Proceedings of the 2nd International Conference on Information Technology for Application (ICITA 2004)

A Novel Network Security Solution

Y. -S. Ma and Eric X. W. Jiang

School of Mechanical and Production Engineering, Nanyang Technological University, Singapore

***Abstract*--The widespread use of computers and networks environments has emphasized the necessity to study the secure operating systems and network security mechanisms. Current commercial solutions for network security, such as firewalls, cannot defend against insider attacks, nor can they support sophisticated trust relationships with external entities. This paper presents a more comprehensive and flexible network security solution. It enforces a mandatory access control policy and encryption on network-related operations and traffic.**

***Index Terms*--Network security; Internal security; TCP/IP stack; CIM**

I. Introduction

INFORMATION technology has enabled sophisticated trust relationships with external entities. This introduces the security issue to ensure confidentiality, integrity and availability of information. Current commercial solutions, generally known as firewalls, protect against attacks by denying network services to outsiders except those secure users, but can neither defend against insider attacks, nor can support sophisticated trust relationships with external entities.

This work presents a network security solution implemented in network Transmission Control Protocol/ Internet Protocol (TCP/IP) stack. The above problems are addressed by mandatory encrypting the information sent on the wire and mandatory controlling access to all the network-related operations in a secure manner.

This paper is organized into eight sections. After this introduction, in Section II, a brief review on the current common practice and the relevant research works is given. Sections III, IV and V present the proposed overall security architecture, and the two major aspects of design framework, network access control and cryptographic protection. Section VI describes the implementation while Section VII shows the experimental results on the costs of security services in the network stack. Section VIII offers conclusions and suggestions for the future work respectively.

II. Review

Typical security problems are inter-process communication threats that can take place while services are being processed by the communication protocols. Current firewalls segment portions of a network for the purpose of preventing damage from unauthorized access or other potential threats from the Internet. Firewall systems can be classified according to their hardware configurations, e.g. routers, circuit proxies, and application/proxy servers. They have been developed to provide functional features such as encryption, authentication, network address translation, virus scanning, URL filtering, bandwidth management, event logging, monitoring, and intrusion notification [1].

Network internal addressing details can be used for “spoofing” and other attacks [1]. Some attacks modify the event logs to erase the perpetrator’s tracks; other attacks review event logs to glean passwords, IDs, and network and processing components [2]. Many intrusions can only be detected by reviewing logs [3]. Although using firewalls is the most common method to control the remote users access to an organization’s internal network, it provides very limited ability to authenticate the internal user and not very flexible. In most cases, it offers an "all-or-nothing" solution only. Trusted computer system principles are vital to inter-organization collaboration, but trusted system concepts by themselves are insufficient to provide network security solutions [4].

Encryption is necessary to achieve confidentiality, integrity, non-repudiation, and continuity. Symmetric and asymmetric key cryptographies are used. Asymmetric key encryption, like PKI [4], is more popular because the sender and the receiver’s private keys can be protected. For wide interoperability, protocols for cryptographic support are commonly used. IETF has proposed the IPsec [1] and TLS [5] protocols for providing cryptographic support at the IP and application layers respectively. The Internet Security Association & Key Management Protocol (ISAKMP) [6] provides a framework for Internet key management and specific protocol support for negotiation of security attributes.

Covert channels occur when some mechanism is used in an unexpected manner to provide a means by which information can flow to an unauthorized entity [7]. In the context of the network subsystem, the shared port number space creates such problems. There is no earlier effort attempted to solve them yet.

A security policy defines the security attributes associated with the entities within a system and the conditions governing actions of and relationships between these entities (users, files, network interfaces, etc) [1][8][16], such as multilevel security, Clart-Wilson, and Biba integrity models. Some mechanisms like Domain and Type Enforcement (DTE) [8][9] provide a framework for expressing and enforcing security policies. Security perimeters are used to ensure a collection of nodes

Dr. Y. -S. Ma is an Associate Professor with the School of MPE, Nanyang Technological University (NTU), 50 Nanyang Avenue, Singapore 639798 (telephone: +65-67905913, e-mail: mysma@ ntu.edu.sg).

