The DIDS (Distributed Intrusion Detection System) Prototype

Steven R. Snapp, Stephen E. Smaha – Haystack Laboratories, Inc.
Daniel M. Teal, Tim Grance – United States Air Force Cryptologic Support Center

ABSTRACT

Intrusion detection is the problem of identifying unauthorized use, misuse, and abuse of computer systems by both system insiders and external penetrators. The growth in numbers and complexity of heterogeneous computer networks provides additional implications for the intrusion detection problem. In particular, the increased connectivity of computer systems gives greater access to outsiders, and makes it easier for intruders to avoid detection. We are designing and implementing a prototype Distributed Intrusion Detection System (DIDS) that combines distributed monitoring and data reduction (through individual Host and LAN Monitors) with centralized data analysis (through the DIDS Director) in order to monitor a heterogeneous network of computers. This approach is unique among current intrusion detection systems. One of the problems considered in this paper is the Network-user Identification (NID) problem, which is concerned with tracking a user moving across the network, possibly with a new user-id on each computer. Initial system prototypes have provided quite favorable results on both the NID problem and the detection of other attacks on a network. This paper provides an overview of the motivation behind DIDS, the system architecture and capabilities, and a discussion about the implementation of the system prototype.

Introduction

Intrusion detection is the problem of identifying individuals who are using a computer system without authorization (i.e., the *external threat*) and those who have legitimate access to the system but are exceeding and/or abusing their privileges (i.e., the *insider threat*). Work is being done in parallel on Intrusion Detection Systems (IDS's) to monitor a single host [9,12,8], several hosts connected by a network [7,6,13], and a broadcast Local Area Network (LAN) [3,4].

The large numbers and complexity of heterogeneous computer networks has serious implications for the intrusion detection problem. Foremost among these implications is the increased opportunity for unauthorized access via the network's connectivity. This problem is intensified when dial-up or internetwork access is allowed, as well as when unmonitored hosts (viz. hosts without audit trails) are present on the network. The use of distributed rather than centralized computing resources also implies reduced control over those resources. Moreover, multiple independent computers generate more audit data than a single computer, and this audit data is dispersed among the various systems. Clearly, not all of the audit data can be forwarded to a single IDS for analysis; some analysis must be accomplished at the local host.

This paper describes a prototype Distributed Intrusion Detection System (DIDS) which generalizes the target environment in order to monitor multiple hosts connected via a network and the network itself. The DIDS components include the DIDS Director, a single Host Monitor per host, and a single LAN Monitor for each LAN segment of the monitored network. Information is gathered and processed locally by each distributed component, with important events and information transported to, and analyzed at, a central location (viz. an Expert System, which is a sub-component of the Director). This architecture provides the capability to aggregate information from numerous different sources. The system is designed to work with any audit trail format as long as certain pieces of critical information are provided by the auditing mechanism.

DIDS is designed to operate in a heterogeneous environment composed of C2 [1] or higher rated computers. The DoD Class C2 (Controlled Access Protection) rating enforces a finely grained discretionary access control that makes users individually accountable for their actions through login procedures, auditing of security-relevant events, and resource isolation. The target environment consists of several hosts connected by a single broadcast LAN segment (presently an Ethernet, see Figure 1). The use of C2-rated systems implies a consistency in the content of the system audit trails. This allows us to develop standard representations into which we can map audit data from UNIX, VMS, or any other system with C2 auditing capabilities. Some abstraction is performed on the raw audit data in order to transform the data into the standard representation. The C2 rating also provides, as part of the Trusted

Computing Base (TCB), the security and integrity of the host's audit records. Although the hosts must comply with the C2 specifications in order to be monitored directly, the network related activity of non-compliant hosts can be monitored via the LAN Monitor. Since all attacks that utilize the network for system access will pass through the monitored segment, the LAN Monitor will be able to analyze all of this traffic.

