SECURE SYSTEM FOR SECURITY PROTOCOL IN TRAFFIC

Mr. Arjun Kumar
ABES Engineering College, Ghaziabad
Master of Computer Application

ABSTRACT

Today security is a very big problem in IT industry. Internet used Local area network and wide area network for communication and transaction purpose. The Internet has emerged as a medium for wide-scale electronic communication involving financial transactions and other sensitive information. In Transaction time data are encrypted mode for data security. Security protocols provide rules that govern such encrypted exchanges. In my paper describes a system for transaction time detecting intrusions on encrypted exchanges over public networks by recognizing the characteristics of security protocols and attacks on them.

INTRODUCTION

A Network is the inter-connection of communications media, connectivity equipment, and electronic devices for the purpose of sharing data and resources. Today is very challenging task of network security, because there are a variety of threats, there is no single solution. We need multiple layers of security, so that if one fails, the attacker would have to deal with the next layer. Network Security Systems are usually composed of many components of both software and hardware which are designed to work together to improve security and minimize maintenance. Our paper combines two common security technologies to provide protection for electronic information exchange over publicnetworks.

SECURITY PROTOCOLS

Sequence of operations that ensure protection of data. Used with a communications protocol, it provides secure delivery of data between two parties. Authenticates user identity. Authorizes access to specific resources based on permissions level and policies. Data encryption has long been used as a means of ensuring the security and integrity of data when transmitted over public networks. Algorithms such as DES, the International Data Encryption Algorithm and the Advanced Encryption Standard make use of keys to encrypt plain text messages before they are transmitted. However, even perfect encryption is not sufficient to prevent communication from being compromised. Encryption is implemented by rules (security protocols) that define and govern the interactions between the parties to encrypted sessions. Security protocols allow key exchange, authentication, and privacy through strong encryption. These protocols define the

content and order of exchanges between the communicating principals. Unfortunately, encryption backed by carefully crafted and thoroughly tested security protocols may still not be sufficient to prevent sophisticated intruders from compromising secure communication. So many paper published in journal and so many scientist work but till now data security is very challenging task. So my paper focuses on cryptographic and security protocol verification and principles for devising secure protocols. While progress has been made, the end is nowhere in sight, as evidenced by the pointed observation in [10]. It is clear that another level of protection must be provided for encrypted data exchanges to detect attacks on the securityprotocols.

ATTACK DETECTION SYSTEM

The Attack Detection System (ADS) [6] is a system that can detect attacks on security protocols within an enclave of valid and recognized parties that communicate using a public network. In this environment, security protocol activity based on the message exchanges between two systems so how transfer the secure message to another user.

A □ □ S : A, B, Na
S □ □ A: E(Kas: Na, B, Kab, E(Kbs : Kab, A))A
□ □ B: E(Kbs : Kab, A)
B □ □ A : E(Kab : Nb)
A □ □ B : E(Kab : Nb– 1)

Chart 1

1. **Characteristics of SecurityProtocol**

The our goal is any how detect hackers is to show that formal definitions of attacks on security protocols can be represented as signatures that can be stored in a knowledge base and compared against ongoing activity to detect attacks. This is done using specific characteristics of protocols.
When
A
B
S
Public
Network
ATTACK
Activity IDE
Monitor
Knowledge Base

Chart 2

Our system recognizes a specific signature of activity that corresponds to a known attack; we signal that an attack has occurred. Additionally, because of the characteristics of our system, we are also able to identify suspicious behavior that may or may not represent an attack.

2. 1CONSTRUCTING SIGNATURES OFATTACKS

An important feature of the our technique is that the detection mechanism does not rely upon knowledge of the payload of the messages exchanged between the principals during protocol sessions. This is because the IDE detects attacks based upon the characteristics of the security protocols themselves. The signatures constructed from protocols and their known attacks are represented by:

(1) Protocolused
(2) The principals (originator and recipient)involved
(3) Datasender
(4) Data received
(5) The concurrent sessions thatoccurs

Consider the canonical **Needham and Schroeder Conventional (symmetric)** *Key Protocol* (NSCKP) [5] shown in chart 1. This protocol requires three principals: A, B and the trusted third party server S. The aim of NSCKP is to establish a secret key *Kab*that is to be used by the principals A and B to encrypt their future exchanges. At the end of a correct run of the protocol, both principals should be in possession of the secret key, Kab, newly generated by the server S. The NSCKP can be represented by the signature given in chart 3 Protocol Session Message # Action SenderReceiver

