An Environment for Security Protocol Intrusion Detection

Alec Yasinsac

yasinsac@cs.fsu.edu

Phone: 850.644.6407

Fax: 850.644.0058

214A James Jay Love Building

Department of Computer Science

Florida State University

Tallahassee, FL 32306-4530

Abstract

Secure electronic communication relies on cryptography. Even with perfect encryption, communication may be compromised without effective security protocols for key exchange, authentication, etc. We are now seeing proliferation of large secure environments characterized by high volume, encrypted traffic between principals, facilitated by Public Key Infrastructures (PKI). PKIs are dependent on security protocols. Unfortunately, security protocols are susceptible to subtle errors. To date, we have relied on formal methods to tell us if security protocols are effective. These methods do not provide complete or measurable protocol security. Security protocols are also subject to the same implementation and administrative vulnerabilities as communication protocols. As a result, we will continue to operate security protocols that have flaws.

In this paper, we describe a method and architecture to detect intrusions in security protocol environments such as Public Key Infrastructures. Our method is based on classic intrusion detection techniques of knowledge-based and behavior-based techniques detection.

Section 1. Introduction.

The Internet is at the center of the economic growth in the United States today. E-commerce, dot-coms, the unprecedented expansion of the NASDAQ, etc. are strong indicators that network traffic and the expansion of the Internet has been the lifeblood of economic growth in the United States. Protecting this vital resource is increasingly difficult under our existing architecture.

There are two distinct paradigms for protecting networks. The classic method is to separate sensitive traffic from the public network. High Assurance circuits are physically encased or link encrypted to ensure that bandwidth is not shared with low assurance traffic and that no outside agent can gain access to sensitive traffic. Partitioning bandwidth in this way provides strong security at high cost, with a simple. It utilizes well-defined and well-understood methods of authentication, integrity protection, and key material distribution, with the primary difficulty that is does not scale well.

Alternatively, sensitive traffic can be carried on public networks, but encrypted at higher layers in the Open Systems Interconnect (OSI) standards protocol stack. At intermediate protocol layers some term this technique "tunneling", and it is widely implemented for medium assurance requirements, such as in Virtual Private Networks. There are applications where protecting privacy is best accomplished at the application layer, protection that can only be provided cryptographically. We have evolved to the point where secure electronic communication relies on the application of cryptography. In 1999, the Department of Defense, once the stronghold of reliance on link encryption, fielded their Electronic Key Management System, acknowledging that application of cryptography is exploding and that the emergence of application layer encryption is inevitable.

Moving protection up the OSI protocol stack results in key distribution and authentication challenges. In all cases, security protocols are used to apply cryptography to resolve these problems. It is not too strong to state that even perfect encryption could not provide security without effective protocols. Unfortunately, security protocols themselves have flaws and are known to be highly susceptible to subtle errors [23], [1], [AN94], [SVO94], [LOWE96], and many more.

Because of the move of encryption toward the application layer, there is an explosion in the growth of Public Key Infrastructure technology, where centralized or partially centralized services provide addresses and keys for participants desiring to establish secure channels with one another. These trusted services are worthy targets for intruders since successful intrusion would have wide-ranging impact. If a central trusted service can be compromised, it might be possible to use that service as an oracle, to compromise communications between any two participants using that service or to masquerade as any participant with virtually no trace. Such attacks are already manifesting [33], with little corresponding effort to address this threat in a dynamic, systematic way.

The security of the information provided by trusted services at the application layer is dependent on security protocols. Extensive work has been done to test [19] and verify [16] security protocols, and significant progress has been made in these areas. Nonetheless, no method provides complete, or even measurable, confidence in security protocols. In fact, based on the nature of security protocols and their environment, it may be impossible to accurately predict their performance through formal analysis or automated testing. In [15] the authors show how attacks can be constructed through interaction of two simultaneously executing protocols, even though both protocols are "secure" when run independently.

