SHOMAR: An Open Architecture for Distributed Intrusion Detection Services

Jeffrey Undercoffer, Filip Perich and Charles Nicholas
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250
{junder2, fperic1, nicholas}@cs.umbc.edu
phone: 410-455-3971
fax: 410-455-3969

Abstract

Distributed Intrusion Detection Systems (DIDS) offer an alternative to centralized intrusion detection. Current research indicates that a distributed intrusion detection paradigm may afford greater coverage, consequently providing an increase in security. In some cases, DIDS offer an alternative to centralized analysis, consequently improving scalabity. SHOMAR, the distributed architecture presented in this paper, provides an open framework that enables secure access to heterogeneous software and hardware components of a distributed intrusion detection system. SHOMAR is built upon a simplified Public Key Infrastructure that provides for authentication, non-repudiation, anti-playback, and access control. This framework supports a broad spectrum of approaches, ranging from hierarchical to peer-to-peer. The system topology and rules governing access to intrusion detection services is based solely upon policy, which is enforced through the use of a capability manager. The protoype system uses Java. The Extensible Markup Language is the sole medium for data exchange between intrusion detection components. SHOMAR provides a distributed service infrastructure independent of the underlying communications network.

1 Introduction

This paper describes the design and implementation of a framework to support distributed intrusion detection systems. The concepts described herein assume a system comprised of intrusion detection clusters, each operationally independent and strategically placed throughout an autonomous system. For need of an acronym this architecture is referred to as SHOMAR meaning "guard or protect" in the Hebrew language.

In [1], attention is called to the vulnerability of centralized Intrusion Detection Systems (IDSs) stating that they may become the subject of intrusion and subsequent compromise or fall victim

[*]Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County. Technical Report **TR-CS-02-14**

to a denial of service attack. Many academic IDSs [16], [12] and [20], as well as commercial IDSs, have a single point of failure represented by the monitoring station, the analytical station, the inference engine or a combination thereof.

The key motivation behind SHOMAR is to provide for a system of distributed intrusion detection clusters which are independent of each other yet collaborate to form a collective Intrusion Detection System.

Bass [3] introduces the notion of fusing data originating from heterogeneous devices as a means of providing "Cyberspace Situational Awareness" to detect and respond to intrusions. In [2] they introduce a system of autonomous agents, each tasked with identifying a specific type of intrusion. The Intrusion Detection Working Group of the Internet Engineering Task Force (IETF) have proposed the Intrusion Detection Message Exchange Format Data Model (IDMEF) and Extensible Markup Language (XML)Document Type Definition [4] to define data formats and exchange procedures for sharing information between intrusion detection systems, response systems, and their respective management systems. Likewise the IETF has proposed the Intrusion Detection Exchange Protocol (IDXP) [7], an application-layer protocol for exchanging data between intrusion detection entities.

SHOMAR provides an alternative to the IETF's IDXP. It provides a uniform infrastructure, a communications protocol and a security protocol for access to heterogeneous hardware and software components. Like the IETF's Intrusion Detection Message Exchange Format it uses a communications language based on XML. SHOMAR goes beyond the IETF initiative by using XML as the sole medium for data exchange of both text and binary data.

Our XML based language is called the Centaurus Capability Markup Language (CCML) [9]. CCML provides an extensible and simple content description transmission format. Unlike the IETF's IDMEF we do not describe a specific data model. This is not of any consequence, due to CCML's extensibility the IDMEF message may be easily wrapped within a CCML message and transmitted. Moreover, any data model may be encapsulated by, and transmitted within, a CCML message. The precise specification of event data specific to intrusion detection systems remains the subject of ongoing research[5] [13] [6]. Given the absence of consensus among researchers we designed our framework to accommodate any data model.

In [21] and [10] we developed Centaurus2 and Vigil respectively. Centaurus2 provides an infrastructure and communications protocol for service discovery, access, and delivery in pervasive computing environments. Centraurus2 assumed that all entities using the system were known in advance and access rights were static. In Vigil we extended Centaurus2 to allow for the delegation of rights to previously unknown, referred to as "foreign", entities. In Vigil a known entity could delegate rights to services to a foreign entity provided that the delegation was permitted by the security policy governing the system.

The concepts employed in Centaurus2 and Vigil are particularly applicable to a framework for distributed intrusion detection because they provide an infrastructure for secure collaboration between producers and consumers of services irrespective of the underlying communications protocol. In SHOMAR, intrusion detection (ID) probes, ID monitors and ID inference engines are considered to be an Intrusion Detection Service, deliverable to any subscriber of that service. The Centaurus2/Vigil architecture serves as the basis for the SHOMAR architecture, however, it was optimized for intrusion detection services.

The rest of this paper is organized as follows: Section 2 provides a synopsis of related research. Section 3 provides the design of the SHOMAR architecture. Section 4 details the communications protocol used by SHOMAR. Section 5 details our implementation to include the operational and security protocols. Section 6 presents an analysis of SHOMAR. We conclude with Section 7.

2 Related Work

According to [2], [17] and [20], research interest in Distributed Intrusion Detection Systems (DIDS) has increased because they afford greater coverage and security. Accordingly, they hold that centralized analysis severely limits the scalability of intrusion detection systems.

Purdue University's Center for Education and Research in Information Assurance and Security (CERIAS) produced the *Autonomous Agents for Intrusion Detection* (AAFID) [2] and [19]. The AAFID architecture consists of *agents, transceivers, and monitors*. In AAFID, agents perform some monitoring function, do not communicate with each other and report to a transceiver. The role of the transceiver is twofold: it tracks and controls the agents that report to it, processing and distributing their data, and it responds to commands issued by its monitor. Like transceivers, monitors have control and data processing roles, however, they differ from transceivers by having control over entities running on multiple hosts while transceivers only have control over entities running locally.

The AAFID architecture, which was prototyped using Perl, follows a strict hierarchical model, as well as a strict data model command structure. The implication is that if new agents implementing new functionalities were defined, AAFID would require modification to the underlying architecture.

In their paper describing the AAFID system, Spafford and Zamboni observe that a scripting language such as Perl is too resource intensive. Ultimately, the use of Perl lead to performance issues. They do no report any empirical or experimental results nor do the detail the manner in which the AAFID system was tested. The do state however, "that AAFID was to serve as an experimental testbed. As such it helped to identify questions that have not been completely answered by past intrusion detection research".

AAFID is in stark contrast to SHOMAR. SHOMAR makes no assumptions as to the command and control structure or the data model employed by the intrusion detection system. The SHOMAR architecture is highly versatile, in that it accommodates a system of systems, a singular hierarchical system, as well as architectures in between the two.

