GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"
at http://www.giac.org/registration/gsec

Intrusion Detection – Evolution beyond Anomalous Behavior and Pattern Matching

Paul J. Barry
Security Essentials Version 1.4
October 11, 2002

Abstract

Intrusion Detection (ID) is a useful tool for a Security Professional. Although advances have been made in ID software, a distinct gap exists between the latest malicious code, which the program is designed to detect and the rules used in the detection. The two main methods for Intrusion Detection are anomalous behavior and pattern-matching. This paper will focus on various techniques, which can be used to improve capabilities of Intrusion Detection software in either capability or performance. In order to keep pace with the latest malicious code, future Intrusion Detection software will have to evolve to a level that not only includes anomalous behavior and pattern-matching, but will required to implement multiple techniques such as Artificial Intelligence, Neural Networks, Artificial Immune Systems, temporal signatures and other learning behaviors.

Background

ID stands for Intrusion Detection, which is the art of detecting inappropriate, incorrect, or anomalous activity [1]. Intrusion Detection as defined by Dirk Lehmann of Siemens CERT is the art of detecting inappropriate, incorrect, or anomalous activity. There are four different types of Intrusion Detection Systems according to Paul Innella of Tetrad Digital Integrity, LLC [2]. The four major types of ID are: 1) Network Intrusion Detection (NID); 2) Host-based Intrusion Detection; 3) Hybrid Intrusion Detection and 4) Network-Node Intrusion Detection (NNID).

James P. Anderson published the first study on Intrusion Detection in April of 1980. In the study, Anderson refers to a Surveillance program which can process Session/Job Records along with Exceptions and apply statistics to these logs along with baseline settings for particular user in order to detect any type of malicious activities. Anderson stresses the fact that the log information from a particular job or session must be presented to the Surveillance program contiguously instead of intermixing information based on an arbitrary timestamp. Utilizing a user's profile, the surveillance system can detect misuse of computer system. Anderson's system was the outline for the first generation of Host-based Intrusion Detection software.

Dorothy E. Denning and Peter Neumann were early pioneers in the Intrusion Detection arena. In 1987 they had provided the framework for an intrusion-detection expert system, which was called IDES (Intrusion Detection Expert System) [4] based off of the 1985 paper Requirements and model for IDES – A real-time intrusion detection system[5].

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The intrusion detection was based on these six components:

- *Subjects:* Initiators of activity on a target system- normally users.
- *Objects:* Resources managed by the system-files, commands, devices, etc.
- *Audit records:* Generated by the target system in response to actions performed or attempted by subjects on objects-user login, command execution, file access, etc.
- *Profiles:* Structures that characterize the behavior of subjects with respect to objects in terms of statistical metrics and models of observed activity. Profiles are automatically generated and initialized from templates.
- *Anomaly records:* Generated when abnormal behavior is detected.
- *Activity rules:* Actions taken when some condition is satisfied, which update profiles, detect abnormal behavior, relate anomalies to suspected intrusions, and produce reports.

This research was important for future generations of Intrusion Detection Systems which implement these six components in varying degrees. The IDES system did not take into account vulnerabilities in targeted systems which would make the processing of rules too slow. The objective was to alert Security Operators who in turn could investigate possible vulnerabilities with the targeted system. Another Intrusion Detection system was the MIDAS system [6] which primarily focused on attack signatures from the audit data versus the normal patterns used in the IDES system. Halme and Kahn [7] proposed a method to use a system which is based on both anomalous behavior as well as patterns in order improve the efficiency of the capturing of Intrusion events.

The IDES lead to NIDES (Next-Generation Intrusion Detection Expert System) [8] from SRI, International. The NIDES system is based on two approaches: (1) intrusions, whether successful or attempted, can be detected by flagging departures from historically established norms of behavior for individual users, and (2) know intrusion scenarios, known system vulnerabilities, and other violations of a system's intended security policy (i.e., a priori definition of what is to be considered suspicious) are best detected through use of an expertsystem rule base. The IDES and NIDES have lead to many other Intrusion Detection Software systems.