In this paper, we describe a mechanism for on-line security protocol analysis and response to attack for security services. This method allows real-time analysis of the security protocol environment to detect intrusions and to uncover flaws in protocols, both fundamental and in their implementation. We begin by giving the background work in security protocol verification and intrusion detection. The following sections describe and illustrate security protocol attack detection and define the architecture for our system. We close with a short summary.

Section 2. Security Protocol Verification

Security protocol analysis research to-date focuses on applying pseudo-software engineering techniques to formally verify that protocols are error-free [DY83], [19], [BAN88], [12], [SVO94], [20], [26], [LOWE96], [LOWE98], [30], [25], [21], [27], [2], [6] and many others. Previous work concentrates on analysis of a single protocol in a laboratory environment, considering only a modeled symbolic representation of protocol execution.

Security protocol analysis techniques have achieved significant success in detecting previously unknown attacks in a wide variety of protocols. Moreover, the research provides a broad understanding of the fundamental and incidental characteristics that tend to make protocols secure or insecure. Much research has targeted abstraction of such characteristics in a systematic way [AN94], [AnN94], [BIRD92], [BIRD93], [29].

Abadi and Needham provide principles for designing secure protocols [AN94]. Their principles include recognition of sequencing of cryptographic operations, placement of identity information in messages, and explicitness of message meaning. While these principles are incomplete [SYV96] they provide rules of thumb to construct signatures for activity that may be suspicious if detected in an operating security protocol environment.

In [29] Syverson gives a taxonomy of replay attacks on security protocols. We utilize this taxonomy to generate threat signatures that can be used to identify threat potential for increased situational awareness. We use this taxonomy to derive a detection strategy and to gather activity information and give an example of an attack on the state of the art security protocol intended to protect Internet traffic.

We also use protocol principles and attack taxonomies to identify behavior recognition and profile strategies. For example, by applying a principle given in [AN94], any protocol sequence that triggers a public key encryption by a principal followed by a request for a signature from that principal using the same public key would be recognized and flagged as suspect. In the same way, we will utilize protocol principles and attack taxonomy to assist us in structuring user and object profiles, e.g. from Syverson's replay taxonomy we know that we need to retain profiles on the number of sessions established between common users. The knowledge gained for previous security protocol research is vital to the success of this project.

Our approach gives a new perspective of security protocols, considering their operation in a production environment, with multiple copies of multiple different protocols executing concurrently. This is a realistic environment where our ability to precisely model detailed, fundamental characteristics with formal methods is limited.

Section 3. Intrusion Detection.

Intrusion Detection traditionally relies on one of two paradigms for analyzing target activity: Knowledge-based or behavior-based analysis. We mirror these techniques for security protocol analysis.

For knowledge-based detection, we define signatures of known attacks as sequences of activity traces. When the signature is detected, the system takes appropriate action based on the nature of the attack. Numerous examples of dangerous behavior in a security protocol environment are given in [37].

Behavior-based analysis relies on statistics or artificial intelligence techniques to establish profiles of user behavior and resource utilization for comparison against ongoing activity. We survey profile characteristics and statistical methods for behavior-based analysis of security protocol environments in [37].

Section 4. State-based Attack Recognition for the IKE Protocol

We devised a method for detecting attacks on security protocols in real time. In [36] we show how known protocol attacks can be recognized using state machines by detecting Lowe's attack on the Needham and Schroeder Public Key Protocol (NSPKP) [LOWE96]. It was later pointed out to us that this is a well-known flaw, so no one will be running this flawed protocol. We agree that, in a perfect world, this protocol would not be encountered in a production environment. However, we recognize that the most common security vulnerabilities in distributed systems are a result of old versions of software with known flaws, showing up in the production environment. We further suggest that since the NSPKP was published in 1978 and the flaw was not found until 1995, there may be a very large number of implementations of the protocol in existence.

