15

Improving the Quality of Secure Distributed Systems

V. Zorkadis
Dept. of Computer Science, University of Ioannina
45110 Ioannina, Greece, zorkadis@cs.uoi.

Abstract

Security and performance along with further properties such as reliability characterize the quality of the services provided by distributed systems. However, security mechanisms may degrade the system performance due to the security-related processing and the exchange of additional data. In this paper, we focus on the tradeoff between security and performance requirements in distributed systems and propose and analyze concepts to optimize the performance behaviour of secure group communication. The performance degradation caused by the security functions can be reduced when employing appropriate mechanisms. The optimization concepts presented in this paper refer to confidentiality and integrity mechanisms and base on the exploitation of the stochastic nature of message arrival processes.

Keywords

Secure Distributed Systems; Quality of Service; Performance Evaluation; Secure Multicast Communication

1 INTRODUCTION

Security services such as data confidentiality, authentication of participants and prevention of unauthorized transmissions and receptions may be required by applications in a distributed systems environment (Li Gong and Nachum Schacham, 1995). They rely on security mechanisms like encryption and checksum exchange that may result in degradation of the system performance due to the related processing and transmission of the additional data. This paper deals only with mechanisms, that are employed to secure the communication among processes in a distributed system environment, and aims to reduce the performance degradation. The paper is organized as follows. This section refers to communication mechanisms in distributed systems, and to mechanisms required to secure the communication. In the second section we discuss optimization concepts, that offer the security functionality and efficiency required in a distributed system environment. The third section analyzes mathematically the performance benefit we can achieve by means of the concepts presented in the second section. Finally, this paper contains conclusions and references.

Distributed systems support various forms of communication such as RPC (Remote Procedure Call) and group communication. As an example of such a communication form

D. Gritzalis (ed.), *Reliability, Quality and Safety of Software-Intensive Systems*
© Springer Science+Business Media Dordrecht 1997

consider the reliable multicast or broadcast protocol for group communication in the Amoeba (Tanenbaum, 1994). Reliable multicast or broadcast means that when a user process sends a message to the group this message is correctly delivered to all members of the group, even though the transmission components may lose packets. The hardware/software configuration required for reliable group communication is shown in 'Figure 1' (Tanenbaum, 1994). One of the machines is elected as a sequencer, which has a special role. One of the possible methods that may be employed to achieve reliable group communication can be briefly described as follows. For a description in detail see Tanenbaum (1994) and the references therein. When an application, for instance in the machine C 'Figure 1', wants to send a message to the group, its kernel sends it first to the machine A which is elected as the sequencer. The sequencer, after it gets the message, it allocates for it the next available sequence number, puts the sequence number in the protocol header, and broadcasts or multicasts it to the group. By means of the sequence numbers and further parameters such as unique message identifiers can the kernels check whether they received all the messages sent to the applications the kernels act for.

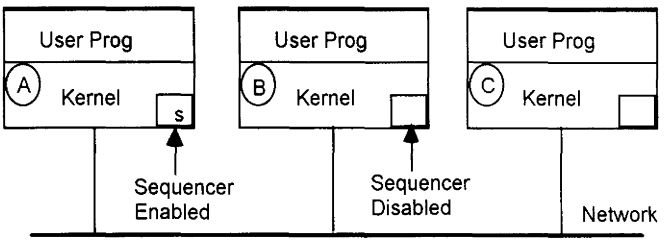

Figure 1 System structure for group communication in Amoeba (Tanenbaum, 1994).

The applications may require for their communication various security services that fall according to ISO/OSI 7498-2 in five classes: authentication, access control, confidentiality, integrity, and non-repudiation. Authentication services provide the corroboration to communicating entities, that their peer entities and/or the source of data received are as claimed. Access control protects against unauthorized use of system resources, e. g., files, processing nodes, communication channels, etc. Confidentiality services protect against unauthorized disclosure of application-related and/or traffic-related data. Data integrity protects against active threats like data modification. Often, in the bibliography authentication is used with the meaning of both ISO/OSI-related definitions of authentication and integrity. In the following discussion, we use authentication to express both ISO/OSI-oriented terms. Finally, non-repudiation provides the recipient and/or the sender of data the proof of the origin and/or delivery and the integrity of the data.