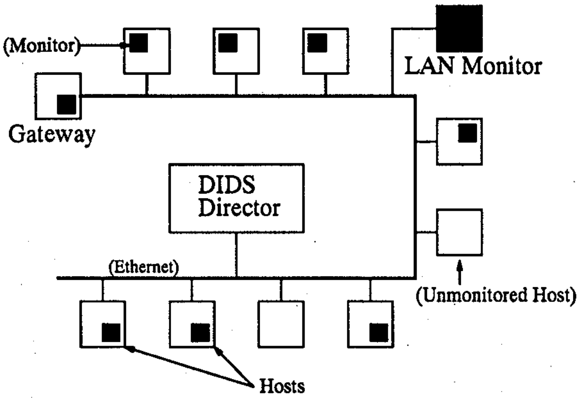

Figure 1: DIDS Target Environment

DIDS Architecture

The DIDS architecture combines distributed monitoring and data reduction with centralized data analysis. This approach is unique among current intrusion detection systems. The major components of DIDS are the *DIDS Director*, a single *Host Monitor* per host, and a single *LAN Monitor* for each broadcast LAN segment in the monitored network (Figure 2).

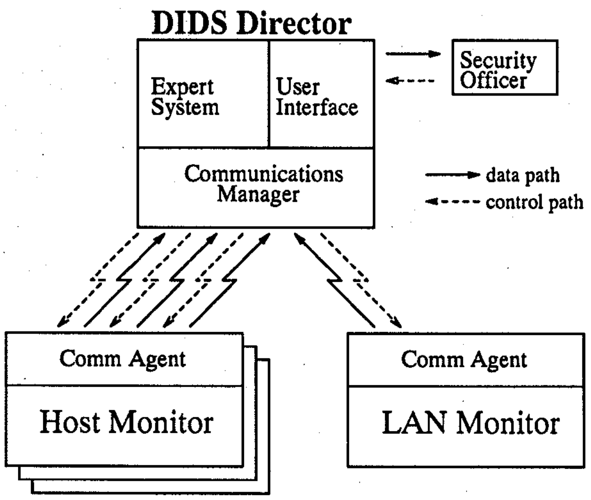

Figure 2: Communications Architecture

DIDS can potentially handle hosts without monitors since the LAN Monitor can report on the network activities of such hosts. The Host and LAN Monitors are primarily responsible for the collection of evidence of unauthorized or suspicious activity, while the DIDS Director is primarily responsible for its aggregation and evaluation.

The Host Monitor collects and analyzes audit records from the host operating system. The audit records are scanned for *notable events*, which are transactions that are of interest independent of any other records. These include, among others, failed events, user authentications, changes to the security state of the system, and any network access such as *rlogin* and *rsh*. These notable events are then sent to the DIDS Director for further analysis. The Host Monitor also tracks user sessions and reports anomalous behavior aggregated over time through user/group profiles. It also searches the event stream for attack signatures, which are sequences of events that are considered to be indicative of attack behavior.

The LAN Monitor's main responsibility is to observe all of the traffic on its segment of the LAN in order to monitor host-to-host connections, services used, and volume of traffic. The LAN Monitor reports on such network activity as *rlogin* and *telnet* connections, the use of security-related services, and changes in network traffic patterns. It is also used to help verify the owners of certain connections between hosts.

Each Host and LAN Monitor has a single *communications agent* that provides an interface between the monitor and the DIDS Director. The agent serves as a buffer between the local monitor and the DIDS Director by handling all communications into and out of the host. This design allows the monitor's analysis components to concentrate on detecting intrusions and not be concerned with communications requirements.

The DIDS Director is divided into three components. The *Communications Manager* is responsible for the transfer of data between the DIDS Director and each Host and LAN Monitor via the local agent. It receives notable event records and system reports from each Host and LAN Monitor, and then sends them to the *Expert System* or *User Interface*. The Expert System is responsible for correlating the information and evaluating the data, and then providing reports on the security state of the monitored system. Based on the reports from the Host and the LAN Monitors, the Expert System makes inferences about the security state of each individual host, and aggregates information to report on the state of the entire system. The DIDS Director's User Interface gives the Computer System Security Officer (CSSO) interactive access to the entire system. The CSSO uses the interface to watch activities on each host, observe network traffic, and request more specific types of information from a monitor.