There are other reasons why we may want to be able to detect known attacks. For example, because of the way Internet standards are developed, it may take a long time to revise standards when flaws are detected. We refer to [21] as a case in point. Dr. Meadows found two flaws in the Internet Key Exchange (IKE) protocol, but due to complexities in the IKE design process, these flaws largely remain part of the standard. Thus, while these flaws are known, they remain as vulnerabilities in the present edition of the IKE protocol suite. We now show ho w the attacks constructed by Meadows using the NRL Protocol Analyzer in [21] can be detected using Security Protocol Attack Detection.

The first protocol is given in Figure 1a, with a known attack given in Figure 1b. This attack is a Man In The Middle (MITM) attack, and no parallel session is necessary. The attacker simply intercepts the first message in a protocol session, replaces the originator's identifier field with the identifier of any valid principal that is not involved in the session, and forwards the message to its intended destination. We are able to construct a machine that recognizes this attack by observing the sequence of send and receive statements at valid principals.

The concurrent execution of security protocols by a group of princ ipals can be represented by a serialized, interleaved execution trace of the steps in each protocol session. This trace provides the environment for dynamic protocol analysis. We recognize that the IKE protocol has many signatures, but for the purpose of simplicity, we start by focusing on the source and destination of the messages. The signature that we will recognize is given in Figure 2.

We model the behavior of this trace with the state transition machine given in Figure 2. The machine represents a protocol signature by mapping the sender/receiver pair for a message into one of the possible segments of a trace. We assume valid participants are members of some population P. In our illustration, A, B, M $\in$ P. From the start state, the machine will transition into S_1 when a valid principal begins a session with another valid principal. When the recipient of the first message receives a message one from another principal, the machine transitions into

S_2. When the recipient of the second message sends a message to the originator of the first message, the machine transitions into S_3 and so on.

In the simplest case, where the steps are executed in exactly the sequence given in Figure 2, without any other activities in between, this model will detect an attempt by Mallory to defeat the IKE Protocol. Recognition of the execution of messages in this sequence does not *guarantee* that an attack has occurred. Fortunately, dynamic attack detection does not rely on guarantees.

The reader may notice that the attack trace of Figure 2 may be masked by legitimate (or malicious) traffic. In [37] we show how to reduce the number of false positives by utilizing protocol and step identifiers and by generating slightly more complex state transition machines. Further, this illustration does not exhaust the utility of information available to a dynamic protocol analysis mechanism. In the illustration given here, we do not consider any information passed as the payload of the message, and note that it is on payload information that traditional protocol verification base their entire analysis. While accessing that information in a dynamic environment will be more than a trivial challenge, its potential is evident.

There are several positive incidental qualities of this simple model. One may consider that transition through the table reflects the likelihood that an attack is under way. When the machine is in the *Start* state (there are no active messages that meet the format), there is no chance that an attack is under way. When the first recognized message is received, it is still unlikely that it is the first message of an attack sequence, but it is more likely than when no messages were being processed. A more significant jump in likelihood may occur when the second message meeting the format is detected. Similarly, as the machine reaches larger state numbers, the likelihood that an attack is underway increases. This monotonic property may be used to signal probability of attack to a system monitor, or may be combined with other sensors to give a network threat picture.

Section 5. A Security Protocol Analysis Architecture

In this section, we detail the architecture for the tool set that will analyze a security protocol environment. This environment will include tools to:

1. Generate security protocol knowledge-bases for known attacks and behavior profiles

2. Capture security protocol traffic in the target environment

3. Analyze security protocol activity against the knowledge base to determine if an intrusion is in progress and/or if there is a protocol flaw.

These tools are presently under development as components of the Secure Enclave Attack Detection System (SEADS). The following descriptions provide a high level view of the architecture and functionality. Detailed descriptions will be published once the system is fully functional.

5.1 Maintaining the Security Protocol Knowledge Base

Intrusion detection technology is based on comparison of ongoing activity to an existing knowledge base to determine if any ongoing activity is malicious. A key element of this architecture is the mechanism that establishes and maintains this knowledge base.