2 OPTIMIZATION CONCEPTS

Encipherment is the main security mechanism by means of which almost all security services may be implemented by its own such as confidentiality or in combination with further mechanisms such as notarization.

The optimization concepts that are proposed in this paper base on encipherment mechanisms or algorithms that use (pseudo) random bit strings such as the OFB operation

mode of DES, (DES, 1977), (Jueneman, 1983), (Diffie and Hellman, 1977), (Chaum and Evertse, 1986) or a corresponding mode of IDEA or a strong random bit string generator (Gait, 1977). This random bit string must not be dependent on the message to be enciphered, so that it may be generated before the message to be enciphered arrives. For instance, when we use the OFB-like mode (DES, 1977), (Jueneman, 1983) we can calculate a random bit string by means of an initial variable and the cryptographic key K. The message is then XOR-ed with this random bit string, which results to the message cipher. The receiver takes the message in clear from the cipher by XOR-ing with the same random bit string.

Table 1 Notation

Notation	*Explanation*
M_i	The ith message
R_i	The ith random bit string
C_i	The cipher of the ith message
I_i	The intitialization variable for the ith message
$G_K(.)$	The random bit string generator by using the key K
$\oplus$	The XOR-Operator

Formally, we may describe the encipherment and decipherment functions as follows (see Table 1 for the notation):

1 Encipherment: The sender generates asynchron the random bit strings R_i. Upon arrival of a message M_i and after the corresponding R_i was calculated, she/he ties both with the XOR-Operation. The result of the XOR-Operation is the cipher C_i, which is sent to the receiver.

$$G_K(I_i) = R_i$$
$$M_i \oplus R_i = C_i$$

2 Decipherment: The receiver generates asynchron the random bit strings R_i. Upon arrival of a cipher C_i and after the corresponding R_i was calculated, she/he ties both with the XOR-Operation. The result of the XOR-Operation is the message M_i.

$$G_K(I_i) = R_i$$
$$C_i \oplus R_i = M_i$$

The above cryptographic functions for encipherment and decipherment allow to exploit the stochastic nature of message arrival processes in distributed systems and therefore to pre-process the (pseudo) random bit strings required to encipher or decipher the communicating messages, i.e., to generate the random bit strings in advance before they are needed immediately for the encipherment or decipherment of messages. Confidentiality and data integrity mechanisms can be constructed that base on these cryptographic functions, so that the performance behaviour can be improved. In the following we describe such security mechanisms for confidentiality and data integrity.

Optimization concept for data confidentiality mechanisms

Group communication protocols (Chang and Maxemchuk, 1984, Garcia-Molina, 1982, Tseung, 1989) such as the multicast or broadcast communication in AMOEBA (Tanenbaum, 1994) or communication that bases on the connection-oriented transport protocol (ISO 8073) are highly reliable. In a highly reliable environment the problem of resynchronization is eliminated, since data loss is handled by the underlying communication mechanisms. The communication mechanisms deliver the messages correctly to all members of the group, even though the transmission components may lose packets (Tanenbaum, 1994). To use a stream cipher with the precomputing property, the communicating peers or the members of a group have to agree at connection set up or the registration of a group communication upon which strong (pseudo) random generator algorithm to use and how to calculate and how often to change the initialization variables. In the case of the reliable group communication in AMOEBA the elected sequencer adds to each message a sequence number and maintains a history buffer with a number of messages sent most recently and their corresponding sequence numbers. This sequence number N_i, which is unique for each message, along with the secret key K may be exploited by the security mechanism to compute the initialization variable I_i and the random bit string R_i required for the encipherment of the message M_i and the decipherment of the corresponding cipher C_i. For instance, it could be $I_i = N_i$ when using IDEA in an operation mode like the OFB of the DES to compute the R_i under the secret key K. The sender proceeds as described above to encipher the message, so does the receiver as well.

