encryption algorithm must be in the length of its key.

As a system architect:

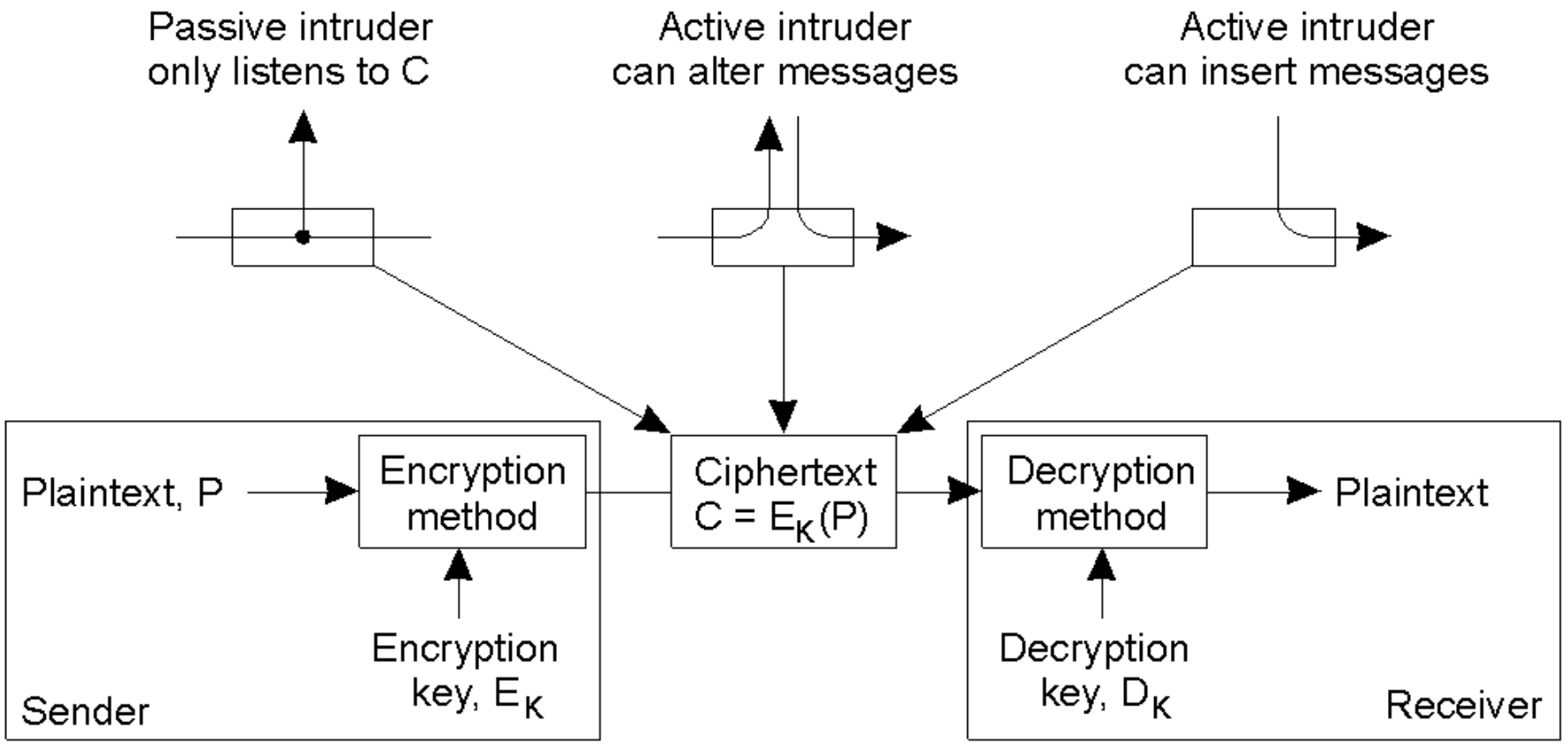
-DO NOT WRITE YOUR OWN CRYPTO FUNCTIONS

-USE OFFICIAL AND STABLE ALGORITHMS

-CHECK REGULARLY FOR CHANGES (MD5 e.g.)

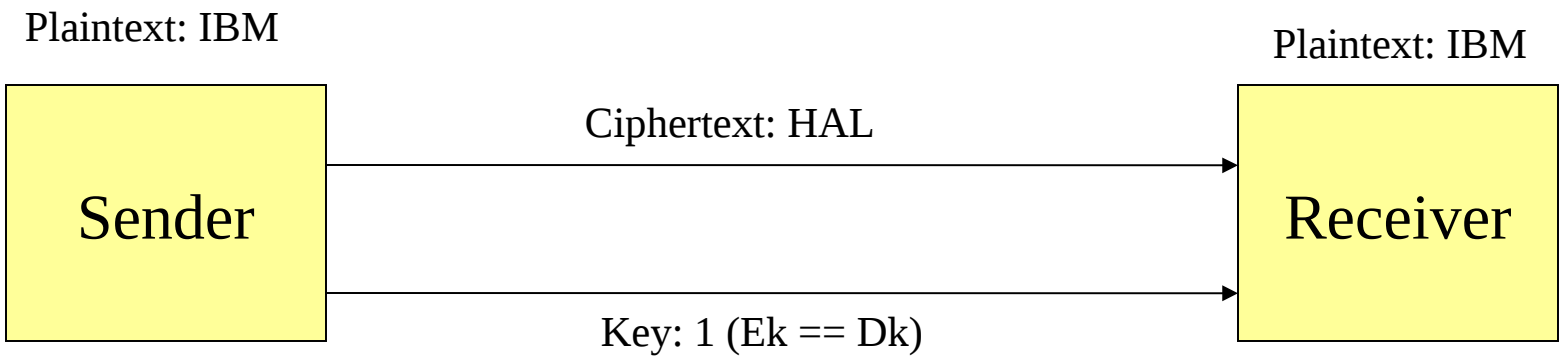
-BE REALISTIC: CRACKED CRYPTO IS THE LEAST OF YOUR WORRIES!

Terminology



From van Steen, Tanenbaum

Symmetric Keys (are like passwords)



There is only one shared key used between sender and receiver. The main problem of symmetric keys in distributed environments are:

-key transport across public networks

-Key maintenance

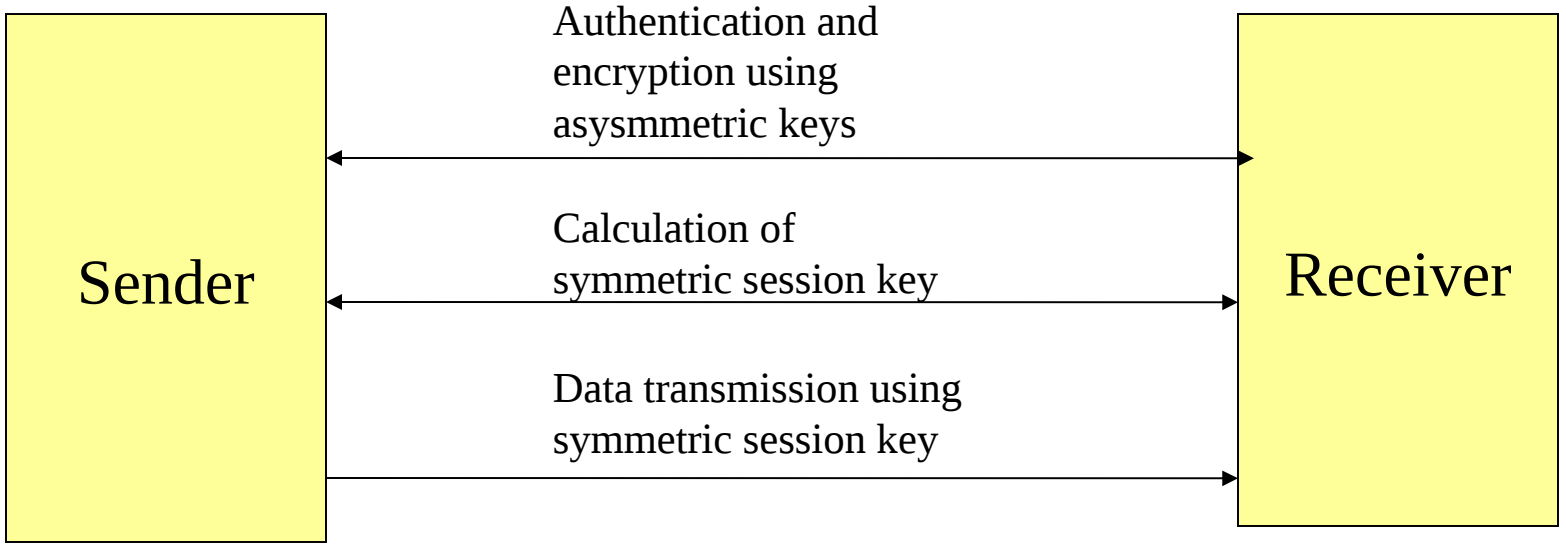
-Number of keys increases with multiple partners

- trusted and well known partners required

- no support for non-repudiation

It is NOT true that symmetric keys are weaker than e.g. asymmetric keys!