The GrIDS (Graph Based Intrusion Detection System for Large Networks) [20] developed at the University of California at Davis was designed to analyze network activity on TCP/IP networks. GrIDS models a network as a hierarchy of *departments and hosts* where hosts consist of a data source and a software controller and departments are collections of hosts and software manager and a graph engine. The graph engine receives input from data sources and builds graphs from the data. This graph is passed up to its parent graph engine. This process is repeated until the top graph engine is reached. Data sources are either network sniffers or an IDS that works on a single host. The software manager is responsible for managing the hierarchy and distributed modules. All GrIDS software components have a standard interface. The intention of GrIDS is that it be extensible, allowing any IDS to be dropped into it.

In GrIDS, all graphs propagate upwards. To prevent a graph from becoming intractable an entire department may be represented as a single node. In GrIDS, a centralized organizational hierarchy server maintains the global view of the entire hierarchy.

The SHOMAR model significantly differs from GrIDS because GrIDS requires centralized management of the hierarchy from the top level down. In GrIDS, careful attention must be paid to the administration tasks of adding hosts, moving departments (LAN segments), or changing the location of a graph engine so that any change happens atomically and the graph is left in a consistent state. In SHOMAR, administration (as will be detailed) is done by making changes to the Capability Matrix in the Capability Manager. SHOMAR, although supporting a hierarchical system of management does not require it. Moreover, SHOMAR facilitates the flow of information regarding intrusive behavior from any point to any point in the system while GrIDS assumes that information flows from the bottom of the hierarchy to the top.

In their paper documenting GrIDS, Staniford-Chen, et al. state that GrIDS was developed as a proof of concept for their approach to scalability and aggregation. They further state that their is considerable amount of work yet to be completed on GrIDS and that a prototype implementation is nearly complete.

EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) is an environment for anomaly and misuse detection [17]. EMERALD development was predicated upon several years of IDS experience at the Computer Science Laboratory at SRI International. EMERALD was designed upon a building block architecture in order to use independent and distributed monitors to detect and respond to malicious activity. These independent components inter-operate to form an analysis hierarchy. The basic architectural structure is comprised of three components; *profiler engines, signature engines, and resolvers*, which together form the EMERALD monitor. Profiler engines perform statistical anomaly detection, signature engines perform signature, or misuse, detection and resolvers respond to suspected misuse by deploying counter measures. An EMERALD monitor can stand alone or may be configured hierarchically to provide inter-domain interconnectivity.

According to [15] EMERALD is in-line with both the Common Intrusion Detection Format (CIDF) [11] and the IETF's Intrusion Detection Working Group's standardization effort [4] [7] to make IDSs inter-operable. Specifically, CIDF is an effort to develop protocols and application programming interfaces so that intrusion detection research projects can share information and resources and so that intrusion detection components can be reused in other systems, while the IETF effort is to define data formats and exchange procedures for sharing information of interest to intrusion detection and response systems, and to management systems which may need to interact with them. Accordingly, the IETF effort and the CIDF define differing data models and exchange protocols, hence it would be difficult for EMERALD to be simultaneoulsy "in-line" with both efforts.

SRI International has considerable research experience in the field of intrusion detection. EMERALD, their most current endeavor, is a culmination of that experience. As previously stated, EMERALD defines the EMERALD Monitor explicitly defining a three component framework (Signature Based Detector, Anomaly Detector, and a Response mechanism) and a target specific object library. At their last reporting [15], SRI was just beginning to experiment with event correlation within the context of EMERALD. In all of the literature describing EMERALD [15], [17] and its attendant sub-systems: eXpert BSM [14], P-BEST [**?**], Live Traffic Analysis [**?**], eBayes [22] and Probabilistic alert correlation [**?**], the authors have not detailed the results of experiments either quantitatively or qualitatively.

EMERALD and its attendant systems did participate in the DARPA Off-Line Intrusion Evaluations, where the highest average detection rate for the top performing system in each category was 64 %.

Like EMERALD, SHOMAR provides a framework for hierarchical integration. However, unlike EMERALD's strict hierarchical arrangement consisting of three component monitors, SHOMAR provides a peer-to-peer framework for ID services integration. Although the IETF and CIDF initiatives specify different data models, SHOMAR supports both because it makes no assumptions regarding the ID entities employing the architecture. Communicating entities, however, must be able to understand each other's data formats.

3 Design

SHOMAR is designed to control access to, and facilitate communications between, intrusion detection services within an autonomous system. Unlike any of the frameworks and architectures detailed in Chapter **??**, or Section **??** of this chapter, SHOMAR provides a secure framework (authentication, access control, anti-playback, and non-repudiation) for ID Systems. SHOMAR consists of four functional components: *SHOMAR Certificate Authority*, *Capability Manager*, *Intrusion Detection Managers* (IDS Manager), and *ID Services*.

Figure 1 illustrates the basic unit within SHOMAR, referred to as an *"ID Cluster"*. The ID Cluster is comprised of an IDS Manager and ID Services.

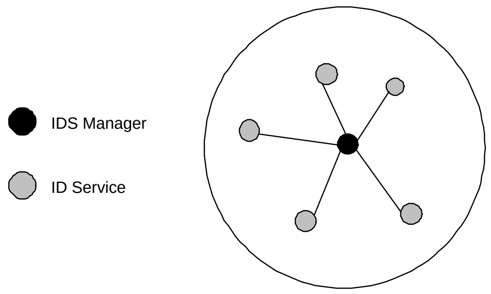

Figure 1. An ID Cluster - the basic unit within the SHOMAR Framework

The IDS Manager serves as the "cluster head" for each IDS cluster. The next component is an ID Service. In SHOMAR, the notion of an Intrusion Detection (ID) Service is abstract, examples of ID services include, but are not limited to: a packet sniffing service, a DDoS detector, a service that reads and reports on a host's system logs, an alarm station, a management station, etc.; no assumptions are made about the capabilities, inputs and outputs of each service. We assume that, in addition to being a producer of some service, an ID Service may also be a consumer of another ID Service's product, provided the producing service permits it. The protocol for access to services is detailed in Section 3.3.

SHOMAR assumes a logical hierarchy across an autonomous system. Therefore, the IDS Manager of a lower level ID Cluster may be an ID Service of a higher level ID Cluster. This hierarchical concept facilitates the aggregation of information across the autonomous system, creating end-to-end situational awareness. For example, each IDS Manager could aggregate the output of its immediately connected ID Services, forwarding information of consequence to other IDS Managers. Figure 2 illustrates the SHOMAR framework within the context of an Open Shortest Path First (OSPF) network.