Concept for data integrity and data origin authentication

We refer to data integrity mechanisms that employ hash functions to compute a message digest (Preenel, 1994, Rivest, 1991), which is signed by the sender to protect against relevant attacks (Li Gong and Nachum Schacham, 1995). The computation of the signature is time consuming compairing with the computation of the hash value, which can be performed very fast. In the case of the reliable group communication in AMOEBA, the sender can compute in advance (precompute) the signature D_i of the sequence number S_i or the unique message identifier concatenated with a random number R_i or a timestamp T_i, that can be valid for the whole duration of the session if the sequence numbers are monotonically increased. When the application process passes the message M_i to be send to the group communication mechanism, the comminication mechanism can compute, by means of a hash function, $H()$, a digest, H_i, of the message concatenated by the sequence number and the timestamp or the random number.

1 The sender precomputes: $D_i = E_{K_S}(S_i \otimes T_i)$ or $D_i = E_{K_S}(S_i \otimes R_i)$, where $E_{K_S}()$ the encipherment function with the secret key K_S of the sender is.
2 When the message to be sent is ready, the sender computes: $H_i = H(T_i \otimes M_i \otimes S_i)$ or $H_i = H(R_i \otimes M_i \otimes S_i)$ and $U_i = D_i \oplus H_i$ and sends to the receiver, the message M_i, maybe enciphered, appended by U_i and by the T_i or the R_i.
3 The receiver upon receipt of the message M_i', the U_i' and the T_i' or R_i', computes the $H_i' = H(T_i' \otimes M_i' \otimes S_i)$, derives $D_i' = U' \oplus H_i'$ and deciphers it with the public key of the sender K_P, $D_{K_P}(D_i') = S_i' \otimes T_i'$ or $D_{K_P}(D_i') = S_i' \otimes R_i'$. If the $S_i' = S_i$ and $T_i' = T_i$

or $R_i' = R_i$ the receiver accepts the message as authentic, since only the member who possesses the secret key K_S can compute the D_i.

Furthermore, the digest of the message is tied up with a sequence number and a timstamp or a random number, that are unique for each message, so replacement of the message by a bogus one is not possible without detection. The use of random numbers does not help detect replay attacks in a future session, since an attacker could store D_i und try to use them in future group sessions by impersonating himself as the group member who possesses the secret key with which the D_i was computed. Therefore, timestamps should be preferred instead of random numbers, unless the use of random numbers is combined with challenge/response mechanisms.

In the next section we analyze the performance benefit we can achieve by using the above optimization concepts.

3 PERFORMANCE ANALYSIS

Model

We model a cryptographic facility as a queueing system and messages to be enciphered as jobs. The above mentioned precomputing possibility is modeled by a preservice property we introduce to a classical queueing system (Kleinrock, 1975). Therefore, we consider an M/G/1 queueing system that has the additional capability to carry out work for a job prior to its arrival, when the queue would be otherwise idle. When the server completes the work for a job, that can be accomplished prior to its arrival, the server enters in an idle period. We will remove this restriction for simulations.

The system with the precomputing property acts exactly as a classical queueing system as long as there are jobs present in the system. At the instant of time the last job in the system leaves, the system begins to accomplish work for the job next to arrive. When the whole work of a job is accomplished prior to its arrival, when regarding constant service times, the system becomes idle. We further partially assume that the time for the XOR-Operation can be neglected, compaired to the time required for the generation of the random bit sequences or the computation of the hash value. This restriction is removed, when we assume that only a part but not the whole of the work required by a job can be accomplished prior to its arrival.

According to the description above, a job can find the system upon its arrival in two possible states:

1 With probability p the server is busy with jobs arrived prior to the new job, which has to wait until all of them arrived earlier are served. In that case the server needs the conventional time, i.e., the same as without precomputing, to completely serve the new arrived job.
2 With probability (1-p) the server accomplished either all or part of the work required by the new arrived job. Thus, its new service time depends on the elapsed time since the last message left the system. It ranges from 0 to x depending upon the duration of this elapsed time. These elapsed times are exponentially distributed due to the memoryless property of the arrival process, since we assumed a Poisson arrival process and therefore an exponential distribution for the interarrival times.

The notation is as follows. The arrival process is Poisson with rate λ, Y is the random variable describing the interarrival times, X the random variable representing the service times of the conventional system and X' is the random variable representing the service time in the

system with the precomputing property, which means the service time a job is aware of since its arrival. The random variables take on values denoted by y, x and x', respectively.