Another IDS around the same era as IDES and NIDES was the Haystack [8] system which became available in 1988. They Haystack system was designed to help the Air Force Security Officers monitor their Unisys 1100/2200 mainframes for misuses in "unclassified but sensitive" data processing. The Haystack system provided information in the following manner: 1) Notable Events were generated as singe events for review by the Security Officers. The security state of the system was reported with a success or failure message; 2) Special Monitoring was captured by the system. Security Officers could tag either "subjects" or "objects" which could be monitored and events generated by these tagged items; and 3) Statistical Analysis was performed on the audit data. There were two types of statistical analysis performed on the data. The first was the "suspicion quotients" which were used to detect how closely a given user's aggregate session behavior matched one of the target intrusion behaviors. The second

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

kind of statistic used in the program was to perform statistical analysis on the user's behavior by looking at the user's trends in previous sessions to the current session.

The next step in Intrusion Detection Systems came with the advent of the Distributed Intrusion Detection System (DIDS) [9]. The DIDS system combined distributed monitoring and data reduction with a centralized data analysis unit. The components of the DIDS are the DIDS director, a single host monitor per host, and a single LAN monitor for each broadcast LAN segment to be monitored in the network. In order to facilitate the monitoring of each user throughout the system the DIDS employed a Network-user identification (NID) which was assigned to the incoming user and used throughout the system to track a particular user's activities. The LAN monitor would build its own "LAN audit trail" which it will then analyze this data using heuristics to determine the likelihood that this activity is abnormal and should be flagged as an event. The Host monitor's purpose is to analyze the log detail and determine if this log line should be forwarded to the expert system. The DIDS uses a rule-based (or production) expert system. The rules are based on a hierarchical Intrusion Detection Model (IDM). The IDM is based on six layers (defined in Table 1).

Level	Name	Explanation
6	Security State	Overall network security level
5	Threat	Definition of categories of abuse
4	Context	Event placed in context
3	Subject	Definition and disambiguation of network user
2	Event	OS independent representation of user action (finite number of these)
1	Data	Audit or OS provided data

Table 1 – Intrusion Detection Model

The IDM levels start out with all of the data from the audit records. More information is gathered and correlated as you reach each subsequent level until the sixth level is reached. At this point, a rating from 0 to 100 is given to the overall network with a high rating implying that the network is less secure.

The IDS systems listed above is not meant to comprise an exhaustive list of all research conducted in the Intrusion Detection field, rather it represents a short glimpse of some of the more prominent Intrusion Detection systems and a brief history of Intrusion Detection systems.

Intrusion Detection Methods and Enhancements

According to Lawrence R. Halme and R. Kenneth Bauer in the "AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques" article [10], the following categories are mentioned in the paper which are: 1) Anomaly Detection; 2) Misuse Detection and 3) Hybrid Misuse / Anomaly Detection*.*

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Their article further breaks down these main categories into subcategories as follows.

- Anomaly Detection
  - Threshold Monitoring
  - User Work Profiling
  - Group Work Profiling
  - Resource Profiling
  - Executable Profiling
  - Static Work Profiling
  - Adaptive Work Profiling
  - Adaptive Rule Based Profiling

- Misuse Detection
  - Expert Systems
  - Model Based Reasoning
  - State Transition Analysis
  - Neural Networks

- Hybrid Misuse/Anomaly Detection

Intrusion Tolerance

Software based on these categories will be necessary to keep Intruders under a watchful position. Not only does the evolution of Intrusion Detection Software depend on these categories, but they must also be able to elevate to higher levels in order to keep pace with new Intruder techniques. The paper on "Characterizing Intrusion Tolerant Systems Using a State Transition Model" [11] provides another approach to the Intrusion Detection arena. Instead of specifically using the well-known attack signatures, it is designed around the functions and services required for protection. The SITAR project (a DARPA-funded research project) [12] is a scalable Intrusion Tolerant Architecture. Figure 1 depicts the basic model used by the Intrusion Tolerant system which was proposed as the framework to describe the dynamic behavior of the system. The basic model describes how the system oscillates between the Good State and the vulnerable state. The Security Officer's duties are to minimize the time that a system is in the vulnerable state. The system is determined vulnerable if a user is able to read, modify, grant or deny information without proper authorization. The Intrusion Tolerant system is mainly designed to focus on the attack vector on a system rather than a particular security exploit used to gain unauthorized access to the system. The Intrusion Tolerant basic model can be used to provide various levels of protection (i.e. If a system requires to protect against a denial of service attack, then the system would want to switch to the degradation state GD, versus protection against file corruption which the system would want to switch to the fail-secure state FS).