To facilitate communications media independence, entities in SHOMAR are identified by handles instead of network addresses. An example handle is of the form *IDDevice1.cadip.cs.umbc.edu* or *IDSManager.cs.umbc.edu.*. Because of SHOMAR's logical hierarchy, the routing of messages follows a tree like structure where each level of the tree is representative of some logical division, for example: *cs.umbc.edu* is the parent of *cadip.cs.umbc.edu*.

The other two components of the SHOMAR framework are the Certificate Authority and the Capability Manager. The Certificate Authority is part of the lightweight PKI. The Certificate Authority generates and signs x.509 version 3 digital certificates for each entity in the system and responds to certificate validation queries from IDS Managers. The Capability Manager maintains an access matrix of an entity's rights, which are based upon group memberships, for all entities

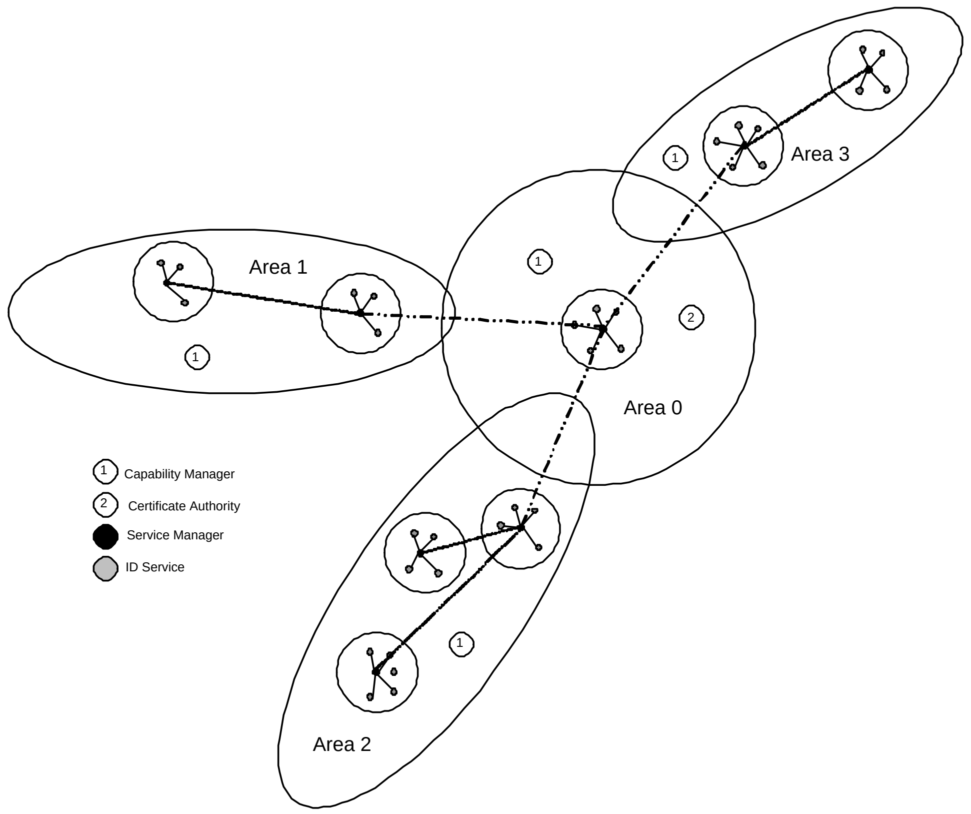

Figure 2. The SHOMAR Framework with an OSPF Network

within the system and responds to requests for group membership. The following subsections detail each component:

3.1 SHOMAR Certificate Authority

In SHOMAR, the Certificate Authority is used to produce x.509 version 3 digital certificates [8]. Here, certificate request and issuance is ancillary to the system. Although accomplished "out of band" when a certificate request from a SHOMAR entity is filled, the entity receives its requested x509 v3 certificate signed by the Certificate Authority and the Certificate Authority's self-signed certificate. The Certificate Authority's self signed certificate is subsequently used to validate other entities' certificates. In SHOMAR, certificates may be stored and protected on a smartcard or stored in a PKCS#11 container.

Certificate generation and signing is typically a one time occurrence for any entity within the SHOMAR System. This is in contrast to a typical PKI where the Certificate Authority makes its registrant's public certificates available in an on-line repository and provides an on-line Certificate Revocation List (CRL) where inclusion indicates that a given certificate is, for one of many possible reasons, invalid. As previously stated, SHOMAR uses a simplified PKI. In SHOMAR, each entity presents its certificate to its Capability Manager when it registers (the registration process is detailed in Section 5.3). Rather than use a CRL to signal a problem with an entity, the entity's entry in the Capability Manager is blocked, consequently preventing all access by that entity to the SHOMAR system. This precludes the necessity of maintaining a CRL, which must be signed by the Certificate Authority each time it is modified.

An IDS Manager verifies the authenticity of its stored copy of the Certificate Authority's certificate by sending the Certificate Authority a validation query. The Certificate Authority replies to the query with $E_{privatekey}((verifier))$. In SHOMAR, the Certificate Authority's certificate SHA-1 message digest is used as the verifier. To verify the validity of its copy of the Certificate Authority's certificate, the ID Manager tests if:

$$\text{(CA's certificate SHA-1 message digest)} = D_{publickey}(E_{privatekey}(\text{(verifier)})) \quad (1)$$

If the test passes, then the copy of Certificate Authority's private key is valid and any object signed by that key is also valid.

This Lightweight PKI, in contrast to the traditional PKI, does not maintain a CRL, does not transmit its key via the network and does not distribute user's certificates or public keys. Rather, the IDS Manager verifies that its copy of the Certificate Authority's certificate remains valid. This is done without transferring any keys or certificates via the network. In turn the IDS Manager will ensure that all certificates that it receives from clients have been signed by the Certificate Authority.

3.2 SHOMAR Capability Manager

The SHOMAR Capability Manager is responsible for maintaining and communicating group membership(s) of all entities in the SHOMAR system. Entities include IDS Managers and ID Services. Group membership may be as general as "*umbc.edu*" (meaning that only entities in the group umbc.edu are allowed access), more restrictive as "*cs.umbc.edu*", or even so granular as only the named Service "*ddos-service.cadip.cs.umbc.edu*", which implies that only the named Service is allowed access.

When the Capability Manager is initialized it reads its x.509 v3 digital certificate and its PKCS#11 [18] wrapped private key from a secure file and stores it into local memory. It also reads and indexes the capability file containing the group membership of all entities within the system, as well as storing the time stamp of the capability file.

When a group membership request is received from an IDS Manager, the SHOMAR Capability Manager compares the current time stamp on the capability file with the time stamp of the last file read, if they are not equal it re-reads the capability file. This feature allows for a dynamic administration, to include rights revocation, of the capability file.