Analysis

Let $f_Y(y)$, $f_X(x)$ and $f_{X'}(x')$ be the density functions (pdf) of the random variables describing the interarrival times, the service time of the conventional system and the service time of the system with the precomputing property and their Laplace transforms be denoted by $F_Y^*(s)$, $F_X^*(s)$ and $F_{X'}^*(s)$, respectively, where $F_Z^*(s) = \int_0^\infty e^{-sx} f_Z(z)dz$. According to the observations above a job finds the system upon its arrival in one of two possible states:

1 With probability p the server is busy with jobs arrived prior to the observed job. The pdf of the service time conditioned on this case is denoted by $f_{X_1'}(x_1')$ and equals the conventional $f_X(x)$, since there was no time for preprocessing.
2 With probability $(1-p)$ the server is either busy with preserving the observed job or idle if the server has completed the preservable work of the observed job. We denote the random variable describing the service time conditioned on this case by X_2' and its corresponding pdf by $f_{X_2'}(x_2')$.

Therefore, the $f_{X'}(x')$ is given by the following equation:

$$f_{X'}(x') = pf_{X_1'}(x') + (1-p)f_{X_2'}(x')$$
$$= pf_X(x') + (1-p)f_{X_2'}(x').$$

To derive the $f_{X_2'}(x_2')$ we observe, first, that with a probability p_1 is valid $x_{np} \le x_2' \le x$, if the server is busy with preserving the obeserved job, because of $y \le x$, where y is the time from beginning of the precomputing until the job arrives. We denote by x_{np} the part of x, that cannot be accomplished prior to the arrival of a job. Secondly, with a probability p_2 is $x_2' = x_{np}$, i.e., the whole preservable work is completed before the arrival of the observed job, if the server is idle, because of $y > x$. In the following we will assume that $x_{np} = 0$. Later, in this section we will analyze a system with $x_{np} \neq 0$.

According to the description above, the server begins with the precomputing of the next job to arrive, at the instant of time the last job leaves the system and continues until either, if considering constant service times for the conventional system, the whole work is completed or the next job arrives. The time that elapses from beginning of the preprocessing until the next job arrives is exponentially distributed with parameter λ, since the arrival process is Poisson.

The $x_2' = 0$ occurs with the probability $p_2 = \int_0^\infty \int_0^\infty f_Y(x+t) f_X(x)dtdx$ and the pdf describing $x_2' = 0$ is given by $u(x_2'$, where $u(z-t)$ is the unit impulse function representing an unit impulse at $z = t$.

The above observations result the following formula:

$$f_{X_2'}(x_2') = \int_0^\infty f_Y(x) f_{X'}(x + x_2')dx + \left(\int_0^\infty \int_0^\infty f_Y(x+t) f_{X'}(x) dt dx \right) u(x_2').$$

Let us now come back to the $f_{X'}(x')$ and deal with the probability p. From the state description at the beginning of this section it is obvious, p is the time percentage the system serves a job already arrived at the system, i.e., without the precomputed part, which equals λ times the average service time in the system with the precomputing property denoted by $\overline{x'}$

Thus, we calculate p as follows:

$$p = \lambda \overline{x'} = \lambda \left[p\bar{x} + (1-p) \int_0^\infty x' f_{X_2'}(x') dx' \right.,$$

where we denote the mean service time of the conventional system by $\bar{x}$.

We obtain the second moment of the service time in the system with the precomputing property by applying one of the following formulas (Kleinrock, 1975, Papoulis, 1991):

$$\overline{x'^2} = \int_0^\infty x'^2 f_{X'}(x') dx' \quad \text{or} \quad \left. \frac{d^{(2)} F_{X'}^*(s)}{ds^2} \right|_{s=0} = (-1)^2 \overline{x'^2}.$$

By the calculation of the waiting time experienced by a job in the system with the precomputing property we must pay attention to the fact that when a new job arrives at the system the number of jobs waiting in the queue and their service times are independently distributed. The service times are then equal to the conventional service time x for all waiting jobs as without precomputing, since there is no time for precomputing between subsequent jobs. Hence, this leads to the following, when taking expectations:

$$\begin{aligned} E\{W_i'\} &= E\{R_i'\} + E\left\{ \sum_{j=i-N_i'}^{i-1} E\{X_j | N_i' \neq 0\} \right\} \\ &= E\{R_i'\} + \bar{x} E\{N_i'\}. \end{aligned}$$

We denote as in (Bertsekas, 1987)

W_i': The waiting time in queue of the ith message.

