Using the IPSec Architecture for Secure Multicast Communication

(Extended Abstract)

Thorsten Aurisch Christoph Karg[1]

Forschungsgesellschaft für Angewandte Naturwissenschaften e.V. (FGAN)
Forschungsinstitut für Kommunikation, Informationsverarbeitung und Ergonomie (FKIE)
Abteilung Kommunikation (KOM)
Neuenahrer Straße 20
D-53345 Wachtberg-Werthhoven
Germany
Telephone: (+49) 228/9435–479 and (+49) 228/9435–513
Telefax: (+49) 228/9435–685
EMail: t.aurisch@fgan.de and chkarg@fgan.de

Topic: Network-centric Applications
Keywords: IP Security, IPv6, Multicast

[1]Corresponding Author

Using IPSec for Secure Multicast Communications[2]

Thorsten Aurisch Christoph Karg

Forschungsgesellschaft für Angewandte Naturwissenschaften e.V. (FGAN)
Forschungsinstitut für Kommunikation, Informationsverarbeitung und Ergonomie (FKIE)
Abteilung Kommunikation (KOM)
Neuenahrer Straße 20
D-53345 Wachtberg-Werthhoven
Germany
Telephone: (+49) 228/9435–479 and (+49) 228/9435–513
Telefax: (+49) 228/9435–685
EMail: t.aurisch@fgan.de and chkarg@fgan.de

Abstract

A key component of the IP Security architecture is the Internet Key Exchange Daemon (IKE). IKE is invoked to establish keys and security related parameters between hosts in order to protected the exchanged application data with them. The IKE concept can not be used for securing group communication based on IP multicast services since it is only addressed to pairwise security. For negotiation and controlled distribution of group security data and membership management the Multicast Internet Key Exchange (MIKE) is introduced. The appropriate daemon will be developed for to the use in military environment. This implies for the MIKE protocols that they are insensitive against packet loss and fulfill special military conditions, e.g. the emission control mode (EMCON) of a user. This paper discusses the architecture and design for MIKE daemon which executes the management functions described above.

1. Introduction

A major task of a distributed Command, Control, and Communication Information System (C3IS) is the collection and transmission of information between the spatially separated parts of the system. For this purpose, usually a TCP/IP-based network is set up. Primary, TCP/IP was designed for unicast transmissions. This kind of transmission works well for peer-to-peer connections where an originator sends his data to exactly one receiver. The unicast model is suitable for an vast variety of internet applications. However, it has weaknesses in the case of data transfer from one sender to a group of receivers. As a matter of principle, such a type of communication is realizable via unicast by sending a copy of the datagram to each receiver. The cost of this method (in terms of bandwidth usage) is proportional in the number of receivers. Hence its applicability is restricted,

[2]Accepted paper of 8^{th} International Command and Control Research and Technology Symposium 2003 (see http://www.dodccrp.org).

especially in military networks where the available bandwidth is small and for this reason a valuable resource. A more bandwidth sparing mechanism is the concept of multicast communication. Multicast allows the transmission of a datagram from one sender to a group of receivers by only one send operation. The bandwidth usage of that operation is optimal since the datagram is only duplicated if necessary. Multicast requires an advanced routing environment which is capable to deal with this additional demand.

A mandatory requirement for military communication is the security of the data traffic. Especially the security services authentication, integrity, and confidentiality are demanded for the transmitted data. In terms of C3IS, a convenient way to establish this requirement is the setup of a virtual private network (VPN), where all components of the information system are connected by secure channels. The necessary mechanisms are provided by the Internet Protocol Security (IPSec) protocol suite [11]. Two protocols are used to provide a secure data traffic. The *Authentication Header* (AH) [9] provides data origin authentication, connectionless integrity and anti-replay service for IP packets. The *Encapsulating Security Payload* (ESP) [10] can be used for encryption, integrity and/or authentication of the payload of IP packets.

IPSec is designed to handle any kind of IP traffic. Hence, it is in particular applicable to multicast traffic. However, the security mechanisms included in the IPSec standard so far are not suitable for group communication. The reason is that the requirements for multicast security are different from those for pairwise security. In particular, the usage of a message authentication code (MAC) for data origin authentication is impossible for multicast groups of three or more members. Since all members share the same symmetric authentication key, a MAC cannot be attached uniquely to a group member. Hence, a member can impersonate another by using the same MAC. A promising approach to this problem is the extension of the ESP header to support group secrecy, group authentication, and source authentication [1].