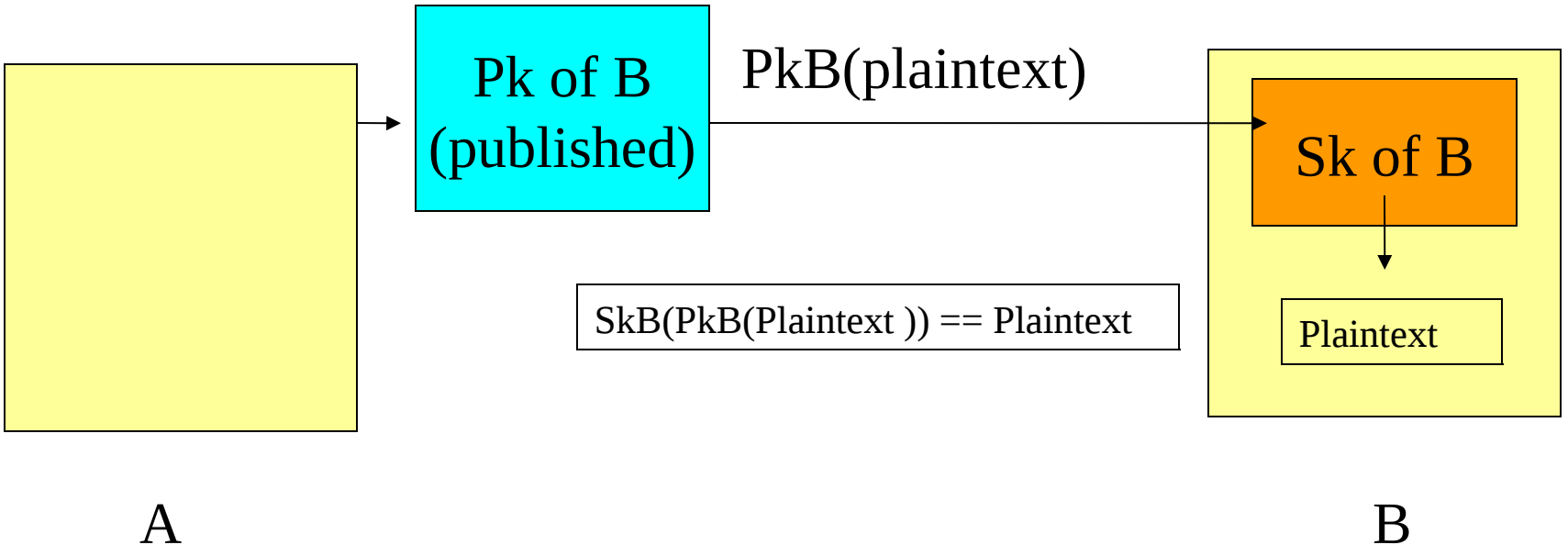
Use of Symmetric Keys in distributed systems



Computationally asymmetric keys are 1000-10000 times more expensive than symmetric keys. For this reason many protocols establish a secure session context using asymmetric keys and use symmetric keys for data transmission (e.g. SSL sessions). Popular symmetric algorithms are: DES, Triple DES, IDEA and the new AES: Rijndael.

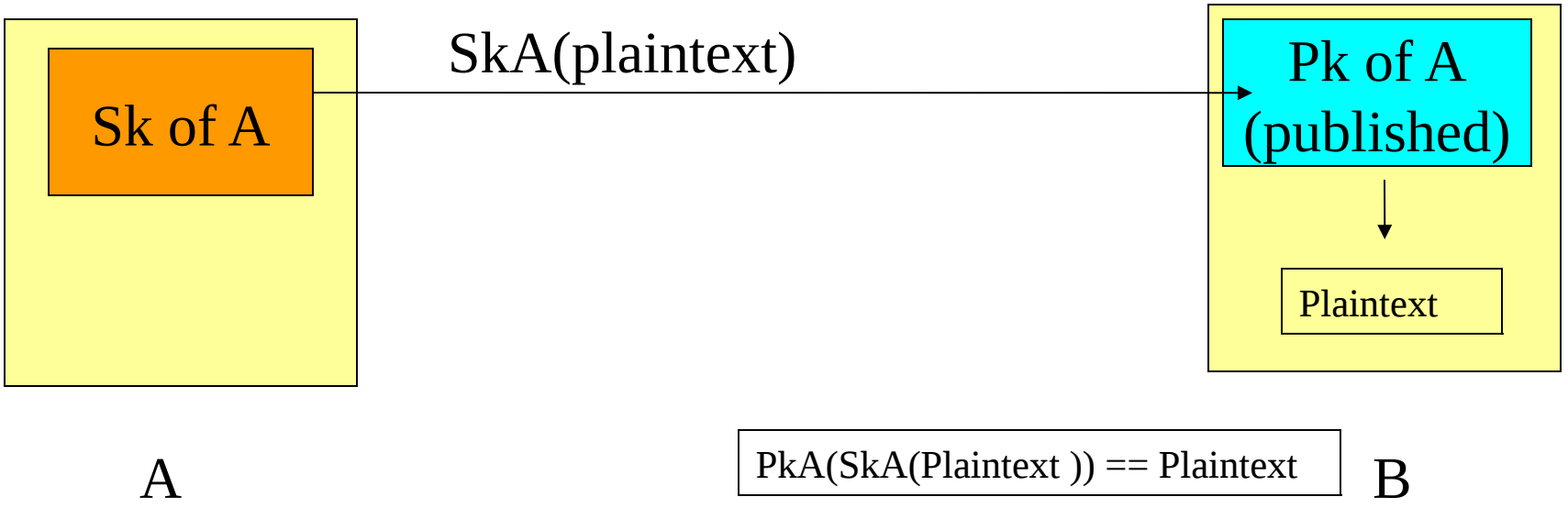
Embedded control devices which support asymmetric key algorithms are still quite expensive on a mass distribution scale.

Asymmetric Keys : Encryption use



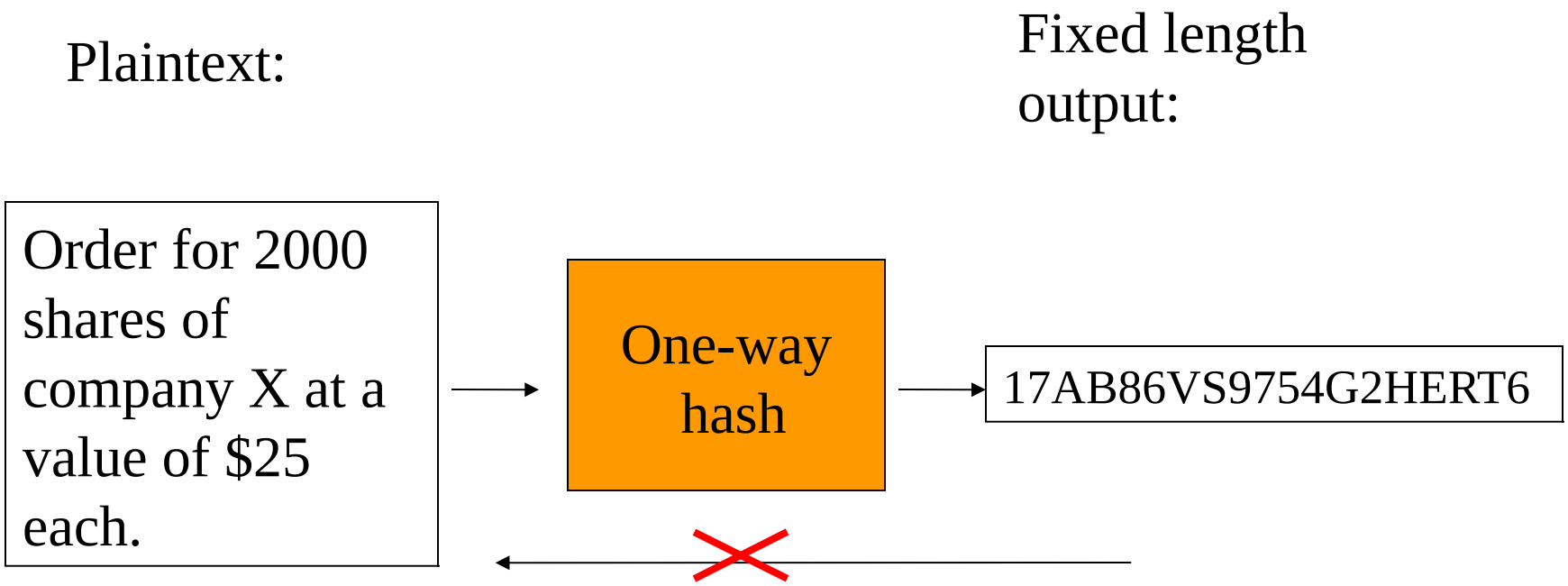
Asymmetric keys are a pair of keys where one stays private (secret) while the other one is made public. To encrypt a message so that ONLY the receiver can read it, the sender uses the PUBLIC key of the receiver. Note that the sender can no longer read the message after encryption with the receivers PUBLIC key! (lost key problem in companies). Popular algorithms are RSA and DSA.