Mr. Eric X. -W. Jiang, was a Msc student of NTU. He is now an IT specialist with the department of Product Lifecycle Management Solutions, IBM Singapore Pte Ltd, 80 Anson Road, IBM Tower, Singapore 079907 (e-mail: jiangxe@sg.ibm.com).

ICITA2004 ISBN 0-646-42313-4

adhering to a particular security policy. Communication between different perimeters requires policy translations [10].

A mandatory security policy is defined to be one where a system security policy administrator tightly controls security logic and attributes. It can implement organization-wide security policies instead of individual users' policies and hence avoids the burden for security on individual user. Some solutions like HannaH [11] and Secure Socket Layer (SSL) [12] libraries provide security functionality in the application layer. However, they cannot achieve mandatory policy support.

Security measures in operating systems has long been a topic of research [13] because mandatory policy support can be achieved by modifying the network stack. The Trusted Computer System Evaluation Criteria (TCSEC) [14] and The Trusted Network Interpretation (TNI) [15] specified the general principles and mechanisms to be applied in any computer system handling sensitive information, but failed to discuss either encryption or protocols [4]. Examples of them include Multi-Level Security (MLS)/TCP [16], Romero et al's model [17], and the DTE mechanism [9]. Distributed Trusted Operating System (DTOS) [13] was invented which includes network access control checks at the socket layer. However, the amount of access control that can be implemented in the socket layer is limited because of the layered nature of the network stack. Recent attempts tried to make firewalls more flexible [9], but this approach is to modify the operating system and make each machine responsible for controlling the input and output operations. The Locked Workstation Expanded Environment Study report [10] proposed an access control similar to their model. So far, the past efforts on integrating mandatory access control into the network stack were very policy-specific; the network stack was modified to directly interpret a specific policy.

III. Proposed Security Architecture

Figure A shows a scenario in which a trusted file server process is running on node A and a trusted client process is running on node B. It can be examined here that what mechanisms can be used to deal with the security threats from untrusted processes running on any of the nodes. Access control mechanisms based on a mandatory security policy prevent untrusted processes from reading or modifying files on the trusted file server. The security policies on nodes A and B

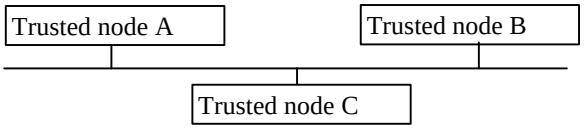

Figure A Example scenario

could prevent any communication with node C. Authentication measures defend the threat of an untrusted user on node B masquerading as a trusted user. Encryption protects the confidentiality of the data on the wire. If an untrusted user on node C or some one who has access to a network component tries to modify the message stream between the trusted file server process and the trusted client process, integrity checks on the packet will fail leading to the packets getting dropped.

The above discussion suggests that the basic requirement for secure communication is to enforce checks on all the network-related operations to ensure that they conform to the policy requirements. Broadly speaking, the network security architecture consists of two major components, network access control and cryptographic protection.

IV. Proposed Network Access Control

A. Network Model

The fundamental design principle of the network model is to separate enforcement from policy in order to achieve flexibility. This goal is achieved by utilizing a security server to make the actual decision. The primary role of the security server is to isolate the rest of the system from the security policy by making all mandatory security decisions. The interface to the security server is the same regardless of

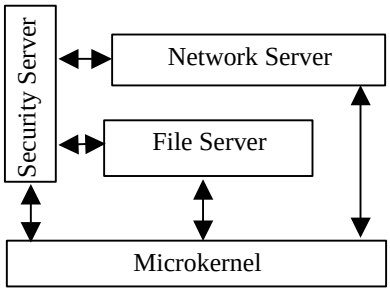

Figure B A typical network security architecture

security policy, thus changing the policy or the implementation of the security databases will have no effect on the rest of the system. As shown in the Figure B, the microkernel and other entities such as the file and the network servers, which need to enforce the security policy, communicate with the security server for making access control decisions.