In response to a group membership request, the SHOMAR Capability Manager sends a message containing the subject's group memberships. The response is digitally signed with the Capability Manager's private key.

Figure 3 shows a high level view of the Certificate Authority, Capability Manager(s), and IDS Managers. It should be noted that there could be multiple Capability Managers where one Capability Manager serves several IDS Managers. In the event of multiple Capability Managers each instance will replicate the capability database. During initialization, each ID Manager learns the location of its Capability Manager from its configuration file and communicates with that Capability Manager directly. At the first communication exchange between the IDS Manager and the Capability Manager, the IDS Manager requests and validates the Capability Manager's certificate, which it receives encoded in a signed CCML message.

3.3 IDS Manager

IDS Managers serve as the *"cluster head"* of their respective clusters and are responsible for ensuring security throughout the system. All intra-cluster communications pass through the IDS

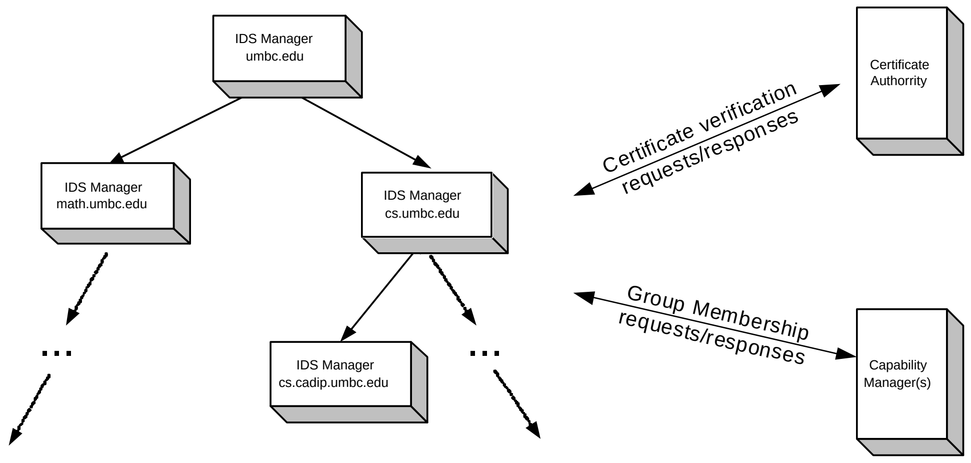

Figure 3. IDS Manager, Capability Manager, and Certificate Authority Overview

Manager who authenticates them on behalf of its attached ID Services.

Logically, IDS Managers are arranged in tree-like structure and form the core of the SHOMAR system. IDS Managers are identified by their "locations", or handles. With the exception of Group Membership requests to the Capability Manager and certificate validation requests to the Certificate Authority all messages are routed through the hierarchy of ID Managers using the handle to determine where to forward each message.

All ID Services rely upon the IDS Manager to broker requests for services between producers and consumers of ID services. Consequently, each ID Service is only concerned with the trust relationship with its immediately connected IDS Manager. In turn, IDS Managers establish trust relationships with their parent IDS Manager as well as other IDS Managers.

When an IDS Manager initializes, it reads its handle of the form: *servicename*, its parent's handle of the form: *IDSManager.cadip.cs.umbc.edu*, its Capability Manager's address, the Certificate Authority's address, and the handle of IDS Managers within its same level of hierarchy from a configuration file. Each IDS Manager starts with its own digital certificate and corresponding private key, and the digital certificate of the Certificate Authority.

Upon start up the following sequence of events occur:

1. The IDS Manager sends a certificate verification request to the Certificate Authority to ascertain that the local copy of the Certificate Authority's certificate is valid.

2. In response, the IDS Manager receives a signed certificate verification response from the Certificate Authority. The IDS Manager verifies the signature of the response according to Equation (1).

3. The IDS Manager sends a certificate request to its Capability Manager.

4. The IDS Manager receives a certificate response from the Capability Manager. It verifies that the certificate contained in the message was signed by the Certificate Authority and also verifies that the signature of the message is valid.

5. If the IDS Manager is not the top most IDS Manager of the domain $(handle \neq parent'shandle)$, register with the parent IDS Manager by sending a CCML registration message that contains a copy of the registering ID Manager's digital certificate that has been converted from its

ANS.1 encoding to Base64 string. The entire message is signed according to Equation (4). The digital signature is also converted to Base64 string and inserted into the CCML message.

6. The receiving IDS Manager extracts the certificate, verifies the signature and verifies the expression using Equation (6) to prevent replay attacks. Here Δ is the maximum round trip time of any message in the SHOMAR system, and is a configurable parameter. When all steps are successfully performed, the receiving IDS Manager registers the sending IDS Manager in its table of pending Services and sends a group membership request to the Capability Manager.

$$\text{Thumbprint}(\text{Certificate}) = D_{CA_{publickey}}(\text{SHA-1}(\text{Certificate})) \tag{2}$$

$$\text{SHA-1}(\text{CCML message}) \tag{3}$$

$$E_{privatekey}(\text{SHA-1}(\text{Original Registration Request CCML message})) \tag{4}$$

$$\begin{aligned} &\text{Message Signature} = D_{publickey} \\ &(E_{privatekey}(\text{SHA-1}(\text{Original Registration Request CCML message}))) \end{aligned} \tag{5}$$

$$\begin{aligned} &TimeStamp(\text{Original Registration Request CCML message}) + \\ &\Delta \cong TimeStamp(\text{ID Manager}) \end{aligned} \tag{6}$$

7. Once the group membership response is received from the Capability Manager the registration is changed from pending to registered and a digitally signed registration acknowledgment containing the parent IDS Manager's digital certificate is sent to the registrant.

8. Once a registration acknowledgment has been received the registering IDS Manager accepts communications from registrant.

9. IDS Managers at any level may register with IDS Managers at any other level of hierarchy. When an IDS Manager receives a CCML message destined for another IDS Manager it forwards the CCML message according to the rules detailed under Security Protocol (Section 5.2). subnet in a similar fashion.

The IDS Manager maintains a table of profiles for all entities registered with it. Information contained in the profile table includes the entity's certificate, group memberships, location (the IDS Manager to which it is immediately connected), name, and permissible access groups. Table 1 illustrates the contents of the profile table:

Element	Contents
handle	the name of the service
parent	the name of the IDS Manager to which it is connected
Address	network address
access groups	access groups to which the entity belongs
member groups	groups in which membership is required for access to the service
registered entities	a list of all entities registered with the particular service
crypto-storage	a class containing the entity's digital certificate, its private key (populated only for the key holder), and functions for cryptographic operations

Table 1. IDS Manager Profile Table

3.4 ID Service

As stated, an ID Service may be both a provider as well as a consumer of a service. All ID Services must register with its IDS Manager prior to accessing any services or making its services available. At registration, an ID Service will, in addition to sending its digital certificate, transmit a list of group memberships required for other ID Services to access its services. The IDS Manager will store the ID Service's certificate, list of memberships required for access to the service and the list of group memberships (acquired from the Capability Manager) for the ID Service in the IDS Manager's user profile table. Once the ID Service has registered with its IDS Manager its services are available to the IDS Manager as well as to any other Service connected to its IDS Manager, providing the other service has been given the appropriate rights.

