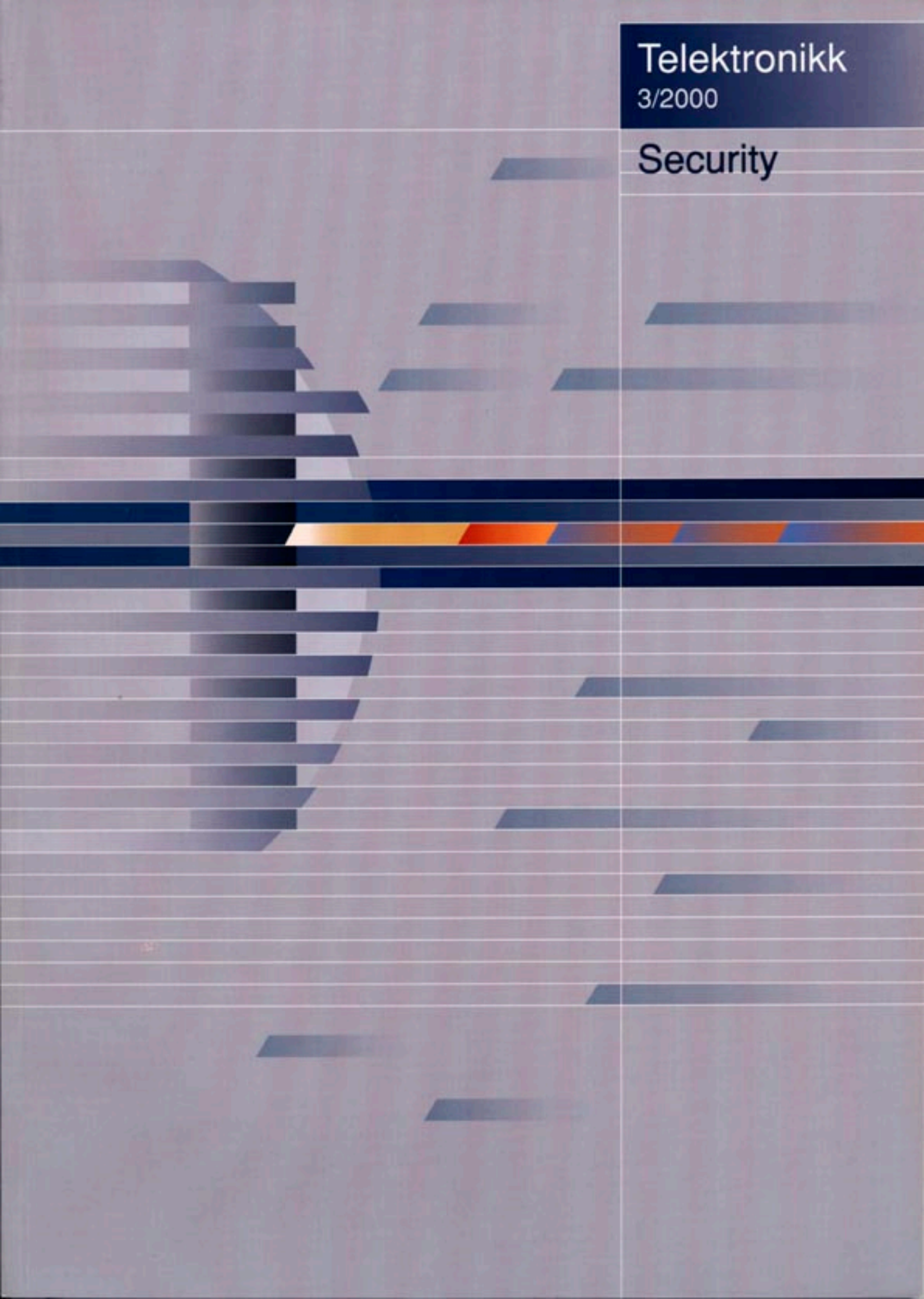


Elektronikk

3/2000

Security



Contents

Teletronikk

Volume 96 No. 3 – 2000

ISSN 0085-7130

Editor:

Ola Espvik

Tel: (+47) 63 84 88 83

e-mail: ola.espvik@telenor.com

Status section editor:

Per Hjalmar Lehne

Tel: (+47) 63 84 88 26

e-mail: per-hjalmar.lehne@telenor.com

Editorial assistant:

Gunhild Luke

Tel: (+47) 63 84 86 52

e-mail: gunhild.luke@telenor.com

Editorial office:

Telenor AS, Telenor R&D

PO Box 83

N-2027 Kjeller

Norway

Tel: (+47) 63 84 84 00

Fax: (+47) 63 81 00 76

e-mail: teletronikk@telenor.com

Editorial board:

Ole P. Håkonsen,

Senior Executive Vice President.

Oddvar Hesjedal,

Vice President, R&D.

Bjørn Løken,

Director.

Graphic design:

Design Consult AS, Oslo

Layout and illustrations:

Gunhild Luke, Britt Kjus, Åse Aardal
(Telenor R&D)

Prepress:

ReclameService as, Oslo

Printing:

Optimal as, Oslo

Circulation:

4,000

Feature: Security

- 1** Guest Editorial;
Øyvind Eilertsen
- 2** An Introduction to Cryptography;
Øyvind Eilertsen
- 10** Advanced Encryption Standard (AES). Encryption for our Grandchildren;
Lars R. Knudsen
- 13** Development of Cryptographic Standards for Telecommunications;
Leif Nilsen
- 21** Show Me Your Public Key and I will Tell Who You Are;
Lise Arneberg
- 26** Security in a Future Mobile Internet;
Henning Wright Hansen and Dole Tandberg
- 34** Mobile Agents and (In-)security;
Tønnes Brekne
- 47** An Overview of Firewall Technologies;
Habtamu Abie
- 53** CORBA Firewall Security: Increasing the Security of CORBA Applications;
Habtamu Abie
- 65** Telenor's Risk Management Model;
Erik Wisløff
- 69** Risk Analysis in Telenor;
Erik Wisløff

Guest Editorial

To a certain extent, security has been regarded as a matter that should be left to military and national security interests. With the advent of worldwide electronic networks, such misconceptions can be dangerous. Luckily, it is now standard procedure to carry out a risk analysis of all new products and systems before they are put into service. A problem is that security considerations are often ignored until the product is almost finished, when someone remembers: "Oh, and we need some security". Security features are among the first to be removed from projects suffering from exceeded budgets. It is an important issue to make sure that security is built into applications from the very beginning. Last-minute introduction of security features will generally lead to both higher expenses and poorer security.

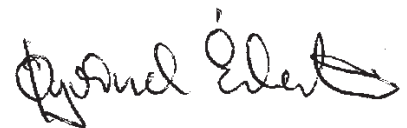
Apart from the actual physical damage resulting from a security breach, the loss of goodwill after a blaring front page in the tabloid press or a prime-time news story must not be overlooked. The latter may actually represent a much more serious and long-lasting loss potential.

This feature section addresses communication security comprising both the security of the users (security services and security as a value added service) and the security of e.g. a telephone operator (security against malicious intruders, competitors, etc.). Physical security (e.g. barbed wire, locks and fences, but also personnel questions) will generally be left out.

Although most users are still not concerned with the security of services, the awareness is certainly growing rapidly, and the ability to offer security-enhanced services will become an ever more important issue for service providers.

Even if the "paper-less office" is still far from becoming a reality, more and more information that previously only existed on paper is now stored on electronic media. While this development makes it easier to share information among those who need it in their daily work, it also increases the potential for outsiders (malicious or not) to intrude. Industrial espionage is a threat that all competitive companies must consider.

In the real world, the so-called "Bribe the secretary attack" is probably the attack with the lowest complexity of them all, whereas cryptographic protection often is the strongest link in the chain. Cryptographic security can be quantified (relatively) easily compared to other aspects that are much less tangible. Thus, we are in the somewhat paradoxical situation that purely academic weaknesses in otherwise secure cryptographic algorithms get a disproportionate interest compared to other security risks. Nevertheless, with the number of available strong algorithms, there is simply no excuse for using weak cryptography.



An Introduction to Cryptography

ØYVIND EILERTSEN

Øyvind Eilertsen (34) is Research Scientist at Telenor R&D, Kjeller, where he has worked in the Security group since 1992. Special interests include cryptography and mobile security.

oyvind.eilertsen@telenor.com

Cryptography constitutes one of the main building blocks of secure communications. While cryptography historically has been largely a military and diplomatic discipline, the development of electronic communication (e.g. the ubiquitous Internet) and the increased use of computers in almost all layers of society have emphasized the need for secure communication solutions also in the civilian sector. A particularly relevant issue is the so-called “new economy”, including electronic commerce, where the parts involved need cryptographic methods to prove their identity and to protect transactions against unauthorized disclosure and tampering.

The purpose of this article is to present an introduction to some of the basic elements of cryptography. We will first look briefly at the history of cryptography and then have a closer look at some cryptosystems that are in use today. We conclude with some remarks about an application of quantum theory with possible far-reaching impact on the future of cryptography.

1 Some Basic Notions

The word *Cryptology* is derived from the Greek words *kryptós* “hidden” and *lógos* “word”. Cryptology comprises *cryptography* (from *kryptós* and *graphein*, “write”), and *cryptanalysis* (*kryptós* and *anályein* (“loosen” or “untangle”). Cryptography is the science of principles and techniques to hide information such that only those authorized can reveal the content. Cryptanalysis describes methods of (unauthorized) solving or *breaking* of cryptographic systems.

A *cryptographic algorithm* or *cipher* is a system (mapping) where a *plaintext* is transformed into a *ciphertext*. This transformation is known as *encryption* (or *encipherment*) and is defined by a *key* that is known only by the sender and the receiver. The inverse mapping is called *decryption* (or *decipherment*).

Breaking a cipher means disclosing the plaintext without prior knowledge of the key. A *perfect* cipher is impossible to break with any (finite) amount of resources.

In its simplest form, the setting consists of the two persons A(lice) and B(ob)¹⁾ who want to communicate securely over an insecure channel. Alice encrypts the plaintext *P* into the ciphertext *C* by means of the encryption function *E* and the key *k*. Likewise, Bob uses the decryption function *D* with *k* to decrypt *C* in order to find *P*. Before the communication can take place, Alice and Bob must have exchanged the key *k* by

means of some secure *key channel*. We write the encryption and decryption as follows:

$$C = E_k(P) \text{ and } P = D_k(C) \quad (1)$$

The adversary E(ve), who is eavesdropping on the insecure channel between Alice and Bob, does not have access to *k* and cannot gain any information from the ciphertext. See Figure 1.

1.1 Ciphers and Codes

The principal classes of cryptographic systems are *codes* and *ciphers*. A *code* is a system where sentences, words, syllables, letters, or symbols are replaced with certain groups of letters or numbers (*code groups*). The code groups (normally 2–5 letters or figures) are listed in a *code-book* together with the corresponding plaintexts. The idea behind the code can be to hide the message from unauthorized persons, to shorten the message in order to save transmission costs (e.g. telegram expenses; bandwidth), or to translate the message into a form that is suitable for transmission (e.g. Morse code). During World War II, the resistance movements in e.g. France and Norway received coded broadcast messages from London where a short sentence had a pre-arranged meaning. “Jean a un moustache très longues” was the code signal sent to the French resistance movement to mobilize their forces once the Allies had landed on the Normandy beaches on D-day.

In this article we will concentrate on ciphers and discuss codes no further.

¹⁾ Alice and Bob were first introduced in [7].

1.2 Cryptanalysis

While cryptography denotes the search for methods and algorithms to protect communication against adversaries, cryptanalysis finds weaknesses in these algorithms (breaks them) to facilitate eavesdropping or counterfeiting. Cryptanalysis is not necessarily “evil”; your friendly cryptanalyst can disclose weaknesses in your systems and propose countermeasures. Much cryptologic research is concerned with finding weak points in existing cryptosystems, and then modify them to withstand these attacks. A frequently used method for testing the security of computer systems is employing a team of experts with knowledge of security “holes” (*tiger team*), and let them try to break into the system.

In earlier times, cryptosystems usually depended on the language in which the plaintext was written. Cryptanalysts employed knowledge of the message language, including the structure, frequencies of letters and words, and the relationships between vowels and consonants. The use of computers for cryptanalysis has generally made such systems obsolete.

Analysis and breaking of modern cryptosystems require advanced mathematical/statistical methods and major computer resources. Typically Terabytes of storage space and operations counted in thousands of MIPS-years are necessary; see e.g. the account of the breaking of RSA keys in Chapter 3.5.

If cryptography is used in a communication system with a significant number of users, one must assume that the details of a cipher cannot be kept secret. If the security is based on the secrecy of the cipher, the system is said to rely on “security through obscurity”, a rather derogatory term within the crypto “community”. This does *not* imply that all secret ciphers are insecure, although this is a common misunderstanding. Obviously, it is much more difficult to cryptanalyze an unknown cipher than a known one, and cryptosystems that protect matters of national security are generally kept secret.

While considering the security of a cipher, secret or public, it is therefore assumed that an attacker knows all details of the cipher, except the actual key in use. This principle was first stated by the Dutch/French philologist Auguste Kerckhoffs in his seminal book *La cryptographie militaire* (1883), and is known as *Kerckhoffs’s principle*. In addition, a common assumption is that an attacker can generate an arbitrary number of corresponding pairs of plaintext and ciphertext for a given key. This is called a *known plaintext* attack, or a *chosen plaintext* attack if the attacker can choose which plaintexts to encrypt.

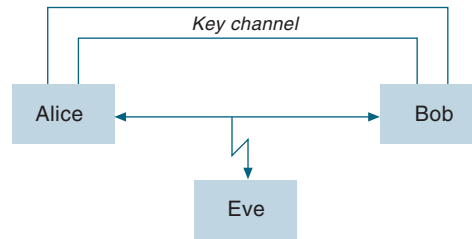


Figure 1 The fundamental cryptographic objective

The *one-time-pad* system, where the key and the message have the same length, was described by the American engineer Gilbert S. Vernam in 1917. If the key is truly random (e.g. resulting from fair coinflips), such a cipher is perfect, as was shown mathematically by Claude Shannon [9] in 1949. The main drawback with this cipher is that the key must be at least as long as the message, and it must be used only once (hence the name). Because of the key exchange problem, one-time-pad systems are used only in environments where security is paramount and the messages are rather short. As an example, a one-time-pad system developed by the Norwegian telephone manufacturer STK was used to protect the “hot line” between Washington D.C. and Moscow in the 1960s.

2 Some History

The history of cryptography is probably as old as the history of writing itself. Of course, during most of the history of writing, the knowledge of writing was enough – there was no need to encrypt messages (or encrypt messages *further*), because hardly anyone could read them in the first place. There are a few examples of Egyptian scribes playing around with the hieroglyphic symbols, possibly to attract the curiosity of the readers, much like rebuses. (Considering the problems modern linguists encountered trying to read hieroglyphs, one is tempted to say that they were thoroughly encrypted in the first place.)

2.1 The Caesar Cipher

One of the most well-known ciphers in history is the *Caesar cipher*, probably employed by the Roman emperor Gaius Julius Caesar. Each letter in the alphabet is “moved” three places to the right, so that *A* is replaced by *E*, *B* with *F*, etc. This is an example of a simple *substitution* cipher, as each letter in the plaintext is replaced with another letter (*monoalphabetic* substitution.)

The Caesar cipher is a simplified case of the *linear congruence* cipher. If we identify the English letters with the numbers 0–25, encryption and decryption is defined as follows: (We assume the reader is familiar with the modulo notation.)

$$E(P) = a \cdot P + b \pmod{26}$$

$$D(C) = \bar{a} \cdot (C - b) \pmod{26}, \quad (2)$$

with $a\bar{a} = 1 \pmod{26}$

The key is the number pair (a, b) , where we have the additional restriction that a must be invertible modulo 26. This is the case if and only if a and 26 have no common factors except 1. We write this as $\gcd(a, 26) = 1$, where \gcd stands for “greatest common divisor”. The Caesar cipher uses $a = 1$ and $b = 3$, so unique decryption is guaranteed.

The main problem with a simple substitution cipher is that the statistical variations in the plaintext language are still present in the ciphertext. The cipher is therefore easily broken with frequency analysis of single letters, pairs (bigrams) and triples (trigrams) of letters. For instance, in ordinary written English, the letter ‘e’ accounts for about 13 % of the letters, the letter ‘q’ is always succeeded by ‘u’, and so on. Obviously, the longer a message is, the more substantial the statistical skews. A plain monoalphabetic substitution that contains more than 40 letters can (almost) always be trivially broken, because only one plaintext is possible. Later cryptographers introduced countermeasures to statistical analysis, e.g. by letting several ciphertext letters represent the most common plaintext letters.

2.2 Polyalphabetic Substitution

A significantly more advanced method is the *polyalphabetic* cipher, which uses several ciphertext alphabets. Around 1470 (according to Kahn [4]), Leon Battista Alberti described a “cipher disk” with two concentric circular metal disks mounted on a common axis. Along the circumference of each of the disks an alphabet is outlined; one for the plaintext and one for the ciphertext. By turning the disks, the ciphertext alphabet is changed. Alberti’s device can be used to construct *polyalphabetic* ciphers. Polyalphabetic ciphers were described by Blaise de Vigenère in 1586 and are often called *Vigenère* ciphers.

The polyalphabetic ciphers were considered unbreakable for almost 400 years, even though several cryptologists in their writings were close to discover a general method for breaking the ciphers. The method was at last described by the Prussian officer Friedrich W. Kasiski in 1863. This did not, however, stop a device principally identical to Alberti’s disk from being patented in Norway in 1883 under the name “Strømdahls kryptograf”. This device was used in the Norwegian defence up to the Second World War. Similar systems were used by the US Signal Corps in 1914.

2.3 The Nomenclator

The most widely used (indeed almost the only) method of protecting secret information from the 15th to the 18th century was the *nomenclator*. A nomenclator is a kind of mix of a codebook and a cipher. It contains hundreds (or even thousands) of letter groups that represent commonly used words and phrases, typically together with an extended monoalphabetic cipher.

2.4 The Rotor and Mathematical Cryptanalysis

Around 1920 several people independently thought of using electrical versions of the cipher disks, so-called *rotors*. By stacking several rotors serially, one can construct polyalphabetic ciphers with arbitrary complexity. Cipher machines include the Swedish *Hagelin* and the *Enigma*, which was patented by the German engineer Arthur Scherbius in 1918.

In the 1920s and 1930s rotor machines were produced commercially in several countries, including USA, Sweden and Germany. The machines were used by both military and civilians

In the 1930s, growing unrest in Poland over the German military build-up led to extensive Polish efforts to encounter the German cryptosystems (The Enigma). With help from the French intelligence service (and the German spy Hans-Thilo Schmidt [5]) the Poles got the details of the Enigma, and led by the brilliant mathematician Marjan Rejewski, they managed to outline the principles for breaking the encryption.

During World War II, all parties used rotor-based cryptographic machinery; the German *Enigma* (several different models), UK *TYPEX* and USA *SIGABA*. The British headquarters of cryptanalysis was located at Bletchley Park north of London. Here, up to 12,000 people worked in the utmost secrecy with the decryption of Enigma cryptograms. Based on the work by Rejewski prior to the war, and further developed by (among many others) Alan Turing, the Allied Forces were able to more or less routinely break and read German secret communication. It has been argued that this work probably shortened the war by at least one year. The different parts of the German army used different variants of the original Enigma design. The *Wehrmacht*, *Kriegsmarine*, *Luftwaffe*, and the military intelligence (*Abwehr*) all had their own version of the Enigma, which were constructed from the same starting point, but all had their own idiosyncrasies that had to be dealt with by the Allied cryptanalysts.

The Enigma was (principally) used for encryption of Morse traffic. In addition, the Germans used the cipher machines SZ 40 and 42 (*Schlüs-*

selzusatz) for teletype traffic, as well as the Siemens T 52 *Geheimschreiber*. At Bletchley Park, the foundation of modern electronic computing was laid down. The first electronic computer (“*Colossus*”) was constructed to break the German *Lorenz* cipher.

In USA, the Japanese ciphers (dubbed RED, BLUE and PURPLE by the Americans) were routinely broken from the mid-1930s. The American ability to read Japanese secret communication probably had significant influence on the outcome of the war in the Pacific, e.g. the battle of Midway.

After World War II, the British government dispensed Enigma-type cipher machines to several former colonies (India, Pakistan, etc.), but the recipients did not know that the cryptography could be broken. This was not generally known until the end of the 1960s. (Actually, the fact that German Enigma messages were being broken and read by the Allies was not published until 1974 [12].)

3 Computers and Modern Cryptography: Crypto goes Public

The electronic computer has had an enormous impact on cryptography. As we have seen, the first electronic computer was developed specifically for cryptographic purposes.

On the one hand, the computer and the international communication networks (e.g. the Internet) make it possible to move data around the globe in seconds. On the other hand, this interconnectivity opens new possibilities for fraud.

In a time when all banking orders were sent as tangible pieces of paper, a fraudster needed to get hold of the specific piece of paper. When the order is sent electronically, however, the fraudster may sit in a different part of the world but nevertheless have access to the data.

An interesting scenario is a thief stealing very small amounts of money from a very large number of victims, a so-called “salami attack”. If the amounts are sufficiently small, the victims will not notice, but the grand total will be significant, because the number of victims is so large. It is e.g. conceivable that a thief may collect fractions of cents from interest calculations. This scenario has often been cited, but it is not known to have occurred. A less subtle variation is actually transferring small amounts from accounts in the hope that the victims will not notice until their next account statement, at which time the thief is long gone.

3.1 The Data Encryption Standard

To keep up with the need to protect banking, health, and other sensitive communication, while allowing intercommunication, the US government moved to make public a cryptosystem. The National Bureau of Standards (now the National Institute of Science and Technology or NIST) invited interested parties to offer candidates. A handful of algorithms were presented, and after a period of scrutiny and adaptations, the Data Encryption Standard was published in 1976.

The winning algorithm was based on an algorithm called *Lucifer* from Horst Feistel of IBM. The 128 bit key of the original design was reduced to 56 bits, and additional so-called S-boxes were introduced. The adaptations were suggested by the National Security Agency (NSA), and this immediately caused great controversy. Non-governmental cryptographers maintained that the NSA either had deliberately weakened the algorithm by reducing the key size to a breakable level, or that they had incorporated a “trap door” that enabled the NSA to easily break ciphertexts.

No substantial evidence was presented, however, and DES was published as a Federal Standard for “un-classified government communication” in January 1977. DES was also adopted by the American National Standards Institute (ANSI) for use in the private sector in USA, and by ISO as an International Standard for protection of banking communication (ISO 8730 and 8732).

3.1.1 The DES Key

DES is by far the most widespread and also the most publicly analyzed algorithm. No traces of trap doors or deliberate weakening have been found, but with the rapid development of computing power the 56 bit key is now under serious attack.

In 1998 The Electronic Frontier Foundation (EFF) presented their DES breaking machine called “Deep Crack”. With this hardware device, it was possible to check all the $2^{56} \approx 7.2 \cdot 10^{16}$ keys in less than 72 hours. To counter this type of attack, the newest version of the DES standard [3] recommends only use of *triple-DES*. In this scheme, three instances of the original DES algorithm is used as follows:

$$C = E_{K3}(D_{K2}(E_{K1}(P))) \quad (3)$$

The sequence encryption-decryption-encryption was chosen to facilitate interoperability with older DES implementations; if $K1 = K2 = K3$ equation (3) reduces to simple DES encryption. A common variation is letting $K1 = K3$, giving a key size of $2 \cdot 56 = 112$ bits.

In 1997, NIST initiated the process of finding a successor to DES, known as the *Advanced Encryption Standard* (AES). The selection process for AES is the theme of an article by Lars Knudsen in this issue of *Teletronikk*.

3.2 Stream Ciphers and Block Ciphers

Symmetric algorithms are usually divided into *block* and *stream* ciphers.

A block cipher breaks the plaintext message into strings (*blocks*) of a fixed length (the *block length*) and encrypts one block at a time. With a given key, a pure block cipher will always produce the same ciphertext from a given plaintext. In many applications this is not a favourable feature, and some kind of feedback must be introduced.

A stream cipher produces a *key stream* $k_1k_2k_3 \dots$. The key stream is combined with plaintext stream $p_1p_2p_3 \dots$ using a simple transformation E and producing the ciphertext $C = c_1c_2c_3 \dots$, with $c_i = E(p_i, k_i)$. Usually, E is the XOR operation, i.e.

$$c_1 = p_1 \oplus k_1, c_2 = p_2 \oplus k_2, c_3 = p_3 \oplus k_3 \dots$$

In a *synchronous* stream cipher, the sender and the receiver must be *synchronized* to allow for correct decryption. This means that they must use the same key stream and operate at the same position in that key stream. If the synchronization is lost, decryption will fail, and re-synchronization, e.g. re-initialization of the stream cipher is necessary.

In contrast, in a *self-synchronizing* stream cipher the key stream is generated as a function of the encryption key and a fixed number of previous key stream bits. If ciphertext bits are deleted or inserted, only a fixed number of plaintext bits will come out garbled before the synchronization is re-established.

Most stream ciphers are based on *linear feedback shift registers*, a structure very well suited for fast hardware implementations. The ciphers that protect the confidentiality of the radio communication in GSM and UMTS are both stream ciphers.

The difference between stream ciphers and block ciphers is a bit "academic". Observe that a stream cipher may be considered a block cipher with block length 1. On the other hand, there are standardized modes of operation that turn a block cipher into a stream cipher.

3.3 Building a (Symmetric) Block Cipher

Obviously, an encryption function E must be complex, preventing unauthorized reversal of the transformation. Modern block ciphers achieve this goal by combining simple functions in clever ways.

3.3.1 Confusion and Diffusion

Ciphers are based on two basic techniques (operations); *transpositions* and *substitutions*. A *transposition* changes the order of the symbols (permutation), without changing the symbols themselves. In a *substitution*, each symbol is replaced by another symbol (from the same or some other alphabet).

3.3.2 Product Ciphers

Almost all modern block ciphers are *product ciphers*. The idea is to build a complex encryption function by composing several simple functions, each offering some, but not adequate, protection. The simple functions are then combined in such a way that the combination is more secure than the individual components. Basic operations include transpositions, linear transformations, arithmetic operations, modular multiplications and simple substitutions.

3.3.3 Iterated Ciphers

An *iterated block cipher* involves a sequential repetition of an internal function called a *round function*. Let F be the round function, r the number of rounds, and let T_i be the temporary output of the round function. Then we get these equations:

$$\begin{aligned} T_1 &= F(T_0, K_1) \\ T_2 &= F(T_1, K_2) \\ &\vdots \\ T_r &= F(T_{r-1}, K_r) \end{aligned}$$

The plaintext is the input to the first round (T_0), and the ciphertext is the output from the last (r^{th}) round (T_r). The K_i are the *round-keys* that are derived from the encryption key K according to the *key scheduling*. In order to make unique decryption possible, the round function must be a bijection for all round-keys K_i .

3.3.4 Feistel Ciphers

A *Feistel cipher* is an r -round iterated block cipher with block length $2t$. The plaintext is the ordered pair (L_0, R_0) , where the t -bit values L_0 and R_0 represent the left and right half-block, respectively. In each round i , $1 \leq i \leq r$, (L_{i-1}, R_{i-1}) is mapped to (L_i, R_i) as follows:

$$L_i = R_{i-1}, \quad R_i = L_{i-1} \oplus F(R_{i-1}, K_i) \quad (4)$$

where K_i is the round-key of round i . DES is an example of a Feistel cipher.

3.3.5 Round Keys (Key Scheduling)

In the round function, the output of the last round is mixed with the current round-key. Typically, these entities are approximately equal in length, i.e. equal to the block length.

The encryption key must be expanded (typically r -fold). Ideally, all round-key bits should be dependent on all encryption key bits to maintain maximal entropy. The expansion of the encryption key to round keys is known as *key scheduling*.

3.3.6 S-boxes

An $m \times n$ *substitution box* or S-box is a mapping that takes an m -bit input and returns an n -bit output. A one-to-one $n \times n$ S-box is a permutation.

S is non-linear if for two numbers a and b , where $a \neq b$, $S(a) \oplus S(b)$ is not equal to $S(a \oplus b)$. S-boxes are typically used to provide non-linearity (often they are only non-linear operations in an encryption algorithm), and are thus very important.

A number of criteria for selection (generation) and testing of S-boxes have been proposed, i.e. the strict avalanche criterion (SAC), which states that flipping one input bit should cause a change in (on average) 50 % of the output bits.

S-boxes are generally visualized and implemented as look-up tables, but some may additionally be computed arithmetically. Smart-card implementations of look-up tables are vulnerable to power attacks (attacks that measure power consumption on the chip when performing encryption). Countermeasures to power attacks include duplicating the look-up tables in RAM; thus large S-boxes are more difficult to protect than small ones. If an S-box can be implemented arithmetically, protection against power attacks becomes much easier.

3.4 Public Keys

In 1976, Diffie, Merkle, and Hellman described the principles for *asymmetric* or *public key* cryptography. The principle is to use different keys for encryption and decryption, thereby avoiding some key management problems. In addition, asymmetric cryptography presents the ability to make *digital signatures*, with approximately the same features as ordinary hand-written signatures. It has turned out that researchers at the British Communications-Electronics Security Group (CESG) discovered these principle

early as 1970; see [2]. This work was, however, not made available to the non-governmental crypto-community until 1997.

In a traditional (symmetric) cipher, the sender and the receiver must have access to a common, secret key. This key must be distributed in a secure way before the communication takes place. In an *asymmetric* cipher, each user has a *private* and a *public* key. Asymmetric ciphers are therefore often called *public key* ciphers. In principle there is a distinction here, as it is conceivable to have an asymmetric cipher without public keys. The distinction is, however, somewhat academic, and no secret-key asymmetric cipher has been published, so we will treat these as synonyms. The private key is only known to the user (it is called "private" rather than "secret" in order to avoid confusion with symmetric ciphers), while the public key is broadcast to all parties the user wants to communicate securely with.

A message that is encrypted with a public key can only be decrypted with the corresponding private key, and *vice versa*. Knowledge of a private key shall not make it possible to reconstruct the corresponding private key.

3.4.1 Usage

Assume that Alice wants to send a confidential message to Bob. Alice encrypts the message with Bob's public key. Now only somebody who has access to Bob's private key (presumably only Bob) can decrypt and read the message.

Assume on the other hand that Alice does not care about secrecy, but she wants every reader of the message to be sure that Alice is the source. She then encrypts the message with her own *private* key. Now everybody with knowledge of Alice's public key can read the message. In addition, since the message can be decrypted with Alice's public key, only Alice can have encrypted it. This is a kind of *electronic signature*. Alice can combine these features by first encrypting the message with her own private key, and then encrypt the result with Bob's public key. Now only Bob can decrypt and read the message, and he can be convinced that Alice is the source.

Asymmetric ciphers are typically much slower than symmetric. Hybrid systems are common, where messages are encrypted with a symmetric cipher, while the (symmetric) key(s) are protected with an asymmetric cipher. An efficient variation of the digital signature above, is to generate a *hash value* (or *checksum*) of the message, encrypt it with one's private key and send it together with the message. A recipient can

decrypt the signed hash value, recompute the hash value of the received message, and compare the two values.

3.5 RSA

While there is a large theory on the construction of symmetric ciphers (see Chapter 3.3), no general method is known for construction of asymmetric ciphers. Existing asymmetric ciphers are based on suitable mathematical problems, of which *discrete logarithms* and *integer factorization* have shown the most promising results. The most widespread public key cryptographic algorithm is *RSA*, which was published in 1978 by Rivest, Shamir and Adleman [7]. (The algorithm is obviously named after its inventors.) The security of *RSA* is based on the intractability of integer factorization.

3.5.1 The Algorithm

To construct an *RSA* public/private key pair, Alice finds two large prime numbers p and q and chooses a number e . The public key consists of the pair (m, e) , where $m = p \cdot q$. Typical values for e are 3, 17, and 65537. Alice then solves the equation

$$e \cdot d \equiv 1 \pmod{(p-1)(q-1)} \quad (5)$$

with respect to d . The private key is d , together with p and q . Encryption and decryption is done by computing powers modulo m :

$$C \equiv P^e \pmod{m} \text{ and } P \equiv C^d \equiv P^{ed} \pmod{m} \quad (6)$$

The decryption works because $ed = k(p-1)(q-1) + 1$ for some integer k , and by Euler's theorem

$$P^{(p-1)(q-1)} \equiv 1 \pmod{pq} \quad (7)$$

If an adversary (Eve) can factor the modulus m to find the prime numbers p and q , she can easily solve equation (5) and find the private key d . Presently, no efficient method of integer factorization is known, but it has not been proved that no such algorithm exists. It is not proved that integer factorization is necessary for breaking *RSA*, but no other efficient attacks are known, except in very theoretical cases. Boneh and Venkatesan [13] present strong evidence that breaking low-exponent *RSA* *cannot* be equivalent to factoring.

As of today, the largest *RSA* modulus whose factorization has been published had 155 digits (512 bits). The main part of the work was done using a network of 285 workstations and PCs, running for 3.5 months, while the final calcula-

tions were done on a Cray supercomputer. The total amount of CPU time was estimated to 8000 MIPS-years.

