Advances in Medical Informatics, J. Noothoven van Goor and J. P. Christensen (Eds.), IOS Press, 1992

# Technical Aspects of Data Protection in Health Care Informatics*

Andreas Pfitzmann, Birgit Pfitzmann
Institut für Informatik, Universität Hildesheim,
Postfach 101363, W-3200 Hildesheim, F. R. of Germany
Phone: +49 5121 883735, Fax: +49 5121 860475

**Abstract.** Data protection comprises availability and data integrity as well as data confidentiality and privacy. We first consider security problems and their causes, and then measures against them. Finally, we recommend actions which should be taken independently of the construction of specific systems.

Our main subject is distributed systems and medical networks: At present, most applications of informatics (computer science) in health care are stand-alone ones, such as databases or 3-D imaging systems. In the future, however, there will be more and more distributed systems. For example, for administrative purposes, lots of PCs in a hospital could be interconnected by a local area network, or PCs at private doctors could communicate with computers at insurances via a public network. For direct medical purposes, there are proposals to use video conferences to consult external experts during operations, or to monitor patients in their homes. The use of chipcards as carriers of emergency data is a distributed system, too.

For security, distributed systems bring about new dangers, but also new possibilities for security measures. Dangers mainly result from the fact that the new systems are more complex, there are more interdependences, and more people have access to at least some part of the system than before. Possibilities are offered, e.g., because more complicated security measures can be performed by computers, and data can be put out of reach of someone or some fault by suitable physical distribution.

## 0 Overview

We first consider the security problems in health care informatics, especially medical networks[1], in detail (§ 1), and then measures against them (§ 2). Since informatical systems used in medicine can be of quite different (technical and social) characteristics, these chapters deal with any kind of informatical systems. Of course, not all problems are posed, and not all measures are applicable, in every system. In § 3, for the case of networks, we therefore list some technical and social parameters which influence the special problems and solutions which apply to a given network.

To show how the material from the previous sections can be applied, in § 4, we present several examples: We classify some specific medical networks according to the parameters from § 3, and sketch which security measures should therefore be taken.

In § 5, we give some resulting recommendations for actions which should be taken independently of the construction of specific systems.

---

* This paper is a revision of "A. Pfitzmann, B. Pfitzmann: Security in Medical Networks; presented at: Handling Health Data in the Future - AIM Working Conference on Data Protection and Confidentiality in Health Informatics; March 19-21, 1990, Brussels (Commission of the European Communities)".

1 **Network** is used as a synonym for "communication network", i.e. a technical system which allows a number of devices ("intelligent" computers, "stupid" phones) to exchange messages. We consider all components of a network, the hardware (lines, switching centers, user terminals) and the software which organizes the message transmission. [52] provides a good introduction to the technical aspects of networks.

The focus on networks results from our own research. We do not claim that networks are the most interesting part of the problem of handling health data, since most of their problems can be solved by standard methods for network security. In particular, we recommend readers to consider problems of statistical databases in more detail [1] Chapter 6, [2], [3], [4]. We also think a very interesting question is whether sensitive data can be reduced considerably by use of CHAUM's credential mechanism [5].

# 1 Security problems in informatical systems, especially networks, in general

Many problems and solutions are similar for informatical systems for quite different purposes and with quite different technical parameters. Therefore, this chapter is not specific to medical systems.

## 1.1 Possible problems

The problems are first formulated for general systems (at least in informatics), then specially for networks.

P1 Complete failure, i.e. the system does not work at all. (This is also called "denial of service", if caused by an attack.)

P2 Incorrectness, i.e. the system works (in a way which affects its environment, or which seems correct at a superficial look), but not correctly.
In networks, the main problems are:
- P2.1 Messages are changed during transmission or in intermediate nodes (such as switching centers).
- P2.2 Users give wrong commands.

P3 Loss of privacy (confidentiality of data).
Especially in networks, there are two kinds of data to consider:
- P3.1 Confidentiality of message contents.
- P3.2 Privacy of traffic data, i.e. who communicates with whom, when, how much etc.
  In medical networks, the main case where the privacy of traffic data is necessary is if patients take part in communication themselves.
  - P3.2.1 Privacy with respect to third parties.
  - P3.2.2 Privacy with respect to the communication partner, i.e. anonymity, e.g., when consulting a psychological advice center.

Note that incorrectness, and also complete failure, can also occur as subproblems of the other problems. For example, if measures to ensure privacy are implemented on a computer, and if this computer or the implementation are not correct, privacy can be lost.

## 1.2 Possible causes of problems

The possible causes, like the problems, are similar for most problems in informatics.

C1 Faults [6].
- C1.1 "Physical" failures (like outside electrical disturbances or inside wearing-out of the material). They usually follow simple statistical distributions, and faults in physically independent systems usually occur statistically independently.
- C1.2 Unintentional human errors.
  - C1.2.1 Design or implementation errors.

C1.2.2 Errors during the operation of the system.
In contrast to physical failures, human errors are often not statistically independent.

C2 Attacks[2] [1], [7].
- C2.1 Outsider attacks (i.e. by people who should play no role in the system, such as eavesdroppers).
- C2.2 Insider attacks (i.e. by people who play a role in the system).
  - C2.2.1 Insiders during the design or implementation of the system (i.e. designers or implementors of the system itself, designers or implementors of the tools used, and so on, since they might design or implement the system wrong, e.g., implant a universal Trojan Horse[3] [1], [2]).
  - C2.2.2 Insiders during the operation of the system.
    - C2.2.2.1 Users.
    - C2.2.2.2 Operators, security managers.
    - C2.2.2.3 Maintenance personnel; in particular, system programmers.

In most cases, the causes with higher numbers are more malignant. In particular, measures which help against attacks also help against faults, and insiders are a much greater problem than outsiders. However, this does not imply that one should consider insider attacks only: Faults can often be handled more efficiently than attacks (which are, hopefully, less frequent), and sometimes security against insiders cannot be achieved on principle.

Of course, combinations of several faults or attacks are conceivable; e.g., someone may implant a Trojan Horse during the design of a system, and later cooperate with it as a normal user of the system.

Note that considering potential attackers everywhere in a system does not mean to suspect that insiders are usually attackers. It only means that a very small percentage of them might be, and it is much better to be safe than sorry.

Especially with medical systems, potential attackers' motives might be, apart from a queer sense of fun: Attackers on privacy might be employers, insurances, blackmailers, secret services (e.g., foreign ones, in order to start blackmail), or relatives. These attacks are more likely than the following ones. Attacks on correctness might be performed because of personal conflicts. Attacks resulting in complete failures might be carried out during wars or by terrorists, or used as a threat during blackmail.

## 1.3 Remarks on why 'advanced informatics' can aggravate problems

Sometimes it is argued that sophisticated technical security measures in systems employing advanced informatics are unnecessary, because they were not taken in previous systems either. However, new techniques can (not "must") aggravate problems. Some reasons are:

---

2 For a given system, **attacker** means somebody who (deliberately) does not behave according to the rules of the system. It is usually assumed that all attackers work together to achieve their goals (e.g., that they exchange all their information), thus we often say "the attacker" to describe the set of all attackers.