Another problem to solve is the configuration of the concerning hosts, this is, the setup of the crypto algorithms to use and the generation and exchange of the necessary keys. In the case of unicast, an automated configuration procedure is already available. Via the Internet Key Exchange (IKE) [8], two machines set up a secure channel by negotiating security associations (SAs). Such an SA contains all IPSec parameters which are necessary to secure the data flow from one host to another. The session keys as part of the SA are generated randomly using the Diffie Hellman key exchange [7]. An automated setup procedure for secure multicast traffic must be more sophisticated. The reason is the fact that the group members may change during the secure communication. This involves an adaptive reconfiguration of IPSec settings. More precisely, if a new member joins the group then he needs to know the IPSec settings to participate in the multicast communication. For security reasons, the session keys have to be changed to prevent the new member from reading data sent before his join. If a member leaves the group, then a key renewal is also necessary since the leaving host shall not be able to take part in the future data transfer.

In our paper, we present the concept of a multicast internet key exchange daemon (MIKED). The task of this piece of software is to set up the parameters for a secure multicast communication under usage of the IPSec protocol suite. Following the unicast case, these parameters are accumulated in a multicast security association (MSA). Since we focus on Internet Protocol Version 6 (IPv6) [6], we choose USAGI Linux [18] as development platform since this operating system contains

an IPv6 capable IPSec module. Furthermore, its sources are open and may be modified for our purposes if necessary.

The paper is organized as follows. In section 2, we give a brief description of technologies our work is based on. The scenario in which the MIKED shall work is presented in section 3. Section 4 is concerned to the multicast internet key exchange daemon and its classification in the IPSec framework. Furthermore, our design goals are illustrated. Finally, in section 5 the architecture of the daemon is presented.

2. **Preliminaries**

2.1 ***The Multicast ESP Framework***

The research on multicast IPSec is an ongoing topic at the IETF. Currently, there exists no standard in terms of a request for comments (RFC) which solves the problem. A promising proposal is the Multicast ESP (MESP) framework [1]. It uses the ESP protocol to provide group secrecy, group authentication and source authentication of multicast packets. Group secrecy is applied by usage of the ESP confidentiality. For access of the data, the group members must possess a shared symmetric key. The group authentication functionality enables the group members to verify whether an ESP payload was originated within the multicast group. Group authentication is applied by usage of a symmetric MAC, whose key is shared among the group members. Using source authentication, a group member can verify the origin of the data and its integrity. This type of authentication requires stronger cryptographic techniques as group authentication. Suitable are digital signatures (for example RSA-SHA1) or timed MACs (for example TESLA [14]).

2.2 ***Key Exchange Protocols***

The group keys are the main components of an MSA. Hence, their negotiation is an important task of the MIKE framework. In principle, there exist two different kind of key exchange protocols depending on how the group key is generated.

- *Key agreement protocols:* The characteristic of this type of protocol is the participation of each group member in the construction of the group key. These protocols are usually generalizations of the Diffie-Hellman key exchange. Examples of key agreement protocols are given in [2, 16]. Key agreement protocols are used if all group members have equal rights and none of them wants to receive a precomputed group key from the other. The cost for this kind of influence on the key generation is large amount of bandwidth usage. The number of messages sent to establish the group key is of quadratic order in the size of the multicast group. Furthermore, an appropriate key size is necessary to guarantee the cryptographic safety of the protocol. Hence, key agreement protocols are only applicable for multicast groups of small size.

- *Key distribution protocols:* An efficient way to manage multicast groups with a large size of members is the usage of a key distribution protocol. In this case the generation and distribution of the group key is taken over by a key server. Therefore, the group members have no influence on the group key. An example of a this type of key exchange is the tree-oriented distribution scheme [19].