Most applications now use 1024 bit *RSA* keys. Compared to a 512-bit key, factorization of a 1024 bit key will demand an increase in CPU time by a factor $5 \cdot 10^6$ and in RAM capacity by a factor of at least $6 \cdot 10^3$.

4 A Quantum Future?

David Kahn notes in [4] that "The war of cryptographer against the cryptanalyst has been won by the cryptographer."

He argues that present-day ciphers are so difficult to break, even with enormous amounts of computing power, because of the exponential nature of the work factor (rule of thumb: When the cryptographer does twice as many calculations, the work factor of the cryptanalyst is squared.) A so-called *quantum computer* may actually reduce the work of the cryptanalyst from exponential to linear again. As an example, an efficient factorization algorithm is described in [10]. Whether a quantum computer will ever be built is quite another matter. Today, this "beast" is well outside the range of practical implementations, even if reports pop up from time to time.

On the other hand, *quantum key distribution* utilises the uncertainty principle to provide key distribution between two parties with no prior common (secret) key [1]. If the parties have access to a separate channel with only a passive adversary (i.e. the adversary cannot actively influence on the signals), the key distribution as provably secure, even against adversaries with unlimited computing power.

5 Suggested Further Reading

David Kahn's *The Codebreakers* [4] is the major work on the history of cryptography up to the 1960s. However, it was first published before the details of the Enigma solutions were published. These events are the theme of a large number of accounts, including Kahn's *Seizing the Enigma* [5] and Gordon Welchman's *The Hut Six Story* [11].

Much of the theoretical background of modern ciphers was laid down by Claude Shannon in the papers *A Mathematical Theory of Communication* [8] and *Communication Theory of Secrecy Systems* [9]. The most comprehensive work on present-day cryptography is the *Handbook of Applied Cryptography* [6] by Menezes et al.

References

- 1 Bennett, C et al. Quantum cryptography, or unforgeable subway tokens. In: *Advances in Cryptology – Proceedings of CRYPTO'82*, 1983, 267–275.
- 2 Ellis, J H. *The Possibility of Secure Non-secret Digital Encryption*. January 1970. (CESG report.)
- 3 NIST. *Data Encryption Standard (DES)*. Federal information processing standards publication. 1999. (FIPS PUB 46-3.)
- 4 Kahn, D. *The Codebreakers*. [Rev. ed.] New York, Scribner, 1996.
- 5 Kahn, D. *Seizing the Enigma*. London, Arrow Books, 1996.
- 6 Menezes, A J, van Oorschot, P C, Vanstone, S A. *Handbook of applied cryptography*. London, CRC Press, 1997.
- 7 Rivest, R L, Shamir, A, Adleman, L. A method for obtaining digital signatures and public key cryptosystems. *CACM*, 21 (2), 120–126, 1978.
- 8 Shannon, C E. A mathematical theory of communication. *The Bell System Technical Journal*, 27, 379–423, 1948.
- 9 Shannon, C E. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28, 656–715, 1949.
- 10 Shor, P W. Algorithms for quantum computation : Discrete log and factoring. In: *Proceedings of the 35th FOCS, Los Alamitos*, 116. IEEE Computer Society Press, 1994.
- 11 Welchman, G. *The Hut Six Story*. M&M Baldwin, 1997. ISBN 0947712348.
- 12 Winterbotham, F W. *The Ultra Secret*. London, Weidenfeld and Nicholson, 1974.
- 13 Boneh, D, Venkatesan, R. Breaking RSA may not be equivalent to factoring. In: *Advances in Cryptology – EUROCRYPT'98*, 57–71, 1998.

Advanced Encryption Standard (AES). Encryption for our Grandchildren

LARS R. KNUDSEN



Lars R. Knudsen (38) is Professor at the Institute of Informatics at the University of Bergen. He received his M.Sc. and Ph.D. degrees in computer science and mathematics from Aarhus University, Denmark, in 1992, respectively 1994. He has written numerous scientific papers in the area of cryptology and is regarded a world expert in block cipher encryption algorithms.

lars.knudsen@ii.uib.no

Introduction

Encryption used to be something which only the secret services and military had a real interest in and which private citizens only knew from crosswords and puzzles in Sunday newspapers. Today encryption is an important part of the information society. Outside of the secret services the interest in encryption started to blossom at the end of the 1970s.

First, IBM (International Business Machines) developed the cryptosystem Lucifer, which later was adapted as a US Federal Information Processing Standard, although slightly modified. This standard was published in January 1977 as the DES (Data Encryption Standard) and is today probably the most used encryption system in the world (at least outside of the secret services). The system is a so-called *secret-key* cryptosystem, where the same information, or *key*, is used to encipher (or encrypt) and decipher (or decrypt) the messages.

Second, the researchers Whitfield Diffie and Martin Hellman discovered (or re-discovered¹⁾) so-called *public-key* cryptography, where the secret key is split into two parts, a public part and a secret part. The public part of the key is made available to everyone; the secret part stays secret with one party. The public key can be used by everyone to encrypt a message, while the secret key can be used to decrypt the cipher-text and restore the message.

The differences between today's secret-key and public-key cryptosystems are many, but there is a need for both of them. Even though the DES has withstood almost 25 years of cryptanalytic attempts to find shortcuts in the algorithm by cryptanalysts from all over the world, time is running out for the algorithm. The main problem is that the DES was designed to accept keys of only 56 bits, which means that there are $2^{56} \approx 10^{17}$ different keys. Even though this number may seem huge, (as an example, 2^{56} seconds are about 2 billion years), it is small enough to enable the design of special-purpose built hardware, which can run through all possible values of the key in a very short time. In 1998 it was estimated that an attacker who is willing to invest one million US dollars, could try all values of the key, one by one, in just half an hour!

With a few encrypted messages on hand, one can simply decrypt these under all possible values of the key. The value of the key which yields some meaningful messages is with a high probability the correct one, and the system is broken.

Technical Detail

In a cryptosystem the message is always first converted to a number. This number is then encrypted by applying some mathematical or non-mathematical operations to it, and the resulting number is then transformed back to cipher-text. The numbers are represented in the binary number system, that is, a number is either a zero or a one. As an example, the number 17 in the decimal number system (the one we use normally) is 10001 in the binary number system. The symbols in the binary number system are called *bits*.

AES – Advanced Encryption Standard

In 1997 the American National Institute for Standards and Technology (NIST) decided that it was time to find a substitute for the DES. Surprisingly (at least to this author) NIST invited parties from all over the world to participate in this process and announced a call-for-candidates for the Advanced Encryption Standard (AES). The conditions for the competition were many and included a whole range of documentation requirements and test results.

The most important requirements for the system are that there must not be any trapdoors (shortcuts), and that the best attack against the system is the trivial one of trying all keys one by one. A more specific requirement is that the secret keys must be of length of at least 128 bits. This means that there will be at least 2^{128} different keys, which is about 10^{39} . The above mentioned special-purpose built machines to try all keys one by one will not have a chance of being applicable in practice before 2030 – 2040, or probably even later.

NIST also invited the world's cryptanalysts to participate in the process. The goal of NIST is that the whole process be as open as it can be, and that all aspects of the design and analysis are made public.

¹⁾ The English intelligence service claims that they invented the public-key techniques around the same time.

The deadline for submitting candidates to the AES was June 15, 1998. Out of a total of 21 submissions, six were discarded because of incomplete documentation. Of the remaining 15, five are from the USA, two from Canada, there is one candidate each from Australia, Belgium, Costa Rica, France, Japan, Korea, and Germany, and then a multinational candidate from Denmark, United Kingdom and Israel. This author represents the Scandinavian colours in this competition.

After one year of gathering information about the 15 candidates NIST decided in August 1999 to pick five candidates for a final and last round. This author was involved in the breaking of two of the 15 candidates and in the finding of serious weaknesses in a third candidate. The five candidates for the final round are in alphabetical order.

- **MARS** by IBM, USA;
- **RC6** by RSA Inc., USA;
- **Rijndael** by researchers from Belgium;
- **Serpent** by researchers from Denmark, UK, Israel;
- **Twofish** by Counterpane, USA.

In April 2000 the last conference on the AES took place in New York, USA, and May 15, 2000 was the deadline for sending in comments and analysis of the five candidates. NIST expects to announce the winner(s) some time in the year 2000.

Serpent

Serpent is a snake; the idea is that Serpent will slither away from all cryptanalytic attacks. My co-authors on Serpent are Ross Anderson from Cambridge University in England and Eli Biham from Technion University in Haifa, Israel. The first version of Serpent (later called Serpent-0) was developed in 1997 and presented at a conference on encryption in Paris, March 1998. The version we submitted to NIST, called Serpent, is a slightly modified version of Serpent-0. Today (July 2000) no one has managed to find any weaknesses of any kind in Serpent.

Secret-key cryptosystems are traditionally constructed by running the message through several so-called substitutions and permutations dependent on the value of the secret key. Substitutions are also sometimes called *S-boxes* and are often implemented in terms of a look-up table, which for every input specifies the function value. The advantage of this approach is that it is relatively easy to choose and use functions with complex mathematical formulae. Permutations are often simple functions which permute (or re-order) the bits of the messages typically, one uses a set of small substitutions each modifying a small piece of the message, but such that the whole text is modified. Subsequently, the pieces are moved

around and mixed. This recipe is then repeated a sufficient number of times, until the resulting ciphertext looks like total gibberish (and often more than that).

Serpent is constructed as above and has 32 iterations or layers. In each layer the 128-bit text is split into 32 smaller parts of four bits each. The four bits are input to a small S-box, which again returns four (other bits). Then the 32 blocks of four bits are concatenated (put together) and the 128 bits are mixed using a permutation. The nice feature of Serpent is that the 32 S-box evaluations can be done in parallel. Most computers today operate on 32-bit words, which enables us to look up 32 S-box values in parallel; that is, on computers with just one processor. This means that the 32 look-ups are much faster than doing 32 conventional look-ups. On 8-bit processors it is possible to do eight evaluations in parallel. The substitutions and permutations are well chosen, such that all known attacks on block cipher have to give up after 7 to 9 layers. Therefore there is a big safety margin in Serpent, big enough to handle even considerable improvements in the known techniques.

On the average PC Serpent is not the fastest algorithm of the final five candidates left in the competition. On the other hand, on other platforms, e.g. in smart card applications, Serpent is one of the fastest; also in hardware Serpent is the fastest of the five. The great advantage of Serpent is that the safety margin protecting against future cryptanalytic improvements is the largest of all five candidates.

Licenses?

One of the great properties of the AES, apart from high security (if Serpent is chosen!) is that the system must be royalty free and free to use for everybody all over the world. It was a condition to participate in the competition that all patents and rights were waived, in case the algorithm should be selected for the AES.

Hidden Trapdoors

One of the favourite subjects in the boulevard press when it comes to encryption system is hidden trapdoors. As an example, when the DES was first published there was a lot of debate on the possibility that the American government had put in a trapdoor enabling them to read encrypted traffic without knowing the secret key. However, I am convinced that no such trapdoor exists for the DES, and I guarantee that no such trapdoors have been put into Serpent. It is very hard to break the encryption systems which are constructed according to the state-of-the-art, but it is even more difficult in my opinion to put a trapdoor into a public cryptosystem without being detected.

Table 1 Timetable for AES

01/97	Announcement of AES
04/97	First workshop in Washington, USA
06/98	Deadline for submission of candidates
08/98	First AES conference in Ventura, USA, with presentation of candidates
03/99	Second AES conference in Rome, Italy
08/99	NIST announces the five candidates for the final
04/00	Third and last AES conference in New York, USA
??/00	Announcement of the AES winner(s)

On the Need of Encryption

In a modern society when we send a (conventional) letter, we expect that only the intended recipient can open the letter and read it. Likewise, we probably expect that documents on our desk are inaccessible to the public. It ought to be exactly the same for electronic information, no matter if it is sent over a public channel or if it is stored on a hard disk. I say "ought to", because I do not think that this is the case today, unfortunately. By far the most electronic mail (e-mail) is sent in clear text today, which at least in principle makes it accessible to everyone. And how many people encrypt the data on their hard disk? Encryption does not require a change in attitude, I think, since I believe everyone would use it for both the above purposes, if all it took was to press a button on a screen. I am perfectly aware that the existence of a good encryption system is not sufficient to enable secure e-mail communication. One needs to solve the problem of how do the sender and receiver share a secret key between them such that no one else knows it? This is not a trivial problem, but there are known solutions.

And the Winner is ...

Who knows? NIST will decide sometime this year (2000). There are rumours that NIST is

Name	Country
Cast256	(Canada)
Crypton	(Korea)
Deal	(Canada)
DFC	(France)
E2	(Japan)
Frog	(Costa Rica)
HPC	(USA)
Loki97	(Australia)
Magenta	(Germany)
<u>Mars</u>	(USA)
<u>RC6</u>	(USA)
<u>Rijndael</u>	(Belgium)
Safer+	(USA)
<u>Serpent</u>	(Denmark/England/Israel)
<u>Twofish</u>	(USA)

Table 2 The 15 AES candidates. The underlined algorithms are the final five candidates

going to choose several candidates for the standard. The advantage of this is that the users get more flexibility, since one scheme could be better suited for a certain platform or application and another scheme for some other platform or application. Furthermore, with several algorithms, one would be able to switch from one algorithm to another in case the first one should become vulnerable to some cryptanalytic attack. This would save us having to go through a long tedious process like the one for the AES will be. The disadvantage of having more than one algorithm is that not everyone would use the same and that the AES therefore should not become as popular as the DES. There are applications, e.g. smart cards, where it is not feasible to implement more than one encryption system.

NIST claims that they will consider all submitted attacks and comments, and that the whole process will be open to the public. This is probably true but with one small modification. I think it is very likely that NIST will receive input from the NSA (the National Security Agency). This input will probably not be accessible to the public. There is of course also politics involved in the AES. Is it likely that NIST will choose a non-American algorithm for a federal American standard? I do not know, but I think that if NIST chooses more than one algorithm, a non-American algorithm will be one of them.

NIST does not have the means to pay money to any of the designers and analysts. Also, the winner(s) will not receive any direct monetary benefit from NIST. All the work of the submitters and people who have spent hours analysing the candidates is unpaid and voluntary.

Why do we do it then? I guess we do it for the fame and recognition one would get in case one's algorithm is selected or even being among the final five. Also, the AES is a chance to give a good encryption algorithm royalty-free to people from all over the world, and to our children and grandchildren.

More Information

More information can be found at the official AES-site <http://www.nist.gov/aes/>; or my site <http://www.ii.uib.no/~larsr/aes.html>, or my Serpent-page <http://www.ii.uib.no/~larsr/serpent>.

Feature Editor's Note:

On October 2, 2000, NIST announced that the Belgian algorithm Rijndael is the winner of the AES contest. NIST said that Rijndael was chosen because of its combination of security, performance, efficiency, ease of implementation and flexibility. NIST tentatively expects the AES standard to become official in April – June 2001.

Development of Cryptographic Standards for Telecommunications

LEIF NILSEN



Leif Nilsen (48) is Senior Cryptologist with Thomson-CSF Norcom AS and Associate Professor at the University of Oslo Graduate Studies at Kjeller (UNIK). He is working with information security, cryptographic algorithms and key management systems. He is Norwegian expert to ISO/IEC JTC1/SC27 and member of ETSI/SAGE.

leif.nilsen@norcom-csf.no

The development of secure information systems encompasses the use of system technical security solutions based on cryptographic techniques. The article provides a basic tutorial on cryptographic algorithms and describes the development of cryptographic standards for use in telecommunications. We review the set of algorithms that has been developed for dedicated use within ETSI standards. This development has been targeted to balance divergent technical and political requirements and has produced a variety of system specific solutions. The article also includes a survey of the most influent standardisation initiatives for general-purpose cryptographic algorithms.

1 Introduction

Moving towards an *information society* where we see a rapid integration of *information systems* and *communication systems*, more and more of public, industrial and private business take place in an open, distributed and digital marketplace. This trend implies an increasing need for security services for protection of both user data and network management data. The ISO Security Architecture [1] defines the following set of security services and also specifies at which layer of the communication protocol the service should or could be implemented:

- Peer Entity Authentication
- Access Control
- Connectionless Confidentiality
- Connection Integrity with Recovery
- Non-Repudiation
- Data Origin Authentication
- Connection Confidentiality
- Traffic Flow Confidentiality
- Connectionless Integrity.

The use of *encryption algorithms* is a basic technique for the implementation of several of the services mentioned above, even if the traditional use of cryptography has been to provide information confidentiality. For an excellent historical overview of the use and importance of crypto

systems, see D. Kahn [2]. In this paper we will discuss several issues related to the development and use of cryptographic standards with emphasis on their use in telecommunication systems.

2 Basic Model of a Cryptographic System

In order to give some tutorial background to how cryptography is used in a communication system, we will briefly discuss the basic model of a cryptographic system as shown in Figure 1.

The sender *A* wants to send the message *M* to the receiver *B*. The communication will take place over an insecure channel, where an eavesdropper *Z* may listen and get unauthorised access to the information passing over the channel.

By applying the encryption device *Enc*, the entity *A* transforms the message *M*, often called the *plaintext*, into an unintelligible message *C*, called *ciphertext*, under the control of the encryption key K_E . The receiver *B* has the corresponding decryption key K_D and is able to recover the plaintext *M* from the ciphertext *C* using the decryption device *Dec*.

In a symmetric cryptosystem $K_E = K_D$ and the exchange of keys has to take place over a confidential key channel. In an *asymmetric* crypto

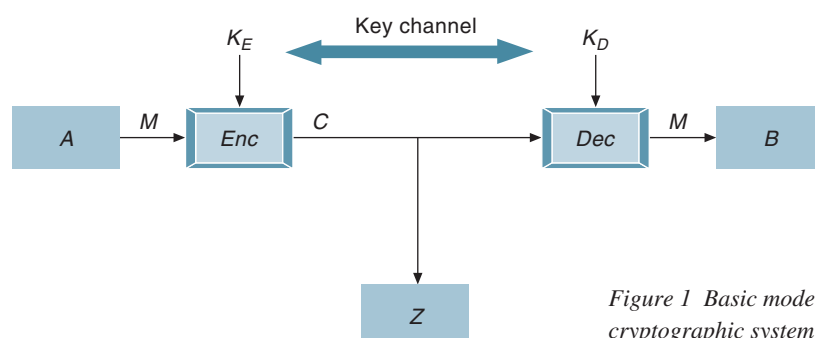


Figure 1 Basic model of a cryptographic system

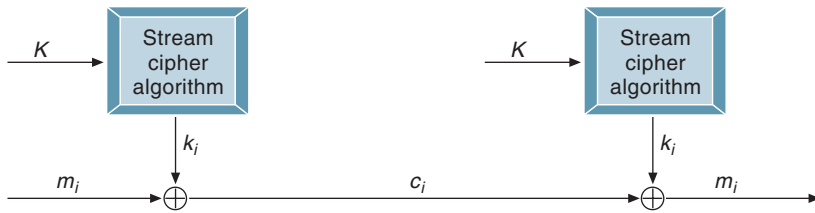


Figure 2 Principles for synchronous stream cipher

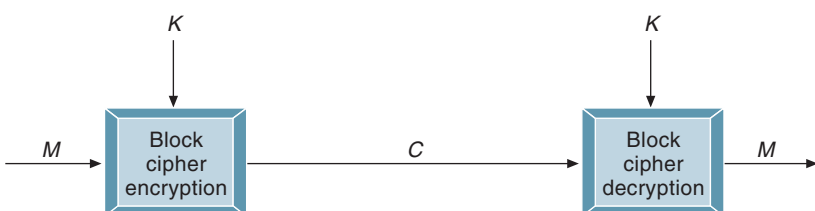
system (also called public key system), it is impossible to find K_D from knowledge of K_E and the encryption key can be a public parameter belonging to the entity B . In this case there is a need for an authenticated key channel from B to A . Current systems often solve this distribution problem using certificates and a Public Key Infrastructure (PKI).

For symmetric systems, the algorithm defining the encryption/decryption transformation can either be a *block cipher* or a *stream cipher* depending on the use of internal memory in the encrypting device.

A stream cipher breaks the plaintext message into successive characters or bits m_1, m_2, \dots, m_n and encrypts each m_i with the i 'th element of a key stream k_1, k_2, \dots, k_n derived from the basic key K and optionally an additional message key. Most stream ciphers operate bitwise by adding the message bit and the key stream bit modulo 2, i.e. $c_i = m_i \oplus k_i$. The receiver can now recover the plaintext bit m_i from the ciphertext bit c_i by adding with the corresponding key stream bit k_i . Note that this requires full bit synchronisation between the encrypting and decrypting devices. The general principles for a stream cipher are shown in Figure 2.

A block cipher breaks the plaintext message into successive blocks M_1, M_2, \dots, M_n and encrypts each block using the same key K . A typical block length is 64 or 128 bits. For a fixed key K , the cryptographic transformation can then be seen as a permutation on the set of all possible blocks. The US Data Encryption Standard DES [3] is a well-known example of a block cipher using a 56 bits key and operating on 64 bits data blocks. Note that a block cipher can operate in

Figure 3 Principles for block cipher



different modes as defined by the international standard ISO/IEC IS 10116 [4]. By operating a block cipher in Cipher Feedback Mode (CBC) or Output Feedback Mode (OFB), a block cipher behaves like a stream cipher. There are also standardised methods for using a block cipher to compute Message Authentication Codes (MAC) [5] and for hashing operations [6]. This means that a strong and efficient block cipher is a handy tool for implementation of several security services. The general principle for a block cipher is shown in Figure 3.

The strength of a cryptographic system relies heavily on the two components, the encryption algorithm and the key management system; neither is simple to design. For the encryption algorithm several security requirements can be summarised in the following: Without knowledge of the secret key, it shall be impossible for an attacker to find the plaintext, given the ciphertext; or from known plaintext/ciphertext pairs to find the secret key. This means that the security of the system does not depend on the secrecy of the algorithm itself. When an algorithm is analysed, it is important to assume that the opponent has detailed information about the specifications.

3 The Strength of Cryptographic Algorithms

Cryptographic algorithms are fundamentally different from other algorithms with respect to their intended goal. Normally an algorithm is designed to solve a specific problem like error correction or finding a path through a complex network. We are then normally able to prove that the algorithm solves the problem completely, even if other and more efficient algorithms may be found. As long as we can meet operational requirements, the algorithm will continue to do its job and it will never be "worn out" or "broken".

The goal of an encryption algorithm is to protect information against all possible attacks of known and unknown enemies. Even if we are able to prove that an algorithm is resistant to a specific attack, we can never be sure that it can withstand a novel attack tomorrow. Perhaps we can design an algorithm that is secure against the computational power of a single enterprise. With the current growth in processing power, the same algorithm could be vulnerable to governmental agencies and dedicated hardware in a few years' time. We see today that some of the most modern and secure crypto systems will be easy tasks for new computational models like DNA computing [7] and quantum computing [8].

A theoretical foundation for cryptographic systems may be based on *information theory* or *computational complexity theory*, but neither of

these approaches can be used to design practical systems that provide provable security in a strong sense. Most algorithms are designed using a system-theoretic approach combining well-established principles with partial security proofs.

All cryptographic algorithms can be attacked by a brute force attack on the key space. Assuming an enemy has knowledge of the algorithm involved, he can try to decrypt the ciphertext with all possible keys. Meaningful plaintext will appear when the proper key is found. If the length of the secret key is n bits, there are 2^n different keys to try. This means that the key length is a simple parameter used to indicate the strength provided by a cryptographic algorithm. However, it is important to recognise that a sufficient key length is a necessary but not a sufficient requirement for a strong algorithm. An algorithm could have a long key, but still be vulnerable to other attacks more efficient than exhaustive key search. A report from 1996 [9] gives guidelines for selection of key lengths as shown in Table 1.

It is important to stress that this table is only relevant for symmetric algorithms. The situation for asymmetric ciphers is much more complex and adequate key lengths for such systems are completely different from those in Table 1. An excellent guide is [10].

4 Encryption Policy

The implementation of encryption solutions is not only a technical matter, but also sensitive political issues are involved. For a long time serious use of cryptography was restricted to diplomatic and military use and encryption technology was considered to be of strategic importance to national security. National and international regulations were developed to monitor the use and dissemination of the technology. Most countries still enforce some kind of export control, but the rules have gradually become more liberal.

The introduction of confidentiality as a standard service in telecommunication systems will not only protect the involved link against fraud and malicious eavesdropping, but it will also prohibit police and security agencies involved in legal interception as part of their combat against terrorism and organised crime. In many countries such interception has been an important tool against serious criminal activity, but normally a special court order must be available before the interception can take place.

In mobile systems like GSM and DECT, the radio link is considered to be an exposed and vulnerable channel and the standards specify encryption of this link. The solutions involved

Type of attacker	Key length needed
Pedestrian hacker	45-50
Small business	55
Corporate department	60
Big company	70
Intelligence agency	75

Table 1 Guidelines for selection of key lengths

have been a carefully chosen design balancing the specific threats involved and the need for general exportability. For such systems there are no export controls on the terminals, but a license is normally needed for network equipment.

5 Open or Secret Algorithms

As stated above, the strength of a cryptographic system should not rely on the secrecy of the system description. In order to provide optimal analysis and confidence in a cryptographic algorithm, it will be important to have public available specifications that have been studied by independent experts. By standing such public scrutiny over a long period of time, an algorithm can achieve the necessary trust and assurance. This is a strong argument in favour of open algorithms and it seems clear that this is the only model for wide acceptance of general-purpose cryptographic algorithms.

Open algorithms may be subject to a variety of research analysis and many results are published as “breaks”, even if the announced attack cannot be conducted within the operational environment of the system. Normally any attack with complexity lower than exhaustive search is reported as “cracking the algorithm”, often resulting in much publicity and worried users. In many cases such results are mostly of academic interest and have minimal impact on the security of the actual system.

However, in the same way as openness is no guarantee for strong algorithms, a secret algorithm does not implicitly mean that the algorithm is weak. Cryptographic algorithms used in military systems for protection of classified information are seldom or never revealed. These organisations have a strong internal competence in design and analysis of algorithms and do not have the same need for an open process. From history they know the difference between attacking a known versus an unknown crypto system. In order not to provide an enemy cryptanalyst with any advantage, they prefer the use of secret algorithms.

6 Use of Encryption in Telecommunications Area

Traditionally the use of encryption technology in the telecommunications area has been seen along the following lines:

1. Protection of network management information. Operator initiated encryption of dedicated connections in order to secure important connections vital to the operation and maintenance of the network itself. This approach may work well within one domain controlled by one operator, but may face problems in multiple domain solutions implementing the Open Networks Provision policy.

2. End-to-end encryption by dedicated users with high security requirements. Users that exchange confidential information over public networks using secure phones or crypto-faxes. Such solutions normally depend on expensive and proprietary equipment. Due to the complexity of key management, the solutions do not always scale well and there are no standards for such connections.

3. Encryption of vulnerable links. Modern mobile networks like GSM, DECT, UMTS and satellite-based networks involve a radio link especially vulnerable to eavesdropping (as well as other security threats). In order to protect user data and signalling information in such systems, encryption techniques have been introduced on the radio link between the mobile terminal and access points to the fixed network.¹⁾

4. Entity authentication. Mobile communications lacks the conventional access point to the network and there is a strong need for authentication of users involved in a call. It may also be necessary for the user to authenticate the network. In most systems such authentication is based on a “challenge and response protocol” in which an entity authenticates itself by proving knowledge or possession of a unique secret key. The challenge and key are then input to some cryptographic algorithm which outputs the correct response.²⁾

7 The Data Encryption Standard (DES)

For many years the US Data Encryption Standard (DES) [3] has been the de-facto standard for commercial encryption. DES was proposed in 1975 and approved in 1977 as a Federal Standard for protection of sensitive, but unclassified information. DES was designed and proposed by IBM, endorsed by the National Bureau of Standards (NBS, now NIST) and later approved by ANSI as US standard. Around 1985 there was work within ISO to establish DES as an international crypto standard, but this project was halted due to political problems. DES is widely used by banks for protection of electronic funds transfer.

DES is a symmetric block cipher, which encrypts a 64-bits data block under the control of a 56-bits key. From the very first moment there was strong criticism against the key length of DES and it was argued that the key space could be searched by strong opponents using dedicated hardware devices. In July 1998, using custom designed chips and a personal computer, the Electronic Foundation built “DES Cracker” [11]. Costing less than USD 250,000 and taking less than a year to build, DES Cracker broke a DES-encoded message in fifty-six hours. There was nothing terribly novel about the decryption machine except that it was built. From Table 1 in Section 3 we see that this result fits nicely with the predictions from 1996.

The immediate response to the shortcomings of the DES key length has been to implement Triple-DES systems, in which the DES algorithm is used in three consecutive steps using two or three different keys. The long-term solution will be to develop a replacement algorithm(s), see section on AES below.

Even if DES is approaching the end of its life cycle, it has been an example of a carefully designed algorithm, which have resisted open analysis over many years. Today we know that it was designed to withstand attacks that were not publicly known back in the seventies [12]. Even new attacks have appeared over the years, but they had little impact on the practical strength provided by DES.

¹⁾ Due to interoperability requirements, this encryption has to be based on standardised algorithms known both to the mobile station and the network.

²⁾ This authentication is often a protocol between the subscriber’s SIM module and the operator and may be based on an operator-controlled algorithm. In this case it is not necessary to have one standardised algorithm.

8 ETSI Standardisation

ETSI is the European Telecommunications Standards Institute. It has approximately 450 members and its main objective is to produce technical standards in the area of telecommunications.

The specification of security standards for a specific telecommunications area or system within ETSI is in principle carried out by the responsible Technical Committee (TC) or ETSI Project (EP). For general security issues there is a dedicated Technical Committee called TC Security. For the design and specification of cryptographic algorithms a Special Committee was installed: the Security Algorithm Group of Experts (SAGE). Unlike other TCs or EPs, SAGE is a closed group with an appointed membership.

The outline procedure for the design of cryptographic algorithms for ETSI standards is that the ETSI TC or EP first decides if there is a need for a standard algorithm. Then the TC/EP drafts a document specifying the requirements for this algorithm. These requirements normally describe issues like use of the algorithm, implementation complexity, performance, resilience, exportability and management of the algorithm and its specifications. The document also specifies if the algorithm should be published or confidential. If needed the TC Security assists the responsible committee to draft the requirements specifications.

In the next phase ETSI SAGE designs and specifies the algorithm according to the requirements. The algorithm is then delivered to the algorithm custodian (in most cases this is ETSI) which takes care of the distribution of the algorithms to the intended users. SAGE also produces a report for the ordering committee describing the work done, the rules achieved and the rules for management of the algorithm.

9 Algorithms Used in ETSI Standards

In this section we will review several standards developed by ETSI and describe the use of cryptographic algorithms in those standards.

GSM – The Global System for Mobile Communications

GSM was the first public standard digital telecommunication system which included substantial use of cryptographic algorithms. The original system employed a standard encryption algorithm called A5 for encryption of user data and signalling information. Only the vulnerable radio link was protected by this encryption and the traffic was in clear within base stations, switches and networks. When GSM began to expand beyond Europe there were difficulties involved in exporting the system to certain countries and

the need for an export algorithm was identified. The original A5 now became A5-1 and a new export version A5-2 was designed by SAGE. Both algorithms should be implemented in the handset and it is up to the network to decide which algorithm to use. Both A5-1 and A5-2 is based on stream cipher technology.

The GSM system also uses an algorithm for authentication of the user and generation of the encryption key. This algorithm is called A3/A8 and is not a standard algorithm in the system. The operator is free to design their own algorithm or they can use an example algorithm from the GSM Association called COMP128. Some security problems have been found in COMP128 and a replacement example algorithm is now available.

For the new data services in GSM, the General Packet radio service (GPRS), a special encryption algorithm called GEA was developed by SAGE. SAGE also developed a special authentication/integrity/key generation algorithm for GSM Cordless telephone System (CTS) in 1999.

DECT – Digital Enhanced Cordless Telecommunications

DECT has security features that are similar to those in GSM. As in GSM it uses an encryption algorithm, the DECT Standard Cipher (DSC), and an authentication and key generation algorithm called the DECT Standard Authentication algorithm (DSAA). Like A5-1 both DECT algorithms were designed before SAGE was set up, but the algorithms have recently been reviewed.

ISDN based audio-visual system

CCITT (ITU) has drafted recommendations H221, H261 and H233 in the area of the use of audio-visual systems and specifies security for these. The CCITT recommendations were adapted by ETSI. Recommendation H233 (“Confidentiality for audio-visual services”) specifies the use of encryption and allows different algorithms to be used. SAGE designed an encryption algorithm especially for this purpose. It is called BARAS (baseline Algorithm recommended for Audio-visual Services).

Multi-application telecommunications cards

A sub-committee of the ETSI TC terminal Equipment (TE) drafted a series of standards for Multi-application telecommunications IC (Smart) card. The specifications included a number of security functions.