Speak the Common Language

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Switching focus from the intrusion detection to intrusion tolerance was one method which could be used by intrusion detection software to alarm Security Operators of intrusion events. Another useful method in the State-based Intrusion Detection arena is the definition of an attack language such as STATL [14]. The STATL language was developed out the necessity of filtering out the domain or particular environment for an attack signature. Other systems depend heavily on their particular environment and are not well suited for extension to other environments. STATL defines the domain-independent features of attack features of attack scenarios and it also provides extensions to allow customization for specific attack signatures. STATL formed its base from the State Transition Analysis Technique (STAT) system. The STATL specification provides a method to completely describe the attack event in the language itself. The STATL language provides the necessary means of representing an attack in terms of states and transitions. The language was designed to track the attack throughout the various stages on a system. Although every aspect of the attack cannot be tracked (i.e. tracking all physical or virtual memory), files or resources that change due to the attack method can be described in the language. The STATL specification is one tool, which can be used by Intrusion Detection systems in their arsenal to describe an attack. The STATL language is simple, yet extensible. In the study, they were able to develop 18 attack scenarios for USTAT, 10 attacks for WinSTAT and 20 attacks for the NetSTAT tool. They found that in encoding the various attacks, they did not encounter any limitations in the language. The STATL language can be applied to Snort Rules as outlined by the “Translating Snort Rules to STATL Scenarios” paper presented by Eckmann [15]. The snort2statl translator program was developed from this paper and was a step at applying the STATL language to an Intrusion Detection program.

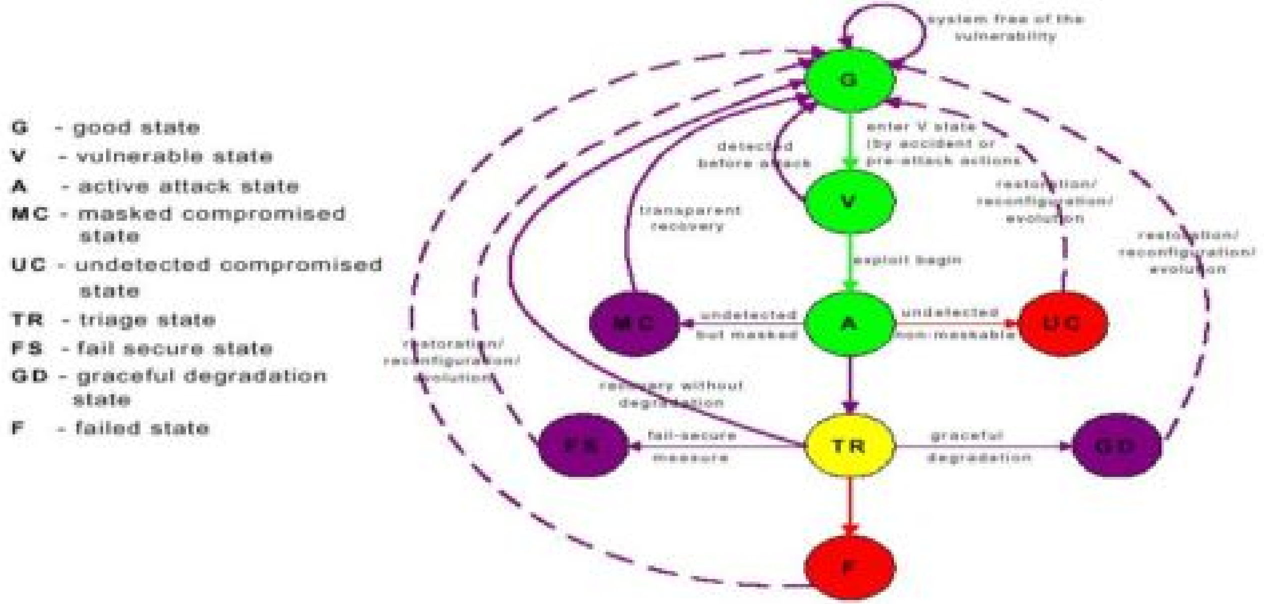
G - good state
V - vulnerable state
A - active attack state
MC - masked compromised state
UC - undetected compromised state
TR - triage state
FS - fail secure state
GD - graceful degradation state
F - failed state
G
V
A
MC
UC
TR
FS
GD
F
system free of the vulnerability
enter V state (by accident or pre-attack actions
detected before attack
transparent recovery
exploit begin
restoration/ reconfiguration/ evolution
restoration/ reconfiguration/ evolution
restoration/ reconfiguration/ evolution
undetected but masked
undetected non-maskable
recovery without degradation
fail-secure measure
graceful degradation