The security protocol knowledge base is comprised of two types of information corresponding to the two detection paradigms of misuse and anomaly detection. For misuse detection, the knowledge base contains signatures that represent attacks and suspicious activity. The signatures are represented as state transition diagrams, as we illustrated in the previous section. Signatures are gathered that are specific to each individual protocol and version executing in the environment.

Signatures are primarily gathered from three sources:

1. Known attacks identified in the literature
2. Attack taxonomies identified in the literature
3. Flaws and suspicious activity gathered during execution

The literature is filled with attacks on published protocols. Translating these attacks into state transition machines is a straightforward process as we have shown with the attacks on the IKE protocol and, earlier, on the NSPKP.

Unfortunately, the protocols that have been most heavily analyzed, are not the ones that will have the heaviest use, e.g. the "Wide Mouthed Frog" protocol of [1] has been extensively analyzed, but is not in widespread use. Protocols such as IKE and the Transport Layer Security (TLS) protocols are much longer and more complex than the protocols proposed in [23] and analyzed in [BAN88/9]. Internet protocols, Public Key Infrastructure Protocols, and protocols devised to protect financial transactions are published as protocol families, with separate protocols for generating certificates, requesting services, requesting connections, etc.

Not only are these protocols larger, but the family approach necessarily leads to less single-threaded execution; said another way, the protocols require branching (IFTHENELSE). A single IF statement can double the size of the search space, so only a dozen such branches can increase the state size by 2^{12}. Formal methods already suffer from state explosion, so the added complexity suggests that formal methods will be less likely to provide complete confidence in security protocols in the future.

There have been some efforts to analyze these current families of protocols [21], [40], [26], [34], [4]. The complexities of these protocol families is illustrated by Paulsen's effort at analyzing TLS. Paulsen only models the TLS handshake protocol, while there may well be interactions between the different TLS components that result in flaws or vulnerability. Further, Meadows did conduct analysis of the entire IKE protocol, but partitioned the analysis by component, with no integrated analysis, which is a dangerous proposition since the components share common keys.

[4] documents an end-to-end analysis of TLS using the CPAL-ES [39]. As reliance on these protocols increases in volume and economic impact, the analysis will intensify and additional flaws will be found. We will continue to utilize the results of this research as attack signatures in dynamic protocol analysis.

We also devise signatures for security protocol activities that are suspect, based on principles of secure protocols and on taxonomies of attacks. For example, we consider Principle #5 in [AN94], and infer that any protocol sequence that triggers a public key encryption by a principal followed by a request for a signature from that principal using the same public key would be recognized and flagged as suspect. This approach gives us insight that allows us to easily detect a risky security protocol situation and improve overall situational awareness.

Finally, we will gather signatures for attacks and suspect activity during operation of the attack detection system. As has been shown with Intrusion Detection Systems, suspect activity that we cannot otherwise predict will occur during normal execution. When these activities are found by the system operators they will be entered directly into the knowledge base.

5.2 Capturing Security Protocol Activity

Our paradigm for security protocol attack detection requires monitoring activity that occurs in a distributed environment. Accumulating distributed information is a hard problem. Our initial approach is to develop a central monitor that accumulates security protocol activity in the target environment. Principals cooperate with the monitor by sending a record of each security protocol transaction (send and receive) to the monitor. Thus, our model consists of a set of principals that intercommunicate with one another using security protocols over an insecure network. They communicate with the monitor via a secure channel. We illustrate this architecture in Figure 3.

Intuitively, the monitor tracks the actions of each participant in every protocol in the environment. Since the protocols are represented as state transition machines, the principal's actions must be in a form that allows the appropriate state transitions to occur. In the IKE protocol example of Figure 2, the only important information for activity records seems to be the message source and destination. In a practical system, much more information is necessary. For example, if multiple sessions are allowed, the monitor must be able to detect to which session each action belongs. We also anticipate that actions may take forms other than simply sending messages. Message receipt and value encryption are two activities that may be important to the intrusion detection mechanism.