To support these functions SAGE was asked to design a cryptographic algorithm called TESA-7. The specification included four modes of use for the algorithm. These are authentication

mode, integrity mode, key diversification mode (i.e. generating an individual key from an identity and a master key) and a secure key loading mode.

So far there has been little use of the Multi-application standards and it has recently been agreed to broaden the use of TSEA-7 to the GSM SIM (Subscriber Identity Module – the smart card of GSM).

UPT – User Personal telecommunications

UPT is a telecommunication service standardised by ETSI that enables users to register on a foreign telephone and then be reached there under their own telephone number. This service requires authentication before it can be invoked.

ETSI SAGE designed a standard authentication algorithm, called USA-4, for this service. However, until now there has been limited use of the UPT standard and hence the USA-4 algorithm.

Hiperlan – High Speed radio LAN

Hiperlan is a standard for high-speed radio LAN over which data is transmitted at high speeds over the air interface. For this standard SAGE developed a dedicated encryption algorithm called HSEA (Hiperlan standard Encryption Algorithm). The export restrictions on the algorithm should be minimal and the algorithm provides a basic level of security. The ETSI project BRAN is currently standardising a successor for Hiperlan. This standard (BRAN) will support higher data rates and it seems that it will employ a standard encryption algorithm.

BEANO – Binary Encryption Algorithm for Network Operators

A few years ago the ETSI TC Security identified the need for an algorithm that could be used to protect the confidentiality of network management data. ETSI SAGE designed a special encryption algorithm called BEANO (Binary Encryption Algorithm for network Operators). BEANO is a strong block cipher algorithm employing an 80 bits key. To overcome the conflicting requirements for broad exportability and a high level of security, the license and confidentiality agreement explicitly limits the use of the algorithm to the protection of network management data. The use of the algorithm for other purposes such as the protection of user data is explicitly excluded.

TETRA – Terrestrial Trunked Radio

TETRA is the new standard for digital private mobile radio communications system. The sys-

tem has been chosen by major Public Safety organisations in Europe as their future mobile communication system, but can also be used in public networks. Security is a big concern in TETRA and the system includes a large number of security features. These are supported by a large number of standard cryptographic algorithms. For the moment there are four standard encryption algorithms defined for TETRA. TEA1 and TEA4 are for general use in TETRA and provide a baseline level of security. The use of TEA2 is restricted to European Public Safety organisations (mainly from the “Schengen” countries). TEA3 is a similar solution for use by other Public Safety organisations. All the encryption algorithms are dedicated stream ciphers designed by ETSI SAGE and their intended use is for the protection of user data and signalling information over the radio link.

Furthermore SAGE has specified one set of TETRA Authentication and key management Algorithms (TAA1). The TAA1 is designed for use in all TETRA systems.

UMTS – Universal Mobile Telecommunications System

The next generation of mobile systems will be UMTS specified by the 3rd Generation Partnership Project (3GPP). The first set of technical specifications was finalised January 2000 and includes the definition of confidentiality and integrity algorithms. Both algorithms are based on a standard block cipher called KASUMI. KASUMI is a modified version of the Japanese block cipher MISTY ([13]) which has been available and publicly analysed for several years.

Due to the high-speed requirements in 3GPP systems, it was important to design an algorithm which was suitable for efficient implementation and could offer a high degree of security. KASUMI makes full use of a 128 bits key operating on 64 bits blocks.

The design and specifications of the standard algorithms for UMTS were conducted by SAGE enlarged with experts from manufacturers in Europe, Asia and US. In addition to the internal evaluation and analysis of the 3GPP algorithms, three different groups of independent experts were involved in an additional external evaluation. The intention is to have these specifications published in the same way as other technical standards, but all issues related to such publication have not been resolved at the time of writing.

³⁾ Information on AES can be found at <http://www.nist.gov/aes>

10 Regional Initiatives – AES and NESSIE

AES – Advanced Encryption Standard³⁾

In 1997 the US National Institute of Standards and Technology (NIST) initiated a process to select a new symmetric-key encryption algorithm to be used as a replacement for DES. The call for candidates stipulated that AES would specify an unclassified, publicly disclosed algorithm available royalty-free, world-wide. The algorithm must be a block cipher supporting a block size of 128-bits and key sizes of 128-, 192- and 256-bits. In 1998 NIST announced the acceptance of fifteen candidate algorithms and requested the assistance of the cryptographic research community in analysing the candidates. This analysis included an initial examination of the security and efficiency characteristics and the outcome was five finalist algorithms. The five finalist algorithms are currently subject for further study before selecting one or more of these algorithms for inclusion in the Advanced Encryption Standard. If all steps of the AES development process proceed as planned, it is anticipated that the standard will be completed by the summer of 2001. It seems obvious that the final AES algorithm(s) will be the preferred algorithm in the years to come.

NESSIE – New European Schemes for Signatures, Integrity and Encryption⁴⁾

NESSIE is a new European 3-year project that started in January 2000. It falls within the Information Societies Technology (IST) Programme of the European Commission and includes participants from industry and the academic world. NESSIE will contribute to the final phase of AES, but has a much wider scope. The main objective of the project is to put forward a portfolio of strong cryptographic primitives that have been obtained after an open call and evaluated using a transparent and open process. These primitives should be the building blocks for the future standard protocols of the information society.

11 Work in ISO

Within the International Standards Organisations (ISO) there are several groups defining security standards. There is one sub-committee (ISO/IEC JTC1/SC27) dedicated to this work, but for many years the development of international standards for confidential algorithms was explicitly excluded from their work program. This rule has recently been changed and a new project for development of such standards is now approved. We can expect that results from the regional initiatives described above will be forwarded to ISO for global standardisation.

12 Conclusions

Cryptographic algorithms are a fundamental building block for a variety of security services. The design of such algorithms is a long and tedious process involving a mixture of skills, and the final solution is often a trade-off between divergent requirements. We have presented some of the basic principles of such algorithms and have described the variety of standard algorithms that exist for different telecommunication systems. We argue that the development of standard algorithms will benefit from the use of open and well-analysed algorithms. AES and NESSIE are examples of new initiatives intending to develop and evaluate new and secure primitives. Future standardisation projects will then focus more on how to embed these primitives securely into a system in order to achieve defined security goals.

References

- 1 ISO. *Information processing systems – Open Systems Interconnection – Basic reference model – Part 2 : Security architecture (first ed.)*. International Organization for Standardization, Geneva, 1989. (ISO 7498-2.)
- 2 Kahn, D. *The Codebreakers. The Story of Secret Writing*. New York, Macmillan, 1967.
- 3 FIPS 46. *Data Encryption Standard*. Federal Information Processing Standards Publication 46, U.S. Department of Commerce/ National Bureau of Standards, National Technical Information Service, Springfield, Virginia 1977.
- 4 ISO. *Information processing – Modes of operation for an n-bit block cipher algorithm (first ed.)*. International Organization for Standardization, Geneva, 1991. (ISO/IEC 10116.)
- 5 ISO. *Information processing – Security techniques – Data integrity mechanism using a check function employing a block cipher algorithm (second ed.)*. International Organization for Standardization, Geneva, 1994. (ISO/IEC 9797.)
- 6 ISO. *Information processing – Security techniques – Hash functions – Part 2 : Hash functions using an n-bit block cipher algorithm*. International Organization for Standardization, Geneva, 1994. (ISO/IEC 10118-3.)

⁴⁾ Information about NESSIE can be found at <https://www.cosic.esat.kuleuven.ac.be/nessie/>

- 7 Beaver, D. Computing with DNA. *Journal of Computational Biology*, 2 (1), 1995.
- 8 Steane, A M, Rieffel, E G. Beyond Bits : The Future of Quantum Information Processing. *IEEE Computer*, 33 (1), 38–45, 2000.
- 9 Blaze et al. *Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security*. Counterpane (2000, June 20) [online] – URL: <http://www.counterpane.com/keylength.pdf>
- 10 Lenstra, A, Verheul, E. *Selecting Cryptographic Key Sizes*. Cryptosavvy (2000, June 20) [online] – URL: <http://www.cryptosavvy.com/cryptosizes.pdf>
- 11 Electronic Frontier Foundation. *Cracking DES : Secrets of Encryption Research, Wiretap Politics, and Chip Design*. Sebastopol, Calif., O'Reilly and Associates, 1998.
- 12 Coppersmith, D. The Data Encryption Standard (DES) and its strength against attacks. *IBM J. Res. Dev.*, 38 (3), 243–250, 1994.
- 13 Matsui, M. New block encryption algorithm MISTY. In: *Fast Software Encryption '97, LNCS 1267*. Biham, E (ed.). Berlin, Springer-Verlag, 1997.

Show Me Your Public Key and I Will Tell Who You Are

L I S E A R N E B E R G

A short introduction to digital certificates and some thoughts on their significance to the digital economy.

Internet and Security – Two Contradictory Terms?

It has been repeated again and again – the lack of security mechanisms on the Internet slows down the development of the new economy. Is it true? Hard to say, really. It is a fact that there are no global trust mechanisms on the Internet infrastructure, you cannot be really sure of whom you are talking to. The only global identification mechanism is the network address, and network addresses may be forged rather easily.

On the other hand, you can build as much security as you like into one specific application. You may issue usernames and passwords and even equip your users with password calculators or one-time passwords to ensure strong authentication between the user and your application. And you may encrypt user sessions by SSL and build crypto based cookie mechanisms to obtain confidentiality and preserve session integrity.

What is the problem then? A number of issues of course. Key lengths is one, secure storage is another. But in my opinion, the lack of common authentication mechanisms is one key issue. The consequences differ slightly depending on which market segment is addressed, the Business to Business (B2B) or the Business to Consumer (B2C) segment.

In the B2B segment, the relation to the customer is often established through other channels than the Internet. For a long time relation, you can afford a (partly) off-line registration procedure and you might take the trouble of managing usernames and passwords for all your users. So, the authentication problem can be coped with. As I see it, the problem is on the customer side. How many different passwords, userids and tokens can you possibly handle? If your daily work involves using five or six web-based services, delivered by as many companies, you might have to remember five or six passwords or handle five or six password tokens. And all of them issued to you in the same role in the same company. A general authentication mechanism would leave the user with one token, valid as authentication mechanism on any service.

The problem in the B2C segment is slightly different because you often do business with someone you do not know in advance. From the business perspective you may have no option but to trust that the name, address and credit card number supplied are actually genuine. It is a fact however, that Internet shops make quite an effort to check for false orders, false credit cards and false shipment addresses, but may still experience losses or fraud.

As a consumer, you may have experienced that it is a lot of work to remember username and password for a number of different Internet shops or services. It would be nice if I did not have to. And it would also be nice if I had a way of checking the authenticity of the service at the other end. Is this really my bank that I am talking to?

Whether the need for more global authentication mechanisms comes from practicality reasons, productivity reasons or genuine lack of trust, the answer to the authentication problem may be digital certificates. As web-based services involves more and more valuable transactions, the issue of non-repudiation and legally valid signatures on documents arises. These services may be built on top of an infrastructure based on digital certificates.

The rest of this paper is an introduction to digital certificates, the infrastructure to make them function and some examples of use.

The General Idea of Digital Certificates

A digital certificate is really an ID card for the digital world. As an analogy, consider an old-fashioned, paper-based ID card. What it does is really to establish a binding between a name and a picture. If you resemble the picture I accept that the name belongs to you. In the same way, a digital certificate establishes a binding between a name and a cryptographic key. If you can prove that you possess that key I accept that the name belongs to you.

Lise Arneberg (40) is Senior Consultant at Telenor Business Systems, in the department of e-commerce. She holds a Cand.Scient. degree in Mathematics/Algebraic Geometry from the University of Oslo, 1985. She has long experience from security and cryptography related issues at Alcatel (research department, telecom and defence communications), and at Scandinavia Online (electronic commerce).

lise.arneberg@telenor.com

In both cases, my trust depends on the issuer of the ID card or the digital certificate. If I recognise the issuer as trustworthy, the ID card has a value. Passports and ID cards issued by banks or the postal service are normally recognised as trustworthy anywhere. While the ones issued by a school or an employer are not generally accepted outside that school or company. The same mechanism applies for digital IDs. If you present a digital certificate and I do not trust or recognise the issuer, it would be of little value to me.

Issuers of digital certificates are normally referred to as Trusted Third Parties (TTPs). In Norway, we have currently three actors on this scene: Bankenes Betalingssentral (BBS), Posten SDS and Telenor. All these three are companies or institutions we are used to trusting. They are all currently running digital certificate services.

Their certificates may be designed for different purposes or services. For remote LAN access, for Internet banking, for security services within an organisation, for tax reporting or for other special applications. And the contents may differ. But generally the content is at least:

- Name of the owner;
- The public key;
- Name of the issuer;
- Validity period and expiry date;
- The issuer's signature.

The issuer's signature ensures the authenticity of the certificate. If any part of the content is altered, a signature check will reveal the fraud. And it is not possible for anybody else to issue false certificates, because you cannot forge such a signature. If we use the passport analogy, the issuer's signature is the seal of Norwegian authorities, as well as the special paper quality that makes a passport so difficult to forge.

Registration and Distribution – the Key Issues for Building Trust

If you have applied for a passport lately perhaps you remember that it is quite a tedious process. You are required to meet personally, show an ID and fill in a couple of forms. To receive the passport, you must once again meet at the police station and show your face and an ID card to verify that you are the correct owner of the passport. Getting a new credit card is much easier. You simply phone and tell that the previous card is broken. After a few days you receive a new one by mail.

These two examples represent different registration procedures and different distribution procedures. The result is that a passport has a higher

level of trust than a credit card (at least in the sense of ID cards). Once again, the analogy can be made to the world of digital certificates. If an authentication process is based on cryptographic keys, how sure can you be that those keys are in the hands of the right person? That depends pretty much on the checks performed during registration, as well as distribution procedures for the certificate and the cryptographic keys. If you have to show an ID card to apply for and receive the certificate, this is a high level of security. If you fill in a form on the Internet and receive the certificate by mail, the level of trust would be low. Requirements for registration and distribution processes is often called a certificate policy (CP). In a sense, the certificate policy defines the likeliness that the certificate is in the right hands.

Two different TTPs may agree to accept each other's digital certificates. They would do so only if the process of registration and distribution are similar and ensures the same level of security or trust. This is called cross certification and is a formal process between to TTPs. If your TTP accepts the certificates of my TTP and *vice versa*, then we can actually trust each other's certificates. And we may check each other's signatures.

And a Little Bit on the Cryptographic Protocols to Make it all Work

Digital certificates are based on public key techniques, mostly RSA. For the slightly rusty reader, I will just repeat that an RSA key pair consists of a modulus, a private key and a public key. If you sign a message with your private key, the signature may be checked using your public key. If someone encrypts a message with your public key, only the private key can decrypt the message.

The modulus and the public key are public knowledge. The private key must be kept secret. Key lengths these days are normally 1024 bits but sometimes the double. Encryption processes involves exponentiation modulo 1024 bits numbers and may be time consuming.

The beauty of public key encryption is that you can easily prove that you possess a key (the private) without exposing it.

Some definitions before we look into the authentication process:

- The Registration Authority (RA) receives certificate applications and verifies the applicant's identity according to what is specified in the Certificate Policy.

- The Certificate Authority (CA) issues certificates and distributes them to the user.
- The CA also publishes all issued certificates in a catalogue or directory, which is available via the Internet.
- When a certificate is revoked for some reason, the CA updates the Certificate Revocation List (CRL).

These are all services operated by the TTP. They are critical applications and are run in a physically secured environment by specially authorised personnel.

In the world of digital certificates, when I want to prove my identity to you, the authentication process would go like this:

1. I claim my identity by sending my digital certificate. Or you look up the certificate in a certificate directory.
2. You check the issuer's name. If you recognise the name as someone you trust, you do the signature check to verify authenticity of the certificate. If not, you look up the directory of your Trusted Third Party to check if this issuer is someone he trusts (i.e. is certified by him). If that is so, you receive the information you need to check the signature of my certificate. If not, you reject my certificate because you have no reason to trust its origin.
3. You check the revocation lists (CRLs) to see if this certificate is still valid.
4. If all is well so far, you send me a challenge.
5. And my response is the challenge signed by my private key.
6. You verify the signature, using the public key of my certificate.
7. If the signature is OK, you will trust that I am the genuine owner of the certificate and now you know who I am.

The reader may have noticed that the steps above require a bit of infrastructure. First of all, digital certificates must be issued by the CA and transported securely to their owners. Secondly, the user must know whom to trust, i.e. (s)he must have access to the public key (or rather the certificate) of the trusted TTP. Thirdly, certificates must be available in a directory. If the certificates are general purpose IDs, this directory must be available to the public. As some certificates inevitably will get lost, stolen or corrupted,

a blacklist or revocation list must be publicly available for everyone to check. And there must be someone who controls and updates the revocation lists.

There is a number of standards describing the details of these steps. Most important is the CCITT X.509 standard for digital certificates. Currently, version 3 of the standard is in use. The PKCS-7 standard describes how to create and verify signatures. The directory is an X.500 service, and the LDAP protocol specifies the interface to the directory. A number of PKCS protocols describe cross certification and other certificate exchange protocols.

On the Practical Side

Suppose your Internet provider or someone else has equipped you with a digital certificate. How do you receive it, keep it and how do you get to use it without knowing the details of the protocols above?

Keys can be kept in SW or in a smart card. Keys in SW must be protected by encryption and you will need a password to decrypt the keys and get access to cryptographic operations. In a smart card, the keys are protected by the operating system of the card and the private key will never leave the card. A PIN is required to open the card for use.

The certificate issuing process may take different forms. It may be done online with the CA and with keys generated locally. In this way, the private key never leaves "home". To secure the certificate generation process, the user is first equipped with two codes used for authentication.

The certificates may also be personalised offline with centralised key generation (mainly smart cards) and then shipped to the user according to policy.

If keys are in SW, there must be a "bridge" between the application (for instance a login script) and the cryptographic keys. Let us call this bridge a security module. It offers a specified interface (API) of security operations to any application on the PC. The application may be able to order a signature, perform the steps of an authentication and perhaps to check the certificate of the entity at the other end. The API may also specify a number of other services such as encryption mechanisms and exchange of encryption keys. With keys in a smart card, all cryptographic operations may be performed at the card, and the application (or the security module) may communicate with the card through the smart card reader.

A number of standard PC applications, such as mail clients and web browsers, is available today with plug-ins that are adapted to a specific security module. For the security module from one major supplier, Entrust, there is a wide range of applications, denoted "Entrust ready", that exploits the security services in their security module. This is all done seamlessly to the user. The user may sign or encrypt mail, or communicate securely with a web service, simply by clicking icons in her regular applications.

For more specific or proprietary applications, security services must be built in by utilising the API of the security module.

Digital certificates can be managed or unmanaged. With a managed certificate, the security module automatically communicates with the CA to handle necessary certificate updates, key backup for encryption keys and other services.

SET – Digital Certificates as Credit Cards

The Secure Electronic Transaction (SET) standard was perhaps the first example of a commercial use of digital certificates. It was developed as a cooperation between VISA and Eurocard. Their goal was to establish secure credit card transactions on the Internet. It is an open standard, based on digital certificates. It was published in 1997 and the first SET transactions in Norway were performed late 1997.

The certificate issuers or CAs of SET certificates are the credit card companies. Both card holders and shops will have certificates. Each credit card company will define their own procedures for registration and distribution of certificates. There is no cross certification between CAs, i.e. you cannot pay with a VISA credit card unless the shop (or merchant in SET terminology) has a VISA certificate too.

The payment process is based on a set of pre-defined messages between the card holder, the merchant and the acquirer payment gateway (also called payment gateway) – the interface to the credit card company. All messages in the protocol are encrypted and signed. The protocol ensures maximum anonymity. The merchant will only know that the card holder is authentic and that the amount is accepted by the credit card company; she will not know the credit card number of the card holder. The payment gateway will only know the amount and not what was purchased. The card holder can be sure that this is an authentic shop with a real VISA (or other) certificate and not just someone who set up a fraud service.

The protocol, it seems, took into account any experience with practical use of credit cards, as well as exploited the new possibilities that came with the new technology. The best of two worlds? Unfortunately, SET has not been a success so far, despite all its beautiful cryptography. Partly because it was early. But also because in its first implementation it had two practical weaknesses.

The certificates were in software and had to be installed on your PC. (Smart card readers were expensive in 1997, about 7 times the prices today, and were out of the question.) Do you know how to take care of a credit card residing on your PC? If it is not in your wallet, how do you keep control of who is using it? Should it be on your home PC or on your work PC? What happens if your children or your cleaning lady are not to be trusted? Now, after a few years, we have got used to the idea of SW certificates, but personally I am still not sure that credit cards should be in SW.

The second practical weakness occurred on the merchant side. A specialised SW called a SET Merchant is required to run the SET protocol. And in 1997, the SET Merchant applications were so expensive that they were out of the question for most Internet shops. Not to mention the requirements for infrastructure. A SET merchant with its certificates is not an application you would want to put on the server directly accessible on the Internet.

So, in spite of SET, most Internet transactions are still paid by old fashioned credit cards. Shops add a bit of security by use of SSL to avoid sending credit card numbers as cleartext on the Internet. But they have to put a lot of effort in manually handling all their credit card transactions. To help this situation, some credit card companies now develop further another option of the SET – without card holder certificates. It is called MOSET (modified SET?) and permits on-line handling of credit card transactions without card holder authentication.

What About the m-Business?

The "mobile Internet" is evolving just these days. There are two approaches to services on a GSM phone. One is the WAP – Internet browsing and Internet services adapted to the format of the cell phone screen and the bandwidth of the GSM network. The second approach is building explicit services into the SIM card of a regular GSM phone. Any way, a phone is much more personal than a PC. It is also more quickly enabled and services will seem more easily accessible than on a PC. So it should be expected that

services accessible by mobile phones will evolve quickly and offer a wide range of services.

With banking services, shopping and subscription services, the issue of authentication arises also in the m-world. Even though the GSM algorithms themselves provide some authentication, this is not automatically available to applications outside the GSM systems.

However, SIM cards are smart cards and as such ideal for storage of cryptographic keys and digital certificates. Telenor Mobil are planning to equip all their new GSM subscribers with a larger SIM card, containing a digital certificate, a security module and a set of application clients, thereby providing a new electronic world with a “global” authentication mechanism.

Paperless Business? Digital Certificates and Legally Valid Signatures

Although so much business communication is done over electronic channels, there is still no mechanism or infrastructure in place to substitute a legally valid, hand written signature on a paper document. What would it take to establish legally valid signatures on a document? In such a manner that none of the signing parties can deny that they have signed, what they have signed and when?

The answer to the “when” is a timestamp service – a trusted party that adds the time to the signed document and then uses its own certificate to sign it.

The answer to the “what” part may be a public notary – a trusted party that stores valuable documents and may verify signatures upon dispute, even after many years.

And the parties must have their digital certificates, containing keys that may be used for signing.

The issue of legally valid signatures still has not reached enough maturity to bring out the implementations. This is due to legislation, but also to infrastructure and trusted actors.

Conclusive Remark

Would it be nice to have only one digital id, one smart card valid in all situations? With credit cards, health information, access to entrance doors at work and whatever. I am not so sure. On the other hand, I may never have that option. So far, the structure or content of a digital certificate must reflect some information about its usage. Your certificate as an employee with a certain role in a certain organisation will be different from your SET certificate or from the cer-

tificate on your GSM phone, which only reflects you as a private person. So we will probably have to cope with a number of digital certificates – just as we cope with a number of magnetic cards. Let us just get used to the idea! And let us pray that someone invents the all-secure-PIN-storage-device – real soon now! (Based on fingerprint identification?)

Abbreviations

B2B	Business to Business
B2C	Business to Consumer
CA	Certificate Authority
CCITT	Comité Consultatif International Téléphonique et Télégraphique (Now: ITU-T)
CP	Certificate Policy
CRL	Certificate Revocation List
GSM	Global System for Mobile Communications
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
PKCS	Public-Key Cryptography Standards
RA	Registration Authority
RSA	Rivest-Shamir-Adleman (algorithm)
SET	Secure Electronic Transaction http://www.europay.com/common/Index.html
SSL	Secure Sockets Layer
TTP	Trusted Third Party
WAP	Wireless Application Protocol

Security in a Future Mobile Internet

HENNING WRIGHT HANSEN AND DOLE TANDBERG



Henning Wright Hansen (28) is Research Scientist at Telenor R&D, Kjeller, where he has been working on aspects related to Internet security since 1996. His research interests include wide aspects of security related to computing, focusing on the security consequences of introducing Internet mobility as well as solutions making the Internet a safer place to be for the mobile users of the future. He is also involved in the TigerTeam activities (practical security testing) at Telenor R&D.

henning-wright.hansen
@telenor.com



Dole Tandberg (38) is Research Scientist in the Security Group at Telenor R&D, Kjeller, where he has been working since 1997. His research interests include security in Internet and mobile distributed systems. He is also involved in the TigerTeam activities and practical security exploits.

Background: 1982–1988 University of Zagreb, Croatia; 1988–1996 ABB Corporate Research; 4 patents holder (2 WW and 2 UK); 1994 Sheffield University (UK), Largest civilian FEM-test research program; 1995 London Conference “New Era Technology” paper holder.

dole-stojakovic.tandberg
@telenor.com

Introduction

Everyone knows it – computers are getting more and more common. Computers are also getting more and more portable. At the same time the Internet has become a common place. The professional computer users demand the possibility to work as usual independent of time and place. And why not? The technology that will make this happen is more or less here.

However, the companies that own and control the computers used by these demanding users have second thoughts. What about security? The Internet is itself a “dangerous” place to be, and even highly skilled personnel make mistakes when implementing security solutions to protect the internal network from script kiddies or may-be even worse, from unscrupulous competitive companies or foreign governments. Who really knows what might be out there?

And now the users want to open up the internal network to be accessed from the Internet. How could this be done in a satisfyingly secure manner? How should the mobility management be implemented to allow for transparent access – both to and from the machines on the road?

This article will try to cover these issues and propose solutions that may become the future foundation for secure Internet mobility in the years to come, with a particular focus on the users and their terminals. Only selected security technologies will be covered, focusing on the mobile users and their terminals.

A Mobile Wireless Network of the Future

The Internet has traditionally offered very poor support for mobility. It is of course possible to move computers connected to the Internet from one local area network to another, but there has been no support for mobility. That is; when moving a computer from one LAN to another, you have to supply it with a brand new IP address. From the network point of view, this is seen as a totally different computer since the IP address is used to identify it. Even today, some legacy operating systems claimed to be “up to date” need to be rebooted in order to use a different IP address.

Mobile IP is about to change this, however, allowing the computer to be moved to keep its old IP address used for identification, and at the

same time obtain and use a new topologically correct IP address. This may be achieved by using e.g. tunneling mechanisms.

Internet mobility may be achieved without altering the routers or other hosts on the Internet, the only thing *really* needed is a Home Agent placed on the home network. The Home Agent is responsible for registering the current locations (IP addresses) of mobile nodes, and forwarding packets destined for the mobile node out to its current location. Authentication is built-in in order to make it difficult to redirect traffic to unauthorised hosts, this will help prevent someone stealing traffic destined to the mobile nodes.

When a mobile node connects back to the home network, an extension is made from the LAN to the Mobile Node. Traffic traditionally being internal and restricted to the LAN only, now ends up being routed over the Internet for everyone to see. Utilising VPN technology is therefore *very* important if privacy, authenticity and integrity of the data exchanged is of any concern. Traditionally VPN systems have been used for protecting traffic between different kinds of networks, but the same technology may be integrated directly on the mobile nodes in a network-to-host scenario. (Or host-to-host for that matter.) The technology used to achieve this is typically IPsec.

At the home network, the extension of the home network towards the mobile nodes is typically combined with the use of firewalls. The firewall is supposed to enforce a security policy separating those “inside” from those “outside” the firewall.

Wireless technology such as the IEEE 802.11 may be used as one attractive method of accessing the Internet or LAN IP networks. The security mechanisms in these standards are however limited in flexibility, and VPN technology is therefore needed on top. WLAN technology is sometimes envisioned to replace traditional wired LAN networks, particularly in dynamic and new environments. Time and money may be saved as the need for cabling is reduced considerably. In future office environments, WLAN technology may be used as a *shared* access technology, e.g. within a large office building. It would be much more efficient to offer WLAN access to all employees in different firms within the building, instead of having each firm imple-

menting its own WLAN system. Visitors may also be given network access through the WLAN within or nearby the office building.

However, using WLAN technology in this manner, the concept of firewalls separating the “internal” from the “external” network is no longer useful. The once physically protected internal network is now more or less publicly available. To maintain the concept of a “secured local area network” completely new firewalls, VPN, as well as other security mechanisms are needed.

This article describes *selected* techniques needed to fulfil the security needs in such a future wireless mobile network.

Secure Networking in the Future

Technical Solutions

Local Adaptive Firewall Services

A commonly used technique used to protect a “private” or “internal” network from an external network such as the Internet is to implement a firewall service. Firewalls started out as simple packet filters, capable of filtering out packets based on IP addresses and information contained within other higher level protocols, such as the TCP and UDP headers.

The simplified firewall illustrated in Figure 1 accepts only inbound connections from TCP port 80, which is dedicated to HTTP traffic, and rejects other types of traffic.

While traditional “static” terminals typically are protected from the Internet by the use of firewalls, mobile terminals of the future are envisioned to roam freely, using the Internet wherever it is available. The firewalls on the home

network will not be able to protect those users. This is the reason why firewalls must be *implemented locally on the mobile terminals*, and not only on the borders of the home network.

It is however not enough to install “traditional” firewalls on these mobile terminals, some requirements that firewalls on mobile terminals should meet include the following:

- They need to be configured dynamically, depending on the current location;
- They should be able to adapt themselves to the current network traffic (e.g. blocking attacks dynamically);
- They need to be easily and securely remotely controlled by their administrator or automatically by a server on the corporate network;
- They need to adapt to dynamic IP addresses;
- They need to automatically adapt to the network interfaces currently in use (ethernet, modems/ISDN, IrDA, Bluetooth, etc.);
- They need to be integrated seamlessly and transparently (for the user) with other applications running on the mobile nodes, rejecting all network connections unless the firewall has been properly activated;
- They need to be integrity protected as well as verifiable, ensuring that the configuration is trustworthy.

To administer these firewalls will be a challenge in the future world of mobile communication.

Some of the same principles presented here are also found in a recently published paper from Steven M. Bellovin, *Distributed Firewalls* [1].

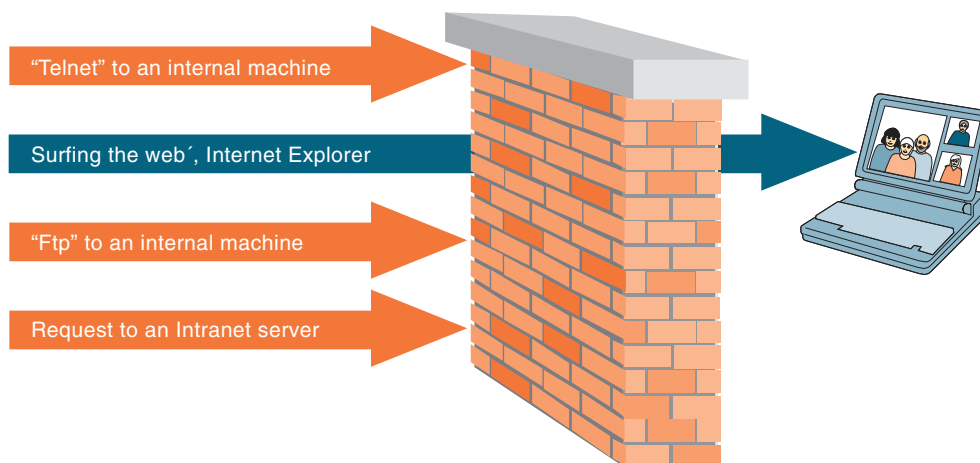


Figure 1 A simple packet filtering firewall allowing web surfing only

Securing Locally Stored Information

Locally stored information needs to be protected from unauthorised access.

The way to properly protect locally stored information while it is not being used is to use strong cryptography, e.g. through properly encrypted file systems. If a terminal is lost or stolen, this may prevent unauthorised access to the information stored. It may still be a security problem that a terminal may be lost or stolen while the locally stored information is in use (currently unencrypted).

Using tamperproof hardware tokens for storing cryptographic keys that are used for accessing the locally stored protected information will help protect against unauthorised access if the terminal is lost or stolen.

While the locally stored information is being used, making certain parts of this information read-only may help prevent unauthorised *modification*. This may help prevent the installation of viruses, Trojan horses, etc. Of course, some parts of the system need to be writeable, and hence there is still a need to have mechanisms to detect and possibly remove viruses, Trojan horses, etc.

In addition, there is a need to have file integrity mechanisms installed, in order to detect all unauthorised modifications made to the file system.

Distributed Secured Storage

Modern state-of-the-art solutions for communication security offer excellent security for the *communication channels* using e.g. IPsec. However, although the use of such technology is well suited to protect private or sensitive data in transit, the same data is often stored without protection at both ends of a secure communication channel.

This leaves the data vulnerable to attacks. In particular, servers connected to the Internet have proved extremely difficult to secure, and even servers that have been properly configured and well protected are frequently broken into.