Figure 1 - State Transition Diagram for ITS[13]

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Another attack language which is a little older than STATL is the Graph-based Intrusion Detection System (GrIDS) language. The GrIDS language was developed in 1996[16]. GrIDS is similar to STATL in that it tracks activity (or transitions in STATL). GrIDS also uses hosts as well as activities to track attacks. GrIDS was designed to monitor network activity between hosts via activity graphs. The aggregated information from these activity graphs is representative of the network activity between hosts. Identifying attacks can be based on thresholds of various activity graphs. Network activity which occurs between hosts in a given timeframe is considered part of the activity graph. In terms of large-scaled networks, GrIDS has potential in the following network attack types: 1) A sweep (like Doorknob rattling); 2) Coordinated attacks and 3) Worms. Since the GrIDS system is based on a graph, it will build one based on network traffic between hosts and it takes into consideration factors such as attributes of the node (host) or edge (traffic). These attributes will provide a mechanism to rapidly identify an attack. The attributes for the graph nodes or edges are not necessarily generated internally by the GrIDS system and the values may be generated externally by other IDSs, or any other device or program which can transport their output to the GrIDS system. The GrIDS system will also allow external correlation functions to be implemented in the language. In addition to external correlation functions, GrIDS will maintain multiple graph spaces (because each attack vector may require their own type of graph in the future) as well as a rule set for each graph. When new information enters the GrIDS system for either the node or the edge, it is analyzed by the appropriate rule set and then a given action is taken. Depending on the rule set, graphs may collapse, others may be created, or none of the information may apply to any graphs currently in the GrIDS system. A given rule set will apply to that particular node and all descendants of that node to ease the use of generating several similar rules. GrIDS will also contain aggregation of data before passing information to the next graph. The aggregation of data is based on modeling departments and then data sent from one department through the use of reduced graphs (i.e. a reduction in the number of nodes). In addition to rule sets, a Policy language is built-in, which allows the definition of unacceptable network behavior. These policies can be compiled into the rule sets of the graphs. The primary goal of the project was to make the aggregation mechanism scalable and to allow the system to be dynamically configurable so that it could be easily deployed to a large-scaled network. There was not a significant amount of time spent on securing communication between GrIDS and other IDS. It was mentioned that GrIDS may not detect intrusions which are small, slow or both. This could be used with other IDSs to assist in large scaled networks. It is not the final solution, however it does present another language to try and define attack methods.

Progressive Signatures and Patterns

Not only does Intrusion Detection Software use a language, but it also must use signatures to detect an attack. One of the newer methods of an attack signature is the paper written by Jones and Li on “Temporal Signatures for Intrusion Detection” [17]. Normal anomaly detection is performed by comparing any type of “normal” activities to network behavior and if this behavior exceeds a statistically significant value, then that activity is tagged as an anomaly and that event is reported. The temporal signature is

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

based on call sequences for a given application during its normal execution phase and a Hamming distance is computed between the “normal” call sequences and the current call sequences to determine abnormal behavior. The temporal signatures were based on work at the University of New Mexico which used the sequences of system calls with the incorporation of timing properties to these sequences of calls. The timing is based on the amount of time between system calls with other factors such as context swap or sleep time deducted from the amount. A multitude of samples are taken on each application with the computation of the time distributions from all of the results are then stored in a “temporal signature” database. The temporal signature method of Intrusion detection could be one of the methods chosen by newer Intrusion Detection Systems. The temporal signature method would take some time to populate the “temporal signature” database; however this method may provide Security Officers with another method to track the attackers.

Although there are some drawbacks to temporal signatures, the next system was designed with high-performance in mind. Sekar, Guang, Verma and Shanbhag proposed a High-Performance Network Intrusion Detection system [18]. The High-Performance Network Intrusion Detection system is said to sport the following characteristics:

- Concise, easy-to-develop intrusion specifications.
- High-speed, large-volume monitoring.
- Robust and extensible.
- Comprehensive evaluation of performance.