NSCKP x 1 send A S
NSCKP x 1 receive A S
NSCKP x 2 send S A
NSCKP x 2 receive S A
NSCKP x 3 send A B
NSCKP x 3 receive AB

Chart 3

Each step of the signature is considered an *event*. Ram sending a message to Mohan is considered as a „send‟ event and similarly Mohan receiving a message from Ram is a „receive‟ event by Mohan from Ram. An important feature of protocol signatures is that they include receives events. Earlier research [6] took into account only the message sending events in the protocol signature. This means that Ram sending a message to Mohan (as in event 1), and correspondingly Mohan receiving the same message (event 2) will be represented as two distinct events in the protocol signature used by the IDE. Consider a scenario during the run of the NSCKP.UponsendingamessagetoMohanaspartofthefirststepoftheprotocol,Ramwill

inform the activity monitor of SEADS about this. Since a public network is being used for the message transfer between Ram and Mohan on insecure lines, the message may be lost or may be intercepted by an intruder. In either case Mohan will not inform the monitor that it actually received a message from Ram. Thus, the sequence of events logged in the monitor will show a message sent by Ram to Mohan, but not received by Mohan, as evident by the lack of the receive notification by Mohan to the monitor. Thus, suppose a previously distributed key Kab has been compromised, through cryptanalysis or other means, and is known by a malicious intruder, Mallory. If Mallory monitored and recorded message three of the corresponding protocol run, consisting of E(Kbs: Kab, A), he can now fool Bob into accepting the key as new by the protocol given in chart 4.

(3) *M(A) □ □B: E(Kbs : Kab,A)
(4) B □ □ M(A) : E(Kab :Nb)
(5) M(A) □ □ B : E(Kab : Nb–1)

Chart 4

After affecting the attack, Bob believes he is following the correct protocol. Mallory is able to form the correct response in (5) because she knows the compromised key Kab. She can now engage in a communication with Bob using the compromised key and masquerade as Ram. We can generate a signature recognizable by the IDE for the above attack on the Needham and Schroeder protocol. The signature is comprised of only three events; two receive events and a send event as shown in chart 5.

Protocol Session Message # Action SenderReceiver
NSCKP x 3 receive A B
NSCKP x 4 send B A
NSCKP x 5 receive AB

Chart 5

Since the malicious intruder (M), is not part of the secure enclave, it will not co-operate with the activity monitor and, hence, will not inform the monitor whenever it sends or receives messages. Thus the above attack signature will consist only of events reported by Bob (a valid principal) to the monitor.

THE RECOGNITION MACHINE

In previous section, we described in detail how the attack signatures are constructed from the description of security protocols. The IDE interfaces with the activity monitor to receive events corresponding to protocol sessions executing within the enclave and compares the events with the attack signatures stored in the knowledge base. The comparison mechanism in the IDE is achieved by using Finite StateMachines.

begin PROTNAME SIGNUM ATTACKTYPE
INITIALSTATE principal (□ /□) principal NEXTSTATE msgNumsessNum
PREVIOUSSTATE principal (□ /□) principal NEXTSTATE
msgNumsessNumend

Chart 6

Each time the IDE receives an originating event from the monitor (i.e. an event that corresponds to the first event of a new protocol session) the IDE constructs a finite state machine recognizer for each signature stored in the knowledge base for that particular protocol. These recognizers remain active until an event occurs that invalidates thesignature.

BASED ON KNOWLEDGE DEFINESIGNATURE

Each signature is stored in the Knowledge Base as a procedure defining a finite state machine. Information in the first line identifies the entry, followed by the state identifiers and the transitions that occur. Chart 6 is a symbolic representation of a signature.

CONSTRUCTION OF THE FINITE STATE MACHINE

When a session begins, the IDE constructs a Finite State Machine (FSM) recognizer for each signature stored in the knowledge base, corresponding to the protocol used in that session. The state transition diagram for attack signature #1 on the NSCKP protocol (as described in section 2.1) is shown in Table 1. Initially the recognizer will be in the start state (SS). As the IDE receives events from the monitor for this particular protocol session it advances the FSM for this signature if the arriving events match those in the attack signature. Upon a transition to the final state in any of the finite state machines corresponding to the attack signatures of the protocol, the IDE signals an attack notification.