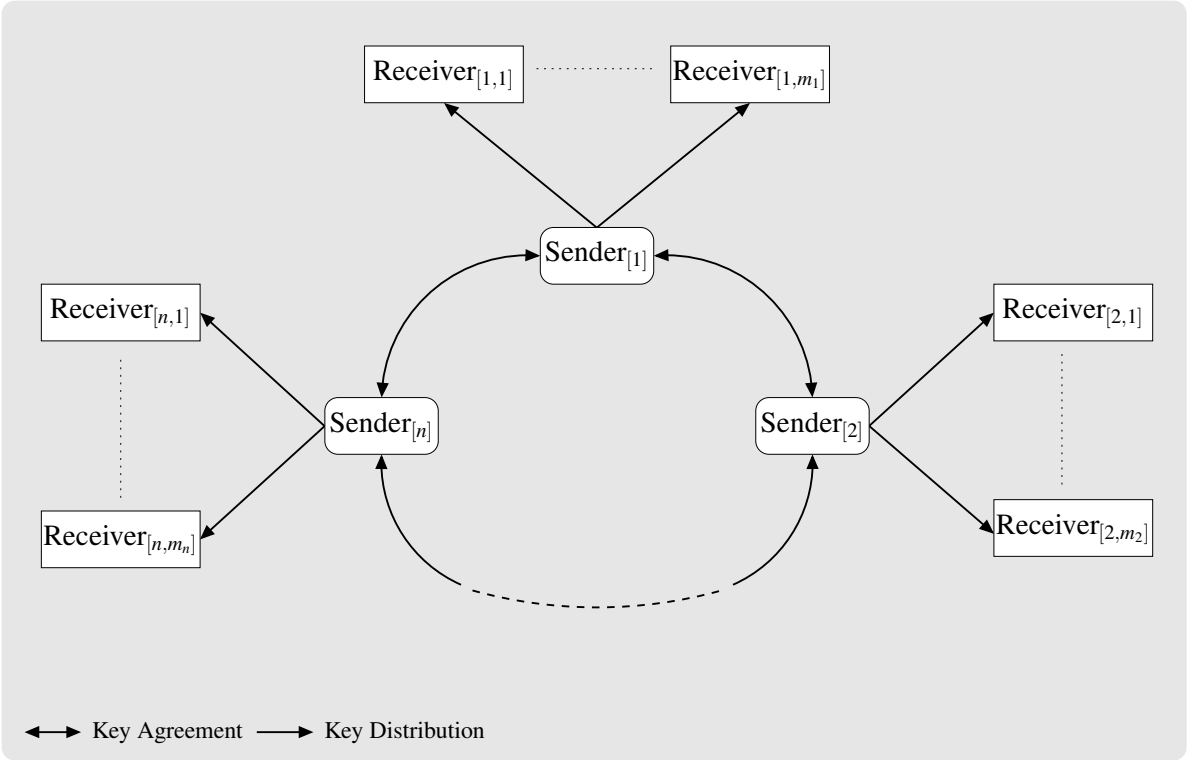

Figure 1: The scenario for our multicast communication.

Note that both types of protocols support forward and backward security of the data traffic. This is, a key renewal is performed in the case of a group member's join or leave.

3. **The Scenario**

In this section the scenario is presented where the secure group communication shall take place. For an illustration see figure 1. In the scenario there exist two types of group members. The first one are those hosts which both send and receive multicast data. The second one are the hosts which only receive multicast data. In the following, the hosts are called senders and receivers, respectively. We make the the following assumptions:

- The senders are connected by a network with wide bandwidth. Their number is small (at most 25 hosts) and they all have equal rights. Hence, a key agreement protocol is used for group key negotiation.
- The number of receivers may be large ($\approx$ 10000 hosts). The hosts are connected via any type of network, especially wireless networks of narrow bandwidth. Therefore, the group key is distributed by an tree distribution scheme. For this purpose, each receiver is associated with a unique sender which acts as key server.

We remark that this scenario is quite general and covers many cases which occur in military communication. An example is a military trial which is held via internet conference. The hosts of the