This system’s basic language consists of variable and type declarations accompanied with a list of rules. The rules consist of a pattern and then an appropriate action. Patterns in this language can be simple or complex and the pattern language is referred to as regular expressions over events (REE). This language also provides the important aggregation component of intrusion detection. One of the aggregation components is the counters which have an aging function which can assign priorities based on historical events, thresholds for the counters and upper and lower limit functions based on the thresholds. The other aggregation component is the table which keeps information similar to a histogram. The tables also contain a purge function for ‘stale’ entries in the tables. Tests were performed on this system and it was 96% effective on intrusion detection and was able to sustain intrusion detection at the rate of 15s/GB, or 500Mb/s using a 450MHz Pentium II running RedHat Linux 5.2. The key feature of this system was the domain-specific language for capturing patterns on a stream of normal as well as abnormal network packet sequences. The system is said to have been time insensitive to the number of rules which would make this system highly scalable. Since the language is extensible, it makes it a prime candidate for implementing new intrusion detection patterns.

The High-Performance Network Intrusion Detection system is one way to squeeze extra performance in the intrusion detection arena, and another method which could be implemented is the Nearest Neighbor Algorithm as described by Wetzel [19]. Wetzel explains the use of Case based reasoning (CBR) in artificial intelligence (AI) to find a match rather than performing an exact match. CBR focuses on memory rather than a

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

set of rules. The similarity matching produced by CBR may increase the number of false positives; however it may provide a mechanism which can be used to quickly identify patterns without having all of the overhead of keeping the rules in memory.

Not only does evolution of intrusion detection software have to identify patterns and anomalies, it also has to start using machine intelligence to learn the behavior of the network. Mr. Nugen [20] discussed machine intelligence in his presentation "Artificial Intelligence in Information Security". The beginning of the presentation by Mr. Nugen discussed the efforts made by programmers to try and make the machine intelligence emulate human intelligence. He then explained Information Security as the disciplines and processes protecting the confidentiality, integrity and availability of information (intellectual property) and resources that are used to manage and protect this information. There were six different levels discussed by Nugen which are: 1) Awareness; 2) Inspect; 3) Protect; 4) Detect; 5) React and 6) Reflect. The intrusion detection software needs to keep cycling through these stages after having learned lessons, when new threats arise, there are new business needs or when new environments (i.e. terrorism, unknown third-party vendor has a link into your company's network) become active. An expert system's rules could be implemented in the network-based intrusion detection system to determine acceptable and unacceptable user behavior. There is an opportunity to utilize neural networks which could learn the normal behavior of humans interacting on the network (especially useful when there is a shortage of expertise to describe the behavior). The various facets of AI were discussed along with a brief description of how they could be used in the information security realm. The Case-based reasoning (CBR) was discussed in the presentation where the best-match is used in searching for a conclusion. The Explanation-Based Learning (EBL) is another facet and it is the generalized from a single example and it was noted that this takes quite a bit of domain knowledge to implement. The Genetic Programming area got its inspiration from the biological evolution with the execution speed of machines. One of the more frightening points made by Nugen was that current hackers probably have access to multi-node machines and have been testing their software on these machines before releasing them onto unsuspecting networks. He stated that it makes it tougher for intrusion detection software because the response time to detect these signatures is a very limited and therefore the intruder has the advantage of time on their side. This limited timeframe stresses the requirement of intrusion detection software to be fast as well as the capability to learn from intruders and provide detection without all of the possible false alarms. Utilizing expert systems and neural networks can be one way of learning about possible attacks.

Doyle, Kohane, Long, Shrobe and Szolovits [21] propose a richer language that subsumes and extends signature and anomaly methods. They state that the weakness of the signature method is the fact that they are too special and that the anomaly methods are too general in nature. The signature and anomaly methods both have strengths and weaknesses, but together they cannot overcome the weakness in the other method. This new language seeks to provide a method to characterize events. Their are: 1) Landmark times (the significant points in the life of the event); 2) Temporal intervals (periods which may indicate significant subevents); 3) Temporal relations (the