Current
State
Event Protocol Session Sender Receiver Message
Number
Next
State
SS receive NCCKP X A B 3 S1
S1 send NCCKP X B A 4 S2
S2 receive NCCKP X **B A**5 **FS**

HOW DETECT ATTACKER

The IDE uses distinct detection methodologies for protocol attacks depending on the number of sessions used in each specific attack. Attacks on security protocols may be over only a single session of the protocol or may utilize information gleaned from multiple runs of the protocol. Thus, attacks may be classified as **Single session** attacks or **Multi-session** attacks.

SINGLE SESSION ATTACKS

Single session attacks are those attacks which may occur in a single session. The signature of such an attack may differ from the protocol itself in only something so subtle as a missing receive statement. In our environment, these subtle differences are easily recognized. Interestingly, we consider the attack on the Needham and Schroeder Conventional Key Protocol (NSCKP) a single session attack even though the attack depends on a previously compromised key from another session. The telling factor is that the attack can be detected by recognition of a single protocolsession.

MULTI-SESSION ATTACKS

Multi-session attacks are those attacks that use information extracted from more than one previous or concurrent protocol sessions. We make the reasonable assumption that such attack sessions must use the information within a certain time period of the reference session(s), from which the information is taken in order to subvert the protocol. For multi-session attacks, the IDE classifies them as either Replay Attacks or Parallel SessionAttacks.

PARALLEL SESSION ATTACKS

A parallel session attack occurs when two or more protocol runs are executed concurrently and messages from one run (the reference session) are used to form spoofed messages in another run (the attack session). As a simple example consider the following One-Way Authentication Protocol (OWAP) [1]:

A □ □ B : E(Kab : Na)

B □ □ A : E(Kab : Na + 1)

Successful execution should convince A that B is operational since only B could have formed the appropriate response to the challenge issued in the first message. An intruder can play the role of B both as responder and initiator. The attack works by starting another protocol run in response to the initial challenge. To initiate the attack, Mallory waits for Ram to initiate the first protocol session with Bob. Mallory intercepts the message and pretends to be Bob, starting the second run of the protocol by replaying the intercepted message. Ram replies to Mallory's challenge with exactly the value that Mallory requires to accurately complete the attack session. The attack is shown in chart 7.The IDE detects parallel session attacks by matching the ongoingactivity

against the attack signatures. The telling factor in this case is the omission of any information from Ram's partners in either session, as reflected in the signature in Table 2.

DESIGN OF THE INTRUSION DETECTION ENGINE

This section provides an insight into the design of the Intrusion Detection Engine. Justification of the major design decisions is also given. The design of the IDE uses the object oriented paradigm. The problem was broken down into smaller components, and appropriate classes were developed to accurately represent theproblem.

Current
State
Event Protocol Session Sender Receiver Message
Number
Next
State
SS send OWAP X A B 1 S1
S1 receive OWAP X+□ □BA 1
S2S2 send OWAP X+□ □AB 2 S3
S3 receive OWAP X B A2 **FS**

Table 2

Attack Session
A □ □ M(B): E(Kab : Na)
M(B) □ □ A: E(Kab : Na + 1)
Reference Session
M(B) □ □ A: E(Kab : Na)
A □ □ M(B): E(Kab :Na + 1)

Chart 7

A major factor in the design of the IDE, was the complexity of the environment being monitored. Within any enclave, we expect to monitor events interleaved from multiple:

□ □ □Concurrent sessions
□ □ □Different principals
□ □ □Different protocols

In addition there is no guarantee that all the sessions will properly conclude. Some sessions may be suspended abnormally and messages may be lost.

ARCHITECTURAL DESIGN

A number of issues had to be taken into account in the design phase of this research implementation. The design was created in order to ensure that all the requirements and specifications were satisfied. In the secure enclave it is possible to have multiple concurrent

sessions of different protocols executing within the enclave. The sessions may consist of the same or different principals. The Intrusion detection engine must be able to keep track of the different protocol sessions executing within the enclave in order to detect any attacks or suspicious activity. Not all attacks on security protocols occur over a single session. As described earlier, multi-session attacks such as replay attacks or parallel attacks may occur within the enclave. These multi-session attacks span multiple different protocol sessions. The Intrusion detection engine must provide a means to keep track of such executing sessions and detect any attacks. Additionally, the detection of attacks has to be communicated to the person or system monitoring the enclave. Detailed reports of all attacks or suspicious behavior must be generated by the IDE. Such reports provide in-depth information about the type of attack and principals participating in the protocol session. The Intrusion Detection Engine receives crucial inputs from the Activity Monitor and from the Knowledge base of protocol signatures. It is important to ensure that interfaces with the Monitor and the Knowledge base are well-defined andreliable.