The data, which the security server needs to make a decision based on the security policy, are generally in the form of a security context, which is a data structure containing all of the security information that must be bound to an entity. This could include the sensitivity level and user identification. Security contexts are treated as opaque blocks of data by tasks other than the security server including the microkernel. From outside of the security server, a security context is referenced by an opaque handle called security identifier (SID). A security identifier (SID) is bound to each entity controlled by the mandatory access control policy. Security identifiers are non-persistent and are meaningful only on the local node. The security server provides functionality for converting a SID to its corresponding security context or a security context to its corresponding SID. All the entities for which security is being enforced are divided into security classes. A security class is a distinct type of object with a distinct set of legal operations, for example, a file, directory, network interface or a node.

When a security decision must be made, the microkernel (or any of the servers) passes the SIDs, the security classes of the subject and the object to the security server. The security server then will computes the allowable access vector for the security contexts and classes associated with the SIDs. For efficiency purpose, policy enforcers cache the results obtained from the security server. In order to support adaptive policies, information about which permissions may be cached and when the access vector's validity will expire is also returned. In addition, the policy enforcers can register a notification port for cache flushes.

B. Network Access

Network access control design is layered to match the network architecture. Firstly, it involves identifying the various network services and resources granted at different layers in the network stack. Then permissions associated with them are identified and the corresponding checks are imposed at points where they are granted. In addition to the normal socket calls, a few system calls are added for additional functionality. These calls are required for specifying the security contexts for an operation or for obtaining the security contexts of the peer.

Unlike the substream approach taken by the DTE work on network security [9], in this model, all the packets going out of a TCP connected socket have the same security attributes. This design binds security attributes to network entities in the lower layers of the network stack such as network interfaces. This approach also includes MLS/TCP. It has the advantage that illegal incoming packets can be detected earlier, thereby reducing the processing time spent on them.

C. Policy enforcement

Traditionally, the access control decision is split into a collection of checks made at different layers and on different nodes due to the layered architecture of the network. In this work, access control decisions are made on a dedicated server and checks are carried out in the network stack; and all the information needed to make the decisions is made available to the security server.

The server node can perform access control checks based on the security attributes of the source process, the source node, the network interface on which the packet was received, and the recipient process. When a process attempts to accept a connection or receive a packet, if the policy prohibits, the packet or the connection is silently dropped. The access control design binds security attributes to network-related entities at different layers as shown in Table A. Similarly, access control is enforced on operations as shown in Table B.

Table A Binding security information to entities at different layers

Entity	Layers in the network stack
Sockets	Socket layer
Port names	Transport layer
Nodes, packets, routing table	Network layer
Network interface	Physical data link layer

Table B Enforcing access control at different layers

System calls	Layer in network stack
socket (), bind (), connect (), listen (), accept (), getsockname (), getpeername (), setsockopt (), getsockopt (), send (), recv (), recvform (), shutdown ()	Socket layer
in_pcbbind (), tcp_input (), udp_input ()	Transport layer
Ip_input (), ip_output (), rtm_* ()	Network layer
	Physical data link layer

By implementing the access control checks right at the point where a particular network service is being granted, leaking any information to unauthorized processes or nodes can be avoided. For example, the permission check on a connection request packet needs to be performed at the TCP layer. If the check is performed above the TCP layer, then a check can be performed only after the TCP connection is established, at which point the information that a socket was listening on that port would have been leaked. If the connection request permission checks fail, then the TCP input routine can act in the same way as if no socket had been bound to that port, and thus the client obtains no information.

Different applications and transport layers could implement different controls varying in strength. For example, TCP supports connection-oriented communication whereas User Datagram Protocol (UDP) supports only datagram communication. Such differences create the necessity for different kinds of policies and enforcement. So, some of the access control decisions are bound to be extremely protocol dependent.