In order to offer sufficient protection of data stored in such servers, one has to protect the data itself, not only the communication channel. One way to achieve this is to use "object security". One example of how this could be done is static web pages on a web-server. Instead of establishing a secure communication channel using e.g. SSL (Secure Socket Layer), these web-pages could be encrypted and/or signed digitally once and for all before being made available on the

server side. This would also reduce the load on the server, as establishing SSL connections is much more processing intensive than establishing traditional HTTP connections.

The pages could then be downloaded using traditional HTTP connections that are not considered secure. However, since the object (the webpage) is itself encrypted, only the authorised recipients would be able to decrypt the contents.

Using PGP (Pretty Good Privacy) to encrypt email, this is what actually happens. The object itself, and not the communication channel, is secured. The encrypted object might be decrypted at its endpoint for use at once, but the object originally received (encrypted) might (should) be stored securely for later use if needed.

There are several benefits using this approach:

- Less processing-intensive: Objects are cryptographically secured only once, not once per connection (as with SSL or IPsec).
- More secure: Even if the server where the object is stored is broken into and the objects on this server are modified or replaced, this will be detected since the signature is no longer valid (content signing).
- More secure: An intruder that has managed to break into a server will not be able to decrypt the protected objects unless he/she is authorised to do so (has the necessary keys).
- More secure: It will be possible to use the network as a secure place for storing user information without worrying whether the server is well enough protected or not. The network could be used for e.g. backing up important data remotely in a secure manner, if the objects are sufficiently cryptographically protected before they leave the mobile terminal.
- Cheaper: Since objects are protected in a less CPU-intensive way, cheaper machines might be used.
- Cheaper: The servers need not be protected as much as servers that hold unencrypted sensitive data.
- Legal benefits: If Telenor offers a "secure storage" for e.g. backing up GSM/UMTS-telephone data (e.g. address books or calendars), it would only need to provide a storage space, not security services to protect the object with regard e.g. to privacy.

- Less prone to attack: Since the sensitive data is encrypted, the potential gain from attacking these servers is reduced.
- Cheaper: The servers themselves need not be monitored as closely as servers containing unprotected information.
- Cheaper: It is no longer that critical to upgrade the systems once new vulnerabilities have been discovered.

Of course, in order to prevent Denial of Service attacks when storing protected objects on a server, there is still a need to secure these servers. The point being made is that it is no longer as crucial as for servers where unprotected sensitive information is being stored.

VPN Access to the Corporate Network

The mobile professional computer user needs to be able to connect to the corporate (home) network while being “on the road”. Whatever access technology used, there is a need to secure the communication channel back to the corporate network. This may be achieved using the VPN technology currently being standardised and implemented. The technology used to build VPN networks is typically IPsec. IPsec offers encryption, authentication and integrity protection services on IP packets, and may be integrated with future PKI and hardware token based security solutions. The user may e.g. be required to use their smartcard in order to gain access to the corporate resources using the network.

The same technologies as described here may be used to secure access to home network expected to be implemented in future homes.

The VPN technology needs to be able to adapt to different network interfaces as well as working transparently with e.g. Mobile IP.

Multiple Shared “Virtual” VPNs

Implementing wireless access technologies such as 801.11 (Wireless LAN), the security thought to be available in a physically protected cable based network is no longer there. In a large office building with many different companies, WLAN access may be offered as a generic network access service to all. On top of this “generic” network access, each company may build a “Virtual” shared VPN network to protect communication within their own defined security domain. Since unauthorised users may be receiving this radio based communication, special care is needed to protect all communication within this virtual network.

This would be a much more efficient and cheaper solution, compared to the case where

each company had to build their own corporate network, WLAN or cable based. In addition, the employees would benefit from having access to the internal corporate network from within WLAN reach of the office building. However, using this Virtual VPN technology combined with mobility, the employees could have transparent secure access to the corporate resources from every access point imaginable.

Currently, firewall solutions are however not well suited to protect such a virtual network. The firewall services need to be integrated on the terminal itself, as stated earlier in this article.

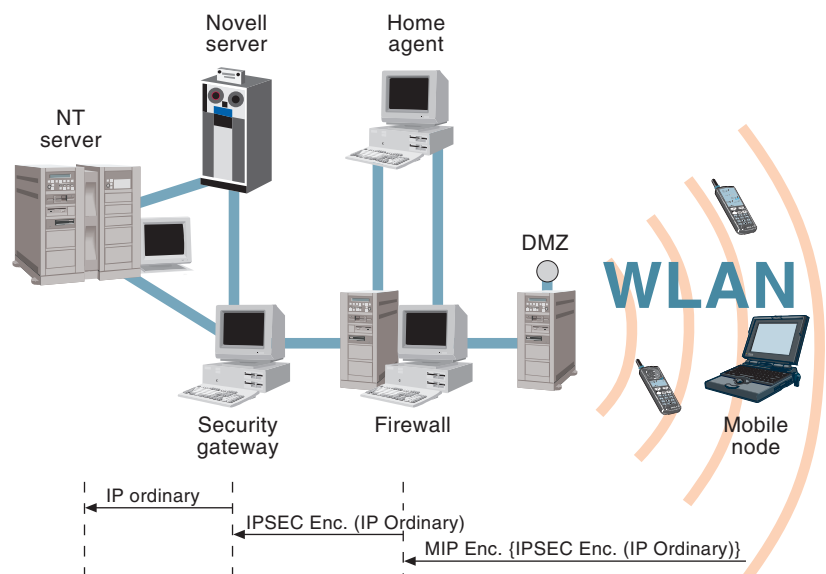
IPsec, smartcards and PKI systems are the ideal security technologies to be used to implement the scenario “Multiple shared Virtual VPNs” described above.

Intrusion Detection Systems

Even though you have implemented a local firewall policy, have integrity checking mechanisms in place as well as updated anti-virus software and partially read-only system installed, there is still a possibility that you may be exposed to successful “hacking attempts”. Even though you may not have been compromised, it is important to detect such activities. (You should know your “enemy”.) Intrusion Detection Systems (IDS) have been designed for exactly this purpose.

There is a need for both network based and host based IDS. In our case, focusing on users using mobile terminals, the network based IDS would be responsible for monitoring all network interfaces against unauthorised activities. The *network based* IDS may pass this information on the other parts of the security system on the terminal, e.g. allowing dynamical modification of

Figure 2 A simple WLAN Virtual VPN technology combined with mobility



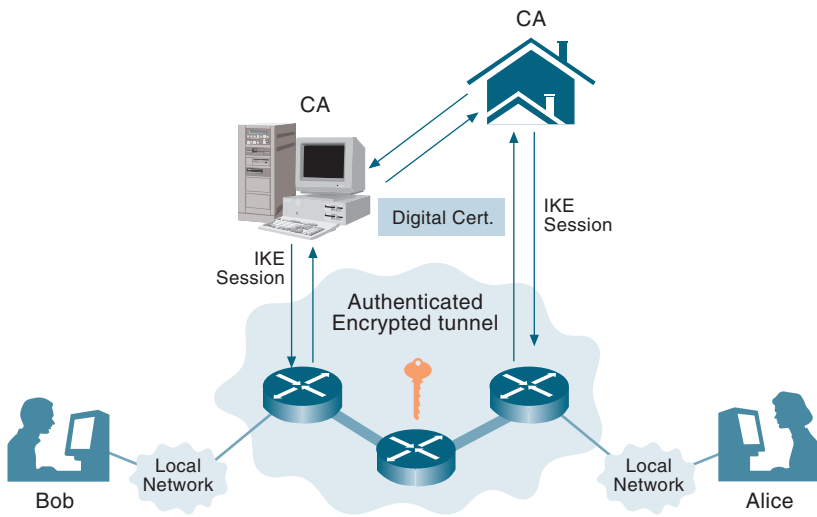
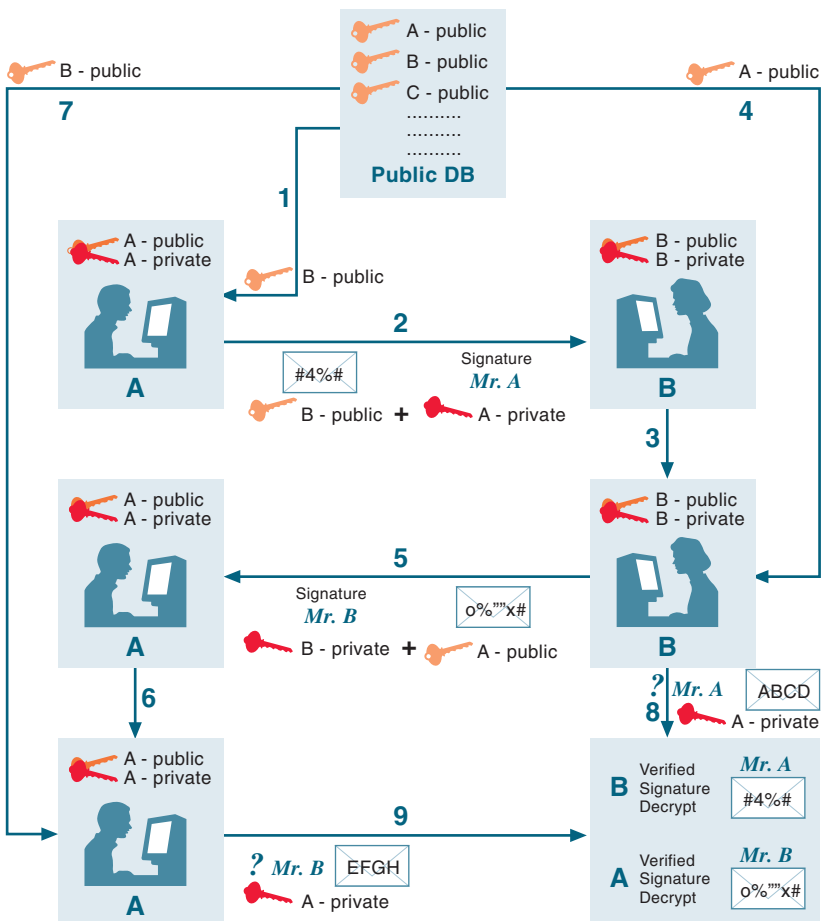


Figure 3 A simple Public Key Infrastructure

the security policy as well as warning the user and maybe the home network as well. The *host based IDS* will monitor access to local resources in an attempt to detect unauthorised activity. It may, as the network based IDS warn the user as well as his home network, and trigger a different and stricter security policy if needed.

Figure 4 A "simple" user-to-user public key cryptography



The mentioned countermeasures have been enforcing a stricter security policy on the local terminal. Other countermeasures may be possible, e.g. trying to track the source of the unauthorised activity. However, one should be very careful implementing and using such countermeasures, as there is a substantial risk of spoofing (e.g. falsifying the IP source address of packets reaching the mobile terminal). This may again lead to sophisticated and hard to prevent Denial of Service attacks.

PKI as the "Security Glue" of the Future

A Public Key Infrastructure (PKI) is a system supporting the use of public key cryptography, often combined with the authorised issuers of public key certificates (also called digital certificates). In a PKI system each user, being a person or a service/machine, would have at least one such certificate.

A public key certificate typically contains information about its legal user such as the name, address, e-mail address as well as the public key. The certificate is signed by its issuer, a certificate authority, in such a manner that all attempts at modifying the information contained in the certificate is easily discovered.

To each public key certificate there is a unique corresponding private key. The public key certificates are part of the public domain and are published through large searchable databases accessible to everyone. The corresponding private keys are strictly personal and are at all times expected to be accessible by its legal user *only*. An ideal protected place to store such private keys would be tamper-proof hardware tokens, such as smartcards.

A PKI would offer a scalable security platform for services such as e-commerce, dynamic VPN networking, secure messaging, non-repudiation and digital signatures. There need not be a pre-defined trust relationship between the communicating parts using a PKI system. As there are several different issuers of public key certificates, there must be a trust relationship between these issuers, hiding the fact that these certificates are issued by different issuers from the users (cross-certification). There may also be many different types of specialised digital certificates, according to the policy and service provided.

AAA Services

Authentication, Authorisation and Accounting (AAA) are services needed in a mobile Internet where different operators want to enable e.g. chargeable roaming between the different domains (ISPs).

The AAA systems supporting scalable Internet mobility are currently under development by the IETF. Radius and Tacacs, the commonly used AAA systems today, have major shortcomings with respect to support for roaming and mobility. The IETF is currently investigating the requirements and working on the solutions needed to support the future mobility services expected to be offered on a future mobile Internet.

A PKI would be an ideal platform to build these AAA systems upon. Having a PKI available would enable easier mechanisms for trust across different security domains, allowing simpler roaming between different operators.

Use of Hardware Tokens

The SIM card (smartcard) in the GSM system has turned out to be a successful way of achieving secure user mobility.

Using a relatively tamperproof secure storage and processing device as a smartcard, access control on the terminal may be enforced in a secure manner. In particular, storing and using private keys used in a Public Key Infrastructure on smartcards currently offers the best combination of user mobility and security available.

Integrating the Security Technologies

Each of the selected security technologies described in the previous section is important in order to protect the mobile users of the future. However, the security technologies described need all to be *co-ordinated* according to a defined security policy. This policy should be defined and managed by the security organisation on the corporate network for professional users, and maybe as a service for its customers from the ISP perspective.

Take the professional corporate and mobile user on the Internet as a scenario. Let us say she has arrived in the UK on a business trip, having left Telenor R&D at Kjeller earlier the same day. Having arrived at the hotel in London, she decides to connect to the Internet using the offered WLAN connection at the hotel. Once connected, she establishes a VPN connection back to Telenor R&D to read the latest mail using a combination of Mobile IP and IPsec together with her smartcard for authentication purposes. However, the *security policy* requires several security services to be established *before* the home network accepts the VPN connection. A local firewall has to be configured properly, a virus scan has to be performed, the integrity of important files and system aspects need to be verified, important security logs need to be audited, etc.

The results of all these checks and configurations need to be securely reported to the home network before access is granted to internal resources at the corporate network. This is actually *required* if the home network is supposed to trust that the mobile terminal has not been compromised. Using the currently deployed state-of-the-art security technology, users establish a secure VPN channel back to the corporate network based on proper authentication only, leaving the mobile terminal to be hacked by anyone without the home network ever finding out.

The challenge is however to *enforce* the required policy, e.g. automatically configuring and checking the mobile terminal as presented above.

A proper security policy may e.g. include:

- What security services are needed;
- The right order to enable the different security services;
- How to configure the different security services;
- How to verify that all services and checks have been performed successfully;
- What to do if some services and/or security checks fail;
- What to do if attempts to compromise the current policy is discovered once the policy has been successfully established;
- How to inform the right entity on the corporate network about the attempt to compromise the current policy.

Different users may have different security policies defined, according to e.g. the type of services the user is authorised to access. For each user, there may also be different security policies defined reflecting the current network access. Three examples of different scenarios include one for using dial-up connections to the corporate network, a second one to be enforced when connecting directly to the Internet and a third one to be enforced if being on a physically secured corporate network.

The different security policies and its “enforcement software” need to be secured as well, making sure any attempts on modification is detected and properly handled according to the defined security policy.

The Policy Control Center

An implementation of a policy enforcement mechanism described in the previous section is currently being developed at Telenor R&D, called a "Policy Control Center". The Policy Control Center is controlled by a secured "Policy Control Center Server" located at the home/corporate network.

For communication between mobile terminals and this server (between the Policy Control Center and the Policy Control Center Server) LDAP over SSL is currently being used with both client and server side certificates.

The Policy Control Center Server contains different policies for different users, as well as different policies for different network configurations. When a mobile terminal is outside the corporate network, we enforce a stricter security policy compared to the case where the mobile terminal is connected via a physically "secured" cable inside the corporate network (and behind the corporate firewalls).

However, when introducing the concept of Multiple shared virtual VPNs as described earlier where a physically secured cable is simply not available, you need to introduce a security policy similar to the one used when connected directly to the Internet.

The initial implementation of the Policy Control Center was running on Linux, where we have investigated among other things the following security technologies to be integrated and used to secure the mobile terminal:

- Local firewall service through the use of "Ipchains". This is a rather basic packet filtering firewall available on Linux systems.
- A simplified network Intrusion Detection System by a combination of "Ipchains" and "Port-Sentry" [2], allowing dynamical blocking of unauthorised network attacks using TCP-wrappers.
- Host based Intrusion Detection System through the use of "LIDS" [3], the Linux Intrusion Detection System.
- Integrity protection on the PCC itself as well as on all system critical parts of the file system through the use of "LIDS" [3]. LIDS enables the strict enforcement of read-only policies, making selected parts of the file system read-only or append only even for the superuser (root).

- VPN access through the use of "FreeS/WAN" [4], a Linux IPsec implementation.
- Disk encryption mechanisms.

All policies need to be *verifiable*. To verify that the policies defined are actually implemented correctly on the mobile terminal, we need to build our implementation upon strong cryptographic protection mechanisms as much as possible. All software components to be used as part of the Policy Control Center need to be integrity protected so that as many attempts as possible on modifying and bypassing the protection mechanisms are discovered. Successfully bypassing of the Policy Control Center could mean that a compromised mobile terminal is able to access the internal corporate network.

We are currently investigating the possibility of integrating local information protection through further use of disk encryption, incorporating the use of smartcards and PKI systems as well as all other security mechanisms required covered earlier in this article, in order to achieve the best possible security for the future users of Internet mobility.

Our current work covers the development of a pure JAVA version of the Policy Control Center, making sure that all kinds of different terminals running JAVA may be secured using our PCC. Although the PCC implementation is platform independent, the policies themselves are however highly platform dependent.

Challenges currently under investigation include for example:

- How to most efficiently and securely write and handle the policies for different platforms;
- How to protect the Policy Control Center itself from unauthorised modification and bypassing attempts;
- Making sure the right policy is selected automatically, how do you know where you are?

Conclusions

Keeping a computer system connected to the Internet secure is a challenging task even for experienced security personnel. New software is frequently being introduced, often with little or no concern for potential security implications, and new vulnerabilities are constantly being found both in old and new software. In particular, without an open development model, mistakes made before are bound to be repeated over and over again without the customers ever hav-

ing the possibility to verify the functionality of the software or detect potential security flaws in the implementation themselves.

Other security mechanisms are needed to secure computers, in particular mobile terminals roaming the Internet; the security focus must be shifted towards the terminals themselves. The Policy Control Center takes the security level steps ahead compared to the currently implemented state of the art security mechanisms available for mobile (as well as other) terminals. In particular, there are two aspects of the Policy Control Center worth repeating; it is able to

- Automatically *configure* the mobile terminal with respect to a predefined security policy;
- *Verify* the state of the mobile terminal with respect to security before access is granted to the internal corporate network.

When designing the actual security policies, “Defence in Depth” should be *the* guideline; add redundant security mechanisms – do not rely on a single protection mechanism that will be a “single point of failure” with respect to security!

The security level that may be achieved by introducing a Policy Control Center however is only as good as the actual security policies themselves; even with a flawless Policy Control Center in place, it will not be able to defend your terminal if there is an “opening” or a flaw in the actual security policies.

Coming technologies and products that introduce some similar concepts as our PCC include COPS within the IETF [5], MEXE within the 3GPP [6], and PGP Desktop Security 7.0 from PGP Security [7].

References

- 1 Bellovin, S. Distributed Firewalls. ;*login: special issue on security*, November 1999.
- 2 *Psionic PortSentry 1.0*. 2000, September 28 [online]. – URL: <http://www.psionic.com/abacus/port Sentry/>
- 3 *Linux Intrusion Detection System*. 2000, September 28 [online]. – URL: <http://www.lids.org/>
- 4 *Linux FreeS/WAN*. 2000, September 28 [online]. – URL: <http://www.freeswan.org/>
- 5 *The Internet Engineering Task Force*. 2000, October 9 [online]. – URL: <http://www.ietf.org/>
- 6 *3GPP*. 2000, October 9 [online]. – URL: <http://www.3gpp.org/>
- 7 *PGP Security*. 2000, October 9 [online]. – URL: <http://www.gpg.com/>

Mobile Agents and (In-)security

T Ø N N E S B R E K N E



Tønnes Brekne (31) is on leave from his position as Research Scientist at Telenor R&D. He is currently a doctoral student at the Department of Telematics at the Norwegian University of Science and Technology in Trondheim.

Tonnes.Brekne@item.ntnu.no
tonnes.brekne@telenor.com

1 Introduction

A mobile autonomous agent is a process that is capable of migrating between different execution environments during its computation. In doing so, it may cross boundaries between different domains, and do parts of its computation on different hosts.

EXAMPLE 1. An autonomous mobile agent could be a program which visits database hosts to search for data, and place orders for query results at those sites where relevant data are found.

There also exist mobile agents that are not autonomous, which follow data, and might even be embedded in data.

EXAMPLE 2. Word and Excel macros embedded in the corresponding document types are examples of mobile agents that are embedded in document data, and that are supposed to lack autonomy, although they sometimes are autonomous.

There is no *inherent* harm in using mobile software. As with other software, security issues should be satisfactorily resolved before using mobile software. Unresolved security issues usually introduce risks which are not under control.

Knowledge about how to secure systems free of mobile agents is fairly widespread – at least among security professionals. What is not so widespread is knowledge about how one secures a system properly against any ill effects caused by mobile agent systems, especially those where the agents supported can be autonomous. This is primarily because such knowledge is currently far from complete.

Code mobility violates several assumptions which are usually considered reasonable for systems without code mobility. Chess lists in [2] a number of previously reasonable assumptions that are violated by mobile code. Each of these violations gives an attacker more possibilities.

Many of the services envisioned as important applications of mobile agents, require the agents to act as representatives of one identifiable legal person. This has important implications for the required level of security, because such action somehow has to be made legally valid. Such services will probably require the agent to be capable of generating a valid digital signature on behalf of its owner: the legal person sending the

agent which is also represented by the agent. In any case some sort of signature traceable back to the agent's owner will probably be required.

Agents acting as representatives of legal persons may also have other requirements they must fulfill.

EXAMPLE 3. Consider an agent that carries out a transaction on behalf of its owner. What happens if a fault disrupts the transaction? Two possibilities are:

- *The agent crashes on the host platform right after it commits to the transaction. Because it crashes, it does not return to its owner to inform the owner of the transaction's completion. If the agent system has "at least once" semantics, then the owner's system might send a new agent with the same instructions once more, potentially duplicating all the transactions that the first agent was supposed to carry out. If the agent system has "at most once" semantics, then the owner's system might not do anything, and get a surprise when the service provider sends its bill.*
- *The agent crashes during the transaction. Depending on the exact type of fault, and the construction of the agent platform itself, the host may not be able to recover the computation. In this case, the host might be able to roll back the transaction. Otherwise this is similar to the case above, with the exception that the transaction was not completed.*

The point of the above exercise is to demonstrate that fault-tolerance issues border on some security issues, and that the fault handling one selects has an influence on what happens when things go wrong. Faults may be induced by attacks (denial of service attacks, for example), so the way one handles faults is relevant not only for dependability, but also in part for the security of mobile agent systems.

There are other issues as well. When an agent enters a domain under the control of another operator than the one it was sent from, there is the question of policy compatibility:

- The agent may have defined which parts of it the host platform may pass on to the host, and which are only for use within the agent platform. This access control pattern may or may

not be compatible with the security policy within which the host platform operates.

- Regardless of whether data are passed on to the host or not, the second issue after policy compatibility is the enforcement of legitimate data usage. This has a lot in common with copyright issues.

Policy data that an agent carries with it, may therefore end up having a lifetime significantly longer than the agent's entire computation. This in turn implies that agent visits may entail irreversible changes to remote security policies or access control structures.

Enforcing security policies is a problem that is generally difficult, particularly when the policies also say something about how data may be used, distributed, etc. There are mainly two sides of this problem in the mobile agent context:

- securing the host against policy violations by an agent running on the host's agent platform; and
- securing an agent against policy violations by the host platform on which it is running.

A lot of work has already been done on the first half of this problem. Schneider recently proposed in [8] a mechanism that handles a class of policies more sophisticated than those that can be modeled by the classic access control matrix. The real challenge is providing the same protection to the agent and its data. This appears to be a very hard problem, as the agent computation is running on a platform completely under the control of a potential attacker. Therefore one must assume that the attacker effectively has read and write access to all parts of the agent.

2 Terminology

For the sake of brevity, the term *agent* will hereafter refer to mobile, autonomous processes (sometimes also called *mobile agents*), or code for processes embedded within data (sometimes also called *embedded agents*), unless otherwise is stated. The main practical difference between the two types of processes covered by the term mobile agent is that code embedded in data most often is not autonomous. A *platform* is the environment within which an agent executes.

Denote by Ψ the set of all possible executions (terminating and non-terminating) by any single process. Each element in Ψ is a string where each symbol represents an event, a state, or a combination of these; the type of representation is not relevant for the purposes of this article.

A process p has an associated set of possible executions Ψ_p .

Let ϕ be the set of all algorithmically definable operations. A very general model of access control is an extension of the access control matrix. The normal access control matrix model consists of a matrix of elements $A[s, o]$, which contains the rights subject s has to object o . An object is merely a named data structure. A subject is an object that happens to represent executable code or a hardware processor of some kind. A right is a reference to an operation in ϕ that may be applied to an object. Note also that s may apply some right $r \in A[s, o]$ to o at any time. This model may be generalized by:

- letting $A[s, o]$ be a set of triples (r, q, c) , where
 - r is a reference to an algorithmically expressible operation selected from ϕ ;
 - q is a state of some sort;
 - c is an algorithm that takes q as a parameter and checks if s may apply r to o ; and
- requiring c to return the decision result as well as a q' that replaces the q stored in $A[s, o]$ prior to the access attempt.

By dropping the informal requirement of algorithmic expressibility, as well as allowing subjects to be both legal persons and automated components, the above model can accommodate both the automated and manual parts of a system (read: both computers and people).

A subject s has a *right* (r, q, c) to an object o if and only if it has the legitimate authority to apply r to o under the constraints defined by c and q . A security policy can be viewed as a system for assigning and managing rights.

An interesting model proposed by Schneider in [8] is one based on execution monitoring. In the execution monitoring model, a security policy is defined by a predicate \mathcal{P} operating on a set $\mathcal{P} \subseteq \Psi$. Since it is not practical to evaluate predicates on sets, another predicate $\hat{\mathcal{P}}$ is defined, which operates on the elements of each set: the execution of a process. $\hat{\mathcal{P}}$ is necessarily at least as restrictive as \mathcal{P} (see [8] for details), and one says that a process p adheres to its security policy if for any given execution σ , $\hat{\mathcal{P}}(\sigma)$ holds.

Sets definable by such predicates are also called *properties*. Sets that cannot be defined by first order logic are called *quasi-properties* for the duration of this article.

A system, or a part of it, is called *secure* if:

1. the operator's risk associated with operating the system is bounded and under control; and/or
2. all executions adhere to the security policy defined using the risk assessment.

There exist a lot of models for access control, but until recently there was only one widespread mechanism employed in implementations: the reference monitor. Execution monitoring can be considered a generalization of the reference monitor model, where a monitor can be built into a process p prior to execution. This monitor observes events, states, or a combination of these as they occur during p 's execution. These actions form a string σ which is the prefix of some execution $\psi \in \Psi_p$. If σ is such that there is no execution in $\Psi_p \cap \mathcal{P}$ having σ as prefix, the process p is terminated. By definition p has violated the security policy by attempting an execution not in \mathcal{P} .

3 Securing Agents

This section lists many of the properties (and quasi-properties) one could wish from mobile agents in a security context. It outlines the challenges that arise when one tries to construct agents that satisfy these properties. The challenges are subdivided according to the source-based partitioning of threats given in Section 1.

Mobile agents are dependent on a host platform, which:

1. supports their computations;
2. interacts with them; and
3. supports their interaction with other "nearby" agents.

In the case of the autonomous agent in Example 1, one possible platform type would be TACO-MA. In the case of the macro agent in Example 2, the platform could be a Word or Excel program combined with underlying network services used to transport Word and/or Excel documents between hosts.

From this starting point, threats can be partitioned according to their source(s):

- agents; and
- agent environments.

Many of the relevant problems have been studied in some depth before, either in the context of viruses (in the sense of Cohen in [3]) or hostile applets. The results for viruses are perhaps especially interesting for agents embedded in document data, as is the case with Excel and Word macros, and languages like PostScript. This article, however, will not focus on the challenges that still remain with respect to malicious software.

In the following, subsections titled "Agent Attacks" deal with attacks carried out by agents on other agents in the host platform environment or on the host platform itself.

4 Availability

Availability is of fundamental importance. If resources are not available to legitimate users, other security properties or quasi-properties are usually of little or no interest. For an agent platform at some host, availability may be associated with an ability to:

- receive incoming agents, and initiate their computation within a reasonable amount of time;
- supply visiting agents with the resources necessary for them to complete their tasks at that host within a reasonable amount of time; and
- do limited recovery from faults, for as many types of faults as is cost-effectively possible.

For an agent at some platform, availability may be associated with an ability to:

- have limited fault-tolerance in the face of host-induced faults;
- have some graceful mode of failure as a (hopefully) last resort, which informs the sender of the failure, and how far the computation had proceeded prior to the failure.

Availability is primarily a dependability issue, but it is also security relevant. Several attacks base themselves on causing resource outages in the attacked system (or a component therein). Attacks of this type are usually called *denial-of-service* attacks. The point of causing resource outages may be to:

1. cause parts or all of the system to stop functioning and cause economic or other damage; or
2. generate a vulnerability, which can subsequently be used to mount the "real" attack.

This subsection will concentrate primarily on denial-of-service attacks, hereafter abbreviated DOS.

4.1 Agent Attacks

The most obvious attacks are those originating with the agent itself. Agents may mount DOS attacks by:

- attempting to consume certain types of system resources such that system performance degrades drastically; or
- disabling system resources by changing crucial data in configurations or executables.

Similar attacks by viruses and Java applets or similar software have been studied in some detail.

The remedies for these types of attacks are usually:

- Only allowing agents that run in interpreted languages, and that may only access virtual resources, thereby allowing the host system a very effective reference monitor-based control of the agent's actions.
- Absolute caps on system resources that may be allocated to any single process or owner of processes.
- Pre-emptive scheduling constructed to ensure fairness for processes with similar priorities.
- Termination of entities that do not adhere to security and/or usage policies.

Apart from these methods, the host platform's "judgement" in a very primitive sense must be employed to filter agents assumed untrustworthy from agents assumed trustworthy. To do this, the digital signing of agents has been proposed (and implemented in some commercial browsers). In order for this to have the intended effect, however, the signature should ideally represent a seal of approval derived from (trusted) third party code inspection of some type. Additionally, it is necessary to employ digital signatures that can be traced to a real-world legal person via a trusted key repository. To the author's knowledge, none of the commercial implementations satisfy both the conditions needed for such a scheme to be effective.

4.2 Platform Attacks

Platforms may mount DOS attacks on an agent by doing one or more of the following things:

1. terminating the agent execution;

2. depriving the agent of resources necessary to continue execution; or

3. inducing faults in the agent execution causing it to crash or enter a deadlock.

There may be other types of DOS attacks that platforms can carry out against agents executing on them, but the attacks described above are reasonably well modeled with the *failstop* failure model. The agent can to a certain degree be made resistant against such attacks with a slightly modified and specialized version of the Norwegian Army Protocol (NAP). Example 3 provides some additional motivation for the introduction of fault-tolerant behavior into agents and their platforms. It is desirable to have a fault-tolerant computation that executes at least once, but not more than once, neither in part nor in whole, provided no more than f faults occur.

4.2.1 The Norwegian Army Protocol (NAP)

NAP represents a solution that gives limited fault-tolerance to a mobile agent, and is proposed by Johansen et al. in [6]. This scheme allows an agent to detect failures and initiate recovery actions, and can to a limited extent cope with the DOS attacks mentioned above.

NAP assumes that each host has a platform capable of running the agent, and that platform crashes can be detected by a well-defined set of platforms. A fail-stop failure model is thus assumed, and in addition, a bounded crash rate is assumed:

There exists a positive integer f such that for any $0 \leq i \leq f$ no more than i crashes of hosts or platforms occur during the maximum period of time taken by an agent to traverse i distinct hosts.

In NAP a mobile agent is an ordered collection of mobile processes.

The i^{th} action, a_i , of an agent is transformed to a *fault-tolerant action*, which is a pair (a_i, a'_i) such that a_i is a *regular action*, and a'_i is a *recovery action*. The execution of a fault-tolerant action satisfies:

1. a_i executes at most once, irrespective of whether it fails or not.
2. If a_i fails, then a'_i executes at least once, and executes without failing exactly once.
3. a'_i executes only if a_i fails.

A failure is caused by some type of fault during the execution of an action. When a failure occurs no earlier than action a_i and prior to action a_{i+1} , the recovery action a'_i starts execution.

A simplified and slightly modified version of NAP (hereafter called NAP') is presented here. Whereas the NAP protocol tolerates f fail-stops during a computation, the modified version should tolerate f fail-stops per action-broadcast pair.