In addition to individual known attacks we utilize attack taxonomies and protocol design principles us to gather the proper information to detect classes of attacks. As an elementary example of how we use attack taxonomies to allow us to gather the proper information, consider reflection attacks. Occurrence of a reflection attack is characterized by instances of two or more concurrent protocol sessions between two principals[1]. We use this fact to help develop a state model for recognizing this activity. In order to recognize multiple concurrent sessions between two principals, we need to gather information about each protocol session including:

a. Session originator

b. Other session participants

c. Other affected participants (e.g. if Alice initiates a protocol session with Sally to acquire Bob's public key from Sally, then Bob is an "other affected participant").

There are many strengths to the monitor model. From a research standpoint, monitors are well-understood, thus allowing us to focus our efforts on detecting intrusions. Additionally, it presents sufficient information to address virtually all classes of attacks, possibly excluding attacks that require intruder collaboration (this assumes that intruders won't play by the rule that requires them to notify the monitor of all activity).

There are several issues with monitors that must be addressed. Among them, the monitor:

1. Generates communication overhead
2. Cannot reliably accumulate intruder actions (intruders may not play by the rules)
3. Is vulnerable to attack itself

[1] We don't suggest here that occurrence of multiple concurrent sessions is a definite indicator of a reflective attack. We merely illustrate our detection concept by showing how we can gather information that is relative to a situation that may lead to a real attack.

Simple design decisions for the monitor minimize some concerns. For example, we reduce overhead by producing a minimal sized activity transaction that is compressed. We address reliability by encrypting activity transactions, and we require strong authentication to reduce the risk of spoofing.

We do not claim that the monitor environment is the only (or even the best) model for gathering security protocol activity information. Broadcast environments, distributed database technology, and data mining technology are other paradigms that may apply. Other research is ongoing to resolve the problem of gathering distributed information for intrusion detection [10] using mobile agents.

5.3 Detecting Intrusions

As we mention earlier, attack detection relies on recognition of characteristics of known attacks on cryptographic protocols as well as recognition of intuitively dangerous activity. The attacks are characterized by sequences of activity traces in the same way that virus scanning technologies match code patterns in files and that network intrusion detection systems [3] match packet type and sequences. In our case, the pattern of protocol sequences produces a signature for the known attack. Protocol traces that match these signatures are always suspect, and in some cases may be sufficient evidence to affect a protective or damage control response in and of themselves, with no corroboration necessary.

A host-based example of suspicious activity given in [3] is that any program that sets UID during execution should be flagged as a high risk. Another example of activity that is always suspect given by Denning [5] is a high rate of password failures by a user. For protocol signatures, we consider protocol patterns categorized as risky, e.g. in [AN94]. For example, consider any protocol sequence that triggers a public key encryption by a principal followed by a request for a signature from that principal using the same public key would be recognized and flagged as suspect. Principle 5 in [AN94] gives us insight that allows us to easily detect a risky security protocol situation and improve overall situational awareness.

Our early efforts use a state-based approach to attack detection. The knowledge base consists of tables representing state machines. The state machines are constructed from known attacks documented in the literature, principles in the literature, principles and rules of thumb that we devise, and user entered diagrams recognized by extensive observation of the system in use. We illustrated this in the IKE Protocol Suite example above. The signature of the attack is represented as a state transition diagram, as shown in Figure 2. The state reflects the progress of the protocol, e.g. how many messages have been successfully for a protocol session. Transitions occur as the direct result of protocol actions. The monitor gathers protocol actions, passes them to the detection engine where appropriate state machines are transitioned. When any state machine reaches a final state, an intrusion is detected.

Section 6. Conclusion.

We present a method to provide active defense for distributed security services. We show how proven intrusion detection technology combined with knowledge gained by formal analysis of security protocols can be applied to this problem. Our method involves addressing behavior relative to protocol activation and use rather than considering activities against objects, as is conducted in classic intrusion detection.