The design in this work enables us to identify the services related to different protocols at the different layers and associate permissions with them. For example, it can be easily specified which sockets could act as a client or a server to a particular socket using the client-associate and server-associate permissions respectively. Hence, it exports fine-grain access rights. It associates access rights with every access control check and each of these access rights could be used for policy specification. Fine-grained access rights are useful both for policy flexibility and for supporting the principle of least privilege.

Resources may be consumed by unauthorized traffic. Rejecting packets at the application layer or even the transport layer is often too late. It shows the vulnerability of a node to denial of service attacks. Since the proposed access control server is implemented at the network layer, it allows to minimize the time spent on unauthorized traffic by discarding illegal packets as early as possible. This design binds nodes and routing table at the network layer. This is because some nodes provide transit services only and therefore there must be some desired controls as part of the routing protocol. Since routing is a network layer function, these controls must involve network layer entities and cannot be left to transport layer endpoints. In addition, by performing the access control

checks at the corresponding layers of the relevant entities, violating network protocol layering can be avoided.

D.Binding Security Information

This section describes how security information is bound to the different network related entities.

1) Nodes

Each node (e.g., host, router, etc.) has a security context. The security context of the destination node is used to verify whether the packets can be sent to that host and similarly, the security context of the source node is used to check if incoming packets from that host can be accepted or not. Performing these checks allows a system to avoid exposing resources to either internal or external un-intended systems.

2) Sockets

A socket acts as a communication end point through which multiple processes can send or receive data. Hence security contexts are bound to sockets for use in send and receive operations on the sockets. They are used in permission checks on operations that manipulate socket private state. There are distinct socket security classes for each (domain, type, protocol) triple.

3) Network Interfaces

Each network interface on a node has a security context associated, which is used in access control checks when sending and receiving network packets through the interface. This design controls the set of outgoing packets and the set of incoming packets that may be sent or received via the network interface. Hence it can support policies that prohibit certain kinds of data from leaving the local node or prohibit certain kinds of data from entering the local node. If a node has multiple network interfaces, they can be assigned different security contexts.

4) Routing Table

The network stack uses the routing table for obtaining information about the route on which a packet should be sent. The network security policy server's configuration specifies a security context for the entire routing table. The routing table security identifier is used to control the ability of processes to observe and modify entries in the routing table.

5) Packets

The security context of a packet is used in permission checks on operations that send and receive packets. Additionally, for connection-oriented sockets (stream sockets), the SID of the connection request packet and the SID of the connection confirmation packet is used to indicate the SID of the remote socket. In this design, a TCP connection is one between two sockets instead of between the traditional dynamic sender and receiver processes. Every packet sent via a stream socket (TCP) inherits its security context from the socket. So the socket used by the sending process acts as the effective sender. Therefore, multiple senders with different SIDs on a connected stream socket can be supported, as long as the policy permits each of them to generate packets with the socket SID. Supporting substreams with different security contexts as done in the DTE approach [9] is not required.

E. Virtualized Port Number Space

The UDP and TCP port number spaces provide a shared resource that can be used as a covert channel [1][7]. Creating a virtualized port number space and implementing security policy on them can prevent this. A regular port space is defined a collection of socket/port-name pairs, in which each port name is unique within the port space. A security union port space is defined as a collection of regular port spaces, where its SID may uniquely identify each member port space. Each transport layer protocol's port space is a security union port space, and has a corresponding security context defined in the network security policy server's configuration. When the transport layer receives an inbound packet, the network server first checks whether the default and any specific member port space in the union port space may receive the packet, if not, the connection is refused or dropped. The two permissions are defined for regular port spaces as shown in Table D.

Table D Permission of regular portspaces

Permission check port space	Entities involved	Description
Space bind	Portspace x socket	Control association between this port space and this socket
Receive	Portspace x psocket	Control ability of a port in this port space to receive this packet

V. Proposed Network Cryptographic Protection

Placement of the cryptographic service within a specific physical layer of the network has far-reaching implications concerning the nature and extent of the service implementation. Three layers can be considered: data link, network and application layers.