The Network-user Identification (NID)

One of the most interesting challenges for an intrusion detection system operating in a networked environment is tracking users and objects (e.g., files) as they move across the network. This is required to provide accountability in a networked environment. On single hosts, the user-id/password mechanism provides some degree of user accountability, but this is lost when multiple uncoordinated user-ids may belong to one human user. For example, an intruder may use several different accounts on different machines during the course of an attack. Correlating data from several independent sources, including the network itself, can aid in recognizing this type of behavior and tracking an intruder back to their source. In a networked environment, an intruder often chooses to employ the interconnectivity of the computers to hide his true identity and location. A single intruder may use multiple accounts on different machines to launch an attack, and that behavior can be recognized as suspicious only if one knows that all of the activity emanates from a single source. Detecting this type of behavior requires attributing multiple sessions, perhaps with different account names, to a single source.

This problem is unique to the network environment and has not been dealt with before in the context of user accountability. Our solution to the multiple user identity problem is to create a *Network-user Identification* (NID) the first time a user enters the monitored environment, and then to apply that NID to any further instances of that user. All evidence about the behavior of any instance of the user is then accountable to the single NID. In particular, we must be able to determine if "smith@host1" is the same user as "jones@host2". Since the Network-user Identification problem involves the collection and evaluation of data from both Host and LAN Monitors, examining it is a useful method to understand the operation of DIDS.

The Host Monitor

For the current prototype, the Host Monitor operates on a Sun SPARCstation running SunOS 4.1.1 or later with the Sun Basic Security Module (BSM) package installed. Through the BSM security package, the operating system produces audit records for virtually every raw event on the system. These raw events include file accesses, system calls, process executions, and logins. A VMS version of the Host Monitor is currently under development as well.

The Host Monitor consists of three analysis components that act in parallel (Figure 3). Each analysis component processes the same data (possibly in different formats) in order to make its decisions. One component looks for notable events, another builds and maintains session profiles for users and groups of users, and the third component looks for attack signatures.

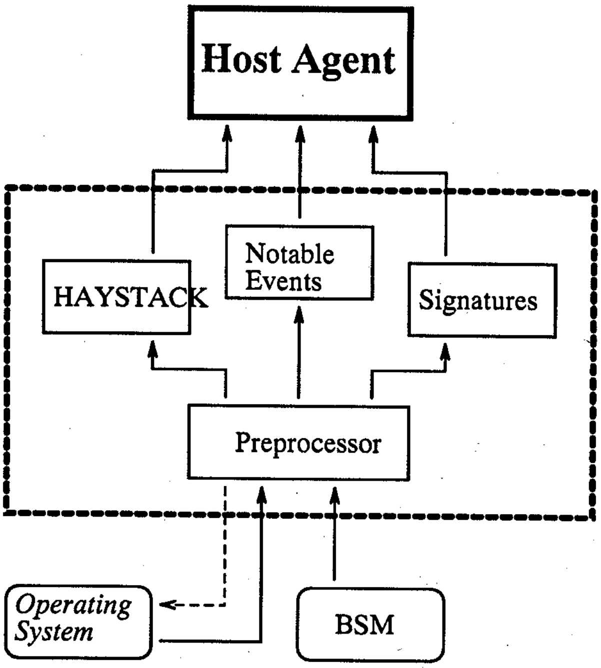

Figure 3: Host Monitor Structure

The Host Monitor *preprocessor* converts the raw audit data generated by the BSM into the abstract events that are processed by each analysis component of the Host Monitor. The preprocessor abstracts from the raw data to remove superfluous information and remove the majority of host specific information. The preprocessor also performs context analysis on each record in order to decide which event type the current record maps into. Since there are more types of raw audit records than abstract events, and the raw audit records appear in different contexts, there is not a simple one-to-one mapping of audit records to abstract events. It is the preprocessor's job to transform each raw audit record into the standard format that the analysis components expect to deal with.