Until recently, the requirement for trusted services essentially resided with the federal government and a few large corporations, where key exchange was most often carried out by courier, with the key material stored on paper tape or diskette. Present technology demands extension of the protection provided by cryptography. This necessitates extension of key distribution and, thus, authentication services. These centralized services are attractive targets for sophisticated intruders. The method we prescribe offers to protect this vital link to our security infrastructure.

Author

Alec Yasinsac (Yasinsac@cs.fsu.edu) is an Assistant Professor in the Department of Computer Science at Florida State University, Tallahassee, FL 32306-4530, 850.644.6407.

References

[1] Burrows, M., Abadi, M., and Needham, R. M. "A Logic of Authentication", In *Proceedings of the Royal Society of London*, A 426:233-271, 1989

[2] S. Brackin, "Automatically Detecting Most Vulnerabilities in Cryptographic Protocols", in The DARPA Information Survivability Conference and Exposition, January 2000, Vol.1, pp 222-36

[3] Crosbie, M.; Dole, B.; Ellis, T.; Krsul, I.; Spafford, E, "IDIOT - Users Guide", Technical Report TR-96-050, Purdue University, COAST Laboratory, Sept. 1996

[4] Justin Childs and Alec Yasinsac, "Analyzing Internet Security Protocols", Submitted to 2001 IEEE Symposium on Security and Privacy

[5] Dorothy E. Denning, "An Intrusion-Detection Model", From 1986 IEEE Computer Society Symposium on Research in Security and Privacy, pp 118-31

[6] G. Denker and J. Millen, "CAPSL Integrated Protocol Environment", In *DARPA Information Survivability Conference* (DISCEX 2000), pp 207-221, IEEE Computer Society, 2000

[7] D. E. Denning and G. M. Sacco, "Timestamps in key distribution protocols," Communications of the ACM, vol. 24, no. 8, Aug 1981, pp. 533-536

[8] Daniels and Spafford, "Identification of Host Audit Data to Detect Attacks on Low-level IP", Journal of Computer Security, Volume 7, Issue 1, 1999

[9] Dolev, D., and Yao, A.C. "On the security of public key protocols". IEEE Trans. Inf. Theory IT-29, 2(Mar. 1983), pp. 198-208. Also Stan-CS-81-854, May 1981, Stanford U.

[10] Gregory, D.; Shi, Q.; Merabti, M., 'An Intrusion Detection System Based upon Autonomous Mobile Agents", pp. 586-591, 14th International conference on Information security, 1998 Aug : Vienna

[11] Y. Jou, F. Gong, C. Sargor, X. Wu, S. Wu, H. Chang, and F. Wang, "Design and Implementation of a Scalable Intrusion Detection System for the Protection of Network Infrastructure", DARPA Information Survivability Conference and Exposition 2000, Jan 25-27, 2000, Vol. 2, pp 69-83

[12] R. A. Kemmerer, "Using Formal Methods to Analyze Encryption Protocols," IEEE Journal on Selected Areas in Communications, vol. 7, mo. 4, pp. 448-457, May 1989

[13] Rajeshekar Kailar and Virgil D. Gligor, "On Belief Evolution in Authentication Protocols", In Proceedings of the Computer Security Foundations Workshop IV, PP 103-16, IEEE Computer Society Press, Los Alamitos, CA, 1991

[14] Sandeep Kumar and Eugene Spafford, "A Taxonomy of Common Computer Security Vulnerabilities Based on their Method of Detection", Technical Report, Purdue University, 1995

[15] J. Kelsey, B. Schneier, and D. Wagner, "Protocol Interactions and the Chosen Protocol Attack", Security Protocols, 5th, International Workshop April 1997, Proceedings, Springer-Verlag, 1998, pp. 91-104, http://www.counterpane.com/chosen_protocol.html

[16] R. Kemmerer, C. Meadows, and J. Millen, ""Three Systems for Cryptographic Protocol Analysis", The Journal of Cryptology, Vol. 7, no. 2, 1993