Performing encryption at the data link layer suffers from the possible exposure of clear-text versions at intermediate nodes and the high cost of key distribution; and is more naturally suited to individual users' perceptions.

In this work, the Internet Protocol (IPsec) [1] is used as a standard authentication and encryption at the network layer. IPsec provides a high-level definition for two IP security mechanisms, the Authentication Header (AH) and Encapsulating Security Payload (ESP). IP AH and IP ESP are not bound to specific cryptographic algorithms. However, at a minimum, implementations must support keyed MD5 (a Message-Digest algorithm [18]) for IP AH and DES for TP ESP [19]. Usually, there is a problem with network layer encryption that it does not support Security Association (SA) between end users in a direct manner. To establish security associations needs to each host to specify a set of attributes including the set of mechanisms (e.g., none, ESP, AH or both) and cryptographic information (e.g., algorithms, keys, initialization vector length, and cryptographic checksum length). Some IPsec implementations [12] tried to get around. This work overcomes the problem by using an application-layer key management daemon, Cisco's ISAKMP

daemon [6], to manage encryption keys and configure IP security associations in the operating system as shown in Figure C.

Negotiations are triggered when a network situation occurs for which there is no established security association. When a process invokes connects or sends, and a security association does not exist for the situation, the IPsec layer performs an *upcall()* to the negotiation server (ISAKMP) requesting an outbound association. After the key exchange is complete, a security association is available for use by the outgoing and incoming traffic.

Groups of communicating nodes need to agree on some common interpretation of security label and policy information as well as the cryptographic mechanisms. A Domain of Interpretation (DOI) module is available in ISAKMP for the purpose. Some nodes could probably implement multiple DOIs and act as gateways between different security perimeters. However, ISAKMP DOI module does not include the packet and other security contexts in the network situation. A new DOI based on the IPsec is defined in this work. This DOI allows the exchange of the security contexts required by this access control model and still supports ISAKMP DOIs; hence the interoperability is preserved.

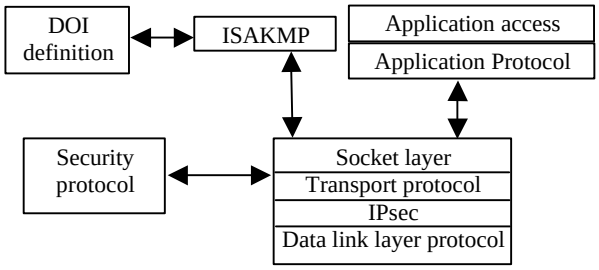

Figure C Placement of ISAKMP in the proposed architecture

Secure Sockets Layer (SSL) [12] model provides a standard for authentication and encryption at the application layer, but it does not ensure the integrity of the transport layer protocol header or the binding between the header and payload. On the other hand, the security provided by the SSL protocol is not transparent to the application. It means that all the networking applications that need security needs to be rewritten to comply with the standard. Hence, it cannot satisfy the intention of a mandatory cryptographic policy. In addition, the SSL protocol does not provide an equivalent Domain of Interpretation (DOI) as provided ISAKMP does for interoperability purposes.

VI. Implementation

The communication between different components in the network security architecture occurs as shown in Figure D. Access control checks are performed on different network-related operations and at the network layer. The functions in the network stack are modified for implementing the access control checks. Access control checks are implemented by making calls to the security server. Security attributes of network-related entities such as nodes and ports are obtained by querying the network security policy server.

Each packet has information about the IPsec protocols to be used and the network situation associated. The network situation consists of the source and destination socket security contexts. The IPsec layer looks for a security association that can be used for this packet. If no security association can be found, it sends a request for setting up a security association to the ISAKMP negotiation server and queues the packet. It also sends the network situation along with the request corresponding to the security context of the node and the default packet security context of the node is used.