Asymmetric Keys : Signature use



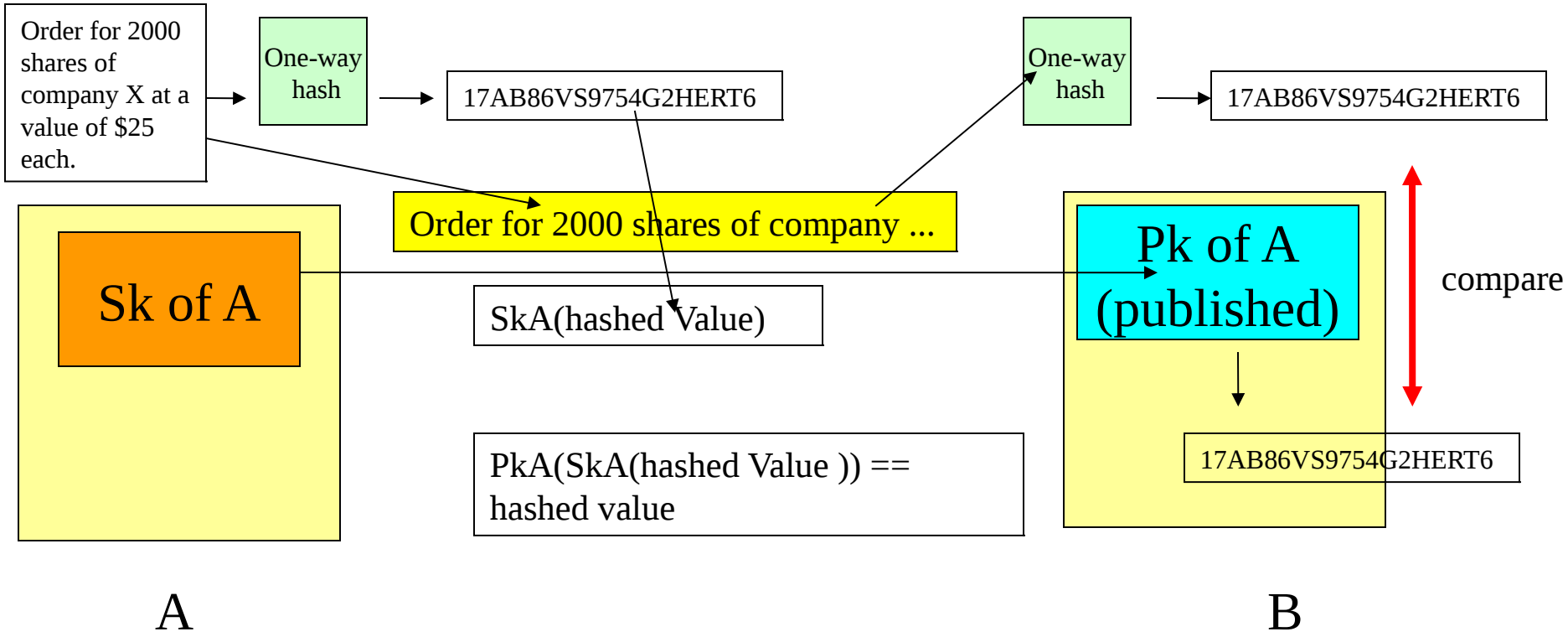
If a sender uses HER SECRET key to encrypt a plaintext, EVERYBODY who has the PUBLIC key of the sender can decrypt the message (no privacy). But everybody will know that ONLY the sender could have encrypted the message because otherwise the senders public key couldn’t de-crypt the message successfully. This identifies the SENDER.

One Way Hash-Codes (digital fingerprints)



A one way hash function creates a fixed length representation of the input (with padding if necessary). The function guarantees that it is computationally infeasible to reverse the process and that no two different inputs will create the same output. A change in the input will cause a completely different output. This can be used to detect TAMPERING during message transfer. Popular one-way hash algorithms are Message Disgest 5 (MD5) and Secure Hash Algorithm 1 (SHA1)

Digital Signatures



The sender creates a digital fingerprint of the plaintext, encrypts it with her secret key and sends it – together with the un-encrypted plaintext to the receiver. The receiver performs THE SAME hash-calculation as the sender on the plain text. The receiver also unpacks the encrypted fingerprint and then compares the two hash values: If they are identical the receiver knows that a) the plaintext is from the sender and b) it has not been tampered during transit. Additionally the plaintext could have been encrypted as well to preserve

The big advantage of public key encryption in distributed environments

- The receiver does NOT get your secret key and can therefore NOT impersonate you or modify your messages etc.
- Distribution of public keys over public networks is not a big problem (see certificates later)

There are some disadvantages as well: a lost secret private key means that messages cannot be decrypted anymore. This is a problem for companies that need to keep records and histories. Also if an employee gets sick, a substitute cannot read her received messages without her secret key. Workarounds like key escrow (the secret key is stored in some repository) of course decrease security by introducing trust. And the increased trust that people put into this technology may be a mixed blessing. What if keys are compromised?

Cryptographic attacks

- Brute force: try every possible key for a given encryption algorithm. An algorithm is good if ONLY a brute force attack will crack it. A special form of attack is a „distributed brute force“ attack which involves large numbers of computers, e.g. on the internet
- Known [plaintext|cyphertext], chosen [plaintext|cyphertext]
- Finding a backdoor for an algorithm through mathematical analysis
- Exploiting weaknesses in random number generation used for an implementation (e.g. the numbers are somehow predictable)

Steganography: invisible ink of the digital time

R 10110001 → R 10110000

G 10001110 → G 10001111

B 01010011 → B 01010010

This is a secret message

54 68 69 73 20 69 73 20 61
20 73 65 63 72 65 74 20 6d
65 73 73 61 67 65 0d 0a

010 10100 01101000 01101001

Least Significant Bit encoding hides a message in the LSB of an image. The image needs to have a certain variability. Additionally, the message can be encrypted. Changes in the LSB of a true color image are not noticeable for the human eye. But computers can detect „unusual“ bit frequencies

The Distributed IT-Security Mantras

- Authentication (Who is it?)
- Authorization (What can she do?)
- Integrity (Is the message un-changed?)
- Confidentiality (Can somebody else than the receiver read the message?)
- Non-Repudiation (The proof that somebody did send a message, e.g. an order)

These goals are clearly directed towards the operation of IT within corporations or governments. Compare this to the civil rights oriented security mantra.

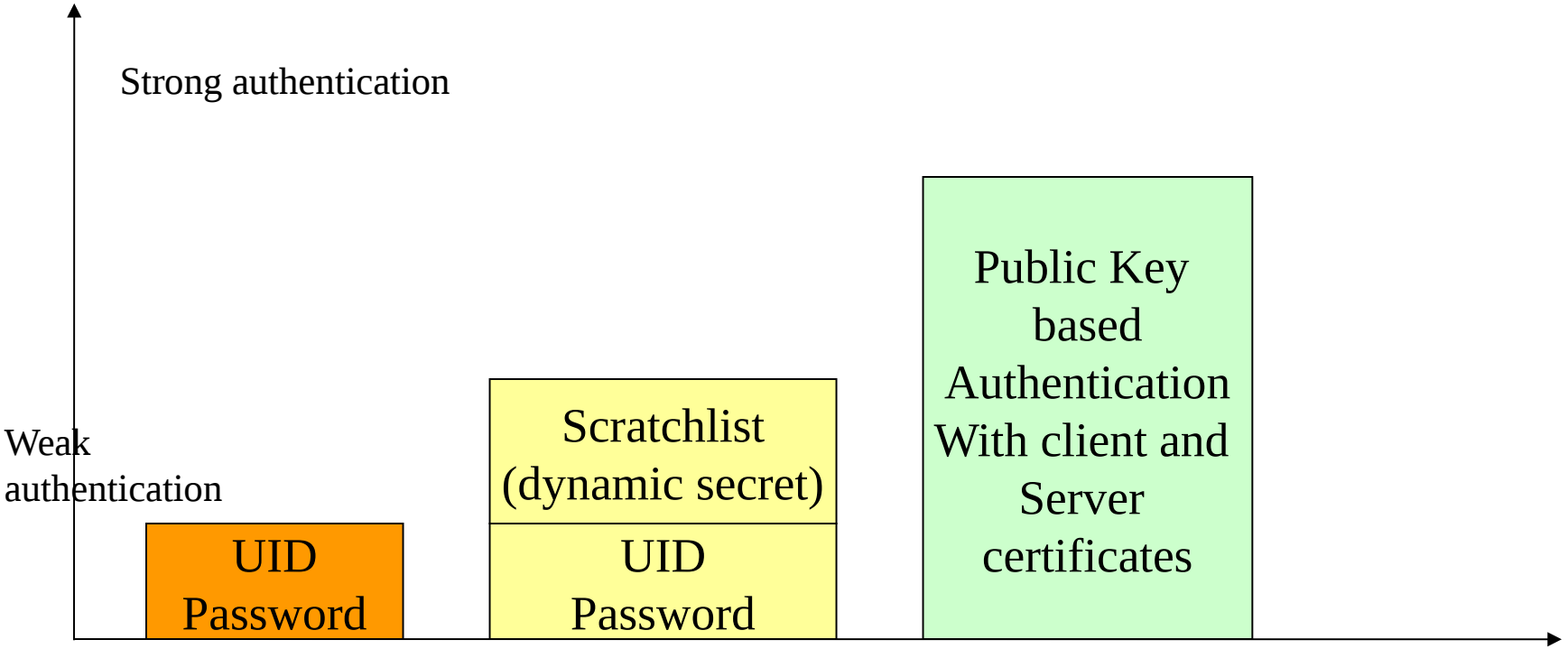
Authentication: Verifying an Identity

By:

1. What you know (passwords, PIN's)
2. What you have (cards, tokens)
3. What you can do (private key for encryption)
4. What you ARE (fingerprints, iris-pattern, blood-test, face recognition etc.)