The notable event detector in the Host Monitor examines each event to determine whether or not it should be sent to the Expert System for further evaluation. Certain critical events are always passed directly to the Expert System (i.e., *notable events*); others are processed locally by the Host Monitor in order to generate profiles, and only summary reports are sent to the Expert System. Thus, one of the design objectives is to keep as much of the processing operations at the local host as possible.

Of all possible events generated by the preprocessor, only a subset is sent to the Expert System for further consideration. For the creation and application of the NID, it is those events which relate to the creation or modification of user sessions that are important. Communications events, authentication events, and privilege changes are especially useful for the NID problem. The Host Monitor consults external tables to determine which events should be sent to the Expert System. Because they relate to events rather than to the audit records themselves, the tables and the modules of the Host Monitor which use them are portable across operating systems. The only portion of the Host Monitor which is operating system dependent is the module which creates the abstract events based on audit records.

The HAYSTACK Component

The HAYSTACK component of the Host Monitor reduces the voluminous system event stream to short summaries of user behaviors, anomalous events, and security incidents [9]. In addition to providing this data reduction, HAYSTACK attempts to detect several types of intrusions: attempted break-ins, masquerade attacks, penetration of the security system, leakage of information, denial of service, and malicious use. HAYSTACK's operation is based on behavioral constraints imposed by official security policies and on models of typical behavior for user groups and individuals. HAYSTACK helps to detect intrusions (or misuse) in two different ways.

HAYSTACK may tag particular security subjects and objects as requiring special monitoring. This is analogous to setting an alarm to go off when a particular user-id is active, or when a particular file or program is accessed. This alarm may also increase the amount of reporting of the user's activity.

HAYSTACK performs two different kinds of statistical analysis. The first kind of statistical analysis yields a set of suspicion quotients. These are measures of the degree to which the user's aggregate session behavior resembles one of the target intrusions that HAYSTACK is trying to detect.

About two dozen features (behavioral measures) of the user's session are monitored on the system, including time of work, number of files created, number of pages printed, etc. Given a list of the session features whose values were outside the expected ranges for the user's security group, plus the estimated significance of each feature violation for detecting a target intrusion, HAYSTACK computes a weighted multinomial suspicion quotient that the session resembles a target intrusion for the user's security group. The suspicion quotient is therefore a measure of the anomalousness of the session with respect to a particular weighting of features. HAYSTACK emphasizes that such suspicions are not "smoking guns", but are rather hints or hunches to the security officer or the DIDS Director that may warrant further investigation.

The second kind of statistical analysis detects variation within a user's behavior by looking for significant changes (trends) in recent sessions compared to previous sessions.

The Signature Analysis Component

The attack signature analysis component of the Host Monitor defines attack-type behavior and recognizes sequences of events that closely match predefined signatures [10,11]. Signature analysis searches for known attack sequences, attacks that exploit known flaws or administrative vulnerabilities, and attacks that a masquerader would employ to change the security state of the system, browse through the file system, store information for future use, or attempt to hide his tracks. Signature analysis builds up a context of activity that is based on the users previous events.

Space	Static/ Dynamic	Partitions
Identity	dynamic	super-user normal
Identity	static	super-user normal
Object location	dynamic	read-only system space writable system space owned user space other user space
User location	dynamic	read-only system space writable system space owned user space other user space
Origin	static	physical terminal local host remote host
Event time	dynamic	business hours off hours
System state	dynamic	normal activity suspicious activity under attack
Session security state	dynamic	normal attacker
Security suspicion state	dynamic	normal excessive

Table 1: Signature Analysis Spaces and Partitions

It then looks at (context, event) tuples in an attempt to identify membership in or resemblance to the described set of signatures. It is the context that determines whether or not the current event triggers an instance of a signature; the same event occurring in a different context may not trigger an instance of that signature. Conversely, in many (but not all) cases it is the event that determines the significance of a change in context.