[17] Ulf Lindqvist and Phillip A. Porras, "Detecting Computer and Network Misuse Through the Production-Based Expert System Toolset (P-BEST)", 1999 IEEE Computer Society Symposium on Security and Privacy, pp 146-61

[18] R.P. Lippman, D.J. Fried, I.Graf, J.W. Haines, K.R. Kendall, D. McCllung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, M.A. Zissman, "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation", DARPA Information Survivability Conference and Exposition 2000, Jan 25-27, 2000, Vol. 2, pp 12-26

[19] Millen, J.K., Clark, S. C., and Freedman, S. B. "The interrogator: Protocol security analysis". IEEE Trans. Sofw. eng. SE-13, 2(Feb. 1987), pp. 274-288

[20] Catherine Meadows, "Formal Verification of Cryptographic Protocols: A Survey," Advances in Cryptology - Asiacrypt '94, LNSC 917, Springer-Verlag, 1995, pp. 133-150, http://www.itd.nrl.navy.mil/ITD/5540/publications/1995/1995meadows-asiacrypt94.ps

[21] Catherine Meadows, "Analysis of the Internet Key Exchange Protocol using the NRL Protocol Analyzer", 1999 IEEE Computer Society Symposium on Security and Privacy, pp 216-34, http://www.itd.nrl.navy.mil/ITD/5540/publications/CHACS/1999/1999meadows-IEEE99.pdf

[22] Catherine Meadows, "A Formal Framework and Evaluation Method for Network Denial of Service", 12th IEEE Computer Security Foundations Workshop, Jun 28-30, 1999, Mordano, Italy

[23] Roger M. Needham, Michael D. Schroeder, "Using Encryption for Authentication in Large Networks of Computers", Communications of the ACM December 1978 vol. 21 #12, pp. 993-999

[24] R. Oppliger. Security issues related to mobile code and agent-based systems. pp. 1165-1170. Computer Communications, Vol. 22, No. 12 (July 1999):

[25] Lawrence C. Paulson, "Proving Security Protocols Correct'", in IEEE Symposium on Logic in Computer Science, Trento, Italy (1999), pp 370-81, http://www.cl.cam.ac.uk/users/lcp/papers/Auth/lics.pdf

[26] Lawrence C. Paulson, "Inductive analysis of the Internet protocol TLS", *ACM Transactions on Computer and System Security* 2 3 (1999), 332-351

[26] A. W. Roscoe, "The Theory and Practice of Concurrency", Prentice Hall, 1997

[27] Dawn Xiaodong Song, "Athena: A New Efficient Automatic Checker for Security Protocol Analysis", 12th IEEE Computer Security Foundations Workshop, Jun 28-30, 99, Mordano, Italy

[28] T. Sander, C. Tschudin, "Protecting Mobile Agents against Malicious Hosts", Lecture Notes in Computer Science, Special Issue on Mobile Agents, Edited by G. Vigna, 1998

[29] Paul Syverson, ``A Taxonomy of Replay Attacks," Proceedings of the Computer Security Foundations Workshop VII, Franconia NH, 1994 IEEE CS Press (Los Alamitos, 1994)

[30] F. Thayer, J.C. Herzog, and J.D. Guttman, "Strand Spaces: Why is a Security Protocol Correct?" In Proceedings of 1998 IEEE Symposium on Security and Privacy, 1998

[31] Brett Tjaden, "A Method for Examining Cryptographic Protocols", University of Virginia Doctoral Dissertation, January 1997

[32] Vigna and Kemmerer, "NetSTAT: A Network-based Intrusion Detection System "Journal of Computer Security", Volume 7, Issue 1, 1999

[33] "Attacks on Encryption Code Raise Questions About Computer Vulnerability", Wayner, Peter, New York Times (01/05/00) P. C2

[34] Wagner, D. and Snyder, "Analysis of the SSL 3.0 Protocol", In D. Tygar, Editor, *USENIX Workshop on Electronic Commerce*, 1996, pp 29-40, USENIX Association

[35] Alec Yasinsac, "Evaluating Cryptographic Protocols:, Ph.D. Dissertation, University of Virginia, Jan 1996

[36] Alec Yasinsac, "Active Protection of Trusted Services", Florida State University, Tech Report #010100.