Since a mobile agent is represented by many similar agents in the NAP' protocol, write $a_{i,j}$ for the j^{th} action taken by agent number i for the $f+1$ agents in a single NAP' protocol execution. Similarly the recovery action for the j^{th} action taken by agent number i is written $a'_{i,j}$. In the following, the *head* will refer to the primary process, which is doing the actual agent computation. The *tail* consists of the remaining backup processes.

It is assumed that the agent execution consists of a series of actions satisfying the following assumptions.

1. At least one action is executed on any given platform.
2. An action may consist of a move to a new platform, possibly entailing movement to a different host.
3. After each completed action, state information is propagated to the backup processes using a reliable broadcast. This state information should be enough for recovery if the subsequent action fails.
4. The actions need not be specifically listed, but are given algorithmic expression.
5. Upon completing an action, reliable broadcast is immediately initiated.

The preconditions of a NAP' execution is the following:

1. The degree of fault-tolerance $f \geq 1$ is fixed.
2. The sender has a platform within his domain capable of supporting (part of) a NAP' computation.
3. The $1+f$ head and tail processes are initiated at the sender's platform, with the initial state of the head distributed to all tail processes.

By convention, the head is written p_0 , and the tail processes are written p_1, \dots, p_f . The platform

supporting a process p_j is written P_j . Note in particular that it is not necessarily the case that $i \neq j \Rightarrow P_i \neq P_j$. Note also that P_j changes during the agent execution if the agent ever changes platform. A broadcast message associated with the completion of action i is written b_i .

The NAP' execution itself is carried out by letting the head execute action $a_{1,i}$, immediately followed by a reliable broadcast of b_i to the tail processes containing the new state information of the head. The head is elected to ensure that the broadcast is delivered to all the tail processes. The broadcast itself proceeds as the one given in [6], but with some differences. The possible outcomes of the broadcast are essentially unchanged:

1. No platform delivers b_i . This happens if either P_0 failed or if p_0 's execution of action $a_{0,i}$ somehow failed. One of the tail processes p_j must therefore begin recovery action $a'_{j,i}$. Immediately after recovery (including the new broadcast) is completed, p_j takes on p_0 's role and spawns a new process p_j to take its previous role.
2. The platform P_1 delivers b_i . This happens only if all non-faulty platforms supporting the agent have delivered b_i . The head process p_0 may thus begin executing action $a_{0,i+1}$.
3. A platform P_j delivers b_{i+1} , but P_0 does not. Thus either P_0 failed or p_0 's execution failed somehow before the broadcast terminated. The P_j with the lowest $j \neq 0$ such that $P_j \neq P_0$ acts as rearguard, and executes the recovery action $a'_{j,i+1}$. Immediately after recovery (including the new broadcast) is completed, p_j takes on p_0 's role and spawns a new process p_j to take its previous role.

The postconditions of a NAP' execution is the following:

1. The head has completed the last action $a_{0,n}$.
2. The last broadcast has terminated.
3. The head and tail processes have terminated.

This description concentrates on the invocation of the fault-tolerance parts. The broadcast protocol details and other details necessary for a complete implementation should not deviate much from that described in [6].

5 Confidentiality

Confidentiality is about controlling the ability of any given subject to extract information from an object. Information here is taken in the widest known sense, the information-theoretic sense. There are three basic methods controlling such information extraction:

1. Rendering data uninterpretable for all practical purposes (which is what encryption does), such that the usual transformation of data to information through interpretation becomes practically impossible.
2. Partitioning data into discrete units, and treating the units as distinct objects in a secret sharing scheme, or more generally: an access control scheme.
3. Blocking the release of non-sensitive data *A* that could reveal information about other, sensitive data *B* due to known or deducible functional dependencies between *A* and *B*. In practice these are inference controls as those used in statistical databases.

For platforms the techniques for implementing these are well-known. The challenge is replicating those mechanisms in mobile agents. It might also be of interest to replicate them for embedded agents.

EXAMPLE 4. An embedded agent might be contained in a partially encrypted document, with instructions to only reveal the encrypted parts to pre-specified platforms. This is something with typical application to Word documents or similar document formats.

5.1 Agent Attacks

The agent attacks are fairly well studied. Some ways of leaking confidential information are

1. attempting unauthorized accesses of other agents' code/data areas;
2. installing a payload on the host computer, executable *outside* the platform as a "normal" process (such as word macro viruses with DOS virus payloads);
3. attempting to manipulate the platform in order to gain access to confidential information (attacking crypto-APIs, for example); or
4. using pure information theoretic attacks to design queries used by agents to collect data for subsequent tracker-type attacks (see [5]) on collected data, a technique which could typically be employed in business (or other) intelligence activities.

The first attack above is fairly effectively halted by a virtual machine-based platform (as for Java applets), *provided* the transformation from an agent's implementation language to the code actually run on the platform is correct, and introduces no new functionality. For agents somehow running as native machine code, one needs hard-

ware support for controlling memory access from the operating system and/or platform such that agents cannot exploit host-specific architectural information to mount attacks.

The second attack above is not necessarily halted by a virtual machine-based platform, although such a platform still forms a fairly effective defense. The reason is that if the agent is allowed to write to a file and can choose part of or all of its name, it can still attempt the installation of a Trojan horse.

The third attack represents the typical attack implemented by a hostile Java applet, where bugs in the implementation are exploited in order to execute unauthorized actions. Defending against this type of attack depends for the most part on a correct implementation that actually logically isolates the agent from manipulating the platform in an unauthorized manner.

The fourth attack is hard to avert. Any investment spent to defend against these attacks depends on how information is made available to the agent.

5.2 Platform Attacks

The complementary case, where the platform is regarded as a potentially malicious entity resembles to a great degree the problem outlined in a paper by Anderson and Needham [1]. The host has effectively complete control of *all* plaintext data, including the agent platform, and any agents executing on it, along with any of their code and data stored at the host in question.

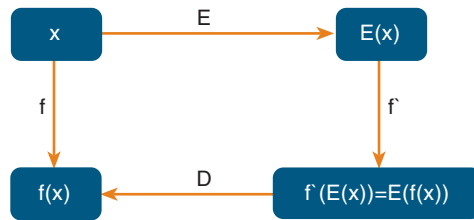
The host can under such circumstances achieve effortless compromise of *all* information, save that which somehow is encrypted in such a form as to still be of use to the agent. This problem is a difficult one to solve. It appears to be crucial to enable mobile agents to:

1. use secrets in their data without revealing them to the platform;
2. securely generate digital signatures; and
3. securely encrypt information extracted from the host platform or decrypt encrypted information meant for that platform.

Properties such as these are only possible if one can encrypt the agent's code without changing its properties when viewed from the outside as a process. In other words, the encrypted agent must ideally be able to:

1. encrypt/decrypt selected inputs and/or outputs;

Figure 1 This figure illustrates the general principle of privacy homomorphisms. The plaintext is x , the encryption function E , decryption function D , and f is a function applied to plaintext data. The function f' is the counterpart of f , when applied to encrypted data



2. interact with its environment using either encrypted or plaintext messages, or a combination of both;
3. sign certain data in complete confidentiality while executing on a potentially malicious host; and
4. support encrypted Turing-universal computation.

If one looks at these requirements, it should become clear that what is actually needed is the ability to apply operations to encrypted data. This problem was originally studied in the form of privacy homomorphisms for databases. The concept of privacy homomorphisms is a good idea, and addresses many of the requirements above as long as strong encryption is not a requirement. In spite of this, the concept could be central to solving the above problem. The few subsequent approaches in the field have to some degree based themselves on the privacy homomorphism concept, even if the resulting system is strictly speaking not a privacy homomorphism.

5.2.1 Privacy Homomorphisms

Let S and S' be non-empty sets with the same cardinality. A bijection $E : S \rightarrow S'$ is the encryption function, with its inverse D being the corresponding decryption function. Denote an algebraic system for cleartext operations by

$$U = (S; f_1, \dots, f_k; p_1, \dots, p_l, s_1, \dots, s_m),$$

where the $f_i: S^{g_i} \rightarrow S$ are functions with arity g_i , the p_i are predicates with arity h_i , and the s_i are distinct constants in S . Denote U 's counterpart for operation with encrypted data by:

$$C = (S'; f'_1, \dots, f'_k; p'_1, \dots, p'_l; s'_1, \dots, s'_m),$$

where each f'_i corresponds to f_i , and each p'_i corresponds to p_i , and each s'_i corresponds to s_i . Thus f'_i has arity g_i and p'_i has arity h_i .

A mapping E is called a *privacy homomorphism* if it satisfies the following conditions:

1. For all f_i and f'_i :

$$f'_i(a'_1, \dots, a'_{g_i}) = E_K(f_i(D_K(a'_1), \dots, D_K(a'_{g_i}))),$$

where $a'_1, \dots, a'_{g_i} \in S'$, and K is a symmetric encryption key.

2. For all p_i and p'_i :

$$p'_i(a'_1, \dots, a'_{g_i}) \text{ if and only if } p_i(D_K(a'_1), \dots, D_K(a'_{g_i})).$$

3. For all s_i and s'_i $D_K(s'_i) = s_i$.

Although this is elegant, it has an inherent weakness, summarized in theorem 3.1 in [5]. In essence, it is impossible to have a secure encryption function E for an algebraic system like U when that system has a predicate p_i inducing a total order on the constants s_1, \dots, s_m , and it somehow is possible to determine the encrypted version of each constant. The following example is from the proof of the theorem in [5].

EXAMPLE 5. Take a plaintext system where $s_i = i \in \mathbb{N}$ for all $1 \leq i \leq m$. Let one function be addition (written $+$), and one predicate be the relation less-than-or-equal-to (written \leq). The corresponding function applied to encrypted data is written $+$, and the corresponding predicate is written \leq' . If the cryptanalyst knows 1 and $1'$, it is possible to decrypt any c' to c by doing a binary search using $+$, $1'$, and \leq' .

5.2.2 Computing with Encrypted Functions

The privacy homomorphism is revisited in work by Sander and Tschudin [7]. They mention two potential candidates for encrypted computation:

1. polynomials encrypted with a particular type of privacy homomorphism; and
2. rational function composition, where one rational function is used to encrypt another.

Only the first scheme, called non-interactive evaluation of encrypted functions, is detailed in their work. Sander and Tschudin present a simple protocol demonstrating how it could work. The protocol is as follows:

1. Alice encrypts f .
2. Alice creates a program $P(E(f))$ which implements $E(f)$.
3. Alice sends $P(E(f))$ to Bob.

4. Bob executes $P(E(f))$ using x as argument.
5. Bob sends $P(E(f))(x)$ to Alice.
6. Alice decrypts $P(E(f))(x)$ to obtain $f(x)$.

The encryption itself uses an additively homomorphic encryption scheme on a ring \mathbb{Z}_n .

EXAMPLE 6. An additively homomorphic scheme is presented in [7]. The encryption function $E : \mathbb{Z}_{p-1} \rightarrow \mathbb{Z}_p$ for a prime p is g^x for plaintext x . g is a generator of $\mathbb{Z}/p\mathbb{Z}$. Addition of plaintext is thus arithmetic addition, while addition of ciphertext is done using multiplication.

The plaintext function is a polynomial with coefficients from \mathbb{Z}_{p-1} . Encryption of f is achieved by encrypting each of f 's coefficients individually. The resulting f' is applied to plaintext data by using so-called mixed multiplication. Mixed multiplication is an operation where the product $E(xy)$ is computed from $E(x)$ and y directly. Note that $E(xy) = (g^x)^y$, so $E(x)$ must be raised to the y^{th} power in order to compute $E(xy)$. This can be done using a double-and-add algorithm, which translates into a square-and-multiply algorithm for encrypted data.

The scheme in Example 6 appears to avoid the inherent weakness of privacy homomorphisms by selecting an encryption function such that there is no known easily computable total ordering of the encrypted elements. Along the way, the ability to do repeated applications of the function on the encrypted data appears to be lost, as the result is encrypted, and Bob can only compute mixed-multiplication. Note in particular that if Bob were able to compute multiplication in the encrypted domain, he would probably be able to decrypt the ciphertext fairly easily. The ability to do multiplication with only encrypted elements requires knowledge of either:

1. the generator g ; or
2. a method of efficiently solving the discrete logarithm problem.

Thus the second scheme trades functionality for increased security compared with easily applicable privacy homomorphisms.

6 Integrity

Integrity deals with the detection of unauthorized operations on data in a system. The restoration of damage after unauthorized operations have occurred is usually considered a part of dependability, while the prevention of the unauthorized operations is typically taken care of by authentication and access controls.

Integrity can typically be boiled down to the unauthorized detection of modification of system objects by editing, creating, deleting, or reordering data in objects. Integrity is usually achieved by the inclusion of redundant information strongly dependent on the data (or system structure) to be protected such that attempts at unauthorized

1. modifications (or creations or deletions) of objects are detectable; and
2. modifications (or creations or deletions) of the redundant information, are detectable.

In addition to the common integrity concept there is also a concept of *sequence integrity*, where a mandatory order is imposed on otherwise individual data/messages/processes in the system. Such integrity is relevant for ordered data, such as video and audio streams. It is also relevant for protocols being executed by an agent, its platform, or the host upon which the platform resides.

6.1 Agent Attacks

As with confidentiality, the agent attacks are fairly well studied, and they are also very similar. Most of them can be realized if the access controls are not precise or not secure (as defined in Denning [4]). Some ways of violating integrity are

1. attempting unauthorized accesses of other agents' code/data areas;
2. unauthorized creation, modification, or deletion of data associated with host software (such as changing mailing lists in MS Outlook);
3. feeding the platform with misleading information or falsifications; or
4. intentionally trying to trigger race conditions or confuse host protocol state machines by manipulating protocol messages used in communication with the platform.

In addition, mobile agents may also become malicious if their state is corrupted during their execution, irrespective of whether that corruption was intentional or not.

The first attack above is also halted by any effective virtual machine-based platform (as for Java applets), under similar conditions as those mentioned in Section 5.1. Similarly, if the agent is running as a native machine code process on the platform, there must be hardware support for controlling memory access from the host's oper-

ating system(s) and/or platform to enforce access restrictions placed on the agents.

The second attack is not necessarily halted by a virtual machine-based platform, although such a platform can thwart a selection of direct attacks. The problem is that the agent may try to do actions allowable *within* its platform environment, that still affect host data external to the platform in accordance with the platform's policy, but in violation of the agent's policy.

EXAMPLE 7. Consider a spreadsheet which includes an embedded agent in the form of macros for a computation. A virtual machine executing those macros might allow write operations to one particular file on the host. An attacker could exploit this by letting the original embedded agent itself be harmless, while the output to the file is a spreadsheet with macros that make up a malicious agent. Such an agent might subsequently be executed with the permissions of a local user on the host. This type of attack is also related with the problems concerning legitimate usage of resources.

The third type of attack can in part be thwarted using systematic trust management. Discerning intentionally falsified information from wrong information given in good faith is at best very difficult, and undecidable, as it encompasses at least one problem with inherent ambiguity (computers are generally not particularly good at deducing intent). Farmer et al. give examples of this type of attacks in [5].

The fourth attack is usually stopped by protocols constructed to take such conditions into account, or by restricting the communication of the agent (as is already done with Java applets).

6.2 Platform Attacks

Platforms can attempt to compromise the integrity of an agent by

1. modifying data carried by the agent or in the agent's workspace;
2. manipulating the agent's execution; or
3. not properly protecting the agent's workspace.

The novel attack here is the manipulation of execution. Not only must the agent's data be protected, but the agent's execution must also be protected. Even if an agent could be executed in an encrypted form, this would still be a problem – the only difference would be that the platform would not know what it was doing. In effect,

some sequence $\sigma = \sigma_1 \sigma_2 \dots \sigma_n$ of actions must not be interfered with, or any interference must be detectable.

6.2.1 Execution Traces

Vigna proposes in [9] a protocol, which in combination with a trusted third party, enables the sender to check the execution of an agent upon completion. The protocol bases itself upon a system where each platform records the execution trace of an agent from the agent's execution on that platform. An *execution trace* is a series of pairs (n, s) , where n uniquely identifies a particular action in the agent, and s records changes in internal variables as a result of the execution of n . In a sense this is nothing more than specifying the format of an "action", as mentioned earlier, so these traces can also be considered executions in the sense mentioned in Section 2.

The protocol itself consists of messages prior to, and after, the agent's execution which transmit data about the agent and its execution. After an execution, the agent's sender should be able to check an execution trace T^p by using the data in T^p to simulate the complete execution of the agent locally, and compare the outcome to that provided by the remote platform. Obviously, this check is only done when there is a suspicion that the remote platform somehow has tried to compromise the agent's computation.

Denote by $E_K(\text{data})$ the symmetric encryption of "data" by encryption function E using key K . Denote by $X_s(\text{data})$ the "data" signed with X 's private key. Denote by $H(\text{data})$ the hash value of "data". Write the identity of the trusted third party as T . The notation $A \xrightarrow{m} B:\text{data}$ denotes the transmission of message m from A to B , where message m consists of "data".

Let A be the sender of the agent in question. Denote by B_1, \dots, B_n the list of platforms the agent is to visit. For the initial platform, the following steps initiate the protocol:

1. $A \xrightarrow{m_1} B_1 :$
 $A_s(A, B_1, E_{K_A}(p, S_A), A_s(A, i_A, t_A, H(p), T))$
2. $B_1 \xrightarrow{m_2} A :$
 $B_{1s}(B_1, A, i_A, H(m_1), M)$

The field $A_s(A, i_A, t_A, H(p), T)$ will also be written agent_{t_A} , and be referred to as the *agent token*. The contents of the messages is given in Table 1.

Before continuing, A must examine M to see if it constitutes an acceptance or refusal on B_1 's part. If it is a refusal, the protocol is finished. If it is an acceptance, the protocol continues for the initial platform as follows:

3. $A \xrightarrow{m_3} B_1$:
 $A_s(A, B_1, i_A, B_{1p}(K_A))$
4. $B_1 \xrightarrow{m_4} A$:
 $B_{1s}(B_1, A, i_A, H(m_3))$

The contents of the messages is given in Table 2.

When all necessary verifications have been done, B_1 can begin executing the agent. This execution continues until the agent requests migration to a new platform B_2 . The migration from any given B_i to B_j requires a separate protocol. Denote by $T_{B_i}^p$ the trace produced by the part of the execution carried out on B_i . The migration to the next platform B_j is initiated by the following three messages:

1. $B_i \xrightarrow{m} B_j$:
 $B_{is}(B_i, B_j, \mathbf{agent}_A, H(T_{B_i}^p), H(S_{B_i}, t_{B_i}))$
2. $B_i \xrightarrow{m'} B_j$:
 $B_{is}(K_{B_i}(p, S_{B_i}), H(m))$
3. $B_j \xrightarrow{m''} B_i$:
 $B_{js}(B_j, B_i, i_A, H(m, m'), M)$

The last field in the third message (M) contains either B_j 's acceptance to run the agent, or its refusal to run the agent. The contents of the messages are given in Table 3.

7 Authenticity

There are two types of authenticity:

1. authenticity of origin; and
2. authenticity of identity.

Authenticity of origin deals with proving or disproving the claimed identity of the creator of some data, be it art, code or something else. Authenticity of origin is a *very* hard problem, which might not have any good solution. With respect to agents, the most relevant cases probably include authenticity of the origin of:

1. data carried by an agent from its sender;
2. data acquired by an agent at some platform;
3. an agent's code.

Message	Sender	Recipient	Message part	Explanation
m_1	A	B_1	$A_s(\$ $A,$ $B_1,$ $E_{K_A}(\$ $p,$ $S_A),$ $A_s(\$ $A,$ $i_A,$ $t_A,$ $H(p),$ $\mathcal{T}))$	Signature generated by A A's identity B_1 's identity Encryption with A 's key K_A The agent's code The agent's initial state Signature generated by A A's identity The agent's identifier Time stamp for agent dispatch Hash of agent's code Identity of TTP
m_2	B_1	A	$B_{1s}(\$ $B_1,$ $A,$ $i_A,$ $H(m_1),$ $M)$	Signature generated by B_1 B_1 's identity A's identity The agent's identifier Hash of previous message B_1 's reply

It is important to note that authenticity of origin is not the same as authenticity of the sender: data may be sent without their creator having to do the sending in person.

Table 1 Initialization of protocol for execution traces

Authenticity of identity deals in general with proving or disproving the identity claimed by a subject. Traditionally, authenticity has been proven using one or more independent techniques, and efforts have been concentrated on the case where humans prove their identity to an (automated) subject of some sort, or where automated subjects need to prove their identities to each other as the initial step in some protocol. Within automated environments, only passive data have been authenticated. What one effectively needs for agents is authentication of process instances.

Table 2 The rest of the initialization of the protocol for execution traces, if B_1 agrees to participate in the protocol

Message	Sender	Recipient	Message part	Explanation
m_3	A	B_1	$A_s(\$ $A,$ $B_1,$ $i_A,$ $B_{1p}(\$ $K_A))$	Signature generated by A A's identity B_1 's identity The agent's identifier Encryption with B_1 's public key A's symmetric encryption key
m_4	B_1	A	$B_{1s}(\$ A $i_A,$ $H(m_3))$	Signature generated by B_1 A's identity The agent's identifier Hash of message m_3

Message	Sender	Recipient	Message part	Explanation
m	B_i	B_j	$B_{is}(\$ $B_i,$ $B_i,$ $\text{agent}_A,$ $H(T^P_{B_i}),$ $H(S_{B_i}),$ $t_{B_i})$	Signature generated by B_i B_i 's identity B_i 's identity The agent token Hash of agent trace at B_i Hash of agent state at migration Time stamp at migration
m'	B_i	B_j	$B_{is}(\$ $K_{B_i}(\$ $p,$ $S_{B_i})$ $H(m)$	Signature generated by B_i B_i 's symmetric encryption key The agent's code The agent's state at migration The hash of B_i 's previous message
m''	B_i	B_j	$B_{is}(\$ $B_i,$ $B_i,$ $i_A,$ $H(m, m'),$ $M)$	Signature generated by B_i B_i 's identity B_i 's identity The agent's identifier Hash of messages m and m' B_j 's reply to B_i

Table 3 Messages for handling migration from one platform to another

7.1 Agent Attacks

Without some mechanism for ensuring authenticity of origin, agents may present false claims of origin for some or all of

1. its own data;
2. the data it acquires during its execution; or
3. data it feeds the platform.

One method of ensuring authentication of origin is to use steganography to mark the object in question. This can be difficult, however, for reasons mentioned in subsection 7.2.1.

Almost all ways of exploiting insufficient authentication of identity are forms of masquerade attacks. In particular, an agent may attempt to masquerade as another agent by exploiting insufficiently authenticated protocol messages. Ensuring this type of authentication in agents properly seems to require either:

- an ability to execute cryptographic primitives in plain view; or
- platform-supported communication with the agent's sender.

The former "solution" runs into the problems mentioned in Section 5.

7.2 Platform Attacks

Without proper authentication of origin, platforms may present false claims of origin for some or all of

1. its own data;
2. data acquired from visiting agents; or
3. data fed to visiting agents.

As with the corresponding agent attacks, this could be based on digital watermarking techniques, if there are any that work with the type of data in question.

Without proper authentication of the platform's identity, an agent may be exposed to any attack involving:

- unauthorized access of the agent's data; or
- simulation of a transaction that was supposed to execute somewhere else, thereby "duping" the agent.

7.2.1 Watermarking and Steganography

One suggested mechanism for authenticating origin, is the digital watermark. Watermarking works by placing information in the data one wants to mark. This information may or may not be visible, although most efforts are directed at invisible watermarks, as one does not want the marking to affect the data in any perceivable way. The information will typically contain the identity of the originator/creator, along with other document-specific information of relevance.

Systems for invisible digital watermarks are often based on steganographical techniques. When a digital watermark is invisible, it may be

1. only perceptually invisible; or
2. both perceptually invisible and hidden as a communication of data.

The digital watermark can be considered a communication where the message is an identity of a creator, possibly along with other information. *Steganography* is the art of concealing the *existence* of a communication, so steganographical techniques are necessary for the second variant of invisible watermarks.

Ideally any steganographical marking used for authentication of origin should:

1. not affect the original data such that legitimate users perceive it as degraded;
2. withstand changes to the original data without being compromised, and becoming uninterpretable for at least the owner; and
3. encode the identity of the originator and other information in a robust manner.

So far so good, but this is still insufficient. This is only of use against the casual attacker. In order for such a marking to actually be of use in this context, it must also

4. withstand tailored attacks designed to destroy that particular type of marking;
5. contain enough information to beyond any reasonable doubt separate the markings of different originators.

This is an impressive list, and any marking that actually satisfies this list will be fairly good. There remains but one property that must be satisfied if the marking is to be of any real use:

6. The first marking must withstand *all* subsequent markings satisfying properties 1 – 5 subsequently applied to the data, and it must be possible to single out the first marking as such – the marking applied first to the data.

This last property is not likely to be satisfied, as it depends on the first marking being affected by all subsequent markings without losing any of its data. To the author's knowledge, no such system exists. Any document with two or more steganographically "strong" markings may therefore be of little or no use as evidence in any court case. Copyright cases would seem to be a relevant example.

Even if one ignores the last property, achieving the first five properties for most data types is in itself a formidable task. The reason for this is that steganographical techniques are notoriously type-dependent when it comes to their influence on perceived data degradation.

EXAMPLE 8. Steganographic techniques that provide little or no image degradation, and may work very poorly when applied to text. This means that image manipulations that pass unnoticed to the naked eye, may stand out as glaring errors when applied to text, and vice versa.

In addition to all this, the five first properties will almost always compete with the actual message for bandwidth. In particular, relatively small objects may be extremely difficult to protect steganographically, which could rule out use of such techniques to some degree.

Currently, steganography has not advanced far enough to sort out all the above problems. Especially property 2 above demands a lot of bandwidth out of that which is available. Steganographical techniques do, however, have a certain promise in this area.

8 Legitimate Usage

Legitimate usage concerns itself with ensuring that an object (which may be a subject) is used subject to constraints imposed by that object's creator. In a sense it contains confidentiality and integrity as specialized forms of legitimate usage. In essence, legitimate usage can be considered the enforcement of a security policy defined for a particular instantiation of an object, *regardless of whether it is beyond the reach of the formal owner or not.*

EXAMPLE 9. Enforcement of copyright on DVD movies, as well as their zoning, is a matter of enforcing legitimate usage. Whether zoning actually is a legal policy or not is another matter.

Example 9 illustrates the point of control fairly well: enforcement of copyright would be possible with known techniques *provided* the object is kept within the boundaries of the system in which it was created.

This problem has many similarities with that of confidentiality in an agent context. It should be immediately obvious that legitimate usage is in general impossible to enforce unless some robust form of authenticity of origin exists. Unless authenticity of origin can be ensured, the policy restrictions placed on an object may be circumvented by simply changing the creator's identity. This is the core of all problems regarding copyright enforcement.

Since there as of today exist no known schemes for authenticating origin based on difficult problems, or problems conjectured to be difficult, such as the discrete logarithm problem, the existence of such a scheme will be assumed for the remainder of this section.

Enforcing legitimate usage where objects move out of their owner's control, has always been a problem. Agents only highlight the difficulties with respect to enforcing confidentiality and integrity. In effect, this is the problem of global security policy enforcement for objects, even though access control mechanisms only control the objects as long as they exist within their system. This problem in itself consists of the following subproblems:

1. authentication of origin, which is assumed solved in this section to highlight the other problems;
2. policy interaction;
3. mechanisms for enforcing very general types of policy (see the generalized access matrix model in Section 2).

Assume some agent a carries with it a set of objects D , which is to be distributed to other systems via agent platforms. Each of the objects D has a usage policy $A[\cdot, D]$ that is to be enforced irrespective of which domain they exist in. $A[\cdot, D]$ is interpreted as: the rights for an arbitrary subject to objects in D .

The platform can control the agent's action by inserting execution monitoring into the agent. This enables the platform a fairly general and detailed control of the agent's actions.

The interesting thing is that embedded agents may come into their own here. Many data types are constructed such that they effectively depend on embedded agents to be displayed and/or used correctly. This means that the platform must use the data by executing the agent. The agent effectively provides a service to the platform. Thus it is possible to implement a certain control of usage by constructing an agent with execution monitoring. This time however, the agent monitors the platform's requests, and terminates if the platform attempts a series of requests that violate the data's security policy.

There are two problems with this approach:

1. it requires some method of enforcing agent execution integrity;
2. it probably requires the data to be encrypted, and thus the agent to be capable of encrypting and/or decrypting data using encrypted code.

9 Conclusion

This article has outlined some of the major challenges in making agents viable as a computing paradigm in contexts where a high level of security is necessary.

Among the main challenges facing constructors of secure mobile agent systems are:

1. enabling the secure generation of strong cryptographic functions by mobile agents;
2. ensuring sufficient fault-tolerance;
3. enforcing agent policies; and
4. handling policy interactions.

Work on these issues has started in earnest only recently. It may be a while before one can conclusively state whether or not mobile agents will be sufficiently securable for applications in, for example, electronic commerce.

References

- 1 Anderson, R, Needham, R. Programming Satan's computer. In: *Computer Science Today : Trends & Developments*. van Leeuwen, J (ed.). Lecture Notes in Computer Science, vol. 1000. Berlin, Springer, 1995.
- 2 Chess, D M. Security issues in mobile code systems. In: *Mobile Agents and Security*. Vigna, G (ed.). Lecture Notes in Computer Science, vol. 1419. Berlin, Springer, 1998.
- 3 Cohen, F. *Computer Viruses*. PhD thesis. University of Southern California (USC), 1985.
- 4 Denning, D. *Cryptography and Data Security*. Reading, Mass., Addison-Wesley, 1982.
- 5 Farmer, W M, Guttman, J D, Swarup, V. Security for mobile agents : Authentication and state appraisal. In: *Proceedings of the European Symposium on Research in Computer Security (ESORICS), number 1146 in LNCS*, 118–130. Berlin, Springer, 1996.
- 6 Johansen, D et al. Nap : Practical fault-tolerance for itinerant computations. In: *Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (ICDCS '99)*. Gouda, M G (ed.). 180–189, 1999.
- 7 Sander, T, Tschudin, C F. Protecting mobile agents against malicious hosts. In: *Mobile Agents and Security, LNCS State-of-the-Art Survey*. Vigna, G (ed.). Berlin, Springer, 1998.
- 8 Schneider, F B. *Enforceable security policies*. Cornell University, Ithaca, New York 14853, Dept. of Computer Science, 1998. (Technical report.) (Revision of July 24, 1999.)
- 9 Vigna, G. Cryptographic traces for mobile agents. In: *Mobile Agents and Security*. Vigna, G (ed.). Lecture Notes in Computer Science, vol. 1419. Springer, 1998.

An Overview of Firewall Technologies*)

HABTAMU ABIE



Habtamu Abie (42) is Research Scientist at the Norwegian Computing Centre. He has previously worked as Senior Engineer and Research Scientist at Telenor R&D (1997–1999) and as Scientific Associate and Scientific Fellow at CERN in Geneva, Switzerland (1989–1993). From 1994 to 1997 he also held a research position at ABB Corporate Research and worked as a Software Development Engineer at Nera AS and Alcatel Telecom Norway AS. He received his Engineering Diploma in Electrical Data Processing from Gjøvik Engineering College (1983), and his B.Sc. (1985) and M.Sc. degrees (1988) in Computer Science from the University of Oslo. He is also working towards his Ph.D. in Computer Science at the Department of Informatics, University of Oslo and UNIK. His past and present research interests encompass security for distributed systems, distributed object computing, architecture and methodology, formal methods and tools, data acquisition and control systems, hard real-time systems, and mobile and personal computing.
habtamu.abie@nr.no

The increasing complexity of networks, and the need to make them more open due to the growing emphasis on and attractiveness of the Internet as a medium for business transactions, mean that networks are becoming more and more exposed to attacks, both from without and from within. The search is on for mechanisms and techniques for the protection of internal networks from such attacks. One of the protective mechanisms under serious consideration is the firewall. A firewall protects a network by guarding the points of entry to it. Firewalls are becoming more sophisticated by the day, and new features are constantly being added, so that, in spite of the criticisms made of them and developmental trends threatening them, they are still a powerful protective mechanism. This article provides an overview of firewall technologies.

1 Introduction

Today's networks change and develop on a regular basis to adapt to new business situations, such as reorganisations, acquisitions, outsourcing, mergers, joint ventures, and strategic partnerships, and the increasing degree to which internal networks are connected to the Internet. The increased complexity and openness of the network thus caused makes the question of security more complicated than hitherto, and necessitates the development of sophisticated security technologies at the interface between networks of different security domains, such as between Intranet and Internet or Extranet. The best way of ensuring interface security is the use of a firewall.

A Firewall is a computer, router or other communication device that filters access to the protected network [16]. Cheswick and Bellovin [6] define a firewall as a collection of components or a system that is placed between two networks and possesses the following properties:

- All traffic from inside to outside, and vice versa, must pass through it.
- Only authorised traffic, as defined by the local security policy, is allowed to pass through it.
- The firewall itself is immune to penetration.