R_i': The residual service time seen by the ith message.

X_j: The service time of the jth message.

N_i': The number of messages found waiting in queue by the ith message.

All long-term average quantities are viewed as limits when time or customer index increase to infinity (Bertsekas, 1987). Thus, W', R' and N' are limits (as $i \to \infty$) of the average waiting time, residual time, and number of jobs found in queue, respectively, corresponding to the ith message. We assume that these limits exist (Bertsekas, 1987), and this is true of almost all systems of interest provided the utilization ρ of the conventional system is less than 1.

Taking the limit as $i \to \infty$ we obtain

$$W' = R' + \bar{x}N' = R' + \bar{x}\lambda W' = R' + \rho W'$$

This last equation leads to the following: $W' = R'/(1-\rho)$.

Note that the average waiting time in the system with the precomputing property is expressed in terms of the mean residual service time in the system with the precomputing property and of the utilization ρ of the conventional system. The mean residual time R' can be obtained by the formula $2R' = \lambda\overline{x'^2}$.

Now, we can proceed with the analysis so as we do when analyzing M/G/1 queueing systems. The average delay T′ of the system with the precomputing property and T of the conventional system are given by the following equations we obtain by means of the formula for the waiting time of the system with the precomputing property and by means of the Pollaczek-Khinchin mean-value formula (Kleinrock, 1975) for the conventional system, respectively:

$$T' = \overline{x'} + \frac{\lambda\,\overline{x'^2}}{2(1-\rho)},\quad T = \bar{x} + \frac{\lambda\overline{x^2}}{2(1-\rho)}\ .$$

Note that in both formulas ρ is the utilization of the conventional system. In the following we analyze constant service times for the conventional system with $x_{np} \neq 0$.

The M/D/1-system with the precomputing property, $x_{np} \neq 0$

We denote the constant service time for the conventional system by $\bar{x}$ and its part that corresponds to the work of a job that can be accomplished in advance by x_p. Proceeding as above we can find the pdf $f_{X'}(x')$ of X' for $x_{np} \le x' \le \bar{x}$.

$$f_{X'}(x') = pu(x' - \bar{x}) + (1-p)\left[\lambda e^{-\lambda(\bar{x}-x')} + e^{-\lambda x_p}u(x' - x_{np})\right].$$

We can find the probability p by using the equality $p = \lambda\,\overline{x'}$, where $\overline{x'} = E\{X'$, since p is the time ratio in percent the system serves a job already arrived. Therefore, the following equation holds:

$$p = \lambda\int_{x_{np}}^{\bar{x}} x' f_{X'}(x')dx'.$$

Now, we can calculate the first and the second moments of the service time in the system with the precomputing property $\overline{x'}$ and $\overline{x'^2}$, respectively:

$$\overline{x'} = \int_{x_{np}}^{\bar{x}} x' f_{X'}(x')dx' = \frac{\bar{x} - \frac{1}{\lambda}\left(1 - e^{-\lambda x_p}\right)}{e^{-\lambda x_p}},$$

$$\overline{x'^2} = \lambda \overline{x'} \overline{x^2} + \left(1 - \lambda \overline{x'}\right)\left[\overline{x^2} - \frac{2}{\lambda^2}\left[\lambda \bar{x} - 1 - \left(\lambda x_{np} - 1\right)e^{-\lambda x_p}\right]\right].$$

The mean remaining service time regarding the job in service, if any, as seen by a job upon its arrival, is given by the formula $2R' = \lambda \overline{x'^2}$. To calculate the mean waiting time W', a job experiences in the system with the precomputing property, we observe, that the jobs that have to wait have the conventional service times, i.e., for these jobs is valid $x' = \bar{x}$. The sum of the mean waiting time W' and the $\overline{x'}$ equals to the mean delay T'. Hence, this leads to the following formula:

$$T' = \overline{x'} + \frac{\lambda \overline{x'^2}}{2(1-\rho)} = x_{np} + \frac{\lambda \overline{x^2}}{2(1-\rho)} = x_{np} + \frac{\rho \bar{x}}{(1-\rho)}.$$

In the next section we will present numerical results based on the formulas above and simulation results by removing the restriction only work for one job to accomplish in advance, when there is no job in the system.