[37] Alec Yasinsac, "Dynamic Analysis of Security Protocols", to appear in the *Proceedings of the New Security Paradigms Workshop* 2000, September 18-22, 2000.

[38] Yasinsac, Alec; Wulf, William A, "Evaluating Cryptographic Protocols", University of Virginia Technical Report, CS-93-66, December 22, 1993, ftp://ftp.cs.virginia.edu/pub/techreports/CS-93-66.ps.Z

[39] Alec Yasinsac and Wm. A. Wulf, "A Framework for A Cryptographic Protocol Evaluation Workbench", Proceedings of the Fourth IEEE International High Assurance Systems Engineering Symposium (HASE99), Washington D.C., Nov. 1999, http://www.cs.fsu.edu/~yasinsac/framewk.pdf

[40] Zhou, J., "Fixing of security flaw in IKE protocols", *Electronics Letters*, Volume 35, Issue 13, 1999, Pages 1072-1073

The Internet Key Exchange Protocol

1. A -> B: HDR_1,SA_A,KE_A,N_A,ID_A

2. B ->A: HDR_2,SA_B,KE_B,N_B,ID_B,

   $K_B^{-1}\{prf(K_{AB},(KE_B,KE_A,CKY_B,CKY_A,ID_B))\}$

Figure 1a

Attack on the Internet Key Exchange Protocol

1. A -> B: HDR_1,SA_A,KE_A,N_A,ID_A

1'. M -> B: HDR_1,SA_A,KE_A,N_A,ID_M

2'. B ->M: $HDR_2,SA_B,KE_B,N_B,ID_B,$

$K_B^{-1}\{prf(K_{AB},(KE_B,KE_A,CKY_B,CKY_A,ID_B))\}$

Figure 1b

IKE Attack State Transition Table

	Signature/Transition	Old State	New State
1	**A -> B**	**Start**	**S_1**
2	**M -> B**	**S_1**	**S_2**
3	**B -> M**	**S_2**	**S_3**
4	**M -> A**	**S_3**	**S_4**
5	**A -> B**	**S_4**	**Beacon**

Figure 2

Secure Enclave Attack Detection System Model

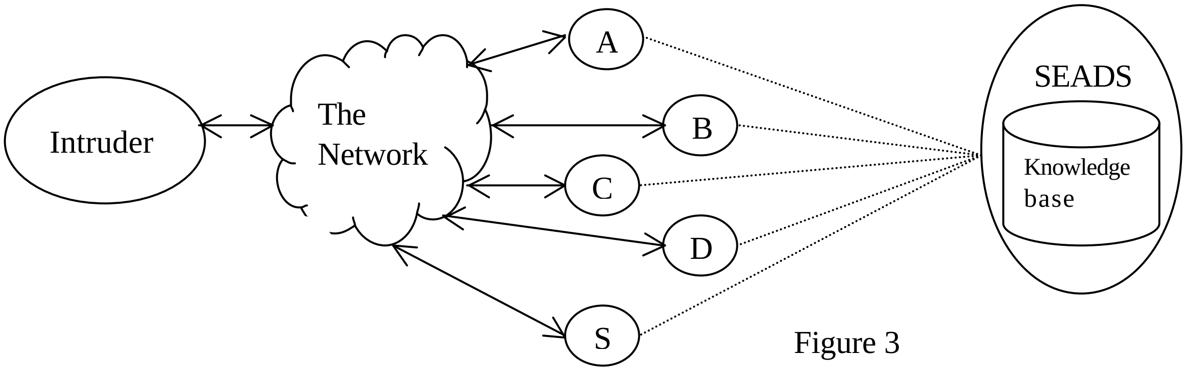

Figure 3