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

shorthand method of expressing relationships to other events); 4) State constraints (the characteristics of objects during the temporal intervals) and 5) Regression functions (these model the criteria for matching templates against data). The latest work on the MAITA system [27] has a goal to extend the TTL in many ways. One of the ways is to augment the range expression for temporal relationships with more general probability distributions of the frequency of landmark times. Linear and Quadratic regression models make up the current constraint language for numeric data, absolute and relative numerical constraints on functions of the data with logical combinations and descriptions and propositions. The paper concluded that most signature and anomaly methods are limited by the reliance on inability of the language to properly express significant patterns. The ability to utilize a language that is based on multilevel abstractions and the capability of expressing uncertainty in the characterization of events allows one to express regularities with the enhancement of abnormalities as part of the language. This enhanced language is meant to increase the likelihood of capturing the intruder by increasing the difficulty level of evading signature or anomalous behavior.language is based off of Haimwitz and Kohane [22], [23], [24], [25], and [26] who developed "trend templates" which were referred to as TTL. The key elements of TTL

Evolutionary techniques for Intrusion Detection

Although advances in pattern matching and anomaly detection are important, examples from other disciplines may provide insight into new intrusion detection methods. Take for example the work performed by Kim and Bentley [28]. Kim and Bentley investigated the use of an artificial immune system for network intrusion detection. One of the key benefits of an artificial immune system is the ability to adapt to a changing environment and dynamically learning the "self" and predicting the "non-self" patterns. A dynamic clonal selection algorithm (DynamiCS) was introduced. The DynamiCS's goal is to distill only the crucial components that yield adaptability to the system. The DynamiCS algorithm was introduced in the paper as a stepping stone towards an artificial immune system that can cope with real environments where self behaviors change after a certain period and a small subset of self antigens is visible at a particular time. The significant features of the human immune system, which provided these desired properties, were discovered. The key properties were central tolerisation, distributed tolerisation, constimulation, affinity maturation, and life span and memory detectors. The DynamiCS was able to implement these with the introduction of the three new parameters of tolerisation period, activation threshold and life span. The experimental results concluded that the system could incrementally learn the globally converged distributions only when a small subset of antigens was injected at each generation. The system performance measured by the antigen detection and self-tolerance rates showed that the number of detector activation's in total primarily controlled this, and that this was directed by the values from the three new parameters mentioned above. At least the study of the artificial immune system may yield new techniques or procedures which could be deployed to an intrusion detection system in the future.

The artificial immune system could be one technique used by future intrusion detection systems and another more powerful system may be the one introduced by Janakiraman, Waldvogel and Zhang [29]. They have proposed the Indra system, a distributed scheme which is based on the sharing of information between trusted peers in a network which would guard against intrusion attempts. The Indra system takes a proactive as well as a peer to peer approach on network security. Usually an intruder will try exploits on several machines until they have compromised a machine. The Indra system can take this information and deliver the attempt across multiple peers which then in turn can react either proactively (e.g., applying patches, temporarily disconnecting the server or both) or retroactively (e.g., disconnect machines that may have been compromised in order to limit further damage). This system runs a special security daemon, the Indra daemon that watches for intrusion attempts and enforces access control to the peer to peer network. In the work performed on Indra, the prototype version relied on trusted key-servers from which Indra gets certificates for its peers. It was stated that this would probably change in the future to a variant on the Web of trust model from PGP [30]. The functionality of Indra is composed of the following modules: 1) Watchers (first level daemons which keep a watchful on suspicious activity); 2) Access Controllers (daemons which provide access control to resources); 3) Listeners (daemons which listen for watchers); and 4) Reporters (daemons responsible for reporting with other hosts, receiving warnings from other hosts, aggregating warnings, and passing warnings to other hosts). Indra provides the ability to add plugins. The report concluded that Indra is a work-in-progress and that the emphasis of Indra was to provide a framework that compliments intrusion detection devices and provides this in a massively networked environment. Indra is reported to offer a scalable solution by providing the security plugins which can be loaded dynamically onto thousands of machines by in an administrative domain.

Conclusion

Network Intrusion Detection systems have always been trying to catch or alert Security Officers of intruders in the network. The growth of technology has increased the complexity of capturing the elusive intruder. Some of the methods mentioned above such as temporal signatures, defining a rich language to handle intruders, making use of artificial intelligence and other techniques may provide the necessary boost for intrusion detection systems to handle the next generation of intruders. Another technique such as tagging network packets so that multiple sensors can correlate a