THE GRAPHICAL USER INTERFACE

In our research, a Graphical User Interface (GUI) was implemented for an overall view of the attacks and suspicious activities detected within the enclave. The GUI allows the reporting of attacks to the user. The user can specify the time duration and the protocol name to obtain a detailed report of all the attacks (on the specific protocol) that took place during that period. The report will include the name of the protocol subject to attack, the principals involved in the session, attack time and other relevant information which will allow the user monitoring the system to research the occurrence of attacks within the system.
The GUI also allows the user to back up the active attack report file to another file.

TESTING

Upon completion of each significant milestone, the IDE was tested to ensure that the product functioned correctly. We approached the testing from four standpoints:
(1) Detection of attacks against protocols in all three categories of single session, replay, and parallelsession
(2) Detection of suspiciousactivity
(3) Effective operation in a highly concurrentenvironment
(4) Effective user interface.
FSMs…
New
Session?
Event
Match?
Attack
Activity

Wait for
Events
Create New
Thread
Channel the
event to the
relevant
monitoring
threadAdvance
FSM
Advance
FSM
Continue Monitoring for
attacks
Notify and write
to attack log file
Stop FSM
/Suspicious
No
Time out
Event: (B->A,NSCKP, session #, message #)
Yes
Yes

CONCLUSION

We have designed and implemented a Knowledge-Based Intrusion Detection Engine to detect attacks on security protocols executing within a secure enclave. This research provides an necessary extra level of protection for encrypted exchanges.

Extensive research on the characteristics of security protocols enabled this detection methodology to achieve its desired functionality. Extracting the description of security protocols into sequences of events allows the IDE to detect attacks on thoseprotocols.

The IDE will detect any attacks or suspicious activity on security protocols executed by valid principals operating within a secure enclave. The detection of the IDE compares protocol activity gathered by the Monitor against the attack signatures stored in the Knowledge base.

A Graphical User Interface (GUI) was also developed in order to facilitate an overall report of attacks that have been detected by the IDE, along with their occurrence times. Collectively, these components represent a fully functional Secure Enclave Attack Detection System.

REFERENCES

[1] John Clark & Jeremy Jacob, “Attacking Authentication Protocols”, High IntegritySystems 1(5):465-474, August1996.
[2] H.Debar, M.Dacier, A.Wespi, “Towards a Taxonomy of Intrusion DetectionSystems”, Elsevier Science B.V 31 (1999) 805-822
[3] Dorothy E. Denning, “An Intrusion-Detection Model”, From 1986 IEEE computer Society Symposium on Research in Security andPrivacy.
[4] Dorothy Denning and G.Sacco, “Timestamps in Key DistributionProtocols”, Communications of the ACM, 24(8), August 1981, pp.533-534.
[5] Roger M. Needham and Michael Schroeder, “Using Encryption for Authentication inLarge Networks of Computers”, Communications of the ACM, 21(12), Dec. 1978, pp.994-995.
[6] Alec Yasinsac, “Detecting Intrusions in Security Protocols”, Proceedings of FirstWorkshop on Intrusion Detection Systems, in the 7th ACM Conference on Computer and communications Security, June 2000, pp.5-8.
[7] National Bureau of Standards (NBS). Data Encryption Standard. DederalInfor-mation Processing Standard, Publication 46, NBS, Washington, D.C., January1977
[8] R.L Rivest, A. Shamir, L. M. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", CACM, Vol. 21, No. 2, Feb 1978, pp.120-126
[9] Otwy, D., and Rees, O. 'Efficient and timely mutual authentication'. OperatingSystems Review 21, 1(Jan. 1987), pp.8-10
[10] J. Kelsey, B. Schneier, & D. Wagner, "Protocol Interactions and the Chosen Protocol Attack", Sec Protocols, 5th, InternatWkshp Apr. 97, Proc. Springer-Verlag, 98, pp.91-104