DATA NETWORK SECURITY
Part 1 Problem Survey and a Model
( preliminary version)

Rolf Blom
Viiveke Fåk
Ingemar Ingemarsson

INTERNSKRIFT
LiTH-ISY-I-0176
CONTENTS

1. Introduction

2. A Communication Scenario; Protection Problems

3. Threats

4. A Communication Model

5. Encryption

6. Authenticators

7. Concluding Remarks
1. INTRODUCTION

Data encryption and related methods may be used to preserve information security in a data network. Here *information security* is defined as the degree to which the destruction, change or loss of information is prevented. Information is defined as the content of the message represented by the data. The information in a block of data is unchanged if the intended result of the transmission of the block is obtained. This means for example that the original message reaches the correct destination where it is interpreted as intended. Undisturbed information does not, in general, require undisturbed data.

The network is supposed to be a public network, accessed by many different users. We are interested in a well-defined group of users who are communicating mainly among themselves. Different groups, however, are also allowed to communicate in a well defined manner. The logical structure of the communication within a group is star-shaped. The information communicated within the group shall be protected against threats from other users of the network, from illegitimate users (wiretappers etc) and from members in the group. The structure of the threats is described in section 3 of this paper.

The network itself and the requirement it imposes are supposed to be unchanged. Encryption and decryption are taking place outside the network. The encrypted data shall comply with the requirements of the network. The communication process in the group consists of time-limited messages which are essentially transmitted from one point to another in the network. This is the basis for the model of the communication which is described in section 4. The model, although simple, enables us to structure the problems in connection with encryption/decryption. This is done in section 5 and 6.
The purpose of the paper is to form a basis for synthesis of security measures by means on cryptological methods. The analysis is general enough to be applied to any data network and any type of user group.
2.1 2. COMMUNICATION SCENARIO; PROTECTION PROBLEMS

When you first look at all the details, that are involved in protection of data communications, you will probably find it hard to make heads and tails of it. Different system architectures, ambiguous use of the nomenclature and other difficulties add to the general confusion.

As an example we can look at an imaginary system with a host computer, a front-end communication computer, a package switching network with concentrators, a small local computer at a branch office and an intelligent terminal. An application in the host computer generates a message, which will be displayed on a terminal in the branch office. The application takes the original string of text characters and adds a check-sum to it. This longer string of characters, is passed to one operating system, where a general block sum is added to the message. The message is then passed to an I/O-handler, which happens to be a remote communications handler. This routine attaches the parametric information about destination and sender directly to the message, adds a sequence number, and sends the bunch of characters to the front-end computer. There the addressing information is removed and transformed, the message is divided into packages and each package is given additional protocol information with addresses, sequence number within message, check sums a.s.o. The packages are sent to the nearest concentrator, where some checks are made, erroneous packages are signalled to be retransmitted, and the whole bunch is finally one by one passed to the local computer. There the checks in the concentrator are performed again as well as some additional ones, which are peculiar to the specific front end - local computer communication. The packages are stripped of their protocol and merged into a single message, which is provided with a new protocol and passed to the intelligent terminal. There all the
remaining protocols with their controls are peeled off one by one and the message is finally displayed on the terminal.

This tiny novel about the life and adventures of a message in a complicated system simply serves to show how difficult it can be to analyze such a situation, if it is viewed in its entirety. The important lesson to be learned is that communication occurs at different levels. What is a mere message at one level is a message plus detailed protocol information at the level above. The link level protocol is common to everyone using that network, but not necessarily to anyone else. The front end computer level protocol is common to everyone communicating with that computer, but not necessarily to users of other computers. The application program’s formats and controls are common to every terminal communicating with that application, but not necessarily to other terminals and applications. Thus, as one goes upwards in the levels, the "message" shrinks and more and more parts are found to be higher level protocols. But each level will add, control, and remove only its own protocol information. Lower level protocols are already peeled off or not yet added, and higher level protocols are just a part of the message. This makes it possible at each level to identify sources, where messages are received from higher levels or generated and protocols are added, nodes, where the protocols are just used and receivers, where the protocol is used and removed from the message.
Figure 2.1 Example of possible levels
3. THREATS