Upon successful registration with an IDS Manager the ID Service also receives a list of all other Services to which it has access. The ID Service may elect to subscribe to another ID Service. When an ID Service subscribes with another ID Service The subscribing service receives an interface to the service to which it subscribed to. The interface is transmitted in a CCML message.

The following process enumerates the sequence of events during ID Service registration to an IDS Manager:

1. The ID Service sends a registration request message to its IDS Manager. The registration message includes a copy of the ID Service's digital certificate and a list of groups wherein membership allows access to the Service. This message is digitally signed by the Client.

2. Upon receipt of the registration request, the IDS Manager follows the same sequence of steps that a parent IDS Manager follows when registering one of its child IDS Managers.

3. The IDS Manager verifies the ID Service by querying the Capability Manager, verifying that the ID Service's certificate was signed by the Certificate Authority, and that the ID Service's registration request was signed by that specific ID Service. Once verified it sends the ID Service a Registration Response Message confirming registration. The Registration Response message contains a copy of the IDS Manager's digital certificate.

4. Upon receipt of the Registration Response the ID Service stores the IDS Manager's digital certificate which is used to validate all communications between the two entities.

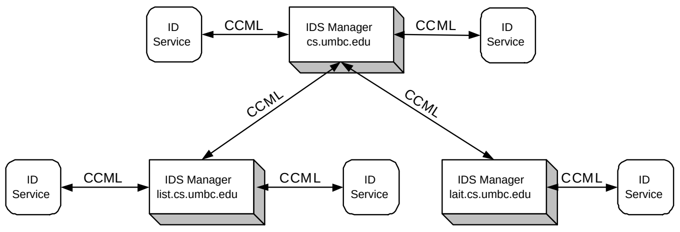

Figure 4. An example IDS Manager and ID Service configuration

Once an ID Service has successfully registered, it may request and receive via the IDS Manager an interface to all services to which it has rights. Recall that an ID Service is only provided with an interface to other ID Services registered with its IDS Manager and to which it has access rights. Figure 4 illustrates an IDS Manager at *cs.umbc.edu* having two registered ID Services and as well as having two other IDS Managers registered with it. Figure 5 depicts the scenario where the IDS Manager at *gvl.cs.umbc.edu* is registered as an ID Service to the IDS Manager located at *umbc.edu*. Although traffic between the two follows the path demarcated by the dashed line the authentication and verification process is only between those two entities and does not involve the intermediate IDS Managers.

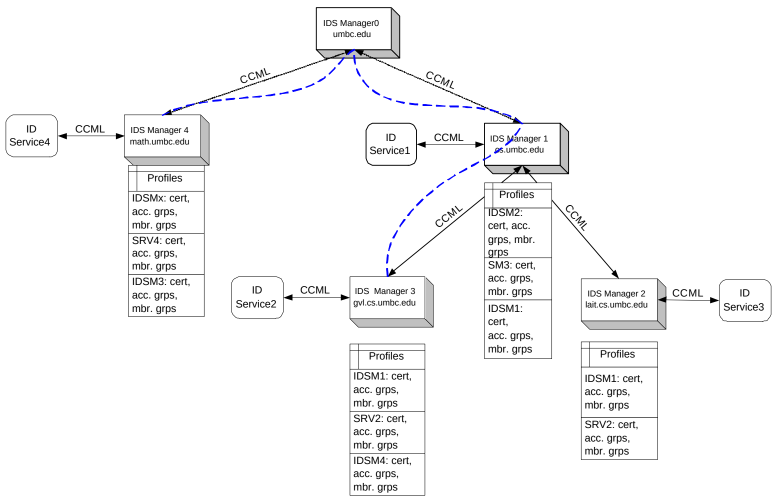
IDS Manager0 umbc.edu
CCML
CCML
ID Service4
CCML
IDS Manager 4 math.umbc.edu
Profiles
IDSMx: cert, acc. grps, mbr. grps
SRV4: cert, acc. grps, mbr. grps
IDSM3: cert, acc. grps, mbr. grps
ID Service1
CCML
IDS Manager 1 cs.umbc.edu
Profiles
IDSM2: cert, acc. grps, mbr. grps
SM3: cert, acc. grps, mbr. grps
IDSM1: cert, acc. grps, mbr. grps
CCML
CCML
ID Service2
CCML
IDS Manager 3 gvl.cs.umbc.edu
Profiles
IDSM1: cert, acc. grps, mbr. grps
SRV2: cert, acc. grps, mbr. grps
IDSM4: cert, acc. grps, mbr. grps
IDS Manager 2 lait.cs.umbc.edu
CCML
ID Service3
Profiles
IDSM1: cert, acc. grps, mbr. grps
SRV2: cert, acc. grps, mbr. grps

Figure 5. Example of Multiple ID Manager Registrations showing ID Manager Profiles

4 Entity Communication in SHOMAR

The information flow in this system is strictly in the form of CCML, which is built on top of XML. The CCML message defines a source, destination, destination location, message type (detailed in Section 5.3), data (a digital certificate, interface features, Service information, etc.), the IETF's IDMEF, and a digital signature.

Figure 6 shows the elements and attributes of a CCML message. Note there may be as many elements of type "attrib" as needed.

The use of a handle for addressing SHOMAR entities enable entity communication irrespective of the underlying communications medium. For example, an ID Service registered to the IDS Man-

```
<ccml version="2.0">
  <system>
    <source name=" " />
    <destination name=" " location=" " />
    <time></time>
    <message>’’message id’’ </message>
    <compression type="none" />
    <action type=" " />
  </system>
  <data>
    <attrib name="" type="CERTIFICATE" value=" " />
    <attrib name=" " type="VERIFIER" value="" />
    <attrib name=" " type=" " value=" " />
    <attrib name=" " type=" " value=" " />
  </data>
  <info>
    <IETF>’’IDMEF’’ </IETF>
    <Description> ’’IDMEF message’’</Description>
  </info>
  <dsig>  </dsig>
</ccml>
```

Figure 6. Elements and Attributes of a CCML Message

ager at LAIT.CS.UMBC.EDU wishing to subscribe to an ID Service registered at LAIT.CS.UMBC.EDU would send the request addressed as follows:

Source name:	IDS Manager_1
Source Location:	LAIT.CS.UMBC.EDU
Destination Name:	IDS Manager_0
Destination Location:	UMBC.EDU

A message addressed as shown is passed through the tree of IDS Managers according to a substring match until the destination location is reached.