3 A **Trojan Horse** is additional (and hidden) functionality, e.g., in a program, which exploits the legitimate authorizations of the user, e.g., the invoking process, for purposes of the creator of the Trojan Horse. For example, a Trojan Horse might send secret information (upon which the program operates) to its creator via a **covert channel**. This is a channel whose usage is not controlled, or whose existence is not even known, such as the exact timing of outputs (cf., e.g., [42], [43]). While a normal Trojan Horse carries out a task fixed by its creator during the design or implementation, a **universal** Trojan Horse can repeatedly receive and execute instructions by its users during the operation of the system.

- Greater dependencies between system parts, e.g., because of
  - centralization,
  - common design of subsystems in various system parts, or
  - common physical weaknesses, for instance a breakdown of the power supply, electromagnetic interference between hardware modules, great electrical disturbances in the atmosphere.
- Greater complexity (hence unintentional design faults are more likely, and both unintentional and deliberate design faults are more difficult to find).
- More dangerous problems can be tackled (such as research on dangerous chemical processes or with dangerous bacteria, using automatic process control and robotics; databases with more private data are manageable).
- Some people trust computers too much, hence they become careless (e.g., they do not even try to check the correctness of results).
- Often nobody is legally responsible for system faults; this, too, can make people careless.
- Attackers, using advanced informatics themselves, have greater possibilities than before (this is particularly important with privacy, since attackers can copy data much faster, and store and evaluate far more, than before).

  In particular, since theft of data means copying only, it can neither be detected nor repaired afterwards. Hence it must be avoided in advance.

## 1.4 Levels of security

No security can be achieved if everybody is an attacker and all machines are faulty. The level of the security achieved by some measures depends on the restrictions on the attackers and faults which must be assumed so that the measures work (attacker model, fault model).

In the following, we use "attackers" as a generic term for "attackers" and "faults".

R1 Trust in organizational restrictions, i.e. the attacker cannot gain (physical or informational) access to certain system parts.

R2 Trust in physical security techniques, i.e. an attacker, although having physical access to certain system parts, cannot tamper with them successfully (this is, e.g., the basic assumption for chipcards).

R3 Restrictions on the number of simultaneous attackers. (More precisely, not the attackers, but the places where they can attack are counted. For example, in fault tolerance one often assumes that there is just one fault at a time; and for some strong security measures in a network connecting $n$ computers, it is necessary that fewer than $n/3$ of the computers are simultaneously controlled by the attacker.)

R4 Computational complexity restrictions, i.e. an attacker has too little time to compute something, although it can theoretically be computed. Cryptographic systems[4] which need this restriction can be classified further, according to how likely the assumptions are true. This, however, is not a strictly scientific classification, since at present, no sufficient lower bound on the complexity of any problem used in cryptography can be proved from scratch.

R4.1 Ad hoc cryptographic systems, i.e. those which are neither supported by a sufficient theory nor thoroughly investigated by the public cryptographic

---

4 Here, **cryptographic system** is used as the generic term, whereas **cryptosystem** is only used for algorithms for encrypting and decrypting data (e.g., DES and RSA). Other cryptographic systems are, e.g., systems for authentication (such as the GMR Signature Scheme) and identification (such as the Fiat-Shamir Identification Scheme). [1] provides a good introduction to cryptography, [11] gives an overview of modern cryptography. For more details, see, e.g., the CRYPTO and EUROCRYPT conference series of the IACR; the proceedings are usually published by Springer-Verlag, Berlin.

research community. A particularly doubtful case of these are systems where the algorithms are kept secret.

R4.2 Cryptographic systems of medium security, i.e. those which are not supported by theory, but have been thoroughly and publicly investigated for a long time, e.g., DES [1], [8], [9].

R4.3 Cryptographic systems like RSA [10], [1], [8] which have been thoroughly investigated and look like being supported by mathematical theory, i.e. like being of class R4.4, although this cannot be proved (yet).

R4.4 'Cryptographically strong' systems [11], [12], i.e. those whose breaking has been proved to be at least as hard as some well-known old and hard mathematical problem, such as the factorization of natural numbers.

R5 Trust in the correctness of designs and implementations.

Some organizational restrictions (R1) are almost always necessary, since most security measures are carried out by machines, and not by the people who are to be protected; at least machines in those people's houses or pockets must usually be assumed not to work against their owners. The same holds for R5, since sufficient measures to ensure the correctness of designs and implementations are not known (cf. § 2.2).

Among the other restrictions, at least R2 and R4.1 are considered as very doubtful by many people, including us.

If the only restriction on the attackers is that they cannot control other people's private machines, and that these machines are correct, we call the achieved security *unconditional*. (There are, e.g., unconditionally secure authentication and encryption schemes.)

# 2 Security measures

The security measures are sorted according to the problems in § 1.1.

However, many measures against incorrectness (mainly those to distinguish insiders from outsiders, cf. § 2.2.2 and 2.2.3) must also be taken if the basic problem is P1 or P3, since it must be ensured that the system implementing the measures against P1 or P3 is working correctly.

We consider measures against attacks in more detail than those against faults[5].

## 2.1 Measures against complete failures (P1)

Most of these measures try to keep the dependencies between system parts small, so that it can be expected that not all parts fail simultaneously.

M1.1 Decentralization (e.g., networks with several paths between any two nodes; duplicated data bases; fault masking by $n$-of-$m$ systems; keeping the people to be protected, such as the patients, involved).

M1.2 Independent operators, no common insiders. (Of course, this may be impossible with respect to users, but users are the most restricted type of insider attackers during the operation of the system, anyway.)

Some problems remain, e.g.:

- Common physical weaknesses (since measures against them are too expensive, such as shielding of everything or renunciation of miniaturization).

---

[5] Of course, there exist quite a lot of fault tolerance measures, in particular as parts of measures M1.1, M1.2, M2.1. [6] provides a good introduction to fault tolerance. For more details, see, e.g., the FTCS conference series of the IEEE computer society.

- Weaknesses in the overall system design, such as overlooked dependencies between system parts. These may be due to common design or implementation tools, cf. C1.2.1 and C2.2.1.

Therefore, some measures must also be taken outside the systems, at least if the application is important or dangerous:

M1.3 Organize the application as to be in a safe state if the information system fails completely. (This must be combined with fail-stop behaviour of the information system, cf. § 2.2.)

M1.4 Maintain usable old-fashioned back-up systems.

M1.5 Refrain from some applications.

## 2.2 Measures against incorrectness (P2)

Here, the basic principle is to test correctness; and if a fault is found, the measures from § 2.1 must be used.