Implementation of security measures in a data network aims at protection against and/or discovers of illegitimate manipulation of the data flow in the network. The threats that occur can generally be classified into the following categories:

a) Passive wiretapping
b) Substitution of messages
c) Insertion of messages
d) Detection of messages

By passive wiretapping we mean that a record of transmitted messages is obtained. Such a record of messages and protocol information can give away sensitive information. For example, cleartext messages can go public, traffic-analysis may reveal a company's modus operandi and hints of how to make an intrusion into the network can be obtained. Passive wiretapping is also the basic threat because it is a necessary tool in effectuating the threats of substitution, insertion and detection of message, it is necessary to know if there is a message or not.

A common collective term for the last three threats b, c and d is active wire-tapping. It is called active, because the threat is that the stream of data is changed in some way. And the purpose of this change of data is to con the intended reciever into doing something different from the right thing. In a data network handling bank transactions we can exemplify the threats by:

b) When a customer makes a deposit, a change of amount of money or of account number in the message is a substitution threat. c) If the message of the deposit is fed into the network another time this is an insertion threat. d) If a withdrawal is made and the message is detected it is a detection threat. The objective of
the wire-tapper is evident in the examples above.

One should not interpret wire-tapper literally in the terms active and passive wire-tapping. The threats are just as actual in any computer or concentrator that is used in the network. Instead of making a "simple" connection to a transmission line the mysterious world of trapdoors and trojan horses in computer programming is used to reach the same goal.
A typical structure of the communication system used by the group of users is shown in figure 4.1.

IT = intelligent terminal (capable of data processing)
NIT = non-intelligent terminal
Typically the communication consists of time-limited messages transmitted one-way between two points in the network. These points may be for example a terminal and the local computer, a terminal and the main computer or a point in the local computer and the main computer. Time-limited communication between two different main computers is also allowed, as indicated by the cotted line in figure 4.1.

The simplest way to describe the transmission of each message is done in the framework of the model in figure 4.2.

![Communication model diagram](source)

**Figure 4.2 Communication model**

Here the node imposes certain restriction on the communication between the source and the receiver. The node may for example include a local computer where the messages are processed. The processing requires certain portions of the message to be clear text (i.e. non-encrypted). Or the node includes the public data network with its requirements on formats, address information etc.

The source output may represent many different points in the communication system. One of the most extreme cases is when the source output is the input data to the terminal. The source output may also be the output from the terminal or the result of a part of the processing in the local computer.

The information in the message from the source is divided into two parts: the node-sensitive and the node-insensitive information. The node-sensitive information is interpreted and used in the node, the node-insensitive
is not. This distinction enables us to distinguish between three types of information protection methods:

1. **Line encryption**, used only between source and node or between node and receiver in figure 4.2.

2. **Message-encryption**, that is encryption of node-insensitive information, can be used all the way through the node.

3. **Verification data**. This can be various forms of data used for example for error detection, verification of message origin etc.

All these methods can be used at each of the levels mentioned in section 2. It should be noted that line encryption, i.e. encryption of every character leaving the source, at one level is equivalent to encryption of only node-insensitive information on the level below. The difference is that in the former case encryption is performed before the information is passed across the interface to the lower level, and thus the responsibility is with the high-level. In the latter case encryption occurs after the interface and the lower level has the responsibility for that protection.

Thus it is possible to find a way through the problems initially mentioned simply by performing the following steps:

1) Identify the levels in the system.

2) Apply the communication model to each of the levels established in step 1. Find their sources, where a "message", i.e. node-insensitive information is given protocol information, i.e. node-sensitive information, their nodes, where node-sensitive information is used, and their receivers, where node-sensitive information
is removed.