When two ID Services communicate, the CCML message (message types are described in Section 5.3) is sent to the entity’s IDS Manager. In turn the IDS Manager verifies the signature, re-signs it, and forwards it to the recipient. Forwarding is based upon comparing the handle of the destination location to the handle of the present location. Messaging between ID Services stay with the the IDS Cluster, however, messaging between IDS Managers can propagate between levels of the system hierarchy.

Although we restrict ID Service communications to those registered to the same IDS Manager, this restriction is solely policy based. The SHOMAR architecture supports communications between any two entities regardless of their placement within the hierarchy.

As previously stated, when an ID Service registers with its IDS Manager, the service transmits a list of groups wherein membership in those groups implies that another service is permitted access to the service. Additionally, the IDS Manager requests a list of group memberships from the Capability Manager to ascertain the access rights of the registering service.

Although a service is only given an interface to services to which it is authorized, all requests to access services are re-verified to ensure that the service possesses the requisite rights. This validation for access rights is shown in Equation (7).

$$IDService.groupmembership \cap OtherIDService.accessgroups \tag{7}$$

5 Implementation

The goal of instantiating security, specifically: authentication, access control, non-repudiation, and anti-playback, as a primary component of SHOMAR, was heavily influenced by the desire to make security as unobtrusive as possible. I believe that the system design is both highly secure and that the security controls are nearly transparent. This task was accomplished through the use of the following tools and mechanisms:

1. A simplified Public Key Infrastructure.
2. X.509 version 3 Digital Certificate.s
3. PKCS #11 containers for private keys stored on computing devices and Smart Cards.
4. Capability Matrix.

SHOMAR was prototyped using Java. All IDS Managers are initialized from the same class. Communication with the network layer is provided by a single class *NetworkMessage.class*. For our prototype we assume TCP/IP network. SHOMAR will operate over different networking infrastructures by modify this class for the appropriate network type.

5.1 Simplified Public Key Infrastructure and X.509 version 3 Digital Certificates

Typically a Public Key Infrastructure consists of a Certification Authority (CA) to include Registration Authorities (RA), certificate holders, users that validate digital signatures and their certification paths from a known public key of a trusted CA, and repositories that store and make available certificates and Certificate Revocation Lists (CRLs).

Accordingly, the simplified Public Key Infrastructure consists of the Certificate Authority having its self-issued and self-signed digital certificate containing its public key. Each SHOMAR entity (IDS Manager, ID Service, and Capability Manager) is issued and possesses a digital certificate and private key as well as the digital certificate of the Certificate Authority.

Generally in a PKI system, certificates are made available in an on-line repository. Consequently, when a user needs an entity's digital certificate, it is requested from such a repository and assumed to be valid once it is received and has verified the certificate signing chain along the entire path to the top level signature authority. In a PKI implementation certificate repositories and CRLs have a high degree of administrative overhead. This overhead and the accompanying network traffic imposed by certificate acquisition and the signature verification is mitigated in SHOMAR by its simplified PKI framework. Rather then implementing certificate repositories and a Certificate Revocation List (CLR), each entity has its own certificate and a copy of the Certificate Authority's certificate. Upon start up, and optionally at configurable intervals, an IDS Manager verifies the Certificate Authority's certificate that it possesses. In turn this certificate is used to verify that each certificate presented to the IDS Manager has been signed by the Certificate Authority. In addition, instead of maintaining a CRL, the Capability Manager(s) has an entry for each valid user on the system. The absence of an entry in the Capability Manager's capability matrix for any entity blocks that entity from any and all accesses to the system.

Smart Cards are used for storage of the digital certificate and private key of some entities and PKCS #11 containers are used for storage of the private keys of other entities. At initialization, the entity, assumed to be a human acting on the entity's behalf, unlocks the card by entering a Card Holder Verification Value (CHV) gaining access to the card. The digital certificate is exported from the card and is available for presentation whenever the Client registers to a ID Manager. On those entities using a PKCS #11 container, an trusted operator enters the passphrase to unlock the container and read the private key into memory.

PKCS #11 describes syntax for private-key information. Private-key information includes a private key for some public-key algorithm and a set of attributes. The PKCS #11 standard also describes syntax for encrypted private keys. A password-based encryption algorithm, as described in PKCS #5, is used to encrypt the private-key information. The private key of IDS Managers, the Capability Manager, and ID Services that do not have smart cards, is stored in a PKCS #11 container in a regular File System. We assume that some trusted operator will enter the pass phrase on system initialization.

5.2 Security Protocol

The IDS Manager is responsible for ensuring the integrity of the SHOMAR system. As stated, each IDS Manager has a copy of the Certificate Authority's certificate. As will be explained below, an IDS Manager to which an ID Service is registered is responsible for authorizing CCML messages destined to the Service. The following describes the security protocol implemented in SHOMAR by the IDS Manager(s).

1. Upon receiving an ID Service certificate, verify the certificate by ensuring that the certificate was digitally signed by the Certificate Authority's private key. When signing a message, compute the signature using Equation (4), convert it to Base64 string and insert it into the message. When verifying the message compare the message digest from Equation (3) with the decrypted signature from Equation (4), and verify the timestamp using Equation (6).

2. If it is the initial registration the registrant generates a Registration Request that includes a copy of the registrant's digital certificate. The certificate is converted from ANS.1 to hexadecimal string and inserted into the registration message. The message is then signed by the sender. Upon receipt of the Registration Request, one of the following five cases will hold and the IDS Manager will respond accordingly.

   (a) If the message is a registration request and is from a child IDS Manager. The parent IDS Manager verifies the certificate and the message signature, and requests the group membership from the Capability Manager. Once everything is verified, it establishes a Service profile for the child IDS Manager. The profile contains the registrant's digital certificate, in ANS.1 encoding, access groups, member groups, name, and location. It then transmits a registration response message containing the registering ID Manager's digital certificate. In turn the child IDS Manager follows the same steps to verify its parent.

   (b) If the IDS Manager is both the source and destination IDS Manager, then the originator of the message is one of its immediately connected services. The IDS Manager verifies the certificate and the message's signature and requests the registrant's group membership from the Capability Manager. Once everything is verified, it establishes a Service profile storing the profile in its user database and transmits a registration response message containing its digital certificate.

    (c) If the IDS Manager is the source IDS Manager but is not the destination IDS Manager, this indicates that the registering Service has already registered with its IDS Manager. The source IDS Manager verifies the digital signature of the message, places its certificate into the message, re-signs the message, and forwards it to the destination IDS Manager.