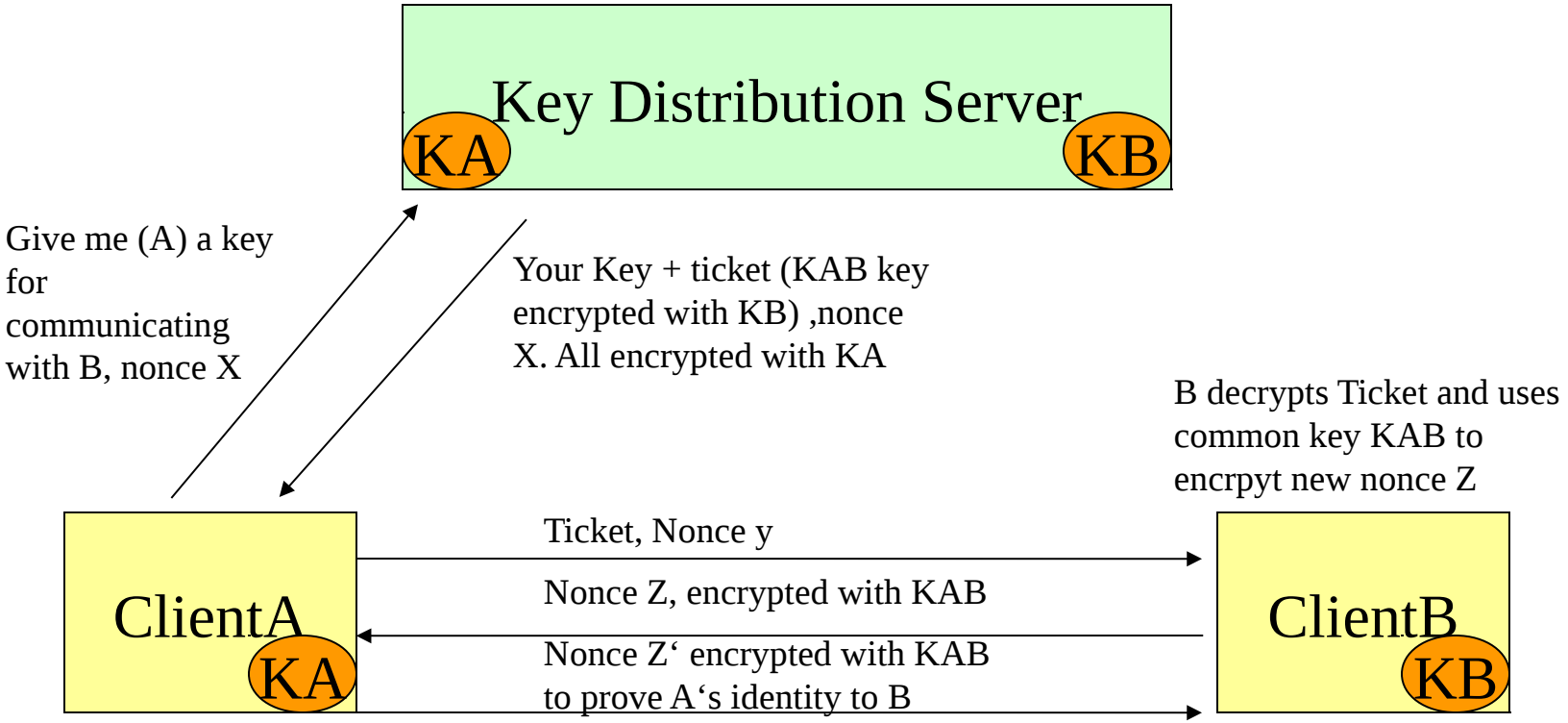
A combination of technologies is often more secure than the use of a single mechanism – except if given to a MIM. (multi-factor authentication). Ask yourself: why do I want to authenticate somebody/something?

Authentication levels and Strength



Please note that the use of SSL in the UID/PW case does not improve the authentication level. The authentication level typically decides about what a client is allowed to see or access. Some systems provide a dynamic step up if a client wants to access some resource which requires a higher level. This is not easy implement correctly in software.

(Partial) Solution to the PW-distribution Problem: Needham-Schroeder Authentication Protocol



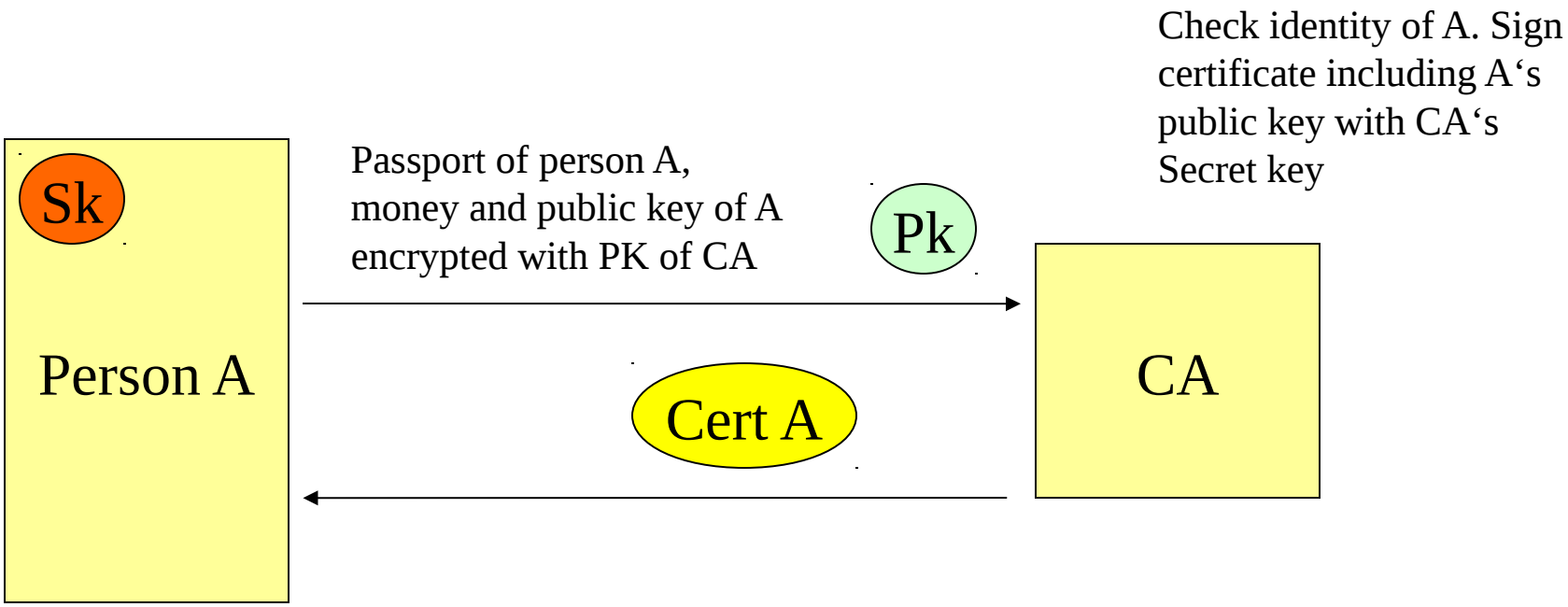
This protocol is used e.g. in Microsoft‘s Kerberos implementation. It requires a secure authentication server which knows the symmetric keys of all principals. A nonce is a request counter that is used to fend off replay attacks. A ticket contains a common key (KAB) and a partner name, both encrypted with the receivers secret key. Note that the secret keys of A or B (or passwords) do NOT go over the wire

No more distributed secrets: Public Keys and Certificates (X.509)

Version	V3
Serial Number	1234 5678 ..
Signature Algorithm	sha1RSA
Issuer	Verisign
Valid From	1.1.2000
Valid To	1.1.2001
Subject	Walter Kriha
	walter@kriha.de
	11.07.1958
	Private Individual,..
Public Key	#12345ABCDEF123
Signature of CA	12EF72A1C590BE..