Such traditional network firewalls prevent unauthorised access and attacks by protecting the points of entry into the network. As Figure 1 shows, a firewall may consist of a variety of components including host (called bastion host), router filters (or screens), and services. A gateway is a machine or set of machines that provides relay services complementing the filters. Another term illustrated in the figure is “demilitarized zone or DMZ” [6]. This is an area or sub-network between the inside and outside networks that is partially protected. One or more gateway machines may be located in the DMZ. Exemplifying a traditional security concept, defence-in-depth, the outside filter protects the

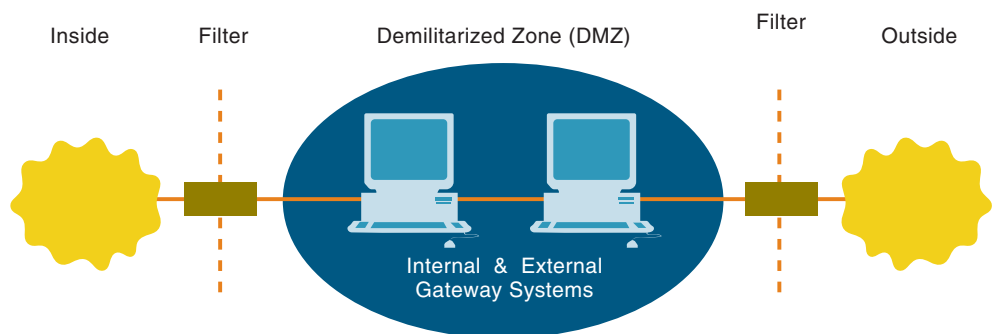


Figure 1 Firewall schematics

*) The work was carried out while the author was a Research Scientist at Telenor R&D.

gateway from attack, while the inside gateway guards against the consequences of a compromised gateway [6, 9]. Depending on the situation of the network concerned, there may be multiple firewalls, multiple internal networks, VPNs, Extranets and perimeter networks. There may also be a variety of connection types, such as TCP and UDP, audio or video streaming, and downloading of applets. Different types of firewall configuration with extensive practical guides can be found in [6, 4]. There are also many firewall products on the market from different vendors. See [8] for an updated list of products and vendors.

This article surveys the basic concept of firewall technology by introducing the various kinds of approach, their applications, limitations and threats against them.

2 Firewalls: Basic Approaches and Limitations

Firewall technology can be used to protect networks, by installing it strategically at a single security screen station where the private network or the Intranet connects to the public Internet, making it easier to ensure security, audit and monitor traffic, and trace break-in attempts. It can also be used to isolate sub-networks, in order to provide additional layers of security (defence-in-depth) within the organisation. There are three basic approaches or services that a firewall uses to protect a network: packet filtering, circuit proxy, and application proxy [6, 10]. Some authors [12, 9] broadly classify these into two kinds of approach: transport level and application level (by including circuit proxy in this category).

2.1 Packet Filtering

Firewalls having this function perform only very basic operations, such as examining the packet header, verifying the IP address, the port or both, and granting and denying access without making any changes. Due to this simplicity of operation, they have the advantage of both speed and efficiency. The filtered packets may be incoming, outgoing or both, depending on the type of router. An additional advantage is that they do their job quite independently of the user's knowledge or assistance, i.e. they have good transparency. Packets can be filtered on the basis of some or all of the following criteria: source IP address, destination IP address, TCP/UDP source port, and TCP/UDP destination port. A firewall of this type can block connections to and from specific hosts, networks and ports. They are cheap since they use software already resident in the router, and provide a good level of security since they are placed strategically at the choke point.

2.2 Circuit proxy

The second approach is the use of what is called a circuit proxy. The main difference between the circuit proxy and the packet filtering firewall is that the former is the addressee to which all communicators must address their packets. Assuming access has been granted, the circuit proxy replaces the original address (its own) with the address of the intended destination. It has the disadvantage of laying claim to the processing resources required to make changes to the header, and the advantage of concealing the IP address of the target system.

2.3 Application Proxy

The third approach involves the use of what is known as an application proxy. An application proxy is more complicated in operation than a packet filtering firewall or a circuit proxy. The application proxy understands the application protocol and data, and intercepts any information intended for that application. On the basis of the amount of information available to make decisions, the application proxy can authenticate users and judge whether any of the data could pose a threat. The price to be paid for this more comprehensive function is that the clients often have to be reconfigured, sometimes a complicated process, with a consequent loss of transparency. Application proxies are referred to as proxy services, and the host machines running them as application gateways.

2.4 Packet Inspection Approach

This approach, in contrast to the technologies so far described, involves inspecting the contents of packets as well as their headers. An inspection firewall carries out its inspection by using an inspection module, which understands, and can therefore inspect, data destined for all layers (from network layer to application layer). It carries out its inspection by integrating all information gathered from all layers into a single inspection point, and then examining it. A state-full inspection firewall is one which also registers the state of any connection it is handling, and acts on this information. An example of a state-full inspection firewall is the state-full packet-filtering mode in Checkpoint's "Firewall-1" [5] or Network Associates' Gauntlet.

Inspection firewalls can provide address translation and hiding, virus scanning, Web site filtering, screening for key words (typically in e-mail), and context-sensitive security for complex applications.

2.5 Firewall Limitations

As pointed out in [9], "Information security professionals often find themselves working against misconception and popular opinions formed

from incomplete data. Some of these opinions spring more from hope than fact, such as the idea that internal network security can be solved simply by deploying a firewall". While it is true that firewalls play an important and central role in the maintenance of network security and any organisation that ignores them, does so at its peril, they are neither the panacea of every security aspect of a network, nor the sole sufficient bulwark against intrusion. Knowing what firewalls cannot do is as important as knowing what they can. The following are limitations one should be aware of.

- A firewall is by its nature perimeter defence, and not geared to combating the enemy within, and consequently no useful counter measure against a user who abuses authorised access to the domain.
- A firewall is no real defence against malicious code problems like viruses and Trojan horses, although some are capable of scanning the code for telltale signs.
- Configuring packet-filtering rules tends to be a complicated process in the course of which errors can easily occur, leading to holes in the defence. In addition, testing the configured rules tends to be a lengthy and difficult process due to the shortcomings of current testing tools. Normal packet-filtering routers cannot enforce some security policies simply because the necessary information is not available to them.

3 Additional Important Features

Firewalls are becoming more complex and sophisticated by the day, and thus more efficient at identifying attempted intrusions and logging them, and automatically notifying the right people. They provide multiple layers of protection and some cache data to improve performance, and support Virtual Private Network (VPNs), Web-based administration, authentication, etc. There is also a tendency to add non-security-related functions to the firewall such as built-in Web servers, FTP servers, and e-mail systems, and even proxy servers for streaming audio and video.

We agree with those who feel that some additions to firewalls make sense and are useful when they enhance security, while others do not make sense and may even be dangerous, especially over time, when they represent a decrease in security and an increase in vulnerability. For example, to add services that increase the administration load adds another potential avenue of attack.

Content Caching

While caching is not traditionally a function of firewalls, it is becoming an increasingly frequent and important feature. An increase in performance is achieved by caching the contents of an accessed location with the result that subsequent requests for access will lead to already cached contents being used, without it being necessary to access the location again (except when it is necessary to refresh).

Logging and Alerts

It is important for a firewall to log events, determine their legitimacy or otherwise, and notify the network administrator. It should be noted that it is essential to protect the integrity of the log, since unauthorised access to, and editing of, the log will, of course, neutralise its raison d'être. Whether the function of protecting the log is fulfilled by the firewall itself or not, is a matter of implementation.

Management

Management ranges from command line to sophisticated GUI-based and secured remote access. Security management and administration, particularly as it applies to different firewalls using different technologies and provided by different vendors, is a critical problem. As more and more security services are introduced and applied to different firewall components, properly configuring and maintaining the services consistently becomes increasingly difficult. An error by an administrator in maintaining a consistent configuration of security services can easily lead to security vulnerability. A firewall should thus provide a security management interface that enables it to be locally or remotely managed in a coherent and comprehensible fashion.

Virtual Private Networks (VPNs)

A VPN is an encrypted tunnel over the Internet or another untrusted network providing confidentiality and integrity of transmissions, and logically all hosts in a VPN are in one Intranet [16]. Some firewalls include VPN capabilities (reasonable extension) to secure networks, so that they can safely communicate in private over the public network. They achieve this by strong authentication and encryption of all traffic between them.

Adaptive Firewalls

The new trend is towards adaptive firewalls that tie filters, circuit gateways and proxies together in series [2]. This gives the firewall administrator greater control over the level of security used for different services or at different points in the use of those services. He may, for example, configure the firewall to give priority to speed of

transfer at the expense of security when this is appropriate. The firewall will then on such occasions reduce security to a lower level, thus allowing for greater speed of transfer, and return it to its original level on completion of the transfer.

Phoenix [15] states that Adaptive Firewall Technology provides fluid, self-adapting control of network access, a key to establishing an effective network security policy by examining every packet (and adapting rules “on-the-fly” based on information in the packet) passing through the network interface.

Quality of Service (QoS)

Some firewalls include QoS features that allow administrators to control what proportion of a given network connection is to be dedicated to a given service. There are those who feel that QoS should be handled by Internet routers, while others insist that this is a matter of access control, and thus should be included in the firewall.

Quoting [2]: “Moreover, some vendors, notably Check Point, have built their QoS engine using the same technology that is in their firewall. The philosophy here seems to be, access control is access control.”

Policy and Firewalls

There are two levels of network policy that directly influence the design, installation and use of a firewall system: higher-level policy and lower-level policy [9]. The former is the network service access policy, which lays down which services are to be accessible to whom, and how they are to be used. The latter is the firewall design policy, which describes how the firewall will implement the network service access policy, and precisely how it will take access decisions in accordance with it. Firewalls typically implement one of two design policies. The firewall may permit any service not expressly denied, or it may deny any service not expressly permitted.

Service access policy may, for example, decree that there shall be no access to a site from the Internet, but allow access from the site to the Internet. Alternatively, it may decree that access from the Internet shall be restricted to certain selected services in the site. The latter is the more widespread of the two.

Today’s business environments are, however, dynamic. Organisations are continually changing to adapt to new circumstances brought about by reorganisations, mergers, acquisitions, etc. Therefore there are regularly new policies to be enforced, and, to remain effective, today’s firewalls must be able to adapt to them.

4 Trends Threatening Firewalls – and Counter Trends

4.1 Trends Threatening Firewalls

Common network denial of service attacks include mail bombs, ping floods, and attacks using known software bugs, all of which are reported to be on the increase. This fact alone means that traditional firewalls performing packet analysis using rules and patterns are no longer adequate protection against network-based attacks, in addition to which, according to recent risk surveys [18, 17], more than half of all breaches today are perpetrated by some legitimate user already behind the firewall.

The traditional assumption that all inside the firewall are friendly and all outside it potentially hostile, is now becoming somewhat outdated. Internet connectivity has expanded, Extranets can allow outsiders access to areas protected by firewalls, and some machines require greater access to the outside than others, which often involves a change in the internal IP address. Another threat is the use of end-to-end encryption since the firewall is unable to peer through the encryption.

In the literature [3], some people have gone so far as to suggest that a more adaptive approach would be to drop firewalls altogether on the basis that they are obsolete, or that the use of cryptography obviates the need for them. Bellovin [3] disagrees with this view, and so do we.

4.2 Counter Trends and Arguments

Bellovin [3] argues that firewalls are still powerful protective mechanisms for the following reasons:

- Most security problems are due to buggy code – in 1998, 9 of 13 CERT advisories concerned buffer overflows and two of the rest were cryptographic bugs – and cannot be prevented by encryption or authentication. A firewall shields most such applications from hostile connections.
- Firewalls are also useful at protecting legacy systems. While applications that require strong authentication should provide their own, there are too many older protocols and implementations that do not. Saying that strong cryptography should be used is true but irrelevant. In the context of such applications, it is simply unavailable.
- More subtly, firewalls are a mechanism for policy control. That is, they permit a site’s administrator to set a policy on external

access. Just as file permissions enforce an internal security policy, a firewall can enforce an external security policy.

As already stated, we concur with the above, and cite the following additional arguments.

Cryptography notwithstanding, the use of firewalls is deeply entrenched in a number of organisations and is part and parcel of their security set up, and will continue to be so for some years yet. While it is true that cryptography is the heir apparent to the firewall, the number of as yet unresolved issues prevents the assembling of a comprehensive solution for securing distributed computing resources around Public Key Infrastructure (PKI) and encryption. In addition, the process of standardisation within the area of PKI is not proceeding particularly rapidly. Thus, even those organisations favouring technologies other than firewalls will just have to bite the bullet and live with them for the moment.

Another factor is the ongoing development of new features and services at present being continually added to firewalls. These reduce a number of the limitations listed above and increase the firewall's flexibility while allowing it to retain its original function unimpaired. Examples, to mention but a few, that illustrate this point are:

- The proposal of a distributed firewall [3], using IPSEC (IP Security), a policy language, and system management tools, that preserves central control of access policy while reducing or eliminating any dependency on topology.
- Phoenix's Adaptive Firewall Technology [15], as noted above, provides self-adapting control of network access, thus establishing an effective network security policy by examining every packet and adapting rules "on-the-fly" based on information in the packet passing through the network interface.
- FORE Systems' Firewall Switching Agent [7], in combination with Check Point's Firewall-1 [5], provides 20 Gbit/s of firewall switching bandwidth while delivering wire-speed routing, switching, and class-of-service delivery.
- OMG's [14] CORBA Firewall Security [12], which brings firewalls to distributed object technology and provides a standard approach by which a firewall identifies and controls the flow of IIOP (Internet Inter-ORB Protocol), which has become the de facto standard interoperability protocol for Internet, providing "out-of-the-box" interoperability with ORBs

(Object Request Brokers), thereby increasing the security of CORBA-based applications [1].

These trends in the development of firewalls make them important mechanisms to ease the transition to flexible and truly distributed security solutions, such as CORBA Security Services [13], thus sparing traditionally-minded network/firewall administrators much discomfort. After all, the laboratory test results described in "Super firewalls" [11] show that today's high-end firewalls are tougher, faster, and easier to use.

5 Conclusions

Notwithstanding the limitations of firewalls and the fact that they are neither the panacea of every security aspect of a network, nor the sole sufficient bulwark against network intrusion, and despite development trends that threaten them, they are still a powerful protective mechanism, and will continue to play an important and central role in the maintenance of network security for some years yet, and any organisation that ignores them does so at its peril.

They continue to change and develop, and new features are regularly added as the need arises. If developments follow the present trend, they will continue to combine configurable access control and authentication mechanisms with their traditional functions, thus providing more powerful and flexible protection for networks to make them secure.

References

- 1 Abie, H. CORBA Firewall Security: Increasing the Security of CORBA Applications. *Teletronikk*, 96 (2), 2000.
- 2 Avolio, F M. *Firewalls: Are We Asking Too Much?* <http://www.crossnodes.com/icsa/perimeter.html>.
- 3 Bellovin, S M. *Distributed Firewalls*. Draft Paper version, 7 July 1999.
- 4 Chapman, D B, Zwicky, E D. *Building Internet Firewalls*. Sebastopol, Calif., O'Reilly, 1995.
- 5 Check Point Firewall-1, version 3.0. White Paper, June 1997. *Checkpoint* (2000, June 19) [online] – URL: <http://www.checkpoint.com/products/whitepapers/wp30.pdf>.
- 6 Cheswick, W R, Bellovin, S M. *Firewalls and Internet Security, Repelling the Wily Hacker*. Reading, Mass., Addison-Wesley, 1994.

- 7 FORE Systems. *Firewall Switching Agent*. White Paper, October 1998.
- 8 Fulmer, C. *Firewall Product Overview*. (1999, December 30) [online] – URL: <http://www.waterw.com/~manowar/vendor.html>.
- 9 ICSA. ICSA Firewall Policy Guide V2.00, Security White Paper series. *ICSA* (1999, December 15) [online] – URL: <http://www.icsa.net/services/consortia/firewalls/fwpg.shtml>.
- 10 Moran, T. Fight Fire with Firewalls. *Microsoft Corporation* (1998, July 27) [online] – URL: http://msdn.microsoft.com/workshop/server/proxy/server_072798.asp.
- 11 Newman, D. *Super Firewalls*. Data Communications, Lab Tests. (1999, May 21) [online] – URL: <http://www.data.com/>.
- 12 OMG. Joint Revised Submission, CORBA/Firewall Security+Errata. OMG Document. *OMG* (1998, July 6) [online] – URL: <ftp://ftp.omg.org/pub/docs/orbos/98-07-03.pdf>.
- 13 OMG. The CORBA Security Service Specification (Rev. 1.2). *OMG* (1998, January 15) [online] – URL: <ftp://ftp.omg.org/pub/docs/ptc/98-01-02.pdf>.
- 14 OMG. *The Object Management Group*. (2000, June 20) [online] – URL: <http://www.omg.org/>.
- 15 Phonex Adaptive Firewall Technology, Multi-Layer Stateful Inspection. White Paper. *Progressive Systems, Inc.* (2000, June 20) [online] – URL: <http://www.progressive-systems.com>.
- 16 Schreiner, R. CORBA Firewalls White Papers. TechnoSec Ltd. (1998, December 15) [online] – URL: <http://www.technosec.com/whitepapers/corba/fw/main.html>
- 17 Thompson, M J. *Corporate Network Security* (1998, September 21) [online] – URL: <http://www.thestandard.com.au/metrics/display/0,1283,750,00.html>
- 18 Warroom Study. *Security in Cyberspace : Hearings before the Permanent Subcommittee on Investigations of the Committee on Governmental Affairs*. United States Senate, 104th Congress, 2nd Session, 1996. ISBN 0-16-053913-7. (<http://www.warroomresearch.com/researchcollabor/infosecuritysurvey.htm>)

CORBA Firewall Security: Increasing the Security of CORBA Applications^{*)}

HABTAMU ABIE



Habtamu Abie (42) is Research Scientist at the Norwegian Computing Centre. He has previously worked as Senior Engineer and Research Scientist at Telenor R&D (1997–1999) and as Scientific Associate and Scientific Fellow at CERN in Geneva, Switzerland (1989–1993). From 1994 to 1997 he also held a research position at ABB Corporate Research and worked as a Software Development Engineer at Nera AS and Alcatel Telecom Norway AS. He received his Engineering Diploma in Electrical Data Processing from Gjøvik Engineering College (1983), and his B.Sc. (1985) and M.Sc. degrees (1988) in Computer Science from the University of Oslo. He is also working towards his Ph.D. in Computer Science at the Department of Informatics, University of Oslo and UNIK. His past and present research interests encompass security for distributed systems, distributed object computing, architecture and methodology, formal methods and tools, data acquisition and control systems, hard real-time systems, and mobile and personal computing.

habtamu.abie@nr.no

Traditional network firewalls prevent unauthorised access and attacks by protecting the points of entry into the network. Currently, however, there is no standard mechanism by which a firewall identifies and controls the flow of Internet Inter-ORB Protocol (IIOP), that has become the de-facto standard interoperability protocol for Internet providing “out-of-the-box” interoperation with ORBs, and is based on vendor-neutral transport layer. The OMG’s intention in proposing its CORBA Firewall Security is to provide a standard approach to the control of IIOP traffic through network firewalls, allowing controlled outside access to CORBA objects, thus increasing their accessibility and security. This article describes and analyses the OMG’s CORBA Firewall Security, paying special attention to such issues as the specific problems associated with it, how current firewall techniques are used to control CORBA based communication, their potential limitations and how these might be overcome, and the various aspects of firewall traversal. In addition, a possible CORBA firewall application scenario is presented. Some CORBA Firewall compliant products are emerging on the market, and this current trend in the implementation of CORBA firewall products will also be described.

1 Introduction

Nowadays networks are subject to continual change and modification as they are adapted to changing circumstances and new situations brought about by reorganisations, acquisitions, outsourcing, mergers, joint ventures and strategic partnerships. In addition, networks are increasingly connected to the Internet. Due to these developments, the maintenance of security has become a far more complicated matter than hitherto. Common Object Request Broker Architecture (CORBA) has become the de-facto standard. Its extensive infrastructure supports all the features required by new business situations of the type mentioned above, and its increasing use in open systems necessitates the development of sophisticated security technologies at the interface between networks of different security domains such as between Intranet and Internet or Extranet. One of the best ways to ensure interface security is the use of a firewall.

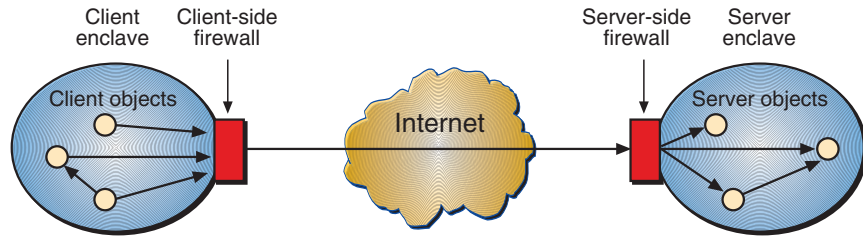
Conventional network firewalls (see [1] for an overview of firewall technologies) prevent unauthorised access and attacks by protecting the points of entry into the network. Currently, however, there is no standard mechanism by which a firewall identifies and controls the flow of IIOP, since IIOP has become the de-facto standard interoperability protocol for Internet providing “out-of-the-box” interoperation with Object Request Brokers (ORBs). The Object Manage-

ment Group (OMG) [10], a non-profit consortium with a current membership exceeding 840 organisations, whose purpose is to promote the theory and practice of object technology in distributed computing systems, is the body responsible for setting standards and specifications for CORBA Firewall Security. The purpose of the OMG’s CORBA Firewall Security is to provide a standard approach to control IIOP traffic through network firewalls, allowing outside access to CORBA objects, thereby increasing their security. This article discusses CORBA Firewall Security with the emphasis on such issues as the specific problems associated with it, how CORBA communication can easily and securely be handled by firewalls, how current firewall techniques are used to control CORBA based communications and their potential limitations, and how to overcome such potential limitations. It also describes various aspects of firewall traversal, IIOP/SSL, callbacks, desired proxy behaviour, chaining or pass-through mode for IIOP/SSL, and CORBA interworking through firewalls.

In addition, this article assesses the CORBA Firewall implementation technologies available on the market, such as WonderWall of IONA, Inprise’s Gateway, ObjectWall of Technosec, NAI’s ORB Gateway, etc. This discussion is essential to an understanding of current trends in the development of CORBA firewall products.

^{*)} The work was carried out while the author was a Research Scientist at Telenor R&D.

Figure 2-1 Object invocations through firewalls over the Internet



2 CORBA Firewall Security – an Overview

CORBA is widely available, provides an extensive and mature infrastructure, and plays a crucial role in integrating many existing enterprises, and has thus become today's most important system integration technology for distributed applications. The increasing use of CORBA in open systems requires sophisticated security technologies to isolate networks or sub-networks that are in different security domains. A security domain here means a network or sub-network under common administrative control, with a common security policy and security level. The domain boundary may be between Intranet and Internet or Extranet. The appropriate means to enforce security policies at the boundaries between security domains are firewalls.

The aim of OMG's CORBA firewall is to improve accessibility to CORBA application servers when there is a firewall separating a server from a client. It makes it easier to enable and control client-firewall-server communication under a broader range of circumstances with significantly reduced administrative burdens. Interoperable CORBA communication is via the General Inter-ORB Protocol (GIOP), which on the Internet is implemented by IIOP (a mapping of GIOP to TCP transport). Because firewalls control IP networking communication, and because ORBs communicate via IIOP, a large part of the activity of the CORBA Firewall is dedicated to the various operations involved in handling IIOP traffic through a firewall [8].

The main function of the CORBA Firewall is thus to recognise an IIOP message, process it, and then allow it to pass through providing fine-grained access control. CORBA firewall technology is applied to both inbound¹⁾ and outbound²⁾ protections. It processes requests from objects outside the firewall wishing to invoke operations on objects inside the firewall, and requests from client objects inside the firewall wishing to use CORBA-based applications outside the firewall on the Internet or Extranet.

As already pointed out, in a CORBA environment, firewalls protect objects from client objects in other networks or sub-networks. A firewall will either grant access from another network to a particular object or deny it. When it grants access, it can do so at different levels of granularity. For example, access to selected objects may be granted, but not to others, and, similarly, access to selected operations within a given object may be granted, but not to others. Firewalls have two distinct functions: inbound and outbound protections. Outbound protection involves allowing authorised client objects to initiate communication with objects outside the enclave (see below), while preventing those not authorised to do so from doing so. Inbound protection involves granting authorised outside client objects access to inside objects while denying unauthorised outside objects access. With no outbound protection, clients would be able to access any outside resource, and with no inbound protection the contents of an enclave would be totally unprotected against the (frequently malicious and destructive) machinations of the world at large.

Figure 2-1 shows the basic concept of CORBA object invocations through firewalls in an Internet environment. An enclave, as shown in the figure, is a group of CORBA objects protected by the common firewall which controls all network communication between them and the outside world. Enclaves can be nested, so that an enclave may contain other enclaves arranged hierarchically on the basis of different access policies. The figure shows a client object calling an operation on a server object in another enclave. The client object can communicate directly with all objects in its own enclave, but must communicate through the firewall with objects outside.

It should be noted that outbound protection involves granting or denying inside objects access to the outside world, but in most cases does not involve restricting which operations they are permitted to invoke on which outside

¹⁾ Inbound protection is used to control external access to internal resources.

²⁾ Outbound protection is used to limit the outside resources to be accessed from within the enclave.

objects. On the other hand, inbound protection involves not only denying or granting outside objects access to inside objects in the enclave, but also restricting which operations they are permitted to invoke on which inside objects.

3 Specific Problems Associated with CORBA Firewalls

Unlike traditional firewalls, CORBA Firewall Security addresses some of the specific problems associated with addressing, callbacks, encryption, cascading of firewalls, transparency, and integration into security systems [8, 11].

3.1 Addressing

Today's standard firewalls with static configurations are not well suited for dynamic CORBA applications because they work on the assumption that a client will always use a fixed, designated port on the basis that a given type of server always listens at that particular port. An HTTP server, for example, always listens at port 80. A CORBA server object, however, does not listen at a specific, designated port. While it is possible to bind a CORBA object to a specific port in the case of simple applications, this is not possible in most cases, because CORBA applications and ORBs generally use objects launched at arbitrarily selected ports.

Since it is not usually possible to predict which hosts and ports will be used for inter-enclave CORBA communication, it is difficult to configure firewalls in this situation. While the Interoperable Object Reference (IOR) provided by the server, containing host/port addressing information, is sufficient within the enclave, since no firewalls are involved, it does not help client objects from outside as they are unable to reach the server object directly. Instead of the actual address of the server, the client needs a proxified address, the address of the firewall. The firewall will process the request and forward it either to the server or to the next firewall. The address of the outbound firewall on the client side can be configured manually, but the IOR containing the address of the inbound firewall on the server side must be supplied by the server.

3.2 Callbacks

Traditionally in a client/server system, there is a very clear and sharp distinction between client and server. The server accepts connections from the client, but not vice versa.

This is not the case in a CORBA object system, a system of communication objects. In many cases it is desirable for a CORBA object server to contact a client object, for example, to facilitate asynchronous information flow. This is achieved by the client creating a callback object, the reference to which, an IOR containing the address of the callback object's inbound firewall, is passed by the client to the server. The server can then contact the callback object through its own outbound firewall and the callback object's inbound firewall.

It should be noted that this is not possible where the client-side ORBs have been downloaded as Java applets since these applets are not permitted to accept inbound connections or to create objects in the client space, and neither do they have any knowledge of the inbound firewall.

3.3 Encryption

Encryption technology is the ideal protection technology, especially for protecting communication over the Internet or other insecure networks against tapping and tampering. In CORBA communication, if the CORBA requests contain confidential information the use of encryption is the only way of protecting it.

Where firewalls are involved, there are three different patterns of encrypted connection: 1) between client and server, 2) between firewall and firewall, and 3) between client and firewall and firewall and server. This means that CORBA firewalls must be able to handle any combination of these three patterns.

3.4 Cascading of Firewalls

In many cases an invocation of an operation passes from client to server through more than one firewall. Figure 3-1 shows such a situation. In the figure we see how an invocation passes

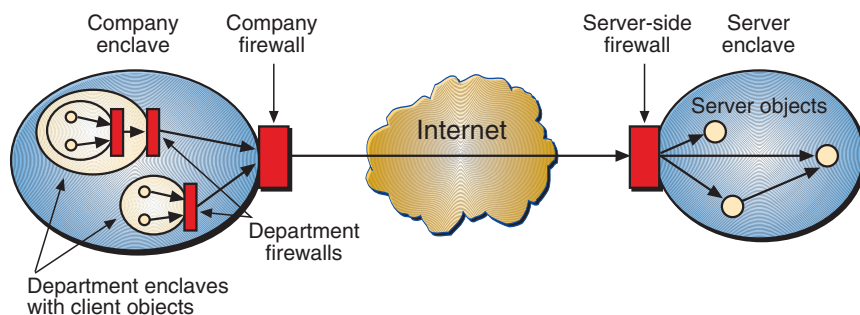


Figure 3-1 Cascading of firewalls in nested enclaves

from a client object in the enclave of a group, for example the Research Lab, over the Internet, to a server object in a server enclave. It passes from the client object to the firewall of the Research Lab, then to the firewall of the department, then to the firewall of the organisation, then over the Internet to the server side firewall, and finally to the server object. Such cascading presupposes the ability of invocations to pass through a series of firewalls even when the latter do not use the same technology.

3.5 Transparency

A CORBA firewall should be transparent to users (application programmers, administrators and end-users). Schreiner [11] has proposed a transparent firewall that allows a client object to use the normal IOR of the server object instead of a proxified address. The client attempts to connect to the server object as though no firewalls were present. The firewall intercepts the request and launches a proxy object, which relays the request to the server object.

In Schreiner's opinion this will make things simpler for programmers and administrators since it obviates the need for proxification and will only work if the incoming side uses an officially assigned IP address. It occurs to me, however, that the use of officially assigned IP addresses will limit the use of dynamic IP addresses. Will the use of interceptor and smart agent technologies be the trend in the development of the new generation of firewall technologies?

3.6 Interoperability

As described earlier, firewalls have two distinct duties, inbound protections that are used to control external access to internal resources, and outbound protections that are used to limit the outside resources that should be accessed from within the enclave. In order for an ORB to establish a connection to an object in another ORB, outbound and inbound firewalls that need to be traversed must be known.

Information about outbound firewalls is specified in the client side ORB, and information about inbound firewalls may also be specified in the case of Intranet and Extranet configurations.

In general, however, the client side knows nothing about the server side, so that the only interoperable way of giving the client access to the information about inbound firewalls is to have this information included in the IORs provided by the server.

3.7 Management

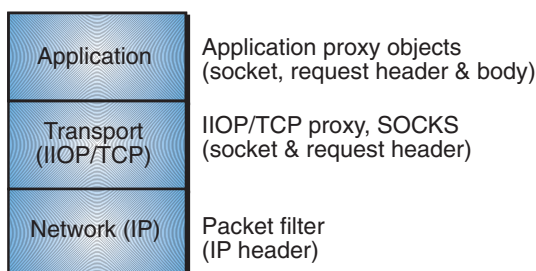
A CORBA firewall needs a secure interface through which administrators can configure it remotely and manage security policies. The functionality of this interface should permit the security policy to be configured dynamically. Preferably the firewall should be auto-configurable. Such functionalities are of paramount importance in the dynamic world of the Internet, since closing down a site is inconvenient and can be costly. The management functionality must also support key management interfaces if the firewall is to be involved in the encryption and decryption process as described above. CORBA firewall management should also be integrated into the organisation's overall security management systems, especially into CORBA Security Services [9] management.

4 The OMG Firewall Proposal

The Joint Revised Submission on CORBA Firewall Security (submitted in response to the OMG Firewall RFP (Request for Proposal)) [8] specifies the use of IIOP in network firewalls for controlling access to CORBA-based applications from Internet, Intranet or Extranet, and specifies and describes how inter-ORB interoperability through such firewalls can be achieved. According to the Joint Revised Submission, firewalls are broadly speaking of two kinds (in contrast to Figure 4-1), transport level and application level firewalls. The former permit or deny access to different resources using different application level protocols on the basis of addressing information in the headers of transport packets, and thus on the basis of where things are coming from or going to, and not what is being accessed. The latter, on the other hand, are in addition restricted to granting or denying access to particular application level protocols, such as IIOP or HTTP, and to those resources known to the application protocol. Consequently, they can grant or deny access on the basis of both addressing information and specific resources.

Figure 4-1 shows the operation of three different types of firewall. Each type operates on the basis of information available at a specific level, application level, transport level or network level. At the transport and network levels, IIOP can be handled like any other TCP/IP-based application protocol. This means that well-known firewall techniques like packet filters and transport-level proxies can be used.

Figure 4-1 3 Levels of different firewalls



On transmission, the message body at the application level is encoded in CDR (Common Data Representation). CDR then translates IDL (Interface Definition Language) data types into a byte-ordering independent octet string. For such an octet string to be decoded back to IDL data type, the interface definition of the object is needed. This information, however, is not as a rule at the firewall with the result that the firewall is unable to decode the message body. This means that an IIOP proxy cannot base filtering decisions on the request body [11].