### 2.2.1 Special measures in the transfer of data

The main measures against incorrectness of data in networks are the following (most of them also help in other systems, in particular, with the storage of data):

M2.1 Error-detecting (correcting) codes[6]:

Against P2.1 only. Not secure against attacks, but fast; e.g., to be used as a line-by-line measure, followed by line-by-line retransmission when necessary.

M2.2 Authentication[7]:

Against both P2.1 and P2.2, but not if P2.2 is caused by a wrong command by an authorized user (i.e. C2.2.2 and parts of C1.2.2). Secure against attacks, if well

---

6 Usually, physical lines are not perfect, i.e. if $A$ sends a message $M$ to $B$, $B$ may receive a message $M^* \neq M$. If this modification is caused by a physical fault, it can, with high probability, be detected (tolerated) by an **error-detecting (correcting) code**. In opposite to authentication techniques, error-detecting (correcting) codes need nothing like a secret key. [41], [44] provide good introductions to coding theory.

7 **Authentication** enables the receiver $B$ of a message $M$ to verify that the message was created by a specific sender $A$, even if an attacker tries to fool $B$.

A scheme which implements exactly this specification is called an **identification scheme**. A scheme which, additionally, enables $B$ to prove to any third party $C$ that message $M$ was created by $A$ is called a **(digital) signature scheme.**

There are unconditionally secure identification schemes, called **authentication codes**, but for most applications they need too long privately exchanged keys [51]. More efficient, and thus more applicable, are cryptographically secure identification schemes like the **Fiat-Shamir Scheme** [40]. The first cryptographically strong (R4.4) signature scheme, the **GMR Signature Scheme**, was presented in [12]. It is nearly as efficient as the less secure RSA-System [10].

Both types of authentication schemes need pairs of keys (or algorithms) ($a$, $t$) privately chosen by the sender $A$, where $t$ is given to $B$. $a$ is used to authenticate a message, $t$ is used to test the authentication of a message. Signature schemes and cryptographically strong identification schemes offer the advantage that $t$ contains no (useful) information about $a$. Thus $A$ can use the same pair ($a$, $t$) for all possible recipients $t$, i.e. the **key distribution problem** becomes relatively simple.

For a general introduction to authentication, see [1], [8], [11], [12].

There are also new signature schemes which are as secure against forgery as the GMR Signature Scheme, but where, if a forgery should nevertheless occur, this can unconditionally be proved (with very high probability) [47], [36]. Thus, there is never any doubt about whether signatures are correct, and the system can be stopped, or security parameters increased, if forgeries occur. Unfortunately, these so-called **fail-stop signatures** are, so far, less efficient than conventional signature schemes in most cases. Recently, even unconditionally secure digital signatures have been invented [39]. There, the key distribution is a complicated protocol, and for many cases they are even less efficient than fail-stop signatures. Thus the feasability of these new schemes depends on the application.

applied. End-to-end measure, since end-to-end and diverse retransmission is often necessary anyway.

M2.2.1 Identification systems[7], if the partners trust each other.

M2.2.1.1 Authentication codes[7]: Unconditionally secure, but inefficient and using enormously long privately exchanged keys.

M2.2.1.2 Cryptographically secure identification systems[7].

M2.2.2 Digital signatures[7], if the partners might disagree on their responsibility (and try to prove it to a third party).

In every case, the keys must be exchanged in a secure way.

Note that none of these systems requires users to be identified by name: Being the owner of a certain key is just a kind of pseudonym. How these pseudonyms are authenticated is application dependent, e.g.,

- there might be a 'telephone book' containing names and keys,
- a central authority, whose signatures are known, might sign keys, or
- if the keys are only used to ensure the internal consistency of a sequence of messages, the keys need not be authenticated by third parties at all.

M2.3 Plausibility tests on higher layers of the system (application dependent) [6].

This is the only measure which also helps against wrong commands by authorized users (but it does not detect "plausible" faults). Parts of this measure can be carried out automatically, but parts must be carried out by the people who would be affected by an incorrectness.

In medical systems, in many cases (e.g., with medical records) this means that correctness should be checked by the patient!

Technically, this is an easy and extremely logical measure, but it might require a change of mind of doctors and a change in legislation.

### 2.2.2 Access control

Access control measures are those which, during the operation of a system, try to prevent people from playing roles which they should not play in the system.

In particular, these measures are intended to keep outsiders out completely, and to prevent normal users from playing operators or changing the system.

A great part of this problem could theoretically be solved by using M2.2, i.e. users would have to prove their authentication to the system. However, without the help of a computer, strong authentication measures are far too time consuming to be performed. Therefore, each user must at least have a tiny computer to carry out this task. This is called an intelligent personal token. In this case, the token must recognize its owner. (This looks like the original problem again, so why use the token at all? The answer is that it is usually easier for users to recognize their tokens than a larger system, hence they are less likely to be fooled into confiding their secrets to fake systems.)

M2.4 Organizational access control measures (against C2.1), e.g.,

- keep terminals at home (does not help against a patient's family, or a doctor's personnel),
- keep chipcards in pockets (but people may lose them, forget to put them into the actual pocket, have no pockets or like them empty; and there are pick-pockets),
- keep data base computers in closed high security zones (but at least in this case, suppliers, maintenance personnel, operators, and cleaning personnel must be considered as possible attackers).

M2.5 Identification of authorized users by PINs, passwords etc. [8]. Some additional measures are necessary:

M2.5.1 One problem is to prevent an attacker from trying all PINs or passwords, at least if they are not much longer than PINs nowadays. All possible measures somehow reduce the number of attempts an attacker can make.

M2.5.2 PINs and passwords must never be transferred through a system owned by someone else. This implies that a chipcard used as an intelligent personal token must have a (small) keyboard of its own.

The same holds if the security of the user depends on an answer of the device: In this case the device must also have a (small) display.

M2.6 Identification of authorized users by biometrics; e.g., recognizing authorized users by their fingerprints [8] Chapters 7.4 - 7.12.

M2.7 Logical separation by secure operating systems. These systems try to guarantee that, among several users of one computer, nobody can read or change data without permission by their owner, or according to another security policy. (See, e.g., IEEE Computer 16/7 (1983) and the Series of the Proceedings of the IEEE Symposia on Security and Privacy.)

However, even the best specified ones, class A1 in [13], [14], are not appropriate for computers dealing with really sensitive data, cf. M2.11.

M2.8 Auditing, i.e. storing who accessed which system part. Auditing presupposes that users can be identified.

This measure is useful if there is a "fuzzy" security policy only, i.e. it is not formalized in advance who may access which system part, but one wants to check it afterwards.

### 2.2.3 Measures to ensure the correctness of the design and implementation

The following measures are meant to ensure the correctness of the design and implementation, i.e. they deal with the causes C1.2.1 and C2.2.1 of problems, in particular with Trojan Horses[3]. For example, it must be guaranteed that system parts which should perform cryptographic algorithms or manage a database or a "secure" operating system work for their owners (cf. § 2.2.2), and not for someone else instead. Unfortunately, none of the following measures is perfect.