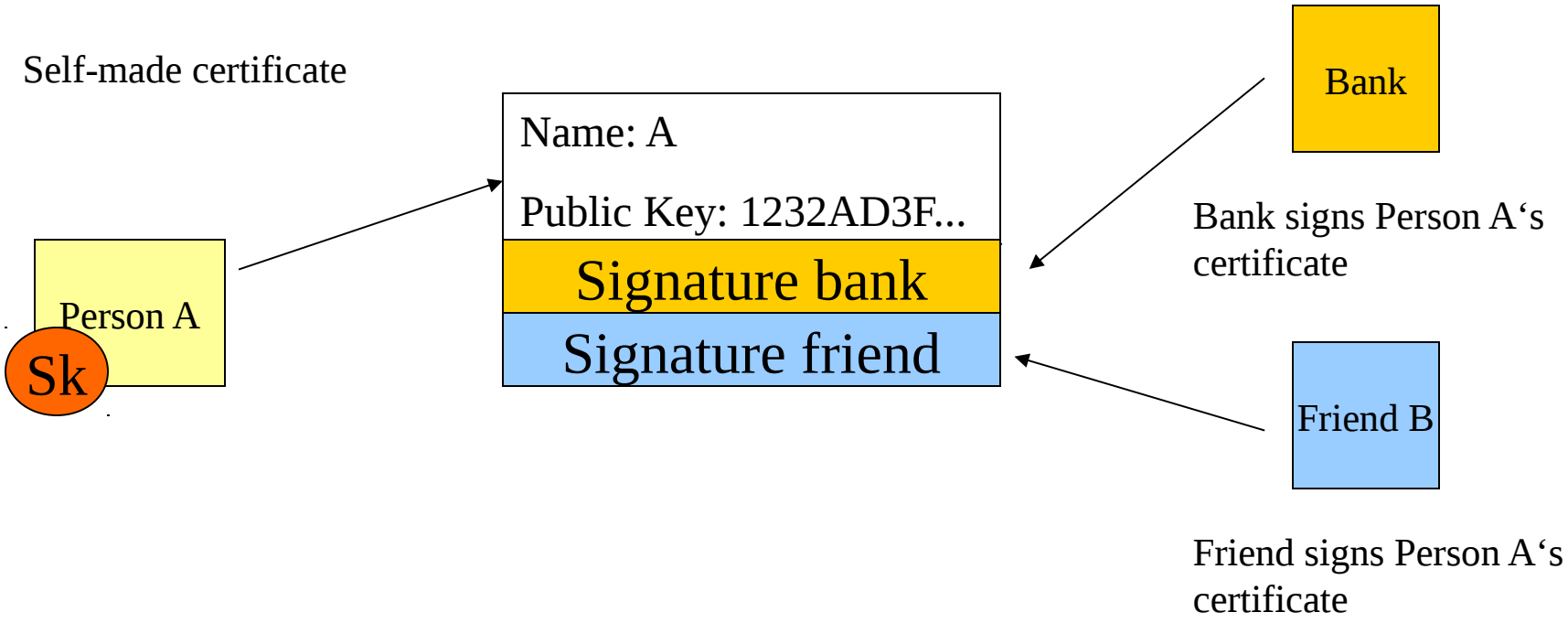
A certificate associates a subject with a public key. To prevent man-in-the-middle attacks, a sender MUST know the receivers public key to encrypt the messages. Extension fields specify e.g. the usage of the certificate, ist revocation location etc. Certificates are a distributed solution for the lack of a common identity server.

Centralized Generation of Certificates: Certificate Authorities (CA)



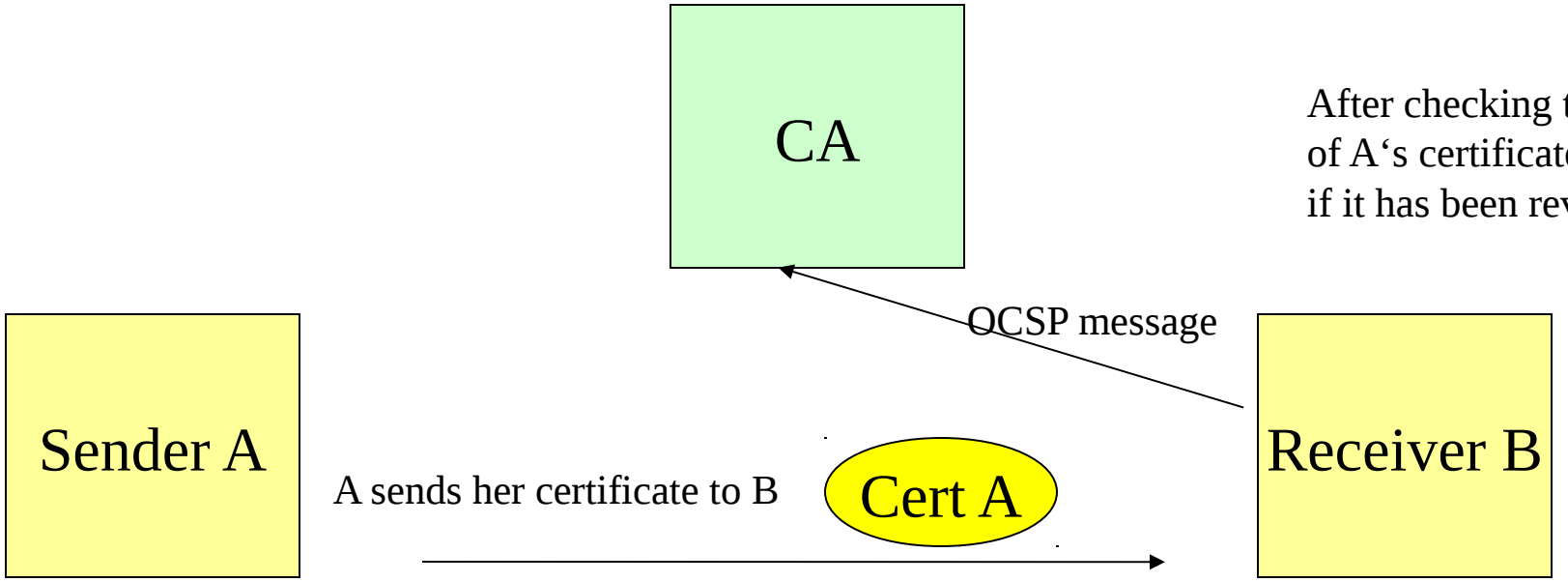
A certificate authority guarantees that client A has public key X for a price. Obviously the price dictates how carefully the CA checks the persons identity. A compromised key of a CA would be VERY serious. A person can now publish this certificate and receive private mail from senders using her public key from the certificate. Why do you trust a CA?

Distributed Generation of Certificates: Certificate Chains



The more people sign person A‘s certificate the higher the likelyhood that a receiver of A‘s certificate will recognize one of the signers as a trusted instance and accept the certificate. Thereby a chain of trust is created. Pretty Good Privacy (PGP), a public domain program for secure messaging works this way. (see Dan Zimmerman‘s PGP book, O‘Reilly etc.)

Certificate Revocation



The Open Certificate Status Protocol allows the checking of certificates by the receiver. Reasons for the revocation are given in a revocation message, e.g. private key compromised, content changed or CA compromised.

(Distributed) Authorization Methods

- Role Based Access Control
- XACML Decision Framework
- Federated Authorization (Oauth)
- ACLs
- Capabilities

Authorization by RBAC

Roles:

- Admin
- UserAdmin

Rights:

- createResource
- addRole

Object and rights

- Table: create, delete..
- Role: add, delete, modify

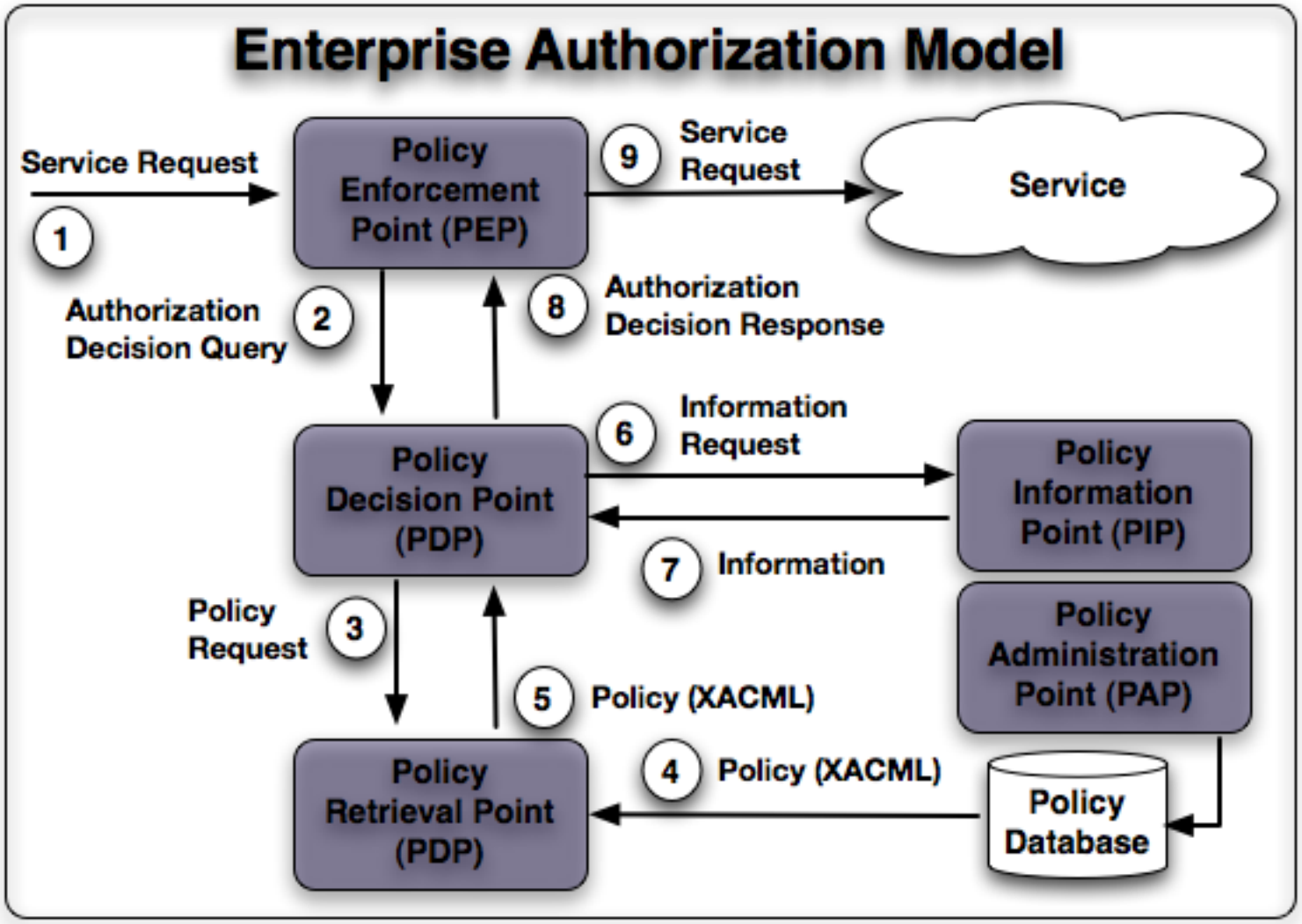
Users:

- Kriha

Users	Roles	Roles	Rights	Rights	Object
Kriha	admin	admin	createResource	create	table

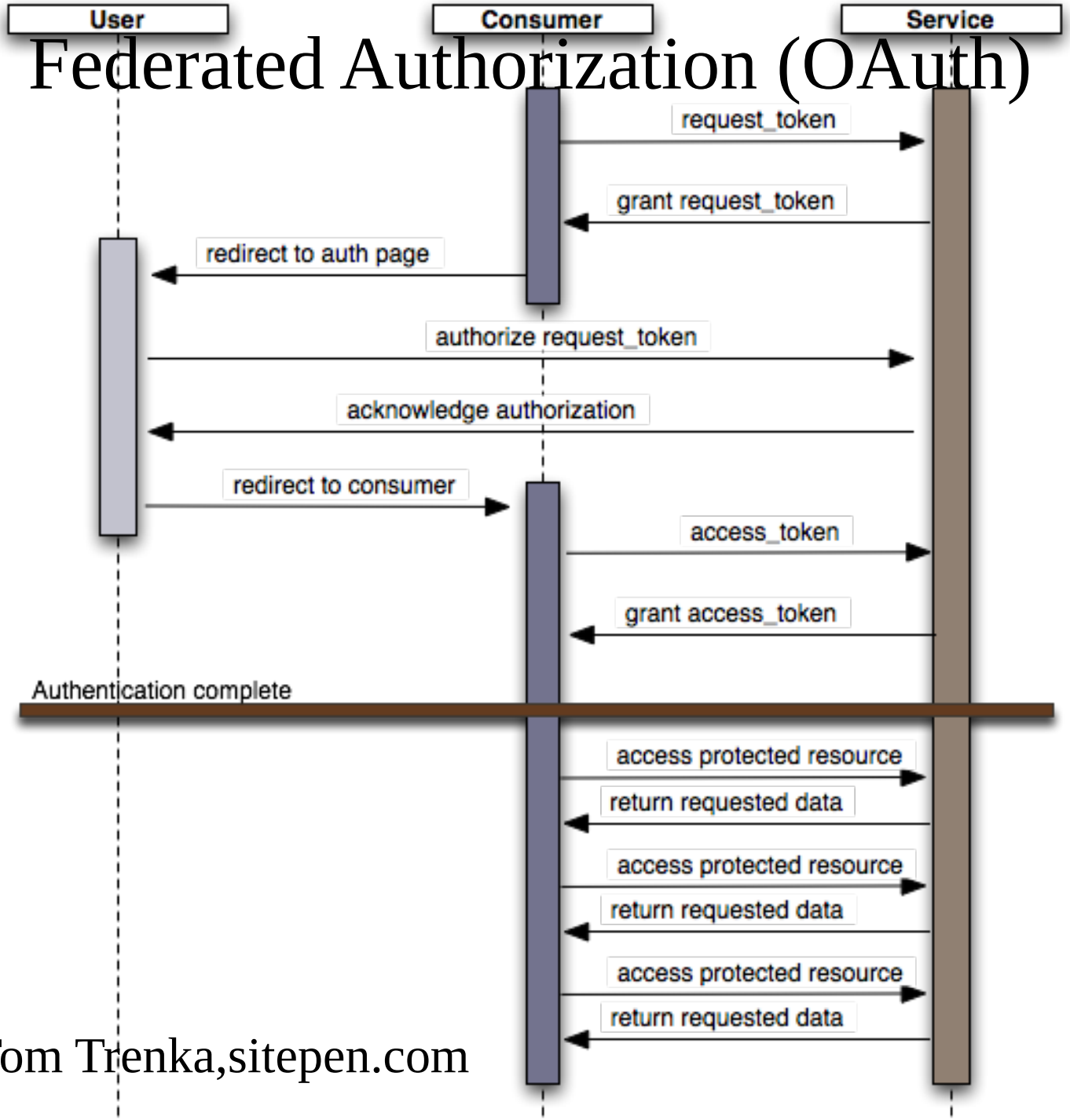
Role based access control is now a standard feature in controlled environments (like companies). A good authorization model allows for “Service Management delegation” in order to let organizational groups maintain their own rights system for their own resources. Resources should never know about users. Sometimes type based authorization is not enough and a rule uses instance qualities to restrict access. These qualities can still be checked externally of resources if the resources provide the necessary interfaces. Without flexible authorization organizations cannot change their structure.

XACML Auth. Decision Framework



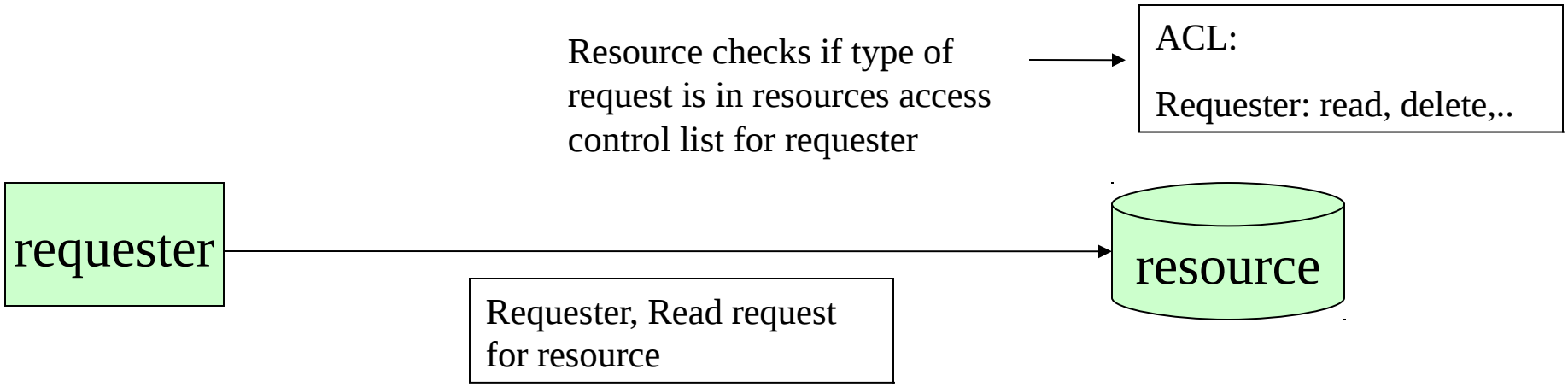
From: National Cancer Institute

Federated Authorization (OAuth)