The context consists of a set of spaces (Table 1) that describe various attributes of the user session up to a particular point in time. The information obtained at login and from each abstract audit event serve to build the context. Each space is divided into a set of partitions. After each event has been processed, a user is in at most one partition within each space. A transition between partitions is caused either by an audit event or by external manipulation (e.g., a message from the Expert System in the DIDS Director).

Spaces are either static or dynamic. Static spaces are defined at login; no transitions between partitions occur during a user session. Dynamic spaces are initially defined at login; transitions between partitions occur during a user session. Some dynamic spaces may be "read-only" for the signature mechanism. Those "read only" spaces can only be manipulated through an external mechanism. A transition between partitions of a dynamic space and the association that transition had with the user privilege level is of greatest interest to the signature mechanism.

The LAN Monitor

The LAN Monitor is a subset of UC Davis' Network Security Monitor (NSM) [3,4]. The LAN Monitor builds its own "LAN audit trail". The LAN Monitor observes each and every packet on its segment of the LAN and, from these packets, it is able to construct higher-level objects such as connections (logical circuits), and service requests that use the TCP/IP or UDP/IP protocols. In particular, it audits host-to-host connections, services used, and volume of traffic per connection.

The LAN Monitor uses simple analysis techniques to identify significant events. The events include the use of certain services (e.g., *rlogin* and *telnet*) as well as activity by certain classes of hosts (e.g., a PC without a Host Monitor). The LAN Monitor also uses and maintains profiles of expected network behavior. The profiles consist of expected data paths (e.g., which systems are expected to establish communication paths to which other systems, and by which service) and service profiles (e.g., what a typical *telnet*, *mail*, or *finger* is expected to look like).

The LAN Monitor also uses heuristics in an attempt to identify the likelihood that a particular connection represents intrusive behavior. These heuristics consider the capabilities of each of the network services, the level of authentication required for each of the services, the security level for each machine on the network, and signatures of past attacks. The abnormality of a connection is based on the probability of that particular connection occurring and the behavior of the connection itself. The LAN Monitor is also able to provide a more detailed examination of any connection, including capturing every character crossing the network. This capability can be used to support a directed investigation of a particular subject or object.

The LAN Monitor has several responsibilities with respect to the creation and use of the NID. The LAN Monitor is responsible for detecting any connections related to *rlogin* and *telnet* sessions. Once these connections are detected, the LAN Monitor can be used to verify the owner of a connection. The LAN Monitor can also be used to help track tagged objects moving across the network. The CSSO can also tap into a network connection to closely monitor a suspicious user's behavior.

The Expert System

DIDS utilizes a rule-based (or production) expert system that is written in CLIPS, a C language expert system implementation from NASA [2]. The Expert System uses rules derived from a hierarchical model that describes data abstractions used in inferring an attack on a local area network. That is, it describes the transformation from raw audit data to high level hypotheses about intrusions and about the overall security of the monitored environment. In abstracting and correlating data from the distributed sources, the model builds a virtual machine which consists of all the connected hosts as well as the network itself. This unified view of the distributed system simplifies the recognition of intrusive behavior which spans individual hosts. The model is also applicable to the trivial network of a single computer.

The low level Expert System objects are the abstract event reports provided by the LAN Monitor and the Host Monitor. These reports are both syntactically and semantically independent of the source.

The goal is to introduce a single identification for a user across many hosts on the network. Upper layers of the model treat the network-user as a single entity, essentially ignoring the local identification on each host. Similarly, above this level, the collection of hosts on the LAN are generally treated as a single distributed system with little attention being paid to the individual hosts.

Events are then placed in context. There are two kinds of context: temporal and spatial. As an example of temporal context, behavior which is unremarkable during normal business hours may be highly suspicious during off hours [5]. In addition to the consideration of external temporal context, the Expert System uses time windows to correlate events occurring in temporal proximity. Spatial context implies the relative importance of the source of events. That is, events related to a particular user, or events from a particular host, may be more likely to represent an intrusion than similar events from a different source.