    (d) If the IDS Manager is neither the source nor the destination IDS Manager, it forwards the message to either its parent or one of its children based upon a substring match of the destination handle and its handle.

    (e) If the IDS Manager is the destination IDS Manager and is not the source IDS Manager, it verifies the certificate to ensure it was signed by the Certificate Authority, verifies the signature of the message, and requests the registrant's group membership from the Capability Manager. Once everything is verified, it adds the registrant's profile to its user database, and sends a registration response to the registrant containing its digital certificate. When the registrant's IDS Manager receives the registration response it adds the sending IDS Manager to its Service database, resigns the registration response and forwards the response to the Service.

3. Upon receipt of a message, the IDS Manager will verify the signature contained in the message, ensuring that it was signed by sender and will verify that the sender has the appropriate rights with the destination. If the signature and rights are valid it will resign the message and forward it to the destination, either up, down, or laterally based upon the destination address.

5.3 Operational Protocol

The following enumerates SHOMAR message types, the initiator, the receiver and their resultant action:

1. *Registration Request* From: Service To: IDS Manager. Contains the registrant's digital certificate and is signed by the registrant. If the registrant is already registered, it is first de-registered and access to all previously subscribed services are terminated and the registrant is re-registered.

2. *Registration Response* From: IDS Manager To: ID Service. Signed by the sender. Transmitted once the group memberships are received from the Capability Manager.

3. *De-Registration* Request From: ID Service To: IDS Manager. Signed by sender. The senders profile is deleted from the IDS Manager, and all subscriptions are removed from ID Service.

4. *De-Registration* Response From: IDS Manager To: ID Service. Signed by the sender, it is assumed that the destination does not receive this message.

5. *Group Membership Request* From: IDS Manager To: Capability Manager. Sent to the Capability Manager from a IDS Manager requesting the group memberships of some entity.

6. *Group Membership Response* From: Capability Manager To: IDS Manager. Signed by the Capability Manager. Contains group memberships of the subject.

7. *Service List Request* From: ID Service To: IDS Manager. Signed by the sending ID Service requesting a listing of available services.

8. *Service List Response* From: IDS Manager To: ID Service. Signed by the sender, informs the Service of available services registered with that IDS Manager.

9. *Subscription Request* From: ID Service To: IDS Manager. Signed by the sending Service requesting subscription to a particular service. In addition to verifying the signature the IDS Manager verifies that the user has access rights to the requested service.

10. *Subscription Response* From: IDS Manager To: ID Service. If the user has access rights to the requested service a subscription response is signed and sent to the Service.

11. *Command* From: ID Service To: ID Service. A request to some Service to change state. Signed by the user and received by the services IDS Manager where both the signature and the user's access rights to the service are verified. Re-signed by the IDS Manager and transmitted to the Service for final arbitration.

12. *Update* From: ID Service (service) To: ID Service (consumer). Signed by the Service and sent to the user via the IDS Manager. The IDS Manager verifies the signature, re-signs the message and forwards it to the User. Additionally, the IDS Manager sends an update to all other users subscribed to the Service.

6 Analysis

We have conducted an analysis of both the operational and security protocols employed by SHOMAR. In analyzing the security protocol we identified the critical steps and any associated vulnerabilities. The operational protocol analysis was conducted on the basis of time and space complexity.

Security Protocol

The SHOMAR security protocol employs digital certificates, digital signatures, and public key cryptography. The Certificate Authority's (CA's) public key, contained in the CA's digital certificate, is used to authenticate the digital certificate of each entity using the system. In turn, the public key of each entity is used to authenticate communications from that entity. It is assumed that the process of generating digital certificates is secure. Each entity starts with the CA's digital certificate and their own digital certificate which were distributed "out of band". No private keys are exchanged during the course of business.

The critical steps of the security protocol and their implications follow:

1. Verifying the authenticity of the copy of the CA's digital certificate stored with an entity.

   If the CA is compromised and the root certificate is altered, or if the copy of the CA's certificate stored at the entity is altered, the authentication check as defined in Equation 1, will fail. This will result in the entity ceasing to function because it has no way of authenticating communications. This constitutes a denial of service and will be readily apparent to and corrected by the system administrator.

2. At system initialization, an ID Manager obtains a copy of the Capability Manager's digital certificate, verifying the authenticity of that certificate.

   If the Capability Manager is compromised and its digital certificate is altered then the certificate verification process of Equation 2, will fail and the ID Manager will not trust any communications from the Capability Manager.

3. Obtain an entity's access permissions from the Capability Manager.

   If the communication containing an entity's access rights is intercepted and altered the verifications according to Equations 4 and 6, will fail and the message will be discarded.

4. During the registration process when an entity sends a copy of its digital certificate to the service manager. The authenticity of the entity and its digital certificate are verified according to Equation (2).

   If the registering entity has presented a counterfeit certificcate, or an altered or stolen certificate (having intercepted a valid digital certificate), the entity will not have a valid private key to sign the registration request as per Equation 4, consequently verification of that message using Equation 5, will fail.

There is some possibility of denial of service type attacks resulting from SHOMAR operating as specified. Specifically, resulting from the compromise of a SHOMAR entity's digital certificate. Otherwise, SHOMAR is secure from fictitious or fraudulent entities.

In the current implementation, communications between SHOMAR entities are not encrypted. However, if needed, this feature could be easily added. Table 1 depicts profile information stored by SHOMAR entities. ID Services are only concerned with a one-to-one relationship with IDS Managers, consequently an ID Service only stores its own profile and that of its IDS Manager. IDS Managers store profiles for every entity to which it is connected. Recall that this profile stores a class called *crypto-storage*. To add message encryption, the registration process would be extended to include the selection of a session key generated by the IDS Manager and transmitted to the ID Service encrypted under the ID Service's public key. The *crypto-storage* class would be modified to contain the shared session key to be used with subsequent communications between the entities as well as some agreed upon symmetric encryption algorithm.

Operational Protocol

SHOMAR has not been quantitatively tested, it has, however, been qualitatively tested for *usability, functionality,* and *scalability* by placing it into service as a *real-time* infrastructure in support of controller type services. Recall that SHOMAR supports any data model, provided that the sender and receiver agree on the model. SHOMAR functions in support of light controller services, audio controller services, temperature controller services, etc.

The implemented framework is configured in a hierarchy approximating the Computer Science and Electrical Engineering Department at the University of Maryland Baltimore County. We have configured our framework to enable trusted communications between approximately 150 entities in a heterogeneous environment consisting of desktop computers, laptop computers and Personal Digital Assistants (PDS) using both a wireless and wired network. In this configuration our framework functions without any performance degradation.