M2.9 Verification (i.e. formal proof that an implementation satisfies the demands of a specification) [15], [16].

On principle, verification of the top level specification is impossible. (There, only validation and/or diverse specification can help [17], cf. M2.10 and M2.12.)

Another problem is that such a proof is usually longer than the implementation which it is to prove. Hence it may be incorrect, too. (And if verification is carried out automatically, the verification tools, in their turn, may be incorrect.)

M2.10 Extensive validation, i.e. testing. With systems of very low design complexity, every possible case can be tested.

However, even with systems of moderate design complexity, testing can only show the presence of faults (or attacks), and never their absence.

Some attacks cannot be detected by M2.9 and M2.10 on principle. These are the so-called passive attacks, such as sending data by a covert channel[3]. There, the system behaves according to its specification; it just exploits the freedom which the specification left to the designer in an undesirable way.

M2.11 Reduction of covert channels[3]; e.g. [18], [19], [20]. (A bandwidth of "only" 0.1 bit/s per covert channel seems to be the practical limit the military aims at for the future [13] p. 81. For channels whose existence is known, exactly 0 is attainable on reasonable assumptions, but at considerable expense [21].)

M2.12 Diverse design [22], [23] (in accordance with the decentralization of the system; in particular, if M1.1 is also applied).

This is no help against 'typical' design faults, i.e. types of faults which are likely to be made even by completely independent designers, in particular, programmers.

In order to make the previous measures easier, M2.13 should be applied.

M2.13 Keep the design complexity as small as possible. This means that well structured designs should be preferred to designs where complicated tricks are used, e.g., to optimize speed or size. (The speed or size which can be gained by such tricks is usually not very significant, anyway; and most hardware is cheap nowadays.)

In case something goes wrong in spite of all previous measures, it must be prevented that nobody is responsible for computers.

M2.14 Everybody taking part professionally in the design of hardware, firmware, or software must sign that their design provides specified services to other modules, as long as explicitly specified assumptions are true. Everybody is liable for their statements.

This measure should be supported technically. Signatures should be digital signatures, and design databases should store them. Starting from a few widely accepted physical assumptions, the expected services of a computer system or a network must be deducible from the statements of the designers. This, and that only professionals take part in the design process, must be checked. This checking can largely be automatized. In [21], this is elaborated a bit further: For error and attack diagnosis, it is reasonable that even technical entities, such as compilers, sign their output.

## 2.3 Measures to ensure privacy (P3)

### 2.3.1 Confidentiality of message contents (P3.1)

These measures are mostly designed for networks, but several of them are also useful with the storage of data.

Of course, message contents cannot be kept secret from the communication partner. However, they can (and in many cases must) be kept secret from everybody else. In particular, these are wiretappers and people with access to intermediate nodes in a network.

M3.1 Physical security measures against wiretappers: These are only possible in few systems (in-house), otherwise they are too expensive or impossible. Note that optical fibers, too, can be tapped, although not quite as easily as copper cables and radio transmission [24] p. 36, [25] p. 108.

M3.2 End-to-end encryption[8,9]:

---

8 **Encryption** enables $A$ to conceal a message $M$ in a way such that only $B$ can read $M$.

For this, the recipient $B$ privately chooses a pair $(e, d)$ of keys (or algorithms) and gives $e$ to $A$. $e$ is used to encrypt message $M$, and $d$ is used to decrypt the encrypted message. This is shortly described by $d(e(M)) = M$. **Private key cryptosystems** are systems where it is easy to compute $d$ from $e$, whereas this is computationally impossible for **public key cryptosystems**. Thus for private key cryptosystems, each pair of sender and recipient needs its own secret pair $(e, d)$, whereas for public key cryptosystems the recipient can publish the key $e$ and use the same pair for all senders. A **hybrid cryptosystem** combines both types: a public key system is used to exchange a key of a private key cryptosystem.

Concrete cryptosystems are mentioned in the text. For a general introduction to encryption, see [29], [1], [8], [11].

9 In a communication network where users $A$ and $B$ are connected via intermediate nodes, encryption can be used in two ways:

One is **end-to-end encryption**, i.e. $A$ encrypts and $B$ decrypts and no intermediate node can read the

M3.2.1 Private key cryptosystems.

Disadvantage: the secret key exchange problem, if there are many potential pairs of communication partners. Systems of two security classes are important:

M3.2.1.1 Vernam-cipher = one-time pad: the only unconditionally secure cryptosystem, but the keys are extremely long [26].

M3.2.1.2 Systems of medium security like DES[10] [1], [8]: not cryptographically strong, but publicly known and well investigated. At present, they are more efficient than any public key cryptosystems. (Speed: hardware implementations (one chip): ≈ 32 Mbit/s; software implementations on MC68000, 8 MHz: ≈ 100 kbit/s, on MC68030, 25 MHz: ≈ 800 kbit/s [9], [27].)

Less secure cryptosystems are unimportant, since the efficiency of DES-like cryptosystems suffices for almost every application. Cryptographically strong private key cryptosystems have no advantages (as far as efficiency is concerned) over public key cryptosystems.

M3.2.2 Public key cryptosystems.

Public key distribution is much easier (although the authentication of keys is necessary and never trivial). On principle, public key cryptosystems cannot be unconditionally secure. Speed of hardware implementations (one chip) at present ≈ 200 kbit/s, software implementations ≈ 1 kbit/s.

M3.2.2.1 There are cryptographically strong cryptosystems, but definitely not against active attacks (i.e. if the owner of the key deciphers some messages chosen by the attacker) [28].

M3.2.2.2 RSA [10], [8], [1]: not cryptographically strong, but well investigated, and in some versions seemingly secure against active attacks, too.

M3.2.3 Hybrid cryptosystems combine the easy key exchange of public key cryptosystems with the speed of private key cryptosystems.

With all kinds of cryptosystems, secret keys must usually be chosen by one of the communication partners and must not be shown to any third party.

### 2.3.2 Confidentiality of traffic data

M3.3 Link-by-link encryption[9] (usually using a private key cryptosystem) keeps all traffic data secret from wiretappers [29].

M3.4 Networks without user observability[11]: They can keep most traffic data secret from

---

messages which *A* sends to *B*.

The other is **link-by-link encryption**, where the traffic between any two adjacent nodes is encrypted by the sending node and decrypted by the receiving node. The disadvantage is that the messages are not encrypted within the intermediate nodes. The advantage is that encrypted data can be sent permanently over the link, i.e. garbage is encrypted if there is nothing real to be sent. Hence an outsider cannot even distinguish whether real messages are sent, i.e. cannot obtain any traffic data.

Both modes are discussed in detail in [1], [29]. In many cases, they should be combined.

10 The key space of DES itself is too small, but there are equally efficient variants which seem more secure, e.g., G-DES [9].