3) Study the possible methods of protection and list carefully what threats that counter at each level.

4) Study the lists made in step 3. Tick off every method, that is indispensable at some level, because it offers protection, which can’t be obtained by any other method at any level. Also tick off this threat and any other threats that are met by the method. Take a look at the rest of the threats, and pick out, if possible, a combination of methods at different levels that is optimal in the sense that it covers the remaining threats at a minimum cost of investments, computing time and inconvenience to the users and maintaining staff of the system.

5) Take the lists from step 2 and use them to identify the points in the system, where the methods found in step 4 should be installed. Make the installation.
5. ENCRYPTION

The distinction between line- and message encryption is not important in this section, where we discuss different requirements on the encryption algorithm and its use.

Encryption algorithms can be divided into two different classes, namely: blockciphers and running key ciphers. A blockcipher takes fixed size blocks of symbols and performs a transformation on the block. The transformation does not differ between blocks, that is the key is the same for all blocks. A running key cipher also works on fixed size blocks of symbols. The blocks may contain 1 or more symbols. On each block a transformation is performed but the transformations differ between blocks. This change of transformation is governed by a sequence of keys. Thus the first data block is enciphered with the first key and so on. We observe that strict synchronism must be kept between the key and the data sequence, when encryption and decryption is done. This is not the case for block ciphers.

In general a running key cipher is considered to be stronger than a block cipher. This is partly due to the fact that a block cipher will transform a typical message the same way every time it is sent, while a running key cipher will not show this property. Some specific countermeasures such as block chaining exist, but they have a cost in that additional data must be added to the message.

To be a good block cipher the block size should allow at least $2^{60}$ different keys. That is the block should contain at least 20 bits. (A tacit assumption has been made that we work on binary data). Due to difficulty in constructing practical enciphering algorithms the blocks will contain substantially more than 20 bits when the number of keys is $2^{60}$. Thus we are in situation where the blocks
of a block cipher contains a large number of bits.

This can be good, when the block size or multiples of it approximately matches the length of messages to be encrypted. But when a small number of bits, for example a character of 8 bits, needs to be encrypted we get an unwanted expansion of the message, which degrades system performance. On the other hand a running key cipher can work on very small blocks without losing cryptographic strength, but then we have a synchronisation problem. Thus there are pros and cons for both methods and which method to use must be answered for each specific situation.
6. AUTHENTICATORS

Authenticators should detect any attempt to alter the sequence of messages. Alteration by removal of a message can be detected only if the messages are held together either by counters or by repetition of a part of a message in the next message. Both these methods should rather be regarded as a kind of protocol than direct authenticators. But both of them also adds information, which must be protected from alteration. Thus, once one of these methods has been applied, authenticators protect against any subsequent, undetected alteration of the message stream.

As was noted in section 4, every message leaving a source consists of two parts. The first part comes from the level above (or from the outside world). It is just a sequence of bits to the source. To that sequence the source adds information, which will be used by nodes and the receiver on the same level. This latter part consists of different data items, where the meaning and purpose is completely clear to the source. One of these items may be an authenticator of the rest or only a part of the message. With disregard of the actual physical placing of the pars, we can picture the message as in figure 6.1.

![Figure 6.1](image)

Figure 6.1

Authenticators can be used for

a) the node-insensitive information only

b) the node-sensitive information only
c) the whole message as it is about to leave the source.

The node-insensitive information can't be further subdivided, and hence authenticators for it should give the same amount of protection to every bit of it. The node-sensitive information consists of pieces of known value, and hence only parts of it may be picked out as worthy of protection. If the whole message is to be authenticated, it is not very likely that any part of it should be left out. In all three cases a certain number of bits will be delivered to a procedure which fabricates the authenticator. This can be regarded as delivering an input \( x \) to a function \( f \) in order to get the output \( y = f(x) \). If someone wants to insert a message \( x \) into the stream of valid data, or if he wants to change a message \( x \) into \( x_1 \), he also has to find the correct authenticator \( y_1 = f(x_1) \). Hence \( f \) can't be a publicly known function of any sort, since that would allow any intruder to compute \( y_1 \) and thus get his message authenticated and accepted. \( f \) can then be assumed to be a function of two variables, \( f(k, x) \). \( x \) is then the message, and \( k \) is a secret key, which is known only to the source and receiver and perhaps also the node.