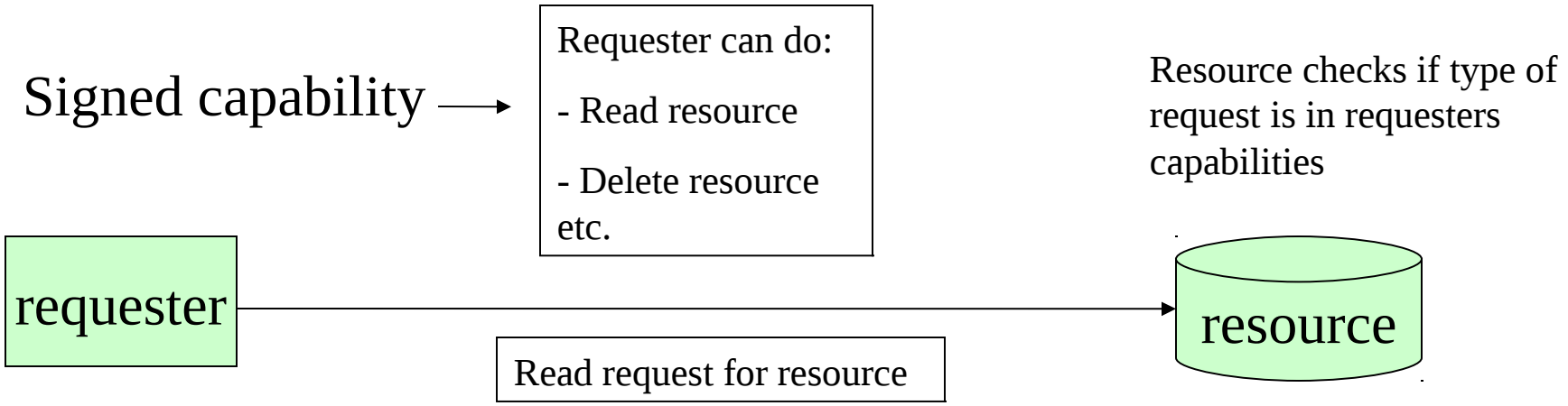
From Tom Trenka,sitepen.com

Access Control Lists (ACL)



Access Control Lists are easy to implement and are very popular today. Both Unix and NT use them. They do not scale very well in case of very large numbers of principals. The OS390 implementation of DCE uses capabilities instead.

Capabilities

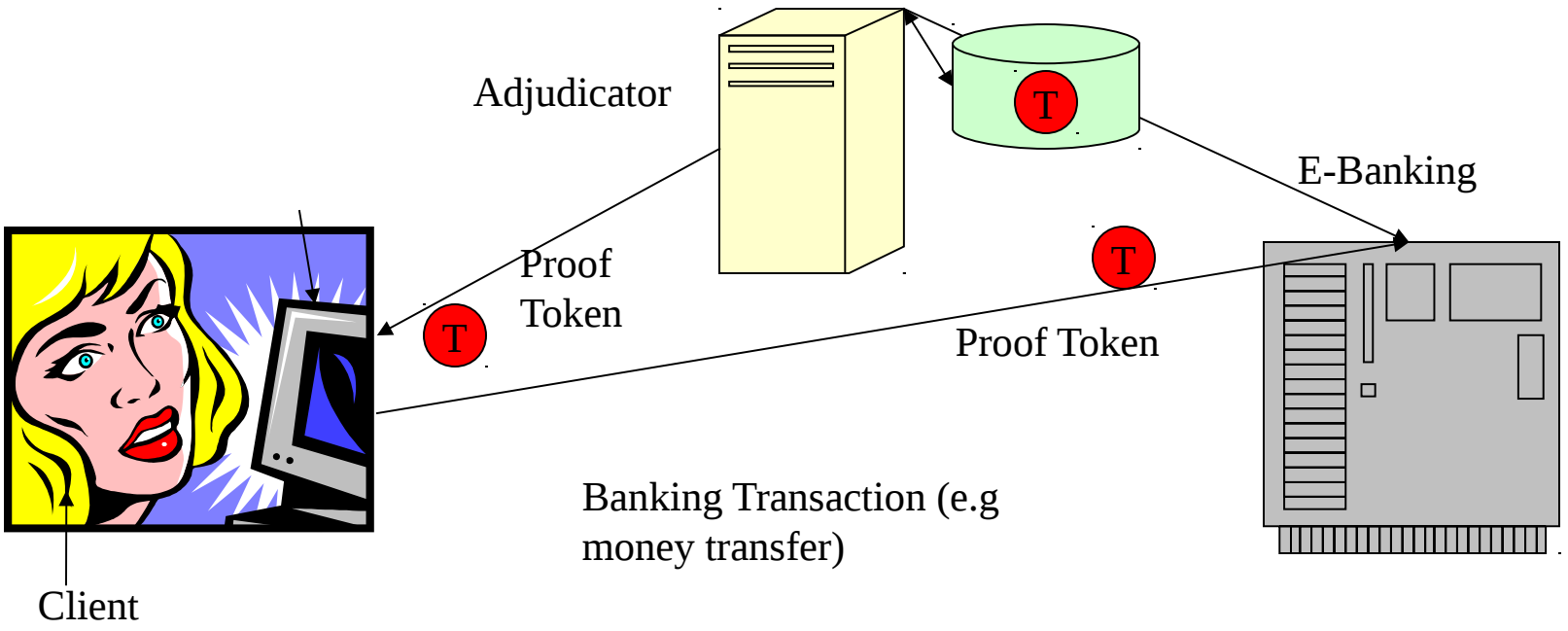


Capabilities work like keys: if you have a key you can open a lock. Capabilities scale well in most environments but have the problem of „lost keys“: they need to be protected from replay attacks and stolen capabilities. And what happens if the rights of an owner change (e.g. by moving to a different department)? Capabilities need to expire automatically. Delegation is easily implemented using Capabilities.

Channel- vs. Object-based Security

Non- repudiation
SSL-channels and architecture
Secure Mail

Non-Repudiation: “It was not me who did…”



Here a mediator (adjudicator) sends a proof token for the transaction to both parties. In banks legally binding procedures have to be certified through the banking commission – if your procedures do not comply you can’t work as a bank.

Non-repudiation is based on digital signatures in addition to extensive auditing of all communication flow. The auditing may create a security problem by itself because important and confidential data is collected and recorded. Who is allowed to see the audit trail data?

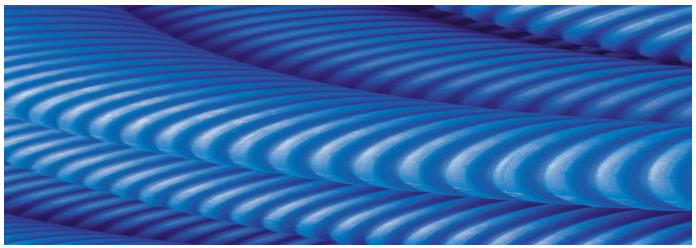
Channel based vs. Object based Security

Sender

Receiver

Information

ssl-channel



Information

integrity/confidentiality

author?