11 **Networks without user observability** keep traffic data secret from attackers in intermediate nodes in the network, and keep communication partners anonymous from each other. Apart from networks based on physical security techniques, there are two basic techniques, MIXes [37] and DC-networks [38]. In general, MIXes are less secure, but more efficient than DC-networks. Details about necessary additions to make the basic concepts secure and about efficient implementations can be found in [46], [45], [48], [49],

attackers in intermediate nodes in the network, and can keep communication partners anonymous from each other.

Of course, if anonymity is not wanted, it can be given up by choosing an appropriate identification or authentication system (cf. § 2.2).

In principle, networks without user observability can be formed as a logical structure on any kind of network. In contrast to encryption, however, this results in rather inefficient systems, and implies considerable organizational overhead, since many users must cooperate. For applications requiring only low performance networks (such as electronic mail), however, this may be acceptable.

For applications with higher performance requirements, the only sensible way is to use an appropriate physical network structure. Since in-house networks without user observability seem less important, one can only hope (and encourage) that in future, public networks are constructed in a way which enables unobservable use.

### 2.3.3 No transfer and storage of unnecessary data

These measures strongly depend on the application. However, they are so important that they should not be missing anywhere. (Here, in particular, the last point from § 1.3 applies.) {In networks, they can be seen as the case of keeping message contents secret from the communication partner.}

Some important measures, in particular in networks, are:

M3.5 Use local intelligence wherever possible. In particular, reduce and evaluate sensitive data at the point where they arise and at the time when they arise wherever possible.

It can also be better to transfer a larger amount of unsensitive data, e.g., from a database to a doctor or patient, than a smaller amount of sensitive data from the doctor or patient to the centre.

If a third party must trust the local computation, tamper-resistant devices can be used. Their security is somewhat doubtful, but if, in the other case, the patients would have to trust tamper-resistant devices at the other end (where tamper-resistance is even more doubtful, since the devices are larger), the situation is not worse than before.

M3.6 No unnecessary identifications. This holds for the users of the system, and, more important still, for patients if they are only objects of communication. In particular, names are very often unnecessary, but also insurance numbers, addresses, and other social data (reidentification problem).

M3.6.1 If data from different sources must be put together for some purposes, special pseudonyms can be used.

M3.6.2 A new and very promising idea to avoid identification, even if authorization by a third party is necessary, is CHAUM's credential mechanism[12] [5], [30]. It could conceivably be used by patients to prove that they are insured in a certain insurance without showing their insurance numbers, or to ensure that patients transfer their medical records to a cancer register, but without telling

---

[50], [7], [53], [54].

12 Assume a number of organizations and customers and a number of standard credentials. Each customer $C$ uses a different pseudonym with each organization. Then **CHAUM's credential mechanism** allows a customer to transform a credential received for pseudonym $X$ from organization $A$ into a credential for another pseudonym $Y$, which is used by $C$ for organization $B$. Thus $C$ can show a credential of $A$ to $B$ without allowing $A$ and $B$ to recognize that the receiver and the submitter are identical. The mechanism also ensures that credentials can only be transferred between the different pseudonyms of one customer.

Credentials can be implemented using the RSA-signature scheme [10]; for details see [5], [55]. A typical application of CHAUM's credentials are anonymous payments [5], [32], [33].

under which pseudonym, or that someone does not have a certain disease, without the medical record containing that person's name. (These measures would need some research work.)

M3.6.3 Another new idea is CHAUM's so-called 'undeniable signatures'[13] [30], which might better be called 'signatures which cannot be shown round'. They can be used if data become more sensitive by the fact that they are signed. For example, everybody can claim that someone else has a certain disease, but it is much worse if they can show a signature of this fact, e.g., by a public health department. In this example, and with 'undeniable signatures', employers, who might need such a signature about an employee, could prove that it is a signature in a court, but could not show it to their friends. (These measures would need some research work, too.)

M3.7 Audit trails (similar to M2.8). This means storing who asked for sensitive data and evaluating these data afterwards.

Of course, this measure is only sensible if the fact that someone asks is less sensitive than the data, e.g., if researchers ask for patient data, but not if patients ask questions about diseases. It implies that authentication is necessary if asking for sensitive data (which should be trivial), and that the different authorized people have different authentications. It is useful if, e.g., someone may ask

- for data in some situations, and not in others (e.g., read the blood group out of the health chipcard of people who had an accident, and not of people who just forgot their chipcards somewhere), or
- only for a modest quantity of data, but where it is previously unknown which data exactly (like a researcher with access to a large data base, where reidentification could be possible).

In contrast to the previous measures, audit trails do not prevent the loss of privacy; they only make it less likely, since it can be discovered afterwards.

M3.8 Legislation should require the explicit authorization by the patient, or at least the notification of the patient, for most data transfers.

## 3 Some network parameters

Before we present examples, which are all taken from medical networks, we list parameters which help decide which security measures are suitable in a specific application. These are the parameters which have been mentioned in the discussion of whether some problem is posed or whether a certain measure can be applied.

### 3.1 Parameters of the technical network structure

- Is a public network used or a special network for just this purpose? In the latter case:
  - In-house system or lines running over insecure ground?
- The bandwidth offered by the network.
- The computing capacity of the user stations.

[13] **Undeniable Signatures** are a system where recipients of a signed statement need the cooperation of the signer, if they want to prove to a third party that the signature is valid. This gives the signer as much control about what is done with the signature as possible. The protocol prevents the signer from cheating during this cooperation. (Whether the signer is willing to cooperate is another matter. Therefore, it must be clear in which situations refusal of the signer to cooperate is considered as proof that the signature is valid.)

## 3.2 Parameters of the application

One class of parameters is, of course, which of the problems from § 1 (mainly, § 1.1) are actually posed in the application, and to what extent. Additional parameters are:

- The bandwidth needed for the application.
- The timing requirements, in particular, whether it is a hard real time, interactive, or batch application.
- Must devices be portable?
- Are mobile devices necessary (i.e. devices which must be able to communicate from everywhere)?
- Is a closed or open network necessary, i.e. is there a limited group of potential users (who might trust each other to a certain extent), or not?

# 4 Security in some examples of medical networks

In the following examples, we use the material from the preceding chapters in the following order: First, we mention to which extent the problems from § 1.1 are posed. Secondly, we mention some important parameters of the example (cf. § 3). Finally, we sketch which of the security measures from § 2 should therefore be applied.

## 4.1 Medical or psychological advice databases queried by patients

In this section, we restrict ourselves to the case of typed questions and answers. The case of telephony (i.e. like anonymous pastoral care by telephone) is similar, but all performance requirements are somewhat higher.

**Problems posed:**

P1 Complete failure: not critical

P2 Incorrectness: not very critical (since for critical applications, people should better see a doctor anyway).

P3 Privacy of both message contents and traffic data: critical, i.e. P3.1, P3.2.1 and P3.2.2 (but the latter only for one out of the two communication partners).

**Parameters:**

- A public network must be used. (This also determines the bandwidth offered and the typical user stations.)
- Bandwidth needed: small.
- Timing requirements: interactive or only batch.
- Fixed stations suffice.
- The network must be open to everybody.