Numerical results

The 'Figure 2' shows the performance benefit we achieve by applying precomputing for various utilization values and for $\bar{x} = 2$ msec in the case of an M/D/1 system with $x_{np} = 0$. As we expect for ρ near 0 the average system time in the system with the precomputing is near 0 and for $\rho \to 1$ we have almost no performance benefit when compairing with the conventional system in percent statements.

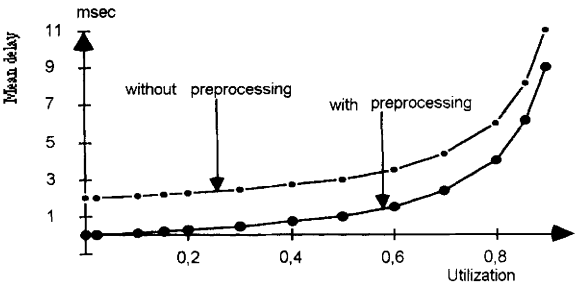

Figure 2 The average delay in the conventional system and in the system with the precomputing property as a function of the utilization of the conventional system.

The 'Figure 3' shows the performance benefit we achieve by applying preprocessing for various utilization values and for $\bar{x} = 50$ msec in the case of an M/D/1 system with $x_{np} = 10$ msec. As we expect for ρ near zero the average system time in the system with the precomputing property 'Figure 3' is near $x_{np} = 10$ msec and for ρ near 1 we have almost no performance benefit when compairing with the conventional system in percent statements. Recall that the performance benefit is constant in absolute amounts.

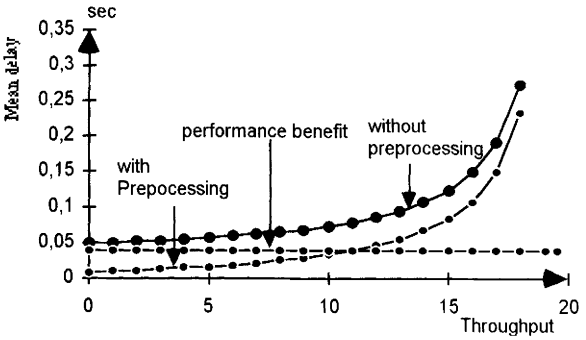

Figure 3 The average delay in the conventional system and in the system with the precomputing property with $x_{np} \neq 0$ and their difference, i.e., the perfomance benefit, as a function of the throughput.

Simulation results

Beside the mathematical analyses we carried out above, we studied the performance behaviour of the system with the precomputing property by simulations as well, where we examined the impact of a different number of jobs, for which work can be accomplished prior to their arrivals 'Figure 4'. Note that the performance impact of the cryptographic functions can be eliminated if we choose a suitable number of messages for which the random bit strings will be generated in advance.

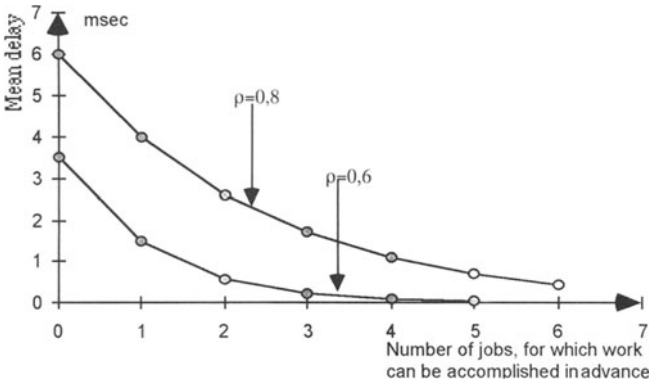

Figure 4 The mean delay in the system M/D/1 with the precomputing property for $x_{np} = 0$ as a function of the number of jobs, for which work can be accomplished prior to their arrival, and as a function of two corresponding utilization values of the conventional system.