Since the mechanisms involved in interaction with a firewall will vary with the type of firewall, it is necessary to have a precise definition of which types of firewall are supported in CORBA if ORB interoperability is to be achieved. In this connection, the current joint revised firewall submission proposes three types of firewall (see below) for use in different situations [8, 11], a TCP firewall for simple and static applications, a SOCKSv5 [16] proxy for client-side firewalls, and a GIOP application level proxy for enforcing fine-grained security policies (especially on the server-side).

4.1 TCP Firewalls

A TCP Firewall is a simple firewall that operates at the transport level, basing its access control decisions solely on information in TCP headers and IP addresses. When a connection request is received on a given port of the firewall, the firewall establishes a connection to a particular host and port. In the course of this process the firewall uses the combination of host address and port (<host, port>) to authenticate the client on the basis of the IP address alone. Having established the connection, the firewall will allow GIOP messages to pass through uninterrupted. In other words, ORB protocols are of no significance to a TCP firewall.

A simple form of ORB interoperability through TCP firewalls can be achieved without any modifications to CORBA. TCP firewalls must be statically configured with host/port address information in order to process access requests. The server can then be configured to replace its own host/port address with that of the firewall in its IOR for use outside the enclave. How this is implemented varies with the situation. One method is to proxyify the IOR manually. The client thus receives an IOR containing the address of the firewall rather than that of the server, and sends GIOP messages to the firewall (which forwards them to the server) under the impression that is where the server is.

Due to the tradition of TCP/IP using one port per service, it is common practice to identify a TCP service by the port number used for the server.

As a result, most of today's firewalls make low-level access control decisions on the basis of port used. Since there is no well-known "IIOP port", this practice does not facilitate ORB interoperability through TCP firewalls. As part of its proposed solutions, the OMG has defined a recommended "well-known IIOP port" and a "well-known IIOP/SSL port". This will enable client enclaves with TCP firewalls to permit access to IIOP servers by enabling access to this port through their firewalls.

The OMG's firewall RFP points out that, while these ports are not mandatory since IIOP servers can be set up to offer service through other ports, the ports serve as a basic guideline for server or TCP, SOCKS or GIOP proxy deployment, and make it possible for client enclaves to identify or filter immediately the traffic as IIOP without processing.

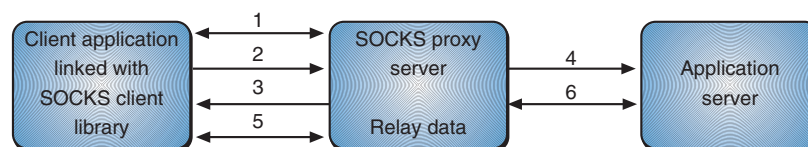
4.2 SOCKS Firewalls

The SOCKS [16] protocol is an open Internet standard (IETF RFC1928) which performs network proxying at the transport layer, mainly on the client-side. SOCKS creates a proxy which is transparent to either party, and which serves as a data channel between clients and servers based on TCP or UDP. In this case a client can have control over which server it wishes to connect to, in contrast to the situation when normal static mapping of TCP connections by a TCP proxy is used. A SOCKS firewall could then reasonably be referred to as a "dynamic TCP proxy".

SOCKS consists of a SOCKS proxy server on the firewall and a client-side library. In the client program the normal network calls of the socket-interface have to be replaced by the corresponding calls of this SOCKS library, and the process is called 'socksification'. The server is unchanged. The socksified client calls the corresponding functions of the SOCKS library. These SOCKS functions communicate transparently with the client and server over the SOCKS proxy server on the firewall.

Figure 4-2 shows a typical scenario of a client communicating with an application server through a SOCKS firewall. There are six phases to the communication process: In the first, the client authenticates itself to the SOCKS proxy server, and if successful, the process proceeds to

Figure 4-2 SOCKS proxy firewall traversal scenario



phase two, in which the client requests a connection to an application server. In the third phase, the SOCKS proxy grants the client's request and, in the fourth, creates a connection to the application server. In phases five and six client and server exchange data transparently over the SOCKS proxy. In practice the SOCKS proxy is relaying data between them. From the server's point of view, the SOCKS proxy server is the client, and from the client's point of view, it is the server. SOCKS also supports authenticated traversal of multiple SOCKS proxy servers.

SOCKS supports strong authentication of clients using GSS-API compliant security mechanisms such as User/Password, Kerberos, SESAME [15], or SSL [17]. The client and SOCKS server can enter into an authentication-method-specific sub-negotiation. If SSL is deployed, the client's certificates can be passed through the connections to allow the SOCKS server and the application server to authenticate the client directly. A SOCKS proxy can base transparent access control on both IP address information and user information stored in its server's and the client's configuration files.

From the perspective of SOCKS, IIOP is simply an example of a TCP-based application protocol. Thus, SOCKS is already capable of serving as a proxy mechanism for IIOP, enabling IIOP traffic to traverse firewalls. So, to handle the simple case of a CORBA client invoking an operation on a CORBA object across a firewall (a special case of Figure 4-2), the only requirements are that the CORBA client must be linked with a SOCKSified TCP library (that provides an identical API for sending TCP/UDP traffic and re-implements these functions to interact with a SOCKS firewall), and that the firewall must support SOCKS (which most existing firewalls do). An additional change is that the client host must be configured to route SOCKS requests to the appropriate proxy server. This is controlled by the client-side configuration file [8].

The information on the security provided by SOCKS firewalls, gleaned from recent research and experiment [13], is as follows: "As a server side firewall, it doesn't protect the server ORB from malicious traffic in the TCP octet stream, and doesn't allow a fine-grained enforcement of security. These can both be added to the SOCKS server, but today's SOCKS doesn't support it. On the client side an outbound firewall normally doesn't need this feature, so this would be a reasonable application of the SOCKS firewall."

4.3 GIOP Proxy Firewalls

The two firewall techniques described above work on the transport level with no knowledge

of the content of the TCP connection between the client and server (with the reservation that SOCKS can be extended to act as an application level firewall). Neither of them is able to check whether the content of the TCP stream is valid IIOP. Hence, neither of them provides any real defence against the malicious client or allows fine-grained enforcement of security policies.

A GIOP Proxy is an application level firewall that understands GIOP messages and the specific transport level Inter-ORB Protocol supported (i.e. a TCP GIOP Proxy understands IIOP messages). A GIOP Proxy Object is a fully-fledged CORBA Object which provides operations for firewall navigation. Note that this CORBA Object does not require a full ORB to be implemented in the firewall, as long as it behaves in a way that is consistent with the semantics of a CORBA Object, and understands the GIOP protocol and a transport mapping (such as IIOP) [8].

A GIOP Proxy firewall relays GIOP messages between clients and server objects. A GIOP message consists of a GIOP header, a message header and a message body. The message header, which is important for the GIOP proxy, contains the operation to be called, the object key to identify the target object, and the requesting client. The GIOP proxy makes access control decisions or fine-grained filtering decisions based on the information in the message header. For example, it could block requests to an object with a particular object key, or it could block requests for a particular operation on a specific object.

To establish a connection to a server, a client establishes a connection to the GIOP proxy.

If the GIOP Proxy is an outbound one, the ORB should be configured manually with the IOR of the proxy object. If the GIOP Proxy is an inbound one, the IOR provided by server to the client should contain the IOR of the proxy object on the firewall. The server places this information in a tagged component, which it then includes in the IOR it provides.

The GIOP proxy first authenticates the client, and then, if successful, connects to the server. Now the client sends a GIOP message to the proxy. The proxy examines this message to see whether it conforms to the security policy, and, if it does, sends it on to the server object. The proxy can, in addition, log the request and the communication if this is desired.

4.3.1 Connection Styles

There are two styles of connection through a GIOP Proxy: normal and passthrough.

- A **Normal** connection is one in which the client connects to the firewall, which in turn connects to the server. The client perceives the firewall as the server, and the server perceives the firewall as the client, neither being aware that it is connected to a mediator. It is the firewall's job to ensure that both the connections are correctly maintained, and to raise the right exception and inform the client in the event of a request being blocked or denied.

A GIOP proxy in this mode can examine the GIOP message and do fine-grained filtering as mentioned above. This gives rise to two security issues. Firstly, the client may not trust a GIOP proxy, and hence would not want the proxy to examine the traffic. Secondly, the client and server may be using a particular authentication and/or encryption mechanism that is unknown to the proxy. Both of these problems can be solved by the concept of a passthrough connection.

- A **Passthrough** connection is one in which the GIOP proxy does not terminate the connection at the GIOP level. The difference between a TCP proxy and a GIOP proxy operating in this mode is that the latter provides security enforcement at the CORBA object level rather than at the transport level. Like the former, it simply forwards GIOP messages without processing or examining them, or raising exceptions, once the connection has been established.

The passthrough mode is mainly used for encrypted communication.

An OMG compliant GIOP proxy has to support both normal and passthrough connections, but may reject the latter if the security policy dictates so.

4.3.2 Callbacks

The OMG firewall RFP also proposes two solutions for handling callbacks over a GIOP firewall: Bi-directional GIOP and GIOP Proxy object.

The proposed **Bi-directional GIOP** solution involves allowing a server to reuse a client's connection to send GIOP request messages. This is a very simple approach because if the client can contact the server, callbacks are possible without further measures on the client side and only works if the server object and the object making the callback to the client are on the same host.

The second proposed solution, more generic and secure than the first, involves the use of a **GIOP**

Proxy object at the client side too, making a callback similar to a normal request (but in the opposite direction) from the server to the client, allowing security enforcement at the client side.

4.3.3 IIOP/SSL

The standard protocol in use today for the encryption of requests sent over the Internet is SSL. SSL supports strong authentication based on asymmetric cryptography and standard X.509 certificates, while symmetric encryption is used on the IIOP message itself. ORB interoperability is frequently based on IIOP/SSL. Therefore, GIOP Proxy firewalls that forward IIOP requests must support the use of SSL as a transport mechanism for secure invocations, and the proxy administrator must have an interface to the proxy in order to be able to specify different levels of access control for different users, client objects and target objects. The proxy should include client and server side authentication for proxified connections, access to client and server X.509 certificates, and access control to proxies.

For proxy firewalls to support the use of SSL a number of requirements must be met. The first is that the certificate of the client must be accessible at each link in the proxy chain, and at the server. The second is that each (inbound) proxy in the chain must be able to impose its own access policy on the traffic passing through it. The third is that Certificate Authorities (CAs) must be known to both parties and trusted by them. In addition, either the certificate policies at both sides must be compatible, or it must be possible for the parties to negotiate a common certificate policy acceptable to both.

Proxies that support the use of SSL fall into two categories: trusted and untrusted.

An untrusted proxy can forward a message from a client by pass-through connection. This means that the proxy has no access to the encrypted message. While this ensures the integrity of the communication between client and server (which is necessary when one or both of the objects are sceptical of the proxy), it gives the proxy little scope for access control, since it is unable to apply its access control list fully.

A trusted proxy, on the other hand, can, in addition to forwarding messages using this same pass-through mechanism, also forward messages by establishing a separate connection to the server. This enables a trusted proxy to apply full access control, which means that when a trusted proxy is used, access control can be applied at the server, or at the proxy on a per operation basis.

5 CORBA Firewall Traversal

5.1 Firewall Tag Components

In a CORBA-based system, client objects connect to server objects by using an IOR. An IOR contains the address of the target object, such as a host/port pair. For an object to be able to pass through firewalls, the IOR must contain access information for these inbound firewalls. In a situation where there are multiple enclaves, i.e. cascaded firewalls, it may be necessary for the IOR to contain access information for all the firewalls to be traversed, although, according to the OMG's firewall RFP, it is strictly speaking only necessary for the IOR to contain access information for the first inbound firewall to be encountered by the object, i.e. the outermost inbound firewall.

The **TAG_FIREWALL_TRANS** component, contained in an IOR, designates a single point of entry into the network of the target object, and may appear several times, once, or not at all in an IOR. The presence of multiple firewall components in an IOR indicates that there are multiple points of entry into the target's network, through any one of which the client can reach the target object. The **TAG_FIREWALL_TRANS** component is encoded as an encapsulated sequence of **FirewallMechanism** structures. This sequence is important since it describes the chain of known inbound firewalls to be traversed, and therefore dictates the order in which they must be traversed. Each firewall mechanism in the sequence contains a **FirewallProfileId** and a sequence of firewall profile data. The latter contains information about the type of firewall supported. The OMG's firewall RFP currently defines three types of firewall, TCP, SOCKS and GIOP proxy.

5.2 Firewall Traversal Algorithm

CORBA Firewall Traversal enables CORBA objects to communicate with each other across different security domains and different combinations of different types of firewall, and hence achieves ORB interoperability through firewalls. Since each type of firewall has its own specific mechanisms for allowing connections through it, it is necessary to be acquainted with these mechanisms, and to know how firewalls of different types work in combination. The rules necessary for the traversal of any combination of the above mentioned types of firewall are laid down in the OMG's firewall RFP [8].

A client object will determine whether it needs to traverse a firewall in order to call a target object, and will do this by examining the IOR it is assumed to possess. If the client object is in the same domain as the target object it wishes to call, it can make a direct invocation. If the two objects are not in the same domain, this is not

possible. If the client object has in its configuration information about an outbound firewall to be traversed, it will send the request to that firewall. In the absence of such information it chooses the first **FirewallMechanism** in the **TAG_FIREWALL_TRANS** field of any firewall component in the IOR.

Having determined which is the first firewall to be traversed, the behaviour the client object exhibits will be dependent upon the type of firewall involved. For further information on the traversal algorithm for each of the three OMG compliant types of firewall, see [8].

5.3 HTTP Tunnelling

HTTP tunnelling is a mechanism for traversing client-side firewalls [4] by encapsulating IIOP packets in HTTP. In the Web environment firewalls are normally configured to pass HTTP traffic, and to block the passage of messages using other protocols, including IIOP. One way of allowing an IIOP message to pass through a firewall not configured to pass IIOP traffic, is to encapsulate, or tunnel, the IIOP message in HTTP. This makes possible communication between CORBA objects through the firewall without any reconfiguration of the firewall.

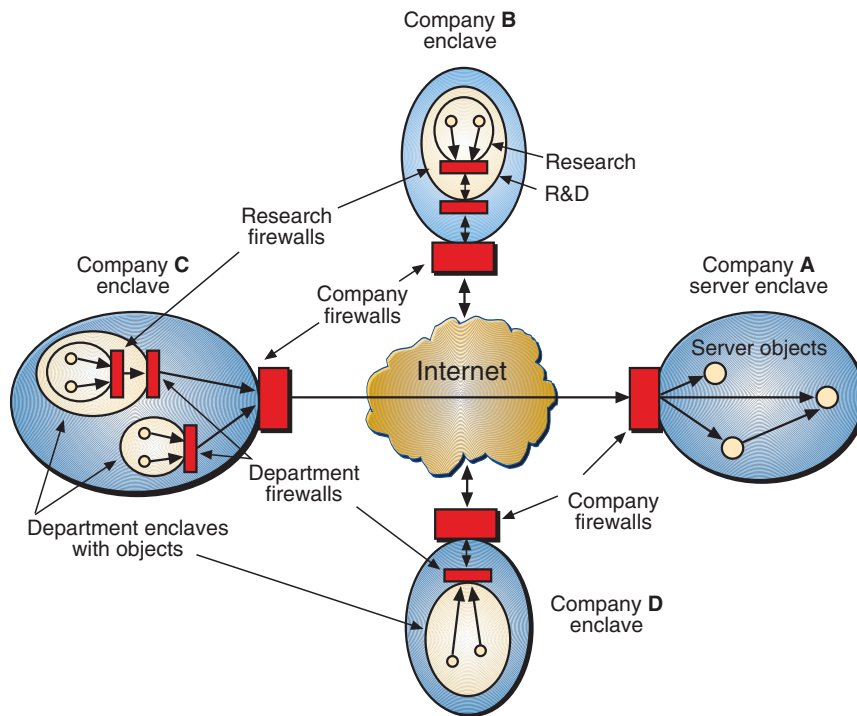
The client ORB encapsulates the IIOP request in HTTP (encodes it into HTTP), which allows it to pass through the HTTP proxy at the client side firewall. At the server side there is an HTTP-to-IIOP-gateway which decodes the request from HTTP to IIOP, and forwards it to the target object. When the target object replies, the gateway encodes it into HTTP and sends it back to the client [11].

There are, of course, additional processing overheads associated with this technique due to the encoding of the IIOP message into HTTP, and the decoding of the HTTP message into IIOP. An additional disadvantage is that it does not support callbacks.

6 CORBA Firewall Application

One of the benefits of CORBA is the ease with which it is able to integrate with legacy systems through object wrappers which define object-oriented interfaces for legacy applications to enable them to interoperate with other objects in a distributed object computing environment. This means that a CORBA firewall can enable legacy systems behind the firewall of an enterprise to interact safely with application objects running on systems outside the firewall. For example, users on the Web can access services of objects that are part of the internal system of an enterprise, and an external third party can access objects for the purpose of remotely monitoring and controlling their activity on behalf of the enterprise.

Figure 6-1 CORBA firewall applications over the Internet



One of the most important characteristics of CORBA is its wide availability and its provision of an extensive and mature infrastructure, which enables it to play the crucial role it does in the integration of distributed systems. CORBA firewalls can therefore fulfil the function of enforcing different security policies at the boundaries between integrated domains.

Figure 6-1 shows a situation in which four different organisations engaging in joint research activities communicate over the Internet using CORBA firewalls. Companies B, C and D use cascaded firewall access, which enables them to enforce different access policies for different departments, while company A uses single firewall access for its server objects, allowing it to enforce access policy at only one single point of entry for the entire company. This means that the research department, for example, of company B can be permitted to access the domain of the research department of company C, while other departments of the same company C cannot. Thus a relationship of limited trust can be established between partner companies. Those organisations using CORBA-based solutions in their information systems will benefit from the use of CORBA firewalls. These include, to mention but a few, healthcare, telecommunications, financial, manufacturing and government entities.

One may reasonably wonder why it should be necessary to have such defence in depth consisting of so many cascading internal firewalls when the company's network is already adequately protected by the outermost inbound firewall. The explanation is quite simple. In spite of the tradi-

tional assumption that all those behind the firewall are friendly, and all those outside it are at least potentially hostile, more than half of all breaches of security are perpetrated by legitimate users behind the firewall, according to recent risk surveys [19, 18].

7 Emerging CORBA Firewall Implementations

CORBA Firewall Security compliant products are emerging on the market and this section gives an overview of these. This discussion is essential to an understanding of current developments in CORBA Firewall Security trends.

7.1 WonderWall of IONA

WonderWall is an IIOP proxy with a bastion host as its basis, whose job is to decide which objects are to be permitted to communicate with which objects across which security domains [4].

It filters all messages arriving on the server's well-known port on the basis of request message header, and provides fine-grained control security. WonderWall supports the exchange of encrypted IIOP messages using Orbix transformer mechanism, and has a facility for logging messages, which allows the tracing of the history of suspicious message exchanges and provides a useful debugging and monitoring facility. A general feature of WonderWall's security practice is that all messages are blocked unless specifically allowed. It uses proxies, reinforced with special security features, to foil sophisticated attacks on the network. In addition WonderWall supports HTTP tunnelling of IIOP.

7.2 Visibroker Gatekeeper of Inprise

Traditionally, to safeguard network security, Java applets have not been permitted to communicate with objects other than those in their own enclave of origin. Gatekeeper is a gateway through which Java applets can communicate across Intranets or the Internet with CORBA server objects outside their own enclave of origin without compromising network security.

Gatekeeper uses an IIOP proxy server, and sends all traffic through a single port [2]. It supports SSL, which it uses to secure the Internet communication between a client object or a Java applet, and the firewall. Gatekeeper supports callbacks, HTTP tunnelling and GUI-based configuration, and provides location transparency.

7.3 Custom Firewall Solutions

These may take the form of TCP proxy firewalls; for example SOCKS or TIS Gauntlet's generic plug proxy. They have the advantage of using IIOP/SSL and providing application transparency, and the disadvantage of lacking application level filtering, having a low level of security and requiring firewall configuration [11].

TIS Plug-gw as a CORBA firewall

The plug-gw of the TIS Firewall Toolkit can be used as a very simple transport level IIOP firewall proxy. It needs proxified IORs, which can be hard if one does not have the ORB's source code and the environment is dynamic. It does not support secure callbacks [12].

SOCKS as a CORBA firewall

As pointed out earlier, SOCKSv5 [16] is one of the current OMG Firewall Joint Submission's suggested firewall mechanisms. In recent experiments SOCKS has been successfully socksified [13].

7.4 ObjectWall of TechnoSec

ObjectWall is a tool kit to be used in combination with other firewall tools, such as SOCKS, to construct secure firewalls for CORBA based applications. It supports transparent proxies at inbound and outbound sides, callbacks, central policy management (i.e. it only grants access to an object if the security policy defined by the administrator allows it) and firewalls with several levels of defence, and provides proxies and packet filters. It also supports the dynamic launching of proxies at runtime.

ObjectWall consists of two parts, Enforcement Modules and Policy Manager. The former are standard firewall tools, packet filter, NAT (Network Address Translation), and TCP level proxies with CORBA interface. The latter, the Policy Manager, has the task of checking whether the requests are in accordance with security policy [14].

7.5 ORB Gateway of NAI

The ORB Gateway functions like a firewall proxy in that it controls the access of CORBA operations to an enclave, but does so with a higher degree of granularity in its control over the CORBA access policy than is typical of a proxy. The access policy is expressed in DTEL++, like OO-DTE, and each request is categorised according to this policy. The domains to which an object is granted access are determined by the category in which the object has been placed on the basis of the degree of trustworthiness of the authentication mechanism. An unauthenticated object may be granted access to a limited domain with few privileges, while strongly authenticated objects will be allowed much freer access.

The ORB Gateway currently supports SSL as an authentication mechanism, but in the future, according to NAI, DCE, IPsec and Kerberos will be used.

The ORB Gateway is a single point of external access to the object services of an enclave which vets access requests on the basis of the nature of the request and of the attributes of the object from which the request comes, and implies control of traffic between ORBs of multiple enclaves rather than gateway or proxy control of traffic with the outside world [5].

7.6 OO-DTE of NAI

Object-Oriented Domain and Type Enforcement (OO-DTE) [6] provides scaleable, role based access control for CORBA systems, and is an object oriented extension of DTE, under which each resource is assigned a type and each process runs under a domain. What types of resource the processes of a given domain are permitted to read and write is specified by the DTE policy. Analogously in OO-DTE a CORBA operation is assigned a type, and the client object and server object (called processes by NAI) each runs in its own domain. The client object can invoke an operation if it is permitted to invoke an operation of that type, and similarly, the server object can implement that operation if it is permitted to implement an operation of that type.

As mentioned above, the language for expressing OO-DTE policy is called DTEL++.

There are two versions of OO-DTE already implemented, a DTE-kernel based system that provides non-bypassable access for CORBA systems, and an above-kernel OO-DTE system that performs access control in an Orbix filter without the non-bypassability feature. The two versions are, however, interoperable and use the same access control policy.

NAI state that they are currently working on the addition of SSL to above-kernel OO-DTE and the improvement of security policy administration, and that the policy distribution and the synchronisation tools presently under development will allow a centrally administered DTEL++ policy to be automatically distributed to CORBA systems within the enclave.

7.7 Gauntlet 5.0 of NAI

Gauntlet 5.0 for UNIX (Solaris & HP-UX) from NAI [7] is an IIOP proxy for Gauntlet. This proxy forwards messages sent between CORBA objects across Gauntlet firewalls, only after ascertaining that they meet CORBA's IIOP standard, thus ensuring that only valid IIOP messages travel across the firewall. This protects the network by ensuring that IIOP-designated ports are used solely for IIOP traffic, and by reducing the exposure of CORBA objects to invalid messages that could disable them.

Gauntlet 5.0 is compatible with IIOP 1.1. It accepts only well formed IIOP messages, which it passes unchanged to both clients and servers in accordance with the policy expressed in its administrative tools and netperm table, and, in the event of communication failure or a message being rejected, it generates an appropriate error message. Gauntlet operates transparently for both inbound and outbound connections and supports callback requests from servers to clients using bi-directional IIOP, and the "NORMAL" mode of IIOP proxy connection, but the current version does not support the "PASSTHRU" mode.

According to NAI information, Gauntlet has been tested for correct interoperability with client and server applications based on the ORBs Orbix v2.3, OrbixWeb, and VisiBroker v3.3 (C++ and Java).

7.8 MPOG of NAI

The Multi-Protocol Object Gateway (MPOG) [3] is an application proxy server that has been installed on Network Associates' Gauntlet firewall. It provides access control for operations on distributed objects, and its architecture allows reuse of policy data base and access control techniques for multiple distributed object technologies such as DCOM, CORBA and Java RMI. It has facilities for message routing based on the CORBA, Java RMI or DCOM interface requested.

For handling security protocols, MPOG currently has two protocol handlers, a non-secure handler and an SSL handler. The non-secure handler uses an unencrypted communication channel which can be used in an environment

where encryption of messaging and authentication of users may not be necessary. The SSL handler supports SSL for authentication between client and MPOG, and between MPOG and server, and for negotiating the supported cryptographic parameters between client and server, in situations where security is an important consideration.

As stated in [3], the MPOG access control mechanism separates the access control decision from the distributed object message handling, and supports OO-DTE domain derivation, trust management, and per-object and role-based access control.

8 Conclusions

Because of CORBA's popularity as a distributed object technology supporting cross-language and cross-platform interoperability and integration of enterprise-wide distributed applications, it is being increasingly used for the development of applications over the Internet, Intranet and Extranet in specialised market areas, such as healthcare, telecommunications, manufacturing and financial services.

The OMG recognises the need to safeguard the security of CORBA objects, and the fact that firewalls are still a powerful protective mechanism that will continue to play an important and central role in the maintenance of Internet security for some years yet. As a result they have specified CORBA Firewall Security with a view to bringing firewall technology to the world of distributed object computing technology, enabling firewalls to identify and control the flow of IIOP traffic, and thus enabling CORBA application objects behind a firewall to interact safely with objects outside the firewall. CORBA firewalls can also make fine-grained access decisions based on the attributes and operations of objects.

Firewalls continue to change and develop, and new features are regularly added as the need arises. If this development follows the present trend, CORBA firewalls will continue to combine configurable access control and authentication mechanisms with their already existing functions, thus providing more powerful and flexible protection mechanisms for the Internet, Intranet and Extranet. In the future it is probable that they may also handle moving agents, and may perhaps be able to provide migration transparency.

OMG CORBA firewall compliant products are emerging on the market, and will continue to do so.

References

- 1 Abie, H. An Overview of Firewall Technologies. *Teletronikk*, 96 (2), 2000.
- 2 Inprise's VisiBroker Gatekeeper. *Borland* (1999, December 15) [online] – URL: <http://www.borland.com/visibroker/gatekeeper/>
- 3 Lamperillo, G. *Architecture of Distributed Object Firewall Proxy*. NAI Labs, December 30, 1999.
- 4 IONA Technologies PLC. WonderWall Administrator's Guide. *IONA* (1997, December 10) [online] – URL: <http://www.iona.com/>
- 5 NAI, Network Associates Labs. An ORB Gateway. *NAI Labs* (2000, June 20) [online] – URL: <http://www.nai.com/>
- 6 NAI, Network Associates Labs. Object-Oriented Domain Type Enforcement (OODTE). *NAI Labs* (2000, June 19) [online] – URL: http://www.nai.com/nai_labs/asp_set/applied/arse_corba.asp
- 7 NAI, Network Associates Labs. Gauntlet 5.0 for Unix, an IIOP proxy for Gauntlet. *NAI Labs* (2000, June 19) [online] – URL: http://www.nai.com/asp_set/products/tns/gauntletunix_intro.asp
- 8 OMG. Joint Revised Submission, CORBA/Firewall Security+Errata. OMG Document. *OMG* (1998, July 6) [online] – URL: <http://ftp.omg.org/pub/docs/orbos/98-07-03.pdf>.
- 9 OMG. The CORBA Security Service Specification (Revision 1.2). *OMG* (1998, January 15) [online] – URL: <http://ftp.omg.org/pub/docs/ptc/98-01-02.pdf>.
- 10 OMG. The Object Management Group. *OMG* (2000, June 19) [online] – URL: <http://www.omg.org/>
- 11 Schreiner, R. CORBA Firewalls White Papers. *TechnoSec Ltd.* (1998, December 15) [online] – URL: <http://www.technosec.com/whitepapers/corba/fw/main.html>
- 12 Schreiner, R. TIS plug-gw as a CORBA firewall White Papers. *TechnoSec Ltd.* (1998, December 15) [online] – URL: http://www.technosec.com/whitepapers/corba/tis_plug_gw/cfwexp1.html
- 13 Schreiner, R. SOCKS as a CORBA firewall White Papers. *TechnoSec Ltd.* (1998, December 15) [online] – URL: http://www.technosec.com/whitepapers/corba/socks/socks_exp.html
- 14 Schreiner, R. ObjectWall : Integrated Protection of Dynamic CORBA Applications. *TechnoSec Ltd.* (1998, December 15) [online] – URL: <http://www.technosec.com/products/Objectwall.html>
- 15 SESAME, A Secure European System for Applications in a Multi-vendor Environment. *SESAME* (2000, June 19) [online] – URL: <http://www.esat.kuleuven.ac.be/cosic/sesame/>
- 16 SOCKS V5. *SOCKS* (2000, June 19) [online] – URL: <http://www.socks.nec.com/>
- 17 SSL3.0 Spec. *Netscape Communications* (2000, June 22) [online] – URL: <http://home.netscape.com/eng/ssl3/3-SPEC.html>
- 18 Thompson, M J. *Corporate Network Security* (1998, September 21) [online] – URL: <http://www.thestandard.com.au/metrics/display/0,1283,750,00.html>
- 19 Warroom Study. *Security in Cyberspace : Hearings before the Permanent Subcommittee on Investigations of the Committee on Governmental Affairs*. United States Senate, 104th Congress, 2nd Session, 1996. ISBN 0-16-053913-7. (<http://www.warroomresearch.com/researchcollabor/infosecuritysurvey.htm>)

Telenor's Risk Management Model

ERIK WISLØFF



Erik Wisløff (39) works in the field of risk management at Telenor R&D. Prior to this he worked in the security domain at AIRMATCOMNOR.

erik-dagfinn.wisloff@telenor.com

Introduction

Telenor defines risk as “the possibility of loss caused by threats or unwanted events” [1]. However, risk is a fuzzy, abstract and highly subjective concept that is inherently difficult to measure and sometimes next to impossible to come to terms with.

One reason for the difficulties in understanding risk is the fact that risk comes from a variety of sources: political, cultural, financial, legal, technical, environmental, competitive and personal framework. Another reason is that one risk may be a business requirement in one context, but a threat to the same business requirement in a different context.

Good risk management will improve the competitive edge of a business, product or service. The improved competitive edge comes from better project management, fewer nasty surprises, fewer accidents, lower costs while at the same time improving quality, meeting deadlines, meeting budgets, better customer communication, etc. Consultancy firms, hoping to land a “hefty contract with lots of money attached”, are quick to point to these benefits of risk management.

Small wonder, then, that some people view risk management more as magic than science, and are deeply suspicious of the claims that risk managers promote. Risk management is neither science nor magic. Risk management is a repeatable business process that systematically examines all the various products, processes, business surroundings and business objectives at all levels of the company. No more, no less.

What is Risk Management in Telenor?

Telenor defines risk management as the business process of managing risks by identifying, analysing and controlling costs related to risks. The rewards of an on-going risk management regime include

- Better knowledge of the risks one faces;
- Better understanding of the interaction between the business and its environment;
- Better understanding of the business' critical success factors.

This understanding could be used in many ways. For instance, it is difficult to cost justify loss reduction measures unless one knows the risks one faces. In addition, it is equally difficult to select reasonable risk financing strategies without knowing the risks one faces. Selecting a reasonable risk balance between different products, services or branches is impossible without a realistic understanding of the business environment. Discontinuing a product or service without regard to the business' critical success factors could turn into a nightmare. Thus, knowledge and understanding lead to an improved competitive edge and the benefits mentioned in the introduction. However, risk management must be an on-going business process in order to harvest the benefits.

Risk management is a flexible business process. You can use risk management to manage risks associated with a single product or service or to manage aggregate business risks.

The primary goal of risk management in Telenor is to minimise the aggregate risk cost, which is the sum of all individual risk costs in the business. The first challenge is to balance future costs caused by an unwanted incident against costs of attempting to prevent the incident from happening in the first place. The next challenge is to balance all the individual risks and costs in such a way that the grand total is minimised.

The optimal protection level point, as a risk manager sees it, is the lowest point in the curve labelled “total costs”. This is illustrated in Figure 1.

A secondary goal is quite simply to identify risks and then selectively “fixing any unacceptable problems”. This is typically a support activity in the business process of managing aggregate risk exposure. It is a necessary activity, but it quickly turns into an isolated series of suboptimisations. However, this is also an important step towards developing a mature risk management oriented business attitude.