**Security measures:**

P1 The availability of the public network will suffice.

P2 Authentication can be provided by having the database sign all answers (together with the corresponding question), i.e. M2.2.2. For this, the only key which needs to be distributed is a public test key of the database.

If somebody is to be legally responsible for the answers given by the database, it can internally contain signatures by the people who entered the data. These would only be shown on demand.

P3.1 Only M3.2, i.e. end-to-end encryption, is applicable. Since people cannot be expected to have exchanged private keys with the database outside the network, a public key cryptosystem should be used (i.e. M3.2.2 or M3.2.3). Hence:

- User stations must have the necessary capabilities. If M3.2.3 is used, software implementations of both cryptosystems should suffice.
- There should be a public standard which cryptosystems to use.
- The database must have published a public encryption key. (The users can transfer their keys for the answers to the database together with the question.)

P3.2 For high security, one must rely on the measures offered by the public network, i.e. it should be a network without user observability (M3.4). (And, of course, the database must not ask for names.)

As measures for the near future, the public network should at least offer link-by-link encryption (M3.3). (Also, all data in the public network should be end-to-end encrypted, since otherwise, somebody who sends encrypted data (see P3.1) might be suspected of having medical problems or even of being a criminal.)

If no network without user observability can be used, one measure should be taken by the database organizers: They should use a common network address for a rather global database or several specialized databases together. This ensures that the network address does not imply the kind of problem someone probably has who dials it. If a distributed database is used, a central MIX[11], owned by the database organizers, can be used to yield the same effect.

If anonymity is indeed offered, and if answers are not free, an anonymous payment system[14] must be used [31], [5], [32], [33]. The value exchange protocol (which ensures that the database obtains the money if and only if the user obtains the answer [34], [35]) can be quite simple, since the database is not anonymous [33] p. 803.

## 4.2 Video transmission during operations

**Problems posed:**

P1 Complete failure: critical, if one completely relies on an external expert watching the operation. However, attacks against a particular patient are quite unlikely, and a global attack would only endanger few operations. (This example is therefore not so critical as a fully automated intensive care unit.) Hence, above all, faults must be considered.

P2 Incorrectness: Small errors in the pictures are not very critical (since pictures are rather redundant). Completely wrong pictures or wrong answers are critical.

P3.1 Privacy of message contents: not so critical. (Possible attackers might be neighbours who know that a certain person is currently operated on, or companies producing horror videos.) (Of course, if the patient's medical record is transferred at the beginning, this is very critical, but this part is not considered here.)

P3.2 Not posed.

**Parameters:**

- In in-house applications, a private local area network can be used. At least for applications spanning more than one city, at present, a public network must be used.
- Since only few user stations are necessary, they can have large computing capacity.
- Bandwidth requirements: quite large (although slow motion video might suffice).

---

14 An **anonymous payment system** allows the customers of a bank to transfer (i.e. to pay and receive) money anonymously. This means that during a transaction, neither the bank nor the payer / payee can identify the payee / payer (i.e. for a payment they need to know each other only by one-time pseudonyms), and that the different transactions of a customer are unlinkable.

Strong anonymous payment system can be implemented using CHAUM's credential mechanism [31], [5], [32], [33].

- Timing requirements: rather critical (real time, but not as fast as most purely technical processes).
- The network (in the logical sense) can be closed, i.e. there is a rather small and well-defined group of people who are allowed to take part in video operations.

**Security measures:**

P1 Lots of standard fault-tolerance measures[5].

P2 Since the question of responsibility may arise, it must be clear in advance whether external experts are responsible for decisions or whether the responsibility remains with the operating team.

In the latter case, weak forms of error detection and authentication suffice (M2.1, M2.2.1.2).

In the former case, M2.2.2 is necessary. This is easy with the answers by the expert, since their bandwidth requirements are small. However, experts can only be called to account for wrong answers if it is clear that they received the right pictures. Therefore, the pictures must also be authenticated. If one trusts a hash function of medium security, and has a hardware implementation of a signature system, this is certainly possible. Otherwise, data reduction would probably be required. If this is impossible, one should refrain from this application.

P3.1 In in-house applications, one might use no measure at all. If a public network is used, one should at least use an encryption system of medium security (M3.2.1.2). Its speed suffices. If there are very few experts, they may all have exchanged private keys. Otherwise, one must use a hybrid cryptosystem (M3.2.3) with a public key of one of the partners. (Cryptographically stronger cryptosystems are not fast enough, if no data reduction is used.)

## 4.3 Monitoring patients in their homes

By this we mean ways of avoiding people falling down helpless or even unconscious without anybody noticing this for days. Either the patient's home can be monitored, e.g., whether the water taps are used regularly, or patients themselves, e.g., their heartbeat.

**Problems posed:**

P1 Complete failure: medium-critical (critical if monitoring is used to keep patients at home who would be in hospital otherwise; with other patients, complete failure only results in the situation which we have today).

P2 Incorrectness:
- False alarm: not critical, but should not happen too often.
- Contrary: like complete failure.

P3.1 Privacy of the message content "alarm" not critical; of all potential other data: critical.

P3.2 Privacy of traffic data:

Case 1: Only the patient's home is monitored. Then P3.2.2 is not posed. P3.2.1 is not too critical, but, e.g., thieves might like to know who needs to be monitored and is therefore probably quite helpless.

Case 2: Patients themselves are monitored. They might like to keep the traffic data of where they currently are secret from the monitoring centre, as long as no alarm occurs, and certainly from everybody else.

**Main security measure:**

In this case, the main security measure is M3.5, i.e. to evaluate the data locally and to send only the value "alarm" or "no alarm".

It implies that there must be some local intelligence, but this is cheap, since only a processor and memory is needed (no CRT, keyboard, or printer).

If patients themselves are monitored, they could even carry all this around.

**Parameters:**

- A public network must be used.
- The amount of data to be transferred is very small, timing requirements are not strict (at least the time an ambulance needs to arrive at the patient must be tolerable).
- The user stations have some computing capability anyway.
- Portable devices are necessary, if patients themselves are monitored, but they can be supported by a fixed station in the patient's home.

**Network security measures:**

P1 Use M1.3, i.e. fail-stop behaviour. This means that missing or wrong data must be considered as an alarm. In this case, as M1.4, the monitoring centre could phone the patient back before sending the ambulance out. Also, since attacks are less likely than faults in this example, some fault tolerance measures should be taken with the local devices and their use of the public network, in order to reduce false alarms.

P2 To detect missing or wrong data, the normal signal "no alarm" should be transferred as 'living zero', i.e. not as nothing or a sequence of zeroes. Otherwise, when the signal "alarm" were to be transmitted, a fault or attack could result in a "no alarm" signal. Instead, the bit sequence from a pseudo random number generator can be used. (This is a special case of M2.2.1.2 for very simple data.) The user stations, which have computing capability anyway, can perform this measure; and since the patient and the centre must have contact before the monitoring starts, the exchange of the secret key (i.e., the seed for the generator) is no problem.