In the context of the Network-user Identification problem we are concerned primarily with the audit data, the event, and the subject. The generation of the first two of these have already been discussed; thus, the creation of the subject is the focus of the following section.

Building the NID

The only legitimate ways to create an instance of a user within UNIX are for the user to login from a terminal, console, or remote source, to change the user-id of an existing instance, or to create additional instances (local or remote) from an existing instance. In each case, there is only one initial login (system wide) from an external device. When this original login is detected, a new unique NID is created. This NID is applied to every subsequent action generated by that user. When a user who already has a NID creates a new login session, that new session is associated with his original NID. Thus the system maintains a single identification for each physical user.

We consider an instance of a user to be the 4-tuple *<session_start, user-id, host-id, timestamp>*. Thus, each login creates a new instance of a user. In associating a NID with an instance of a user, the Expert System first tries to use an existing NID. If no NID can be found which applies to the instance, a new one is created. Trying to find an applicable existing NID consists of several steps. If a user changes identity (e.g., using the UNIX *su* command) on a host, the new instance is assigned the same NID as the previous identity. If a user performs a remote login from one host to another host, the new instance gets the same NID as the source instance. When no applicable NID is found, a new unique NID is created.

There is still some uncertainty involved in attempting to solve the Network-user Identification problem. If a user leaves the monitored domain and then reenters it with a different user-id, the uncertainty is resolved by creating a session "thumbprint". The thumbprinting technique allows us to examine certain characteristics of two different network connections in an attempt to correlate them and determine if two connections belong to the same session. Similarly, if a user passes through an unmonitored host, the need again arises to use the thumbprinting technique in an attempt to match a connection entering the host with a connection leaving the host. Multiple connections originating from the same host at approximately the same time also creates uncertainty if the user names on the target hosts do not provide any helpful information. The Expert System can make a final decision with additional information from the Host and LAN Monitors that can (with high probability) disambiguate the connections.

Conclusion

Our Distributed Intrusion Detection System (DIDS) addresses the shortcomings of current single host IDS's by generalizing the target environment to incorporate multiple hosts connected via a local area network (LAN). Most current IDS's do not consider the impact of the LAN structure when attempting to monitor user behavior for attacks against the system. Intrusion detection systems designed for a network environment will become increasingly important as the number, size, and complexity of LAN's continue to increase. Our prototype has demonstrated the viability of our distributed architecture in solving the Network-user Identification problem. We have tested the system on a network of Sun SPARCstations and it has correctly tracked network users in a variety of scenarios. Work continues on the design, development, and refinement of rules, particularly those which can take advantage of knowledge about specific kinds of attacks. In addition to the current Host Monitor, which is designed to detect attacks on general purpose multi-user computers, we intend to develop monitors for application specific hosts such as file servers and gateways. In support of the ongoing development of DIDS we are planning to extend our model to a hierarchical Wide Area Network environment.

Acknowledgments

The DIDS project is a joint effort between the United States Air Force Cryptologic Support Center, Haystack Labs, the University of California at Davis, and Lawrence Livermore National Labs.

References

1. Department of Defense, *Trusted Computer System Evaluation Criteria,* National Computer Security Center, DOD 5200.28-STD, Dec. 1985.
2. J. C. Giarratano, *CLIPS User's Guide Volume 1 - Rules,* NASA Lyndon B. Johnson Space Center, Information Systems Directorate, Software Technology Branch, Houston, TX, January 1991.
3. L. T. Heberlein, G. Dias, K. Levitt, B. Mukherjee, J. Wood, and D. Wolber, "A Network Security Monitor," *Proc. 1990 Symposium on Research in Security and Privacy,* pp. 296-304, Oakland, CA, May 1990.
4. L. T. Heberlein, K. N. Levitt, and B. Mukherjee, "A Method to Detect Intrusive Activity in a Networked Environment," *Proc. 14th National Computer Security Conference,* pp. 362-371, Washington, D.C., Oct. 1991.
5. T. Lunt, "Automated Audit Trail Analysis and Intrusion Detection: A Survey," *Proc. 11th National Computer Security Conference,* pp. 65-73, Baltimore, MD, Oct. 1988.