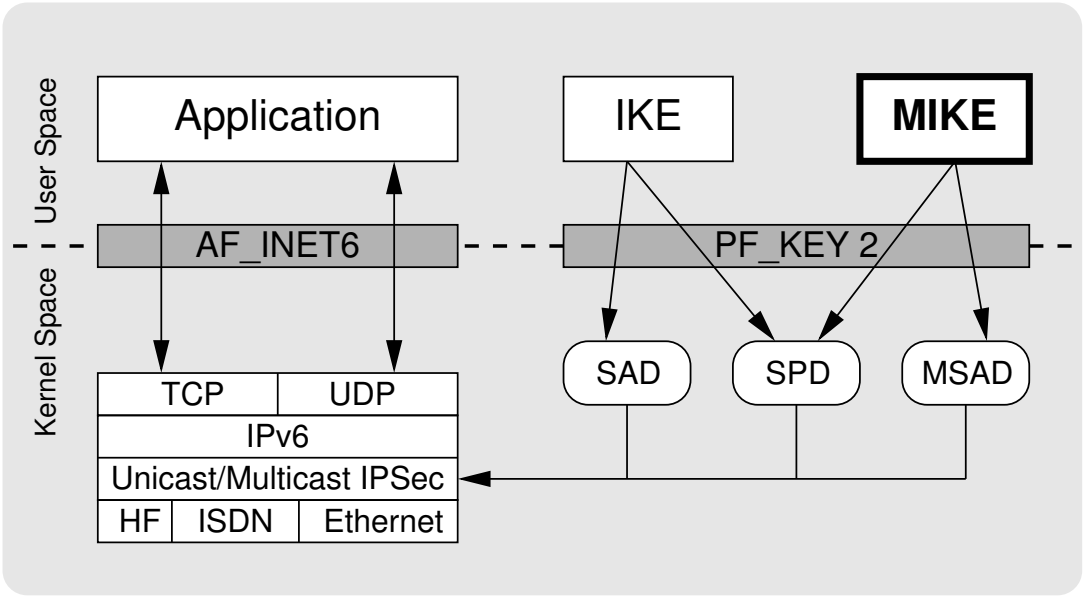

Figure 2: The MIKE daemon as part of the IPSec framework.

judge, the advocates, the accused and the witnesses are the senders. The receivers are the hosts of staff personnel (secretaries, etc.) and listeners.

4. The Multicast Internet Key Exchange Daemon

The MIKED must be regarded as part of the IPSec framework (see figure 2). It runs in the user space and has the task to negotiate the multicast security associations (MSA) which are necessary to establish a secure communication for a given multicast group. The MSAs are stored in the multicast security association database (MSAD) by calling the appropriate methods of the PF_KEY API [12]. A MSA is uniquely referred by a tuple consisting of the source address, multicast address, security parameter index and the protocol to be secured. The parameters stored in such an association are as follows.

- *Source and destination address:* In the case of secure group communication the destination address is equal to the multicast address of the group.
- *Source and destination port:* The specification of the ports allows a service dependent application of security services.
- *Security parameter index:* This index is part of an unique reference of the MSA.
- *Security protocol:* In the case of multicast the protocol is either MESP or a combination of AH and ESP.
- *Cryptographic algorithms:* Three different kinds of algorithms are necessary. For confidentiality, a symmetric encryption cipher is required (for instance 3DES or AES). Group

authentication is done via a symmetric MAC (for instance, MD5 or SHA1). For source authentication either a asymmetric crypto algorithm (for instance, RSA-SHA1) or TESLA is deployed.

- *Sequence number:* The sequence number may be used as protection against replay attacks.
- *Life time of the MSA:* An IPSec MSA must be renewed in periodic intervals. The life time describes the temporal duration of such an interval. Note, that in case of a member join or leave the key must renewed even if the interval is not expired.

To work successfully in the above scenario, MIKED must support two modes. If running on a sender host, the daemon must both be able to take part in the key agreement procedure and take over the secure key distribution of the client hosts. If running on a receiver host, the daemon's task is to connect a key server to get the MSAs for secure group communication.

In the design of MIKED we have the following goals in mind:

- *Independence from the underlying multicast mechanisms:* MIKED shall work in any multicast environment. Hence, we do not require a special kind of multicast routing or reliable multicast. All datagrams are sent via UDP to a multicast address.
- *Separation of key management and application:* The MSA management is totally separated from the applications whose data transmission shall be secured. Actually, the key exchange is transparent to the application. This allows the usage of application software, especially COTS products, without adaptions for multicast security.
- *Usage of existing standards:* The implementation of MIKED shall be based on existing standards as far as possible. For example, we deploy PF_Key API for configuration of the IPSec kernel module. Furthermore, the necessary changes in the kernel of the operating system shall be minimized.
- *Consideration of military situations:* Since MIKED shall work in an military environment, we have to take account of military circumstances such as emission control (EMCON) or narrow bandwidth in tactical networks.
- *Robust IPSec MSA exchange protocols:* The key exchange protocols mentioned in section 2.2 as main part of the MSA establishing process guarantee cryptographic safety. However, they are not applicable directly, since they leak of concrete protocol specifications. Hence, the exchange protocols have to be embedded in network protocols on base of TCP/UDP. Requirements for these protocols are robustness and thinness in the sense of packet size.
- *Flexibility:* We regard MIKED as an open platform which shall offer a large diversity of key exchange protocols. This is important for research and evaluation of protocols with respect of their applicability in military surroundings.