P3.1 The measure against P2 already provides end-to-end encryption (private key cryptosystem). The signal "alarm" is just sent as a deviation from the normal sequence.

P3.2 If only the patient's home is monitored, and no continuous transmission is necessary (as in the water tap example, where a delay of hours can happen anyway), one should use transmission by phone in irregular intervals (not radio transmission, which could be measured from the street).

If patients themselves are monitored, the data should be sent (of course encrypted) by radio with very low intensity to a fixed station in the patient's home. This station transmits them further. The home could even be shielded to avoid the radio transmission being measured from the street.

For all further measures, the same as in § 4.1 holds.

## 4.4 Exchanging data to settle accounts with an insurance

In this text, this example is not considered in detail. The network aspect of its security is simple, but not the main part of the problem. Reasons why the network aspect is easy are:

- The bandwidth required is extremely small and the timing requirements are weak, i.e. every cryptosystem is fast enough.
- There is only a limited number of possible pairs of communication partners. This makes key distribution easier.

However, the main problem is the large quantity of private data which are collected in the insurances, if detailed medical records are exchanged. There are two approaches to this problem, which must probably be combined:

- One is strict access control in the insurances.
- The other is to avoid detailed medical records being exchanged. The organization of this measure depends on how far insurances trust doctors. If they do, there is no problem. If they do not, the controlling function of the insurance can either be replaced by tamper-

resistant devices, or one can try to use the credential mechanism (cf. M3.5, M3.6.2).

## 4.5 Further examples

Further examples of the use of medical networks might be
- establishing and querying cancer registers and other large databases for research purposes;
- providing employers with health data about employees;
- chipcards with emergency data.

All of these are similar to the example in § 4.4 in the sense that there, too, the main problem is not the use of a network (in the technical sense), but the collection of data that may arise at the end, i.e. the main problems are data reduction and access control (§ 2.3.3, § 2.2.2). Of course, the reductions which can be made, and, to a smaller extent, the access control measures applicable, are different in the different examples.

We think that advanced informatics provides innovative ideas for data reduction in these examples, but they would require to be worked out.

# 5 Recommendations

Here, we mention some actions which should be taken independently of the construction of any specific medical network or other informatical system used for health care, or which must be initiated long before the system is to be constructed. (For recommendations concerning the construction of specific systems, see the whole rest of the paper.) We propose technical and legislative actions:

A1 Cryptographic systems should be standardized. (This needs a fundamental change in direction, since a few years ago, the international standardization of cryptosystems has been abandoned for political reasons, i.e. by pressure of the secret services.) As a suitable set of cryptographic systems, we propose (cf. § 2):
- G-DES as a fast private key cryptosystem,
- RSA, enhanced with an appropriate redundancy predicate against active attacks, as a public key cryptosystem, and
- GMR as a conventional signature system.

Prestandardization activities should be started concerning credentials[12], undeniable signatures[13], and perhaps fail-stop signatures[7].

A2 Some cryptographic protocols (using the cryptographic systems from A1) should be standardized, in particular
- key exchange protocols,
- message formats for data to be encrypted or authenticated, and
- for some applications, measures to hide traffic data.

A3 Since a device used as an intelligent personal token must have a (small) keyboard and display, and ordinary chipcards are therefore not suitable for many purposes, the development of such devices should be encouraged.

A4 It must be prevented that nobody is responsible for computers. Since not only specifically designed hard- and software will be used in medical systems, there must be commercial products complying with measure M2.14.

A5 The most promising tool to avoid unnecessary sensitive personal data is CHAUM's credential mechanism [5]. How it can be used to maintain security against fraud, but to avoid detailed medical records, e.g., in insurances, should be investigated by an interdisciplinary team.

A6 To complement A5 and several measures for specific networks, legislative action should be taken to define medical records as property of the patient, and not of doctors or insurances, or at least to give the patient specific rights like that of inspection or correction.

**Acknowledgements.** We are happy to thank Michael Waidner for many helpful comments.

## 6 References

[1] D. Denning: Cryptography and Data Security; Addison-Wesley, Reading 1982; Repr. with corrections, Jan. 1983.
[2] D. Denning: Commutative Filters for Reducing Inference Threats in Multilevel Database Systems; Proc. 1985 IEEE Symp. on Security and Privacy, April 1985, Oakland, Computer Society, 134-146.
[3] D. E. Denning, J. Schlörer: Inference Controls for Statistical Databases; Computer 16/7 (1983) 69-82.
[4] K. Dittrich, J. Schlörer: Anonymisierung von Forschungsdaten und Identifikation anonymer Datensätze; Datenschutz und Datensicherung DuD /1 (1987) 30-38.
[5] D. Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044.
[6] T. Anderson, P. A. Lee: Fault Tolerance - Principles and Practice; 2nd rev. ed., Dependable Computing and Fault-Tolerant Systems Vol. 3, Springer-Verlag, Wien 1990.
[7] A. Pfitzmann, M. Waidner: Networks without user observability -- design options; Eurocrypt '85, LNCS 219, Springer-Verlag, Berlin 1986, 245-253; revised version in: Computers & Security 6/2 (1987) 158-166.
[8] D. W. Davies, W. L. Price: Security for Computer Networks; (2nd ed.) John Wiley, New York 1989.
[9] A. Pfitzmann, R. Aßmann: Efficient Software Implementations of (Generalized) DES; Proc. SECURICOM 90, 8th Worldwide Congress on Computer and Communications Security and Protection, March 13-16, 1990, Paris.
[10] R. L. Rivest, A. Shamir, L. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems; Communications of the ACM 21/2 (1978) 120-126, reprinted: 26/1 (1983) 96-99.
[11] G. Brassard: Modern Cryptology - A Tutorial; LNCS 325, Springer-Verlag, Berlin 1988.
[12] S. Goldwasser, S. Micali, R. L. Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks; SIAM J. Comput. 17/2 (1988) 281-308.
[13] Department of Defense Standard: Department of Defense Trusted Computer System Evaluation Criteria; December 1985, DOD 5200.28-STD, Library Nr. S225,711.
[14] German Information Security Agency: IT-Security Criteria; 1st version; Köln, Bundesanzeiger 1989.
[15] C. A. R. Hoare: The Mathematics of Programming; 5th Foundations of Software Technology and Theoretical Computer Science, LNCS 206, Springer-Verlag, Heidelberg 1985, 1-18.
[16] L. Lamport: A Simple Approach to Specifying Concurrent Systems; Communications of the ACM 32/1 (1989) 32-45.
[17] P. G. Neumann: Flaws in Specifications and What to Do About Them; Fifth Intern. Workshop on Software Specification and Design; May 1989, Pittsburgh, Pennsylvania; Software Engineering Notes 14/3 (1989) xi-xv.
[18] K. Loepere: The Covert Channel Limiter Revisited; Operating Systems Review 23/2 (1989) 39-44.
[19] C. Tsai, V. D. Gligor, C. Sekar Chandersekaran: A Formal Method for the Identification of Covert Storage Channels in Source Code; Proc. 1987 IEEE Symp. on Security and Privacy, April 27-29, 1987, Oakland, 74-84.
[20] C. Tsai, V. D. Gligor: A Bandwidth Computation Model for Covert Storage Channels and its Applications; 1988 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Washington 1988, 108-121.
[21] W. Clesle: Schutz auch vor Herstellern und Betreibern von Informationssystemen; Diplomarbeit, Institut für Rechnerentwurf und Fehlertoleranz, Universität Karlsruhe, June 1988.
[22] A. Avizienis, J. Kelly: Fault Tolerance by Design Diversity: Concepts and Experiments; Computer 17/8 (1984) 67-80.
[23] J. C. Knight, N. G. Leveson: A Reply to the Criticisms of the Knight & Leveson Experiment; Software Engineering Notes 15/1 (1990) 24-35.
[24] J. Horgan: Thwarting the information thieves; IEEE spectrum 22/7 (1985) 30-41.
[25] H. J. Highland: Random Bits&Bytes; The Charade of Computer Security; Computers & Security 6/2 (1987) 108-109.
[26] C. E. Shannon: Communication Theory of Secrecy Systems; The Bell System Technical Journal 28/4 (1949) 656-715.