If a permission request for receiving a data packet or a connection request fails, the message or connection is silently dropped without any notification to the receiver. This is to avoid a covert channel. This work supports extremely fine granularity of access rights; it provides 33 accessing rights compared with only 2 by the DTE implementation [9]. Fine-grained access rights are useful both for policy flexibility and for supporting the principle of least privilege.

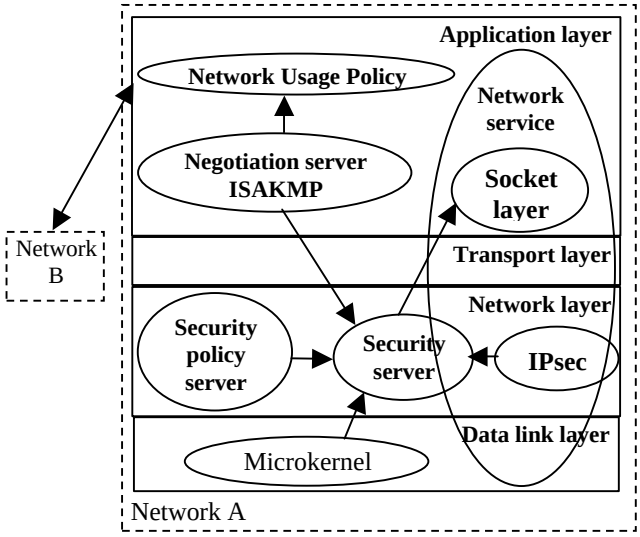

Figure D Implemented security architecture

VII. Results

All tests were performed on a 1G MHz Intel Pentium 4 with 512MB of memory as the client and a 1G MHz with 512MB memory of IBM PowerPC Unix workstation 44P-170 as the server, using a 100 MB switched Ethernet.

The connection establishment data is gathered via test runs (as shown in Table E). SA setup and the associated IPsec queue checking are responsible for 99.8% of the cost. This significant time loss is due to the asynchronous design of this check. A better implementation of IPsec could probably eliminate this asynchrony completely, reducing connection establishment time by up to 1.0 second. The performance of the connection establishment improves if the required security associations are already present in the kernel's cache. In this situation, security-imposed costs cause connection establishment time to increase from 3.4 ms (without network

security) to 8.6 ms, an additional delay that is not noticeable to humans. It should be noted that setting up a security association is required only if a security association for the particular network situation does not exist.

Latency is assessed by measuring application-to-application roundtrip time of a TCP packet carrying 1 byte of data (see Table F). The network security measures in this implementation increase latency by about 40% for the minimum size packets. The encryption time is related to the packet size. For 1, 512 and 1024 byte packets, the latency ratios are 1.4, 1.5 and 1.6 respectively. It seems that for long packets the latency would approach to a stable ratio of 2. If cryptographic hardware is used, i.e. encryption is not included; the costs for access control and SUPs suffer only about 10% penalty on top of the basic network access time.

Table E Times for connection establishment

Operations	Without SAs and access vectors in cache (ms)			With SAs and access vectors in cache (ms)		
	Min	Max	Median	Min	Max	Median
Server/Client SA setup	2x1870	2x3370	2x2760	0	0	0
Server/Client IPsec queue check	2x31	2x494	2x355	0	0	0
Access control checks	4.8	5.3	5.0	0.33	0.46	0.41
IPC calls for SUPs	3.3	3.6	3.5	3.3	3.6	3.5
Base cost	3.2	3.5	3.4	3.2	3.5	3.4
Encryption	1.09	1.63	1.34	1.09	1.63	1.34
Total connection establishment time	3814	7742	62	7.92	9.19	8.65
Observed connection establishment time	5920	6250	6130	8.2	8.9	8.6

Table F Round-trip time for an I-byte TCP message

Operation	**Time (in ms)**
Base (with no network security measures)	3.7
Access control including Security union port spaces (SUPs)	0.4
Encryption	1.1
Total	5.2

VIII. Conclusions And Future Work

In conclusion, this paper presented a network security solution with mandatory access control and mandatory cryptography in the network stack. The approach used separates policy from enforcement. The novelty of this design is the transparent mandatory security enforcement for all applications, including internal or trusted external uses. Security-aware applications can specify preferences using the system calls provided in this work. The interoperability with the original protocols is preserved. It can partially solve the problem of covert channel.