4 CONCLUSION

In this paper we presented optimization concepts for security mechanisms in distributed systems and we analyzed their performance behaviour by means of queueing theory and simulations. We proposed the use of stream ciphers that base on strong random number generators or the OFB mode of operation of DES or a corresponding mode of IDEA which permit precomputing of the required random bit strings for the encipherment and decipherment of the communicating messages. We showed that the performance degradation caused by the security mechanisms can be reduced and maybe eliminated if the sender, the sequencer and the receivers compute the random bit strings in advance for an appropriate number of messages. Similarly, we proposed a concept for a data integrity and origin authentication mechanism, which permits the sender to precompute a signature of a sequence number and a timestamp with which the message is tied by a hash value.

5 REFERENCES

Bertsekas, D., Gallager, R. (1987) Data Networks, Englewood Cliffs, NJ: Prentice-Hall.

Chang, J. and Maxemchuk, N.F. (1984) Reliable Broadcast Protocols, ACM Transactions on Computer Systems, Vol. 2, 251-73.

Chaum, D., Evertse, J.-H. (1986) Cryptanalysis of DES with a Reduced Number of Rounds, Proc. of CRYPTO 1985, Advances in Cryptology, Springer Verlag, Lecture Notes in Computer Science **218**, 192-211.

DES (1977) Data Encryption Standard, National Bureau of Standards, Federal Information Processing Standards Publication (U.S.), FIPS PUB 6 .

Diffie, W., Hellman, M. E. (1977) Exhaustive Cryptanalysis of the NBS Data Encryption Standard, Computer, Vol. 10, No. 6, 74-84.

Gait, J. (1977) A new Nonlinear Pseudorandom Number Generator, IEEE Transactions on Software Engineering, No. 5, 359-363.

Garcia-Molina, H. (1982) Elections in a Distributed Computing Systems, IEEE Transactions on Computers, Vol. 31, 48-59.

ISO 8073. Connection Oriented Transport Protocol Specification.

ISO/OSI 7498-2: Security Architecture.

Jueneman, R. R. (1983) Analysis of Certain Aspects of Output Feedback Mode, Proc. of CRYPTO, Advances in Cryptology, Plenum Press, 99-127.

Kleinrock, L. (1975) Queueing Systems, Volume I: Theory, John Wiley and Sons, New York.

Li Gong and Nachum Schacham (1995). Multicast security and ist extension to a mobile environment, J. Wireless Networks, 1, 281-95.

Papoulis, A. (1991) Probability, random variables and stochastic processes, MacGraw-Hill.

Preneel, B. (1994) Cryptographic Hash Functions, Eur. Transactions on Telecommunication, Vol. 5, pp. 431-48.

Rivest, R. (1991) The MD4 Message Digest Algorithm, Proc. of Crypto 90, Springer Verlag, 303-11.

Tanenbaum, A. S. (1994) The Amoeba Microkernel, Distributed Open Systems, edited by F.M.T Brazier and D. Johansen, IEEE Computer Society Press, 11-30.

Tseung, L. N. (1989) Guaranteed, Reliable, Secure Broadcast Networks, IEEE Network Magazine, Vol. 3, 33-7.

6 BIOGRAPHY

Vasilios Zorkadis received the 'Diploma' degree in electrical engineering in 1983 from the 'Aristotle' University, Thessaloniki, Greece. He carried out postgraduate studies in computer science (1984-1986) at the University Karlsruhe, Germany, and he received the Ph.D. degree (Dr. rer.nat.) from the same University in 1994.

He is currently Visiting Professor at the University of Ioannina, Greece. He spent several years in Munich as a Managing Director of TDS GmbH (1986-1989) and in Karlsruhe as a Visiting Researcher (1990-1995) at the FZI Research Center for Computer Science.

His research interests include various topics in computer networks, security, distributed systems, and mobile communication systems.