[27] A. Pfitzmann, R. Aßmann: More Efficient Software Implementations of (Generalized) DES; Interner Bericht 18/90, Fakultät für Informatik, Universität Karlsruhe 1990.

[28] M. Blum, S. Goldwasser: An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information; Crypto '84, LNCS 196, Springer-Verlag, Berlin 1985, 289-299.

[29] V. L. Voydock, S. T. Kent: Security Mechanisms in High-Level Network Protocols; ACM Computing Surveys 15/2 (1983) 135-171.

[30] D. Chaum, H. van Antwerpen: Undeniable Signatures; Crypto '89, LNCS 435, Springer-Verlag, Heidelberg 1990, 212-216.

[31] H. Bürk, A. Pfitzmann: Digital Payment Systems Enabling Security and Unobservability; Computers & Security 8/5 (1989) 399-416.

[32] D. Chaum: Privacy Protected Payments – Unconditional Payer and/or Payee Untraceability; SMART CARD 2000: The Future of IC Cards; Laxenburg (Austria), 19.-20.10.1987, North-Holland, Amsterdam 1989, 69-93.

[33] B. Pfitzmann, M. Waidner, A. Pfitzmann: Rechtssicherheit trotz Anonymität in offenen digitalen Systemen; Computer und Recht 3/10,11,12 (1987) 712-717, 796-803, 898-904; revised version in: Datenschutz und Datensicherung DuD 14/5,6 (1990) 243-253, 305-315.

[34] H. Bürk, A. Pfitzmann: Value Transfer Systems Enabling Security and Unobservability; Interner Bericht 2/87, Fakultät für Informatik, Universität Karlsruhe 1987.

[35] H. Bürk, A. Pfitzmann: Value Exchange Systems Enabling Security and Unobservability; Computers & Security 9/8 (1990) 715-721.

[36] G. Bleumer, B. Pfitzmann, M. Waidner: A Remark on a Signature Scheme where Forgery can be Proved; Eurocrypt '90, LNCS 473, Springer-Verlag, Berlin 1991, 441-445.

[37] D. Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms; Communications of the ACM 24/2 (1981) 84-88.

[38] D. Chaum: The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability; Journal of Cryptology 1/1 (1988) 65-75.

[39] D. Chaum, S. Roijakkers: Unconditionally Secure Digital Signatures; Crypto '90, August 1990, Abstracts, 209-217.

[40] A. Fiat, A. Shamir: How to Prove Yourself: Practical Solutions to Identification and Signature Problems; Crypto '86, LNCS 263, Springer-Verlag, Berlin 1987, 186-194.

[41] R. G. Gallager: Information Theory and Reliable Communication; John Wiley, New York 1968.

[42] B. W. Lampson: A Note on the Confinement Problem; Communications of the ACM 16/10 (1973) 613-615.

[43] K. Loepere: Resolving Covert Channels Within a B2 Class Secure System; Operating Systems Review 19/3 (1985) 9-28.

[44] R. J. McEliece: The Theory of Information and Coding; A Mathematical Framework for Communication; Encyclopedia of Mathematics and its Applications, Gian-Carlo Rota (ed.), Vol. 3, Addison-Wesley Publishing Company, Reading, Massachusetts, Second printing, with revisions, 1979.

[45] A. Pfitzmann: How to implement ISDNs without user observability - Some remarks; Fakultät für Informatik, Universität Karlsruhe, Interner Bericht 14/85.

[46] A. Pfitzmann: Diensteintegrierende Kommunikationsnetze mit teilnehmerüberprüfbarem Datenschutz; IFB 234, Springer-Verlag, Heidelberg 1990.

[47] B. Pfitzmann: Fail-stop Signatures; Principles and Applications; Proc. Compsec '91, 8th world conference on computer security, audit and control, Elsevier, Oxford 1991, 125-134.

[48] A. Pfitzmann, B. Pfitzmann, M. Waidner: Datenschutz garantierende offene Kommunikationsnetze; Informatik-Spektrum 11/3 (1988) 118-142.

[49] A. Pfitzmann, B. Pfitzmann, M. Waidner: Telefon-MIXe: Schutz der Vermittlungsdaten für zwei 64-kbit/s-Duplexkanäle über den (2•64+16)-kbit/s-Teilnehmeranschluß; Datenschutz und Datensicherung DuD /12 (1989) 605-622.

[50] A. Pfitzmann, B. Pfitzmann, M. Waidner: ISDN-MIXes – Untraceable Communication with Very Small Bandwidth Overhead; Proc. Kommunikation in verteilten Systemen, IFB 267, Springer-Verlag, Heidelberg 1991, 451-463; slightly extended version in: Proc. IFIP/Sec'91, Brighton, England, May 1991, Elsevier Science Publisher.

[51] G. J. Simmons: A Survey of Information Authentication; Proceedings of the IEEE 76/5 (1988) 603-620.

[52] A. S. Tanenbaum: Computer Networks; 2nd ed., Prentice-Hall, Englewood Cliffs 1988.

[53] M. Waidner: Unconditional Sender and Recipient Untraceability in spite of Active Attacks; Eurocrypt '89, LNCS 434, Springer-Verlag, Berlin 1990, 302-319.

[54] M. Waidner, B. Pfitzmann: Unconditional Sender and Recipient Untraceability in spite of Active Attacks – Some Remarks; Fakultät für Informatik, Universität Karlsruhe, Interner Bericht 5/89, March 1989.

[55] D. Chaum: Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms; Intern. Conf. on Cryptology, Sydney, Australia, January 1990, AUSCRYPT '90, LNCS 453, Springer-Verlag, Berlin 1990, 246-264.