Telenor's Risk Management Model

Telenor's risk management model illustrates the business process of managing risks. The risk management model is based on a refinement of the classical ‘analyse – act – evaluate’ cycle. As

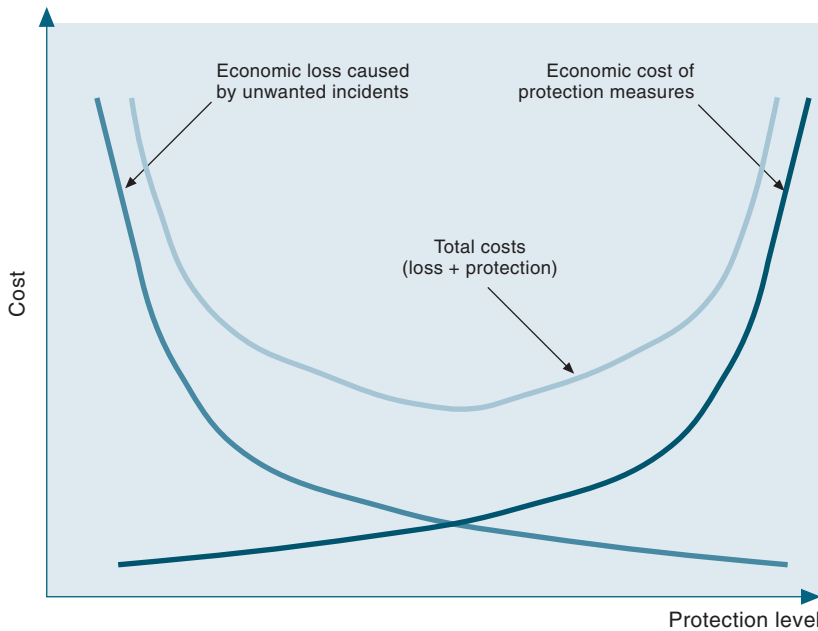
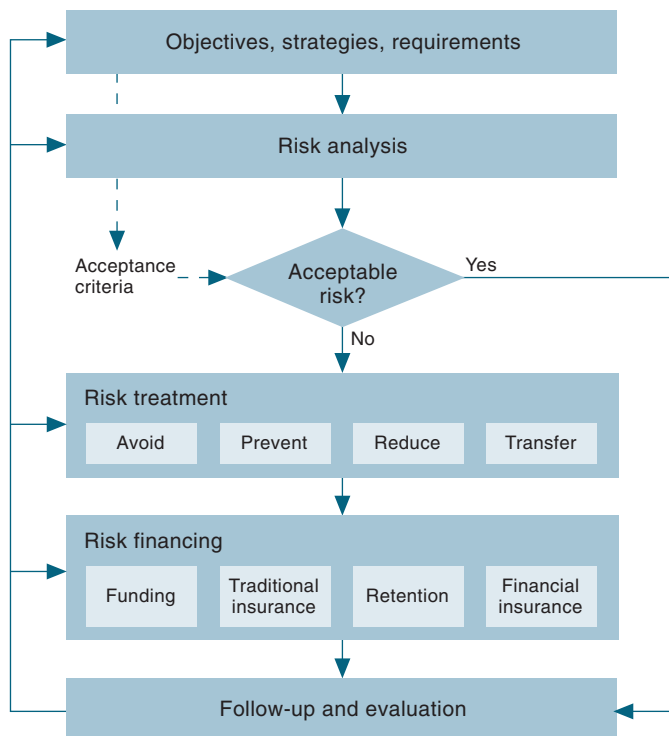


Figure 1 Minimising total costs

illustrated in Figure 2, the model consists of five major steps.

Risk management should not be isolated from the rest of the business. An obvious drawback of the model is that it does not visualise an important risk management practise: communication. All relevant risks must be communicated to the involved stakeholders, and stakeholders must communicate their interests to the risk manager.

Figure 2 Telenor's risk management model



Objectives, Strategies and Requirements

The first step in the risk management model is to define the business objectives, strategies and requirements.

Contrary to popular belief, the object of risk management is not to avoid risk at all cost. The object is to avoid or transfer unnecessary or unacceptable risks, while accepting selected risks. Taking calculated risks is an excellent business practise. Accepting risks blindly, or trying to remove all risk, is a very bad habit. Therefore, risk management cannot be effective without consciously deciding which level of risk is acceptable.

The applicable goals, strategies and business requirements define the background – be they wide-ranging business goals or narrowly defined product requirements. This background is a necessary foundation for the acceptance criteria.

The acceptance criteria come from the objectives, strategies and requirements, and they will be used to decide whether a risk should be accepted or not. Acceptance criteria can be described qualitatively (“we will not break any laws”) or quantitatively (“We will not accept more than n instances of abc ”). Acceptance criteria can also be used to develop risk indicators and decide the trigger levels for the risk indicators.

Understanding the objectives, strategies and requirements is vital for the risk management cycle. This understanding must be communicated to the next stage in the model, the risk analysis.

Risk Analysis

Risk analysis is the second step in the risk management model, and is an essential tool in risk management. The goal of a risk analysis is to identify and analyse risks, compare the risk exposure with the acceptance criteria and suggest loss reduction measures to the unacceptable risks. This gives the decision-maker the necessary background to make a decision on how he or she wants to treat risks. Acceptable risks should be monitored. Risk analysis is discussed in some detail in another article, and further reading is available at Telenor's Risk management homepage [A] or Telenor's TeleRisk homepage [B].

There are different ways to do a risk analysis – qualitative or quantitative, with formal methods or without formal methods – but the idea is to have a repeatable process that gives high quality answers at a reasonable cost.

Mitigation Strategies

The third step in the risk management model is to select a mitigation strategy. One can achieve risk reduction by applying a relevant mitigation strategy toward the unacceptable risks. Telenor's four mitigation strategies are

- Avoiding the risk altogether. If a risk is totally unacceptable and risk reduction is not possible by any other means, then it is necessary to avoid the risk, for instance by discontinuing a product or service.
- Preventing the risk from materialising. This mitigation strategy corresponds to reducing the frequency or likelihood of a risk materialising. An example of this strategy is the widespread use of virus control to prevent malicious software from harming a PC.
- Reducing the consequence if risk does materialise. Backup of servers is a classical example of consequence reduction. Even if malicious software does take out a server, up-to-date backups minimise harm.
- Transferring the risk to a third party. Insurance, contracts and disclaimers are traditional methods of transferring risk to a third party.

Usually a mix of the mitigation strategies will give the best result. The importance of the financial mitigation strategies is highlighted by the situation where it is impossible to avoid or prevent an unacceptable risk, and the decision-maker accepts the risk. At this point, the offensive risk manager should prepare fallback strategies and suggest financial mitigation measures.

Risk Financing

The fourth step, risk financing, is necessary whether the mitigation costs are payable up front or they turn up when a risk materialises. Risk financing in Telenor primarily falls into one of the four main strategies:

- Funding the mitigation is usually used if "something" is done to prevent, avoid or reduce a risk. The funding can be self-financing or financed by the customer. It is possible to build up reserves without having to pay taxes when the funding is unfunded or funded by way of a Captive.
- Traditional insurance is commonly used. An insurance company usually accepts the costs associated with a risk materialising. Of course, they expect payment for accepting this risk.
- Retention is the sum or cost the business has to carry by itself in order to get a lower insurance premium.

- Financial insurance is somewhat opposite to traditional insurance. With financial insurance, an insurance company charges a lump sum to cover aggregate risks. Sometimes the costs of the materialised risks are less than the lump sum. In this case, they return the difference minus a fee. However, should the costs be higher than the lump sum, then the insurance firm cannot reclaim the difference from Telenor.

Financing is necessary for risk reductions and acceptable risks. If the risk is too low, it is possible to remove risk reduction controls thus lowering the cost of financing risks.

As a rule it is necessary to keep the mitigation cost lower than the cost suffered if a risk materialises. The risk manager might know a lot about risk and risk mitigation, but the finance officer is the financial expert. Therefore, there are close links between risk management in Telenor, the finance officer and insurance in terms of Telenor Forsikring AS, the Captive of Telenor.

Monitor and Review

The final step is monitoring and reviewing. Risk management is a continuous process, an ongoing cyclical activity. The background, risks discovered during the risk analysis, risk mitigation and risk financing must be monitored continuously and reviewed regularly.

In addition to this, the risk manager must communicate regularly with all stakeholders and any other interested parties.

Supporting Framework

Risk management is heavily dependent on senior management involvement. The board of directors approved Telenor's original risk management policy 17 March 1995. This policy is replaced by the current policy, approved by the board of directors, dated 18 September 1999 [2].

An important benefit of good risk management is that one is on top of the situation. There is less uncertainty, one addresses only the unacceptable risks and there will be fewer crises and nasty surprises. However, even a good framework needs support. Telenor's risk management framework is supported by methods, guidelines, leaflets, word lists, template files, etc. The Risk Manager Forum plays an important role as an arena where the risk managers meet and discuss topics of interest.

Concluding Remarks

The perception of risk management appears to be changing globally, as does the concept and definition of "risk".

In the coming years it is likely that risk management will have two parallel perspectives. The first perspective is the defensive side, where one attempts to minimise aggregate risk costs. Today this is the primary focus of risk management in Telenor. The second perspective will be the offensive side, where risk managers actively support risky business ventures by identifying risks, opportunities and exits, preparing fallback strategies while aggressively supporting an increased risk exposure. The offensive perspective reflects the adage that it is not possible to make money without taking risks.

Risk is traditionally associated with a negative impact and unwanted incidents. Today, many risk management forums attempt to embrace the concept of risk as both a positive and negative factor. The argument for this varies, but more or less appears to follow these lines: risk is not necessarily the incident itself, rather: it is the consequence of an incident. Thus, if an incident occurs, the outcome is either a downside (loss) or an upside (gain). The opposition points out that while this may be correct, one is better off if this is called uncertainty management with two distinct and separate processes: the traditional risk management minimising the downside, and opportunity management maximising the upside.

Telenor has set out to take advantage of these emerging trends by setting up a project to promote a corporate business risk management methodology. This methodology will include risk management practices for “uncertainty” management, while taking advantage of both the defensive and offensive perspectives to risk management.

References

- 1 Faglig plattform for risikoanalyser i Telenor. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/dokumentene/24/10/telerisk20.doc
- 2 Konsernprosedyre for risikostyring i Telenor. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/dokumentene/67/49/konsernprosedyre_for_risikostyring.doc

Suggested Reading

- A *Telenor Infotorg* website for Risk management, select Risikostyring in the menu bar at http://134.47.108.143/staber_og_selskaper/sikkerhet/index.htm.
- B *Telenor Infotorg* website for TeleRisk (risk analysis), select TeleRisk in the menu bar at http://134.47.108.143/staber_og_selskaper/sikkerhet/index.htm.
- C *AS/NZS 4360*, which embraces the concept that risk is neutral (can have an upside and downside). *AS/NZS 4360* is a much cited risk management standard.
- D *BS7799*, which only views risk in the traditional sense (downside). *BS7799* is on track to becoming an ISO standard.
- E *CORAS*, an IST project proposal for a platform for risk analysis of security critical systems at <http://www.telenor.no/fou/prosjekter/coras/coras.htm>.
- F *Risk management* site at <http://www.riskmanagement.com.au>.

Risk Analysis in Telenor

ERIK WISLØFF



Erik Wisløff (39) works in the field of risk management at Telenor R&D. Prior to this he worked in the security domain at AIRMATCOMMONOR.

erik-dagfinn.wisloff@telenor.com

Risk analysis is an essential activity in the risk management business process. Risk analysis identifies threats, assesses how often they will materialise, assesses the impact the threat will have if it does materialise, presents the individual threats in an easily understood risk exposure statement and identifies possible loss reduction measures. The object of a risk analysis is to support a decision-process by explicitly stating the risk exposure.

Important deliverables of the risk analysis are the identified threats, the risk exposure of each threat and appropriate loss prevention measures.

Among the benefits of performing a risk analysis is a better understanding of the target of evaluation (TOE) and the possibility to cost justify loss protection measures. However, an on-going Risk Management regime must be in operation to harvest the full benefit of a risk analysis.

Telenor's Risk Analysis Model

Telenor's risk analysis model is described in Faglig plattform [1]. The model is in part inspired by Norsk Standard for risikoanalyse [2], and is compatible with [2] and the more recent

Australian standard [3]. The model is illustrated in Figure 1.

The risk analysis consists of several sequential steps, beginning with a description of the TOE. The result of the analysis is a written report that usually includes recommended loss reduction measures.

Describing the Target of Evaluation

The first step is to define the target of evaluation (TOE). The TOE can be anything – a business process, a physical object, a logical object or social interactions. A precise description usually includes

- The name of the TOE and the stakeholders;
- An unambiguous purpose for the analysis;
- A rough draft of the TOE and how the TOE interacts with the surroundings;
- All relevant facts, conditions or circumstances of importance to the analysis. This is information about the TOE or the TOE's surroundings, and may be technical, political, legal,

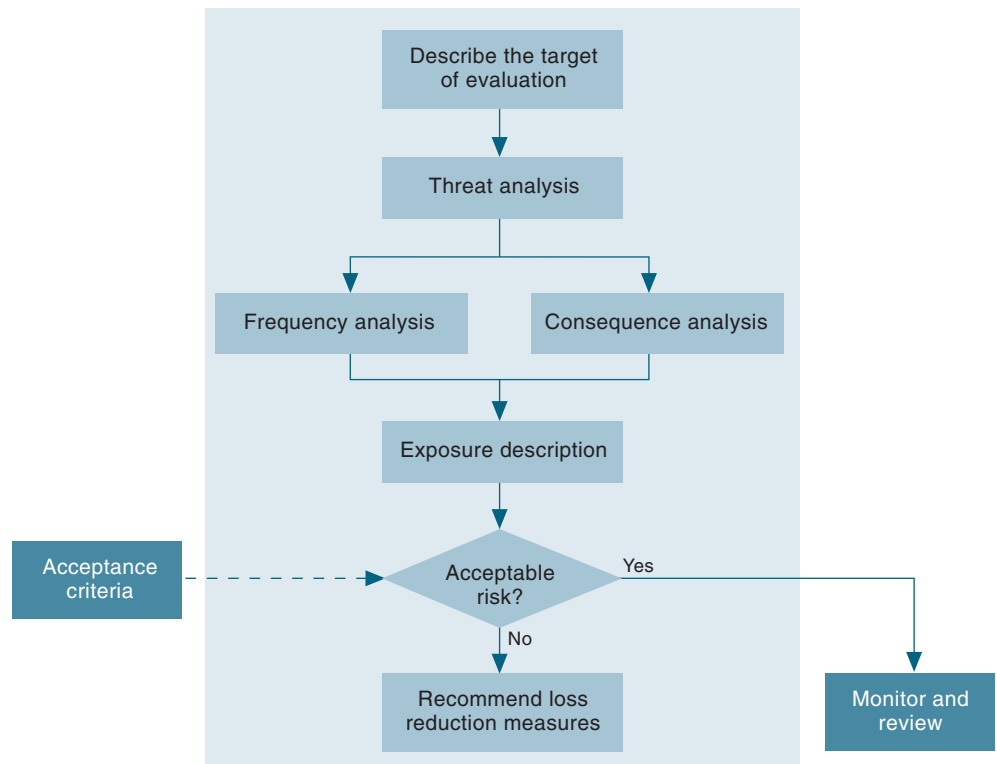


Figure 1 Telenor's risk analysis model

financial, social, environmental, humanitarian and/or organisational information;

- A clearly stated boundary between what is included and what is not included in the risk analysis.

The quality of the analysis is heavily influenced by this first step. If a critical part of the TOE is forgotten or deliberately omitted the analysis may be invalidated. Missing or inadequately described parts of the TOE usually produce confusion and arguments instead of a good understanding of the TOE. Arguments are also common when people do not accept the limits of the analysis.

This step is important, because it synchronises people's understanding of the TOE and lays down the ground rules for the threat identification phase.

Unfortunately, the layperson is seldom prepared to spend time on this step because "everyone knows what the TOE is". Maybe, but it is a rare occasion when everyone knows what the TOE actually is, understands what it really does, correctly describes the critical success factors, precisely describes the customer, etc.

Threat Analysis

The second step is the threat analysis, which Telenor splits into two half steps.

The first half step involves identifying the threats to the TOE. A threat is a present or future vulnerability, activity, accomplishment or event that could have a negative future impact on the TOE.

It is essential to use a structured approach in the threat identification phase. Significant threats are usually overlooked when the threat identification phase is unstructured, thus lowering the credibility of the analysis. An unstructured approach also leads to repeatedly returning to the threat identification process, thus increasing costs.

Telenor recommends Hazard and Operability studies (Hazop) [4] as a basis for threat identification. Hazop is a technique for structuring a brainstorming process, and is well suited when analysing complex objects. A skilfully executed Hazop will supply an exhaustive list of threats, what causes the threats to materialise and to a certain extent the consequences of the threats. However, Hazop is not recommended if the analyst does not have previous experience with this technique.

For the layperson, Telenor recommends using specially designed threat identification tech-

niques [5] even though this is a less structured approach to threat identification.

The next half step is an analysis of what causes a threat to occur. A separate brainstorming session may be necessary unless the causes were established during the threat identification phase. In this session one tries to answer the question "what can cause this threat to materialise".

The depth of the causal analysis is determined by the length of the causal chain. The direct cause of the threat is sometimes enough, but it may be necessary to establish a chain of causes before the causality is sufficiently examined.

Frequency and Consequence Analysis

The third and fourth step consists of analysing the frequencies and consequences related to each threat. The model shows these steps side by side, because it is a matter of personal preference and practicality whether one is completed before the other begins, or if they are analysed in parallel.

The frequency analysis examines each threat to determine how often the threat is likely to occur. The frequency analysis should be quantitative, but lack of time and hard data usually prevents this. The preferable alternative is to quantify a range of frequencies – for instance high, medium, low – and allocate each threat to one of the labelled ranges. If this is impossible, a qualitative description of the likelihood of each threat is called for.

The consequence analysis focuses on the damage a threat can set off, preferably expressed in economic terms. An indirect consequence occurs when the triggered threat sets off a chain of events before the consequence shows up, whereas a direct consequence is set off by the triggering threat. Sometimes it is necessary to look for both direct and indirect consequences before the total loss is determined. In any case, it is essential to determine the consequences because no loss prevention measure should cost more than the loss it prevents.

The consequence analysis should also outline the mechanisms or barriers that are supposed to prevent the damage. This knowledge is useful when selecting measures to minimise the consequences when a threat is set off.

Exposure Description

The previous steps analysed threats, frequencies and consequences. The next step is to present the threats in terms of risk exposure. The risk exposure is a description of the impact a materialised risk will have. There are three main points to consider:

Firstly, the aggregate risk exposure for a given time period should be presented when the previous analyses are quantitative. Aggregate risk exposure is defined as

$$\begin{aligned} \text{Aggregate risk exposure} \\ = \Sigma_T (\text{frequency} * \text{consequence}) \end{aligned}$$

where Σ_T is the summation of the T threats (in this article a threat includes vulnerabilities and unwanted events), frequency is the number of expected incident in a given time period and consequence is the economic consequence per incident.

Secondly, the exposure description should be as clear, concise and informative as possible. Therefore, while the aggregate risk exposure is precise, it is not informative in terms of individual threats. To this end, the individual risks can be presented in a table, a matrix or in a verbal narrative.

Thirdly, the exposure description must follow a format similar to the decision-maker's acceptance criteria. Acceptance criteria are ideally expressed by the decision-maker before the risk analysis starts, and they represent the level of risk the decision-maker can accept.

There is usually not enough data to support stating the aggregate risk as a single number. In addition, many of the finer points of the analysis are lost when the result is aggregated into a single number. Tabulated risks are effective only when the decision-maker is comfortable with this format, and verbal descriptions are often too verbose. Therefore, Telenor recommends using a risk matrix to present the acceptance criteria and the risk exposure; see Figure 2.

The two axes are frequency and consequence. The granularity of the axes must be suitable for the purpose of the analysis. Usually four or five suitably labelled intervals are sufficient. Each threat is then plotted according to the result of the frequency and consequence analysis.

It is vital to ensure that the verbal label one assigns to the intervals is acceptable to the reader. For instance, the label *Insignificant* is probably offensive when the consequence of a threat is life threatening injuries or permanent disability.

It is also necessary to explain what the labels mean. For instance, *Possible* might mean "between 1 and 10 occurrences per decade", and *Substantial* could be "between NOK 100,000 and NOK 250,000 per incident". The expected economic risk of a threat plotted in the cell possible/substantial in this example is between

Frequency

		Improbable	Possible	Usual	Common
Consequence	Insignificant				
	Substantial				
	Serious				
	Disastrous				

Figure 2 Risk matrix

NOK 250,000 and NOK 10,000 per year. The analyst will have to assign a qualitative meaning to the label if it is not possible to quantify the threats. For instance, *Disastrous* might mean "National media will have a feeding frenzy leading to significant loss of reputation to Telenor, discontinued service of the TOE and customers claiming substantial monetary compensation in addition to a significant number of customers fleeing from other Telenor products or services".

A decision-maker usually hesitates to implement remedial action unless the cost of loss prevention is less than or equal to the risk exposure. Therefore, the risk exposure should be stated in economic terms whenever possible – in addition to the risk matrix.

In a real world of risk analysis it is often necessary to assign both qualitative and quantitative definitions to the labels.

Deciding Whether a Risk is Acceptable

Deciding what is acceptable is the decision-maker's responsibility. As mentioned previously, the decision-maker should express the acceptance criteria before the analysis begins. When this is done, the risk analysis team can determine whether a risk is acceptable or not, without setting up a meeting with the decision-makers.

A recommended format for the acceptance criteria is the risk matrix. The unacceptable risk exposure is quite simply shaded, in Figure 3 the unacceptable risk exposure is shaded in a darker colour. The decision-maker's intentions are easily understood: Any threat plotted in a shaded area is unacceptable.

Recommending Loss Reduction Measures

The plotted acceptance criteria, which describe the level of risk a decision-maker accepts, and the risk exposure of each individual threat will reveal whether

		<i>Frequency</i>			
		Improbable	Possible	Usual	Common
<i>Consequence</i>	Insignificant				
	Substantial				
	Serious				
	Disastrous				

Figure 3 An example of a risk matrix with decision criteria

- The risk exposure is too high; in which case it is necessary to prevent loss. A loss reduction measure is an action taken or a physical safeguard installed before a loss occurs.
- The risk exposure is acceptable; in which case one should monitor the risk. No loss reduction measures should be recommended for an acceptable risk.
- The risk exposure is too low; in which case one should consider removing unnecessary loss prevention measures. The argument for this is to shift a resource from unnecessary measures to more productive risk reduction measures.

The recommended loss reduction measures should be grouped according to the risk mitigation strategies (avoid, prevent, reduce, transfer).

Telenor’s Corporate Framework for Security Risk Analysis

Telenor’s senior management has formally approved company-wide policies making risk analysis mandatory. So risk analysis is supported by senior management. The challenge to the practitioner is to balance the depth of analysis with the expected benefits of performing the analysis, and to avoid “paralysis by analysis”.

Telenor’s risk analysis framework is twofold. On the one hand it consists of corporate-wide policies, guidelines and tools. On the other hand there are the policies, guidelines and tools unique to each business area. Below, we outline the corporate risk analysis tools, biased towards the security domain.

Security Risk Analysis

The corporate policy for systems security [6] instructs all owners of information systems to perform a security categorisation of their system, and perform a risk analysis according to the security category. The three security categories, with corresponding depth of analysis, are

- A Especially security critical
 - a thorough risk analysis is required, possibly with a series of focused risk analysis of specific vulnerabilities.
- B Security critical
 - a Standard risk analysis is required.
- C Not security critical
 - a Minimal risk analysis is required.

This categorisation offers a win-win situation where managers can use the security category to filter and prioritise information, and the owner of an especially security critical system will get management’s attention while managers of other systems are relieved of unnecessarily detailed reporting. Thus the total effort a system owner must put into a security risk analysis is also minimised since the bulk of Telenor’s systems falls into category C or B.

In practice, the categorisation is a simple and straightforward task, requiring an interdisciplinary team which assesses the security profile of the system according to the assessment procedure. The procedure explores five domains; economy, marketplace, dependencies, the legal framework and the security profile of the system. These five domains are explored from both the customers’ and Telenor’s point of view through a number of questions. Each domain is assigned an integer value between 1 and 5, the values are added and, depending on the sum, the security category is either A, B or C.

The current security categorisation system was deployed in late spring 1999 and has given some surprises. The biggest surprise is that users generally perform a categorisation faster in real life than expected, while the quality of work meets or exceeds the expectations.

An interesting result is that the categorisation procedure which translates the security attributes confidentiality, integrity and availability into the five domains appears to be an eye-opener for users unfamiliar with security.

There are interesting variations in how the users perform the categorisation. Some users meticulously work through the full set of questions for each of the five domains, document all answers in detail and then work out a “correct” value for each domain and finally select the appropriate security category. Other users skip questions they do not like, slap an integer value they are happy with on each domain, and summarily select the appropriate category – and are done with the job. In one case users in different business areas have categorised almost identical sys-

tems as category B, with only a one point difference, using very different approaches. When questioned informally, the meticulous user said that they learned a lot about their system and its environment, and that the resulting report would be used as part of the system documentation. The other user reported that the classification procedure was acceptable, but that they did not learn very much from the exercise and that the main benefit was that their security officer was satisfied.

Risk Analysis During Product Development and Introduction

The product development and introduction process, or P3 as it is more commonly referred to in Telenor, calls for risk analysis both of the project risks and a security risk analysis of the future product or service.

The project risk analysis follows an intuitive process, and is quite straightforward.

On the other hand, the security risk analysis begins with a security categorisation which decides the depth of analysis. A review of the security risk analysis is required for each formal phase evaluation the project is subjected to.

Given enough time and that the product or service is categorised A or B, the security risk analysis is repeated one or more times before the product or service is launched.

Risk Analysis in Practice

All risk analyses are company confidential in Telenor. Confidentiality is necessary because one is very vulnerable after a risk analysis. Not only has the analysis uncovered weaknesses, it has also exposed and documented the mechanisms that can be used to exploit the weakness.

Instead of looking into specific risk analysis, this article will highlight the methodology behind Telenors Minimal Risk Analysis and some of the lessons learned.

The Minimal Risk Analysis

Users complained that the previous corporate risk analysis framework was too demanding, time consuming, inflexible and too focused on security risk analysis. Informal discussions with disapproving users pinpointed problems such as untrained analysts using semi-formal analysis methodologies, guidelines written by experts for expert use and too voluminous template files. In addition to this, the users had little or no chance of performing a risk analysis within real-world time constraints. In short; they felt the corporate risk analysis standard was impractical and theoretical.

The Minimal Risk Analysis (MRA) was developed as a response to these complaints. The primary objective was to propose a generic risk analysis framework that would give a methodical user a high success rate. The secondary objective was to have the Minimal Risk Analysis take less than one week to plan, execute and document. This contrasts the then current corporate methodology requiring a specialist analyst using perhaps 200 – 300 man-hours in a two to four month time scale.

The resulting design of the MRA is three-part:

- 1 A process guide; directing the user through a proven sequence of activities;
- 2 A template file; giving the user a skeleton report;
- 3 Threat assessment guidelines; assisting the user in uncovering threats.

The Minimal Risk Analysis is object oriented. The TOE is described as an object consisting of sub-components which interact with and are influenced by external objects. As a minimum, each object or sub-component will be described by its internal structure, its function and its interaction with other objects or sub-components. The following threat identification phase systematically explores each object to identify threats.

The Minimal Risk Analysis differs from the Standard Risk Analysis in three areas:

- Formal methods are not mandatory in a Minimal Risk Analysis. This means that risk analysis expertise is not required, though it is helpful. However, the lack of formal methods does not mean that the analysis is unstructured. The user has a guideline for the work process, a template file for the report and structured aids to the security threat identification.
- Acceptance criteria are not developed before the risk exposure is written. With a finalized risk exposure the decision-maker quickly sees which risks are acceptable or unacceptable.
- Causal analysis is omitted from the threat analysis phase since the causal analysis is not very useful for the acceptable threats. A causal analysis is occasionally necessary to identify cost-effective treatment strategies for unacceptable threats, but treating an unacceptable risk is often quite straightforward.

Practical use shows that a Minimal Risk Analysis of acceptable quality can be performed with-

in a one week time-frame, using approximately 20 – 30 man-hours in addition to the threat identification meeting.

As yet, no formal evaluation of the effectiveness of a Minimal Risk Analysis has been undertaken.

However, user feedback indicates that the Minimal Risk Analysis methodology is acceptable in the “real world”, while the quality of the risk analysis appears to be adequate. In addition it is becoming clear that the first few times a user is in charge of a Minimal Risk Analysis they feel they would benefit from some kind of expert support.

Typical Problems

Risk analysis should ideally be performed whenever something changes in the TOE environment. This can be personnel changes, reorganisations, changes in technology or market behaviour, new laws and so on. A more practical approach is to perform a new risk analysis whenever there are significant changes either in the TOE or in its environment. In real life, people are busy designing new products, solving problems, reporting to management and so on. So the risk analysis tends to be postponed, sometimes indefinitely because the TOE causes so many problems that there is barely time to fix one before another crops up ...

Another problem is finding the right people, the people who do know what the TOE is. They, or management, give other tasks higher priority. Ironically, occasionally these highly prioritised tasks are problems – making the best experts work reactively ...

Risk analysis is an interdisciplinary activity. Sadly, on occasions the quality of a risk analysis is below the quality the decision-maker requires because the interdisciplinary focus is lost in the “need for speed”.

Cause and Effect

Differentiating between cause, effect and consequence is not always easy.

Imagine this chain of events; you receive software with malicious code by e-mail, you open the e-mail and run the software. Unfortunately, the virus protection does not catch the malicious code, the malicious code destroys your files and you have no access to the backup files. Worse yet, your friendly IT-support person is unavailable due to acute sickness, you cannot reach the deadline for the bid you are working on and you lose an important contract. Your boss then promises you a “lengthy, uncomfortable meeting with much shouting, waving of arms and pointing of fingers”.

What is the threat in this scenario? What is the cause? What is the consequence? Is there an effect? Anything goes as long as a single threat has one or more causes and leads to one or more consequences. Moreover, one cause can trigger several different threats, while one consequence can be triggered independently by a number of threats.

Actually, it is often a matter of perspective. A good description of the TOE and an unambiguous purpose for the analysis will go a long way toward giving the answer. This is one of the reasons why a precise and correct description of the TOE is important.

Management Decisions

Management decisions should be based on explicitly expressed decision criteria. The management may be eager to express their decision criteria early on. This is a good sign – provided they actually manage to come up with decision criteria relatively easily.

However, when the business environment or TOE is very complex, the decision criteria are easier to express at a late stage of the analysis. This is also the case when it is apparent that the managerial logic is fuzzy or inconsistent. When the risk exposure is presented in the risk matrix, the risk analysis team should grab hold of the principal decision-maker, and have him/her point at the unacceptable cells. This usually works. The interesting thing is that the manager is usually grateful because he/she gets a clearer understanding of their “gut feeling”.

Risks and Problems

It is worth noting that a risk is not the same as a problem. A risk is something that has not happened, and it may not happen at all. A problem is something that has happened, that causes concern and which (normally) cannot be ignored. Sometimes a threat analysis meeting is bogged down by participants looking for solutions to something they see as a problem. On the other hand, decision-makers sometimes dismiss a risk because the risk is not yet a problem – but this is not necessarily the same as accepting the risk.

Work in Progress

Currently several key issues are being addressed;

- The Standard risk analysis package is being revised, to make it more user friendly.
- A methodology for performing quantitative risk analysis has been developed and has been tested in a live risk analysis in Telenor. The test was successful, and the methodology with supporting simulation tools are being im-

proved and prepared for practical use. This work is a joint effort between Telenor and Norconsult.

- New threat assessment guidelines are in development. Guidelines for project risk, delivery risk and business process risk will be deployed during 2000.
- Telenor R&D is the project manager of a consortium proposing to EU that the consortium, through the IST framework, "... provide an integrated methodology to aid the design of secure systems and thus establish trust and confidence in our products".
- A methodology for Business Risk Analysis is being developed and is expected to be deployed in 2001.

References

- 1 Faglig plattform for risikoanalyser i Telenor. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/dokumentene/24/10/telerisk20.doc
- 2 Norsk Standard. *Krav til risikoanalyser/ Requirements for risk analyses*. (NS5814.)
- 3 Australian standard. *Risk management*. (AUS/NZ 4360:1999.)
- 4 Risikoanalyseteknikker. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/dokumentene/24/44/risikoanalyseteknikker_v1-0.doc
- 5 Trusselidentifikasjon sikkerhetsrisiko. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/dokumentene/24/58/trusselid_sikkerhet_ver_10.doc
- 6 Konsernstandard for systemsikkerhet. *Telenor Infotorg*. (2000, June 26) [online] – URL: http://134.47.108.143/staber_og_selskaper/sikkerhet/handbok/retningslinjer/vedlb08.htm