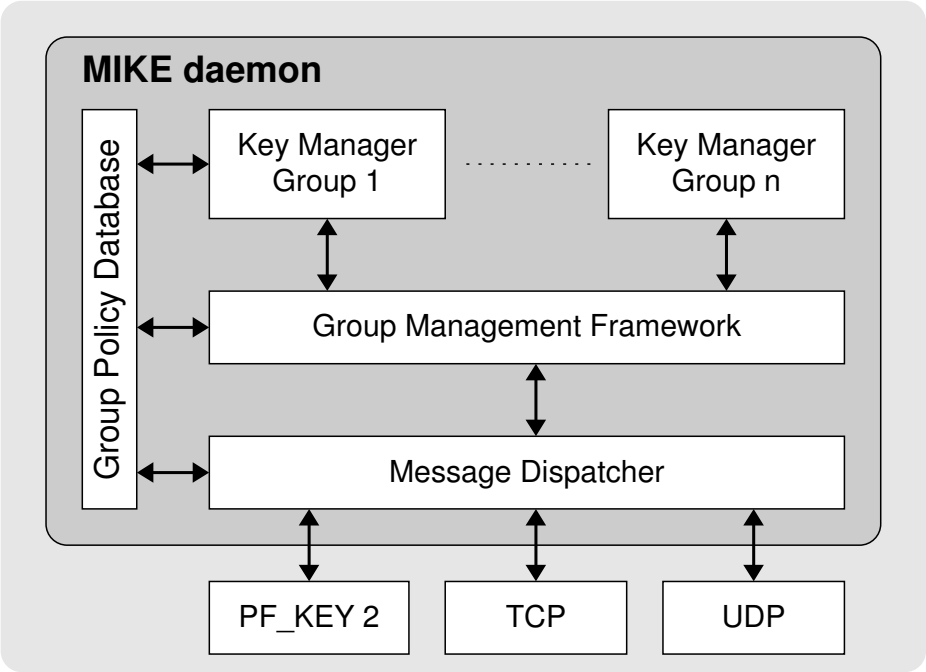

Figure 3: The architecture of the MIKE daemon.

- *Code reusability:* MIKED is in the first place a research vehicle. Besides a prototypical implementation which works in the real world we want to perform simulations on the key exchange protocols with respect to their applicability in military environments. Hence, many parts of code must portable such that it can be used in both the daemon and the simulation environment.

5. Architectural Overview

For the implementation of MIKED we propose an architecture as displayed in figure 3. The daemon consists of three layers. At the bottom is the *Message Dispatcher* which is responsible for the communication with the other group members. We differ between two dispatchers depending on the job to be done.

- The "real world" dispatcher goes into action if the key exchange is performed over the internet. This dispatcher has access to the operating system's TCP/IP stack and to the IPSec kernel module.
- The "simulation" dispatcher is used if the MSA negotiation of a multicast group is simulated on a single host. This dispatcher is equipped with facilities to simulate packet loss and transmission delays.

Both the real world and the simulation message dispatcher are connected to a certain number of group management frameworks which reside in the middle layer. The number of frameworks

depends on the type of dispatcher. The real world dispatcher is connected to exactly one group management framework, namely to the one which is responsible for the host's multicast key management. In the case of the simulation dispatcher, the number of group management frameworks matches the number of simulated hosts. Note that the type of dispatcher is transparent to the group management framework. This is, the framework cannot detect to which type of dispatcher it is connected.

The *Group Management Framework* is so to say the heart of the MIKED. It has the central task of coordination of key management for all multicast groups the host has joined and the respective message distribution. Furthermore, the framework ensures that the negotiated MSAs are transmitted from the key managements to the message dispatcher in order to be stored in the MSAD. The functioning of the framework depends on the type of host the daemon is running. If the host is a sender then the framework has to react on both key agreement and key distribution protocol messages. Otherwise the group management is restricted to key distribution client processing.