6. T. F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, C. Jalali, H. S. Javitz, A. Valdes, and P. G. Neumann, "A Real-Time Intrusion-Detection Expert System (IDES)," Interim Progress Report, Project 6784, SRI International, May 1990.
7. T. F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. G. Neumann, and C. Jalali, "IDES: A Progress Report," *Proc. Sixth Annual Computer Security Applications Conference*, Tucson, AZ, Dec. 1990.
8. M. M. Sebring, E. Shellhouse, M. E. Hanna, and R. A. Whitehurst, "Expert Systems in Intrusion Detection: A Case Study," *Proc. 11th National Computer Security Conference*, pp. 74-81, Oct. 1988.
9. S. E. Smaha, "Haystack: An Intrusion Detection System," *Proc. IEEE Fourth Aerospace Computer Security Applications Conference*, Orlando, FL, Dec. 1988.
10. S. R. Snapp, "Signature Analysis and Communication Issues in a Distributed Intrusion Detection System," MS Thesis, Division of Computer Science, University of California, Davis, Aug. 1991.
11. S. R. Snapp, B. Mukherjee, and K. N. Levitt, "Detecting Intrusions Through Attack Signature Analysis," *Proc. Third Workshop on Computer Security Incident Handling*, Herndon, VA, Aug. 1991.
12. H. S. Vaccaro and G. E. Liepins, "Detection of Anomalous Computer Session Activity," *Proc. 1989 Symposium on Research in Security and Privacy*, pp. 280-289, Oakland, CA, May 1989.
13. J. R. Winkler, "A UNIX Prototype for Intrusion and Anomaly Detection in Secure Networks," *Proc. 13th National Computer Security Conference*, pp. 115-124, Washington, D.C., Oct. 1990.

Author Information

Steven R. Snapp received a Bachelor's Degree in Computer Science from Colorado State University in 1989, and a Master's Degree in Computer Science from the University of California at Davis in 1991. He has been working on the DIDS project since its inception in 1990, and is currently a software engineer at Haystack Labs in Austin, Texas. Steve was the lead author of an earlier paper on DIDS that won an Outstanding Paper Award from the 14th National Computer Security Conference held in Washington, D.C. in October 1991. Steve can be reached via electronic mail at Snapp@Dockmaster.NCSC.mil.

Stephen E. Smaha is the President of Haystack Labs, Inc. He is the designer and implementer of the HAYSTACK Intrusion Detection System, which is in use at Air Force bases and other U.S. Government sites. Steve is the architect of the DIDS system, and has lectured widely on computer security and intrusion detection. He has graduate degrees in Computer Science and Philosophy from Rutgers University and the University of Pittsburgh, and a BA from Princeton. Steve can be reached via electronic mail at Smaha@Dockmaster.NCSC.mil.

Lt. Daniel M. Teal received a Bachelor's Degree in Electrical Engineering from the Massachusetts Institute of Technology in 1989. He was commissioned by the Air Force and assigned to the AFCSC as a Communications and Computer Security Engineer. He is the lead technical program manager for the DIDS project, and serves as a technical consultant on UNIX security for his directorate. Dan can be reached via electronic mail at Teal@Dockmaster.NCSC.mil.

Captain Tim Grance is a computer scientist for the Air Force Office of Communications-Computer Systems Security located at the Air Force Cryptologic Support Center, Kelly Air Force Base, Texas. He works in the Engineering Division Counter Measures Development Branch as the DIDS project manager. His primary investigative areas are intrusion detection, network security, and UNIX security. He has over 11 years experience in computers and security, and holds a Bachelor of Science degree in Computer Science from the Georgia Institute of Technology, Atlanta, Georgia. Tim can be reached electronically at Grance@Dockmaster.NCSC.mil.