If messages can be inserted and altered, they can also be intercepted and analyzed. Just as \( y \) is a function of \( k \) and \( x \), all possible pairs of \( x \) and their \( y \) are a function of \( k \). If this function is invertible, we can compute \( k \) from known pairs \( (x, y) \). The ideal is if \( (x, y) = f^1(k) \) is a one-way function, which means that it can't be inverted no matter how many valid \( (x, y) \)-pairs that are known. If this ideal can't be achieved, we have to resort to holding the bastions as long as possible. This means that the computation of \( k = f^{-1}(x, y) \) should be as time-consuming as possible. Once it is done it should turn out to have many possible solutions. Every pair \( (x, y) \), however, should have a different set of solutions.
If the latter didn't hold, we could use any of the keys in the first solution and be sure to get the correct $y^1 = f(k, x^1)$. All this can be summarized thus:

The authenticated data $x$ are sent through a function to get $y = f(k, x)$

$k$ is a secret key, which is chosen from so many alternatives, that no one is likely to guess the correct value.

$f$ is so constructed that it is highly unlikely that $f(k_1, x_1) = f(k_2, x_2)$ if $k_1 \neq k_2$ or $x_1 \neq x_2$

$k = f^{-1}(x, y)$ should preferably not be computable.

If $k = f^{-1}(x, y)$ is computable, it should have so many solutions for each pair $(x, y)$ that it still is unlikely to pick the right key from the solutions. Also, in order to weed out one remaining correct key from different $f^{-1}(x_i, y_i)$, a great many pairs $(x_i, y_i)$ should be needed.
7. CONCLUDING REMARKS

To summarize the ideas presented previously in the paper, consider figure 7.1. It shows in a schematic way the threats and counter measures that have been discussed.

![Diagram of data network threats and counter measures](image)

Figure 7.1 Threats and counter measures in a data network

Let us take the counter measures one by one and discuss the effect it has on the possibility to carry out the different threats. Line encryption gives good protection against passive wire-tapping although it probably can't hide information about whether or not a message is sent on the transmission line. However, which message that is sent, is not revealed. In spite of this a chance attach of active wire-tapping may succeed, if no other counter measures exist. For example, injection of previously recorded messages or detection of messages can remain unnoticed. It will certainly remain unnoticed if no message sequence information exists. Also substitution of a part of the encrypted message that does not contain node-sensitive information may not be discovered. Thus line-
encryption and for the same reason, message-encryption should be combined with use of authenticators. Another reason for this is that normal transmission errors also will be detected by the authenticity control.

Message-encryption will give the node-insensitive information protection against passive wire-tapping, but as is said above it should be combined with use of authenticators. If no line-encryption exists the node-sensitive information is revealed to a wire-tapper, which will give him an opportunity to learn how the network operates. Even if the node-sensitive information is protected in a way that makes it impossible to carry out any active wire-tapping threats without discovery, the fact that node-sensitive information is in clear text may make it easy to jam the system into a deadlock. This is a threat that must be seriously considered when the administrative routines of the network are designed.

As have become obvious from what is said above, authenticators is a fundamental counter measure. It can't protect against passive wire-tapping, but it is basis for protection against all active wire-tapping threats.

Up to now, we have talked about encryption without a single reference to how keys for enciphering and deciphering should be maintained and distributed in the network. The same holds true for parameters in authenticator functions. This is quite obviously a very important problem, but its solution can't be given in a general form. The distribution and handling of, let us call it, security parameters, in the network, must be considered in connection with the operation standards of the network.