As to the future work, in the current implementation, the code that selects the cryptographic measures to be adopted is not separated out from the code that enforces the cryptographic requirements. This separation needs to be done. The current implementation assumes that different operating system nodes on the network use a common set of security contexts. A policy-mapping service needs to be implemented.

References

[1] R. Atkinson, “Security architecture for the Internet Protocol,” RFC 1825, Internet Engineering Task Force, Aug. 1995.

[2] S. M. Bellovin, “Problem areas for the IP security protocols,” in Proc. Sixth Usenix UNIX Security Symposium, July 1996.

[3] N. MacAuliffe, “Is your computer being misused? A survey of current intrusion detection system technology,” in *Proc. Sixth Annual Computer Security Applications Conference*, 1990.

[4] S. T. Walker, “Network security: The parts of the sum,” in Proc. *IEEE Symposium on Security and Privacy, pp.* 2-9, 1989.

[5] T. Dierks and C. Allen, “The TLS protocol,” Internet draft, Transport Layer Security Working Group, May 1997.

[6] D. Harkins and D. Carrel, “The resolution of ISAKMP with Oakley,” Internet draft, IPsec Working Group, July 1997.

[7] C. G. Girling, “Covert channels in LAN’s,” *IEEE Trans. Software Engineering*, vol. 13, pp. 292-296, Feb. 1987.

[8] V. Varadharaj an, “Network security policy models,” in A USCR *YP T90* (J. Seberry and J. Pieprzyk, eds.), vol. 453 of *Lecture Notes in Computer Science.* pp. 74-95, Springer-Verlag, 1990.

[9] T. J. Fraser, M. J. Petkac, W. G. Morrison, M. L. Badger, B. Uecker, E. L. Cohen, J. Grillo, K. A. Oostendorp, K. C. Djahandarl, T. P. Horvath, C. Ko, D. L. Sherman, “DTE firewalls phase two measurement and evaluation report,” TIS Report #0682, Trusted Information Systems Inc., July 1997.

[10] Secure Computing Corporation, “Locked workstation program LWS ex-panded environment study report,” Technical Report, Secure Computing Corporation, Jan. 1996.

[11] Security First Technologies, “A white paper on HannaH and network security”, http://www.securewaxe.coin/products/hatinah/hannahpaper.html

[12] A.O. Freier, P. Karlton, and P. C. Kocher, “The SSL protocol,” Internet draft, Transport Layer Security Working Group, Nov. 1996.

[13] T. Fine and S. E. Minear, “Assuring Distributed Trusted Mach,” in *Proc. 1993 I-EEE Symposium on Research in Security and Privacy*, pp. 206-218, May 1993.

[14] National Computer Security Center, *Department of Defense Trusted Computer System Evaluation Criteria.* 1985. DoD 5200.28-STD.

[15] National Computer Security Center, *Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria.* 1987. NCSC-TG-005.

[16] D. Futcher, R. Sharp, and B. Yasaki, “Building a multi-level secure TCP/IP,” in *Proc. 14th National Computer Security Conference, pp.* 78-87, 1999.

[17] S. Romero, C. Schaufler, and N. Bolyard, “BSD IPC model and policy,” in *Proc. 16th National Computer Security Conference*, pp. 97106, 1993.

[18] P. Metzger and W. Simpson, “IP authentication using keyed MD5,” RFC 1828, Internet Engineering Task Force, Aug. 1995.

[19] D. Piper, “The Internet IP Security Domain of Interpretation for ISAKMP,” Internet draft, Network Working Group, Cisco Systems, Feb. 1997.