Message size in SHOMAR varies according to content, however, all messages contain a 128 byte digital signature Base64 encoded as a 172 byte string. Registration and Registration Response messages include the 1255 to 1291 byte digital certificate Base64 encoded as an ≈ 1700 byte string. Note: the Base64 encoding of binary data results in an expansion of the ratio 3:4.

Messaging complexity for ID Service to ID Service Communications are:

$$(2 * \mid number\ of\ messages \mid)$$

because the IDS Manager receives, verifies and forwards the message from source to destination. The best case messaging complexity for IDS Manager to IDS Manager is 1 where the source and destination are one logical level apart. The worse case messaging complexity from IDS Manager to IDS Manager is given as follows:

Let *N* = the number of IDS Managers.
Let *H* = the height of the logical IDS Manager tree.
Let *M* = Messaging Complexity.
Assume that the branching factor of the IDS Manager tree is ≥ 2.

$M = 2 * H$ where $H \leq \lceil log_2 N \rceil$

In any security system, a single point of failure represents a serious concern. In many IDSs, some enterprise monitor, alarm station, management station or analytical unit represent a single point of failure. In SHOMAR, the Certificate Authority and the Capability Manager are singular units. However, once an IDS Manager and its immediately connected ID Services start, the need for the Certificate Authority or the Capability Manager is minimized because they are only used to authenticate an entity and convey that entity's rights upon system initialization.

7 Conclusion

SHOMAR provides an open framework for distributed intrusion detection services through a system of IDS Managers that enforce security (access control, authentication, non-repudiation and anti-playback). SHOMAR enables service integration and aggregation across an autonomous system while making no assumptions as to the nature of intrusion detection services or the data model used by those services. SHOMAR exclusively uses XML for data exchange between IDS components. SHOMAR utilizes a simplified public key infrastructure to minimize run time key and certificate administration while at the same time providing a high degree of assurance to the intrusion detection entities.

References

[1] Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel, and Ed Stoner. State of the Practice of Intrusion Detection Technologies. Technical Report 99tr028, Carnegie Mellon - Software Engineering Institute, 2000.

[2] Jai Sundar Balasubramaniyan, Jose Omar Farcia-Fernandez, David Isacoff, Eugene Spafford, and Diego Zamboni. An Architecture for Intrusion Detection using Autonomous Agents. Technical report, Cerias Purdue, November 2001.

[3] Tim Bass. Intrusion Detection Systems and Multisensor Data Fusion. In *Communications of the ACM*, volume 43, pages 99–105, April 2000.

[4] D. Curry and H. Debar. Intrusion detection message exchange format data model and extensible markup language (xml)document type definition. draft-ietf-idwg-idmef-xml-06.txt, December 2001. expires June 27, 2002.

[5] Jon Doyle, Isaac Kohane, William Long, Howard Shrrobe, and Peter Szolovits. Event Recognition Byyond Signature and Anomaly. In *2nd IEEE-SMC Information Assurance Workshop*, June 2001.

[6] Rich Feiertag, Cliff Kahn, Phil Porras, Dan Schackenberg, Stuart Staniford-Chen, and Brian Tung. A Common Intrusion Specification Language, June 1999. http://www.isi.edu/ brian/cidf/drafts/language.txt.

[7] B. Feinstein, G. Matthews, and J. White. The Intrusion Detection Exchange Protocol (IDXP). draft-ietf-idwg-beep-idxp-04, January 2002. expires July 9, 2002.

[8] R. Housley, W. Ford, W. Polk, and D. Solo. RFC 2459 Internet X.509 Public Key Infrastructure Certificate and CRL Profile, Janaury 1999.

[9] Lalana Kagal, Vladimir Korolev, Sasikanth Avancha, Anupam Joshi, Timothy Finin, and Yelena Yesha. Highly Adaptable Infrastructure for Service Discovery and Management in Ubiquitous Computing. Technical Report TR CS-01-06, Department of Computer Science and Electrical Engineering. University of Maryland Baltimore County, Baltimore, MD, 2001.

[10] Lalana Kagal, Jeffrey Undercoffer, Anupam Joshi, and Tim Finin. Vigil: Enforcing Security in Ubiquitous Environments . In *Grace Hooper Celebration of Women in Computing 2002*, 2002.

[11] Cliff Kahn, Dan Bolinger, and Don Schackenberg. Communication in the Common Intrusion Detection Framework v 0.7, June 1998. available at: http://www.isi.edu/ brian/cidf/drafts/communication.txt.

[12] Wenke Lee and Salvatore Stolfo. A Framework for Constructing Features and Models for Intrusion Detection Systems. *ACM Transactions on Information and System Security*, 3(4):227 – 261, November 2000.

[13] Ulf Lindqvist and Phillip A. Porras. Detecting Computer and Network Misuse through the Production-based Expert System Toolset (P-BEST). In *IEEE Symposium on Security and Privacy*, pages 146–161, 1999.

[14] Ulf Lindqvist and Phillip A Porras. eXpert-BSM: A Host-based Intrusion Detection Solution for Sun Solaris. In *Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC 2001)*, pages 240–251, New Orleans, Louisiana, December 10-14 2001. IEEE Computer Society.

[15] Peter G. Neumann and Phillip A. Porras. Experience with Emerald to Date. In *First USENIX Workshop on Intrusion Detection and Network Monitoring*, April 1999.

[16] Vern Paxson. Bro: A system for Detecting Network Intruders in Real Time. In *Proceedings of the 7th Symposium on USENIX Security*, 1998.

[17] Phillip A. Porras and Peter G. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *1997 National Information Systems Security Conference*, oct 1997.

[18] RSA Laboratories. PKCS 11-Cryptographic Token Interface Standard, January 1994.

[19] Eugene H. Spafford and Diego Zamboni. Intrusion detection using autonomous agents. *Elsevier Computer Networks*, 34:547 – 570, 2000.

[20] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. GrIDS-A Graph Based Intrusion Detection System for Large Networks. In *Proceedings of the 19th National Information System Security Conference*, 1996.

[21] Jeffrey Undercoffer, Filip Perich, Andrej Cedilnik, Lalana Kagal, and Anupam Joshi. Centarusu2: A Secure Infrastructure for Service Discovery and Delivery in Pervasive Computing. *MONET: Special Issue on Security*, 2002.

[22] Alfonso Valdes and Keith Skinner. Adaptive, Model-based Monitoring for Cyber Attack Detection. In H. Debar, L. Me, and F. Wu, editors, *Recent Advances in Intrusion Detection (RAID 2000)*, number 1907 in Lecture Notes in Computer Science, pages 80–92, Toulouse, France, October 2000. Springer-Verlag.