Information

Digital Signature

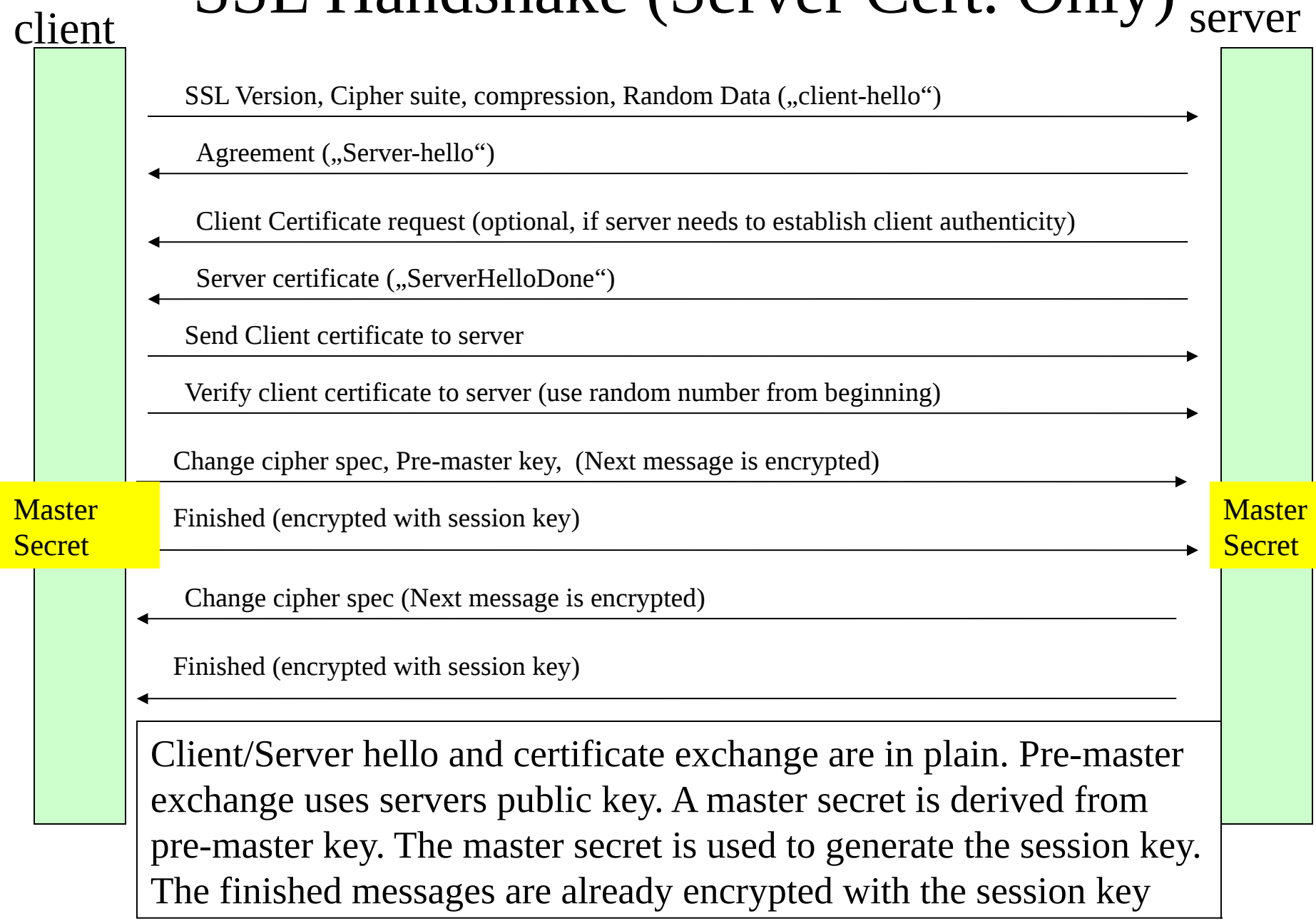
encrypted

Information

Digital Signature

encrypted

SSL Handshake (Server Cert. Only)



Client/Server hello and certificate exchange are in plain. Pre-master exchange uses servers public key. A master secret is derived from pre-master key. The master secret is used to generate the session key. The finished messages are already encrypted with the session key

SSL Dangers

- Certificates of Certificate Authorities need to be transported to the client over a secure channel (pre-installed etc.) Other wise man-in-the-middle attacks are possible. The domain name of the server should be included in the certificate
- Participants need to be sure about what NAMES in a certificate really MEAN
- Danger lies in the fallback and protocol negotiation features of SSL. A bank server e.g. needs to enforce 128 bit encryption with trusted algorithms and cancel the connection if the client tries to fall back to something less secure
- As always: the implementation e.g. the random number generation may be flawed.

PKI suffers from some basic problems like non-existing global names, various Certificate Authorities and key management problems. SSL does NOT create non-repudiation

Object-based Security: S/MIME

From: walter@kriha.de

Subject: Test

Content Type: multipart/signed;protocol=„application/pkcs7-signature“; micalg=sha1; boundary=boundaryXYZ

--boundaryXYZ

Content-Type: text/plain

(plain-text message)

--boundaryXYZ

Content-Type: application/pkcs7-signature

Content-Transfer-Encoding: base64

(signature)

--boundaryXYZ

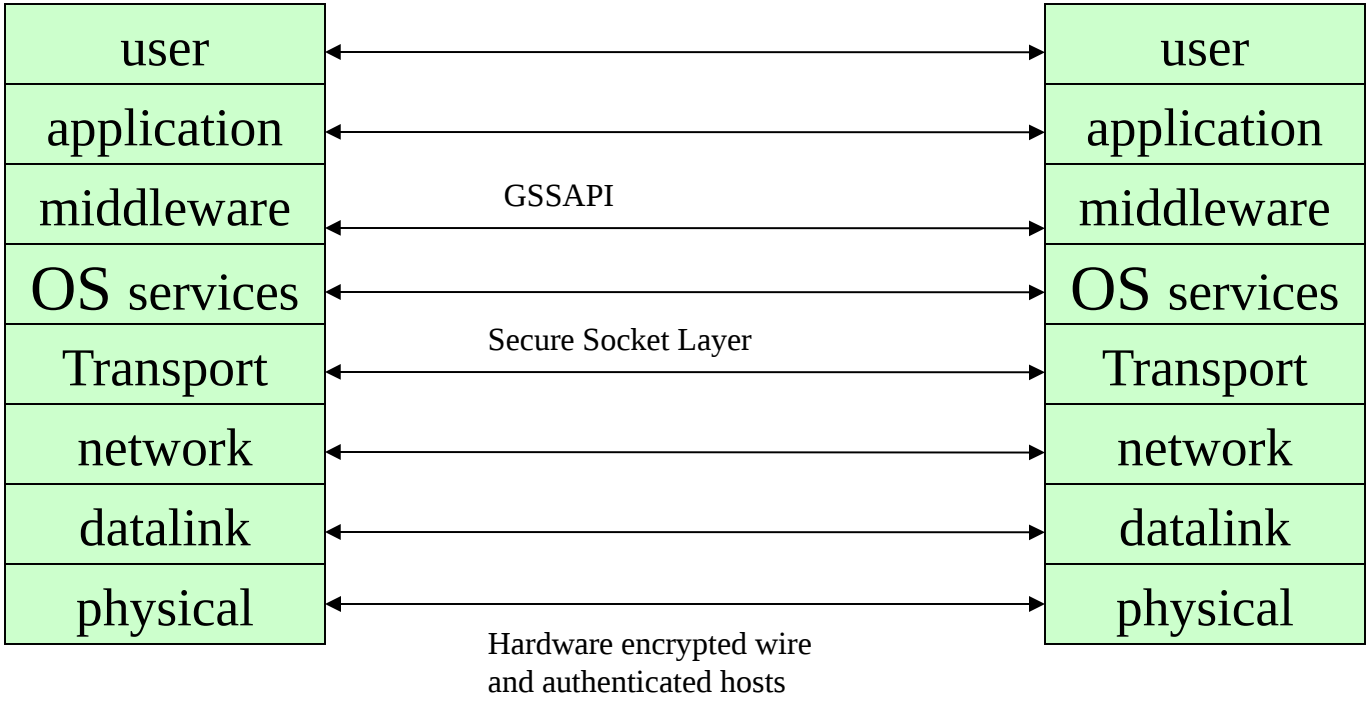
S/MIME allows secure e-mail over the store-and-forward architecture of SMTP. Note that the first FROM/SUBJECT items are in plain text. Mail partner discover S/MIME capabilities of each other through mail sent between them. S/MIME supports intermediaries (relays etc.) perfectly.

The end-to-end argument

„Any communications system involves intermediaries, such as network devices, computers and programs which are unarware of the total context of the communication being involved. These intermediaries are therefore incapable of ensuring that the data is processed correctly“ (Voydock 1983)

The original end-to-end argument can be found in Salter,Reed and Clark 1984). IPSec, SSL and S/MIME work on different levels with the ones closer to the application deal better with proxies etc.

End-to-End security (top-down)



Security can be implemented on many layers. The deeper the layer the more TRUST is needed in the upper layers. End to end means that the specific layer can control the security to the receiving layer, e.g. by handing only encrypted content to the next lower layer.

Resources (1)

- Van Steen/Tanenbaum, Chapter 8
- Studie “Gesicherte Verbindung von Computernetzen mit Hilfe einer Firewall”, Andreas Bonnard, Christian Wolff, Siemens AG (für Bundesamt für Sicherheit in der Informationstechnik BSI)
- Internet Cryptography, Richard E. Smith, www.visi.com/crypto
- WWW Security FAQ, www.w3.org/Faq (with short bibliography)
- Cryptography FAQ, www.faqs.org/cryptography-faq
- RISKS, Forum on Risks to the Public in Computers and Related Systems http://catless.ncl.ac.uk/Risks (real life stories on the social and political consequences of security flaws

Resources (2)

- The EU commissions report on the US/UK Spy project Echelon – How the US and UK do industrial espionage against Europe.
- Simson/Garfinkel, Database Nation
- Bruce Schneier, Applied Cryptography (the bible of cryptography). Surprisingly good to read and understand!
- Bruce Schneier et.al., Practical Cryptography – even better for software developers. Explains problems of PKI very well.
- Diffie et.al., Privacy on the line (explains why encryption is a civil right that organizations like the NSA try to subvert) (Yes, THAT Diffie from Diffie-Hellman Key exchange)
- www.cert.org , your most important source for information on new security breaches etc. Register for the newsletter! Also an excellent source on security technology
- Frederick Thomas Martin, Top Secret Intranet – How US Intelligence built Intelink – The worlds largest most secure network. Good to read.

Resources (3)

- Improving the Security of Your Site by Breaking Into it:

  http://www.fish.com/~zen/satan/admin-guide-to-cracking.html

  A good introduction into cracking systems. Fun reading too.
- The strange tale of the denial of service attacks against grc.com, Steve Gibson, 2001. The power of distributed DOS attacks. (www.grc.com) Really funny to read!
- Introduction to SSL, www.developer.netscape.com/docs/manuals/security/sslin/contents.html
- Coulouris et.al., Section 7. Contains a good explanation of kerberos ticket mechanisms. (Used in OSF/DCE and Microsoft products)