The *Key Managements* are located in the upper layer of the MIKE architecture. A key management component manages the MSA negotiation of exactly one multicast group. To describe the functioning we differ again between sender and receiver hosts. In the former case, the key management must be able to perform a key agreement protocol as well as a key distribution protocol in order to serve the assigned receivers. In the latter case, the key management is limited on the communication with the associated sender to receive the MSA for the joined multicast group.

All the above components have access to the *Group Policy Database*. This database holds all relevant information concerning policy and security of MIKE. The kind of information depends on the requesting component.

- The message dispatcher gets IP filtering rules to sort out datagrams for or from hosts which are excluded from group key management.
- The group management framework gets information about the multicast groups for which key management is permitted. Furthermore, the framework is supplied with all needed parameters for the successful processing of the set in key exchange protocol(s).
- An important task of the key management is to authenticate the outgoing messages and verify the incoming messages on integrity and source authentity. The needful keys are stored in the group policy database.

The implementation of MIKED is based on the USAGI project [18]. USAGI stands for Universal Playground for IPv6 (USAGI). The goal of the project is to deliver the production quality IPv6 and IPSec (for both IPv4 and IPv6) protocol stack for the Linux system. Several times a year, the USAGI project publishes a tar archive which contains patched sources of a linux kernel of recent date and a collection of IPv6 capable UNIX tools (inetd, telnet, etc.). The MIKED itself is developed in C++ under usage of the Standard Template Library (STL). For cryptographic methods the Crypto++ library is used [4].

References

[1] Mark Baugher, Ran Canetti, Pau-Chen Cheng, and Pankaj Rohatgi. *MESP: A Multicast Framework for the IPsec ESP*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, March 2003. (draft-ietf-msec-mesp-01.txt).

[2] Mike Burmester and Yvo Desmedt. A secure and efficient conference key distribution system. In A. De Santis, editor, *Advances in Cryptology – EUROCRYPT ' 94*, Lecture Notes in Computer Science #950, pages 275–286. International Association for Cryptologic Research, Springer-Verlag, 1995.

[3] Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas. Multicast Security: A Taxonomy and Efficient Constructions. In *INFOCOMM '99*, 1999.

[4] Crypto++ Library: a free C++ class library of cryptographic schemes, 2002. Webpage: `http://www.eskimo.com/~weidai/cryptlib.html`.

[5] Stephen Deering. *Host Extensions for IP Multicasting*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, August 1989. RFC 1112.

[6] Stephen Deering. *Internet Protocol, Version 6 (IPv6) Specification*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, December 1998. RFC 2460.

[7] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, (22):644–654, 1976.

[8] Dan Harkins and Dave Carrel. *The Internet Key Exchange (IKE)*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, November 1998. RFC 2409.

[9] Stephen Kent and Randall Atkinson. *IP Authentication Header*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, November 1998. RFC 2402.

[10] Stephen Kent and Randall Atkinson. *IP Encapsulation Security Payload ESP*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, November 1998. RFC 2406.

[11] Stephen Kent and Randall Atkinson. *Security Architecture for the Internet Protocol*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, November 1998. RFC 2401.

[12] D. McDonald, C. Metz, and B. Phan. *PF_KEY Key Management API, Version 2*. Internet Engineering Task Force, Webpage `http://www.ietf.org`, 1998. RFC 2367.

[13] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[14] Adrian Perrig, Ran Canetti, J. D. Tygar, and Dawn Song. The TESLA Broadcast Authentication Protocol. *RSA Security CrytoBytes*, 5(2), 2002. Webpage `http://www.rsasecurity.com/rsalabs/cryptobytes`.

[15] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[16] Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-Hellman key distribution extended to groups. In Clifford Neuman, editor, *3rd ACM Conference on Computer and Communications Security*, pages 31–37, New Delhi, India, March 1996. ACM Press.

[17] Andrew S. Tanenbaum. *Computer Networks*. Prentice-Hall, 3rd edition, 1996.

[18] USAGI (UniverSAl playGround for Ipv6) Linux IPv6 Development Project, 2002. Webpage: `http://www.linux-ipv6.org`.

[19] Chung Kei Wong, Mohamed G. Gouda, and Simon S. Lam. Secure group communications using key graphs. In *Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 68–79, 1998. Appeared in ACM SIGCOMM Computer Communication Review, Vol. 28, No. 4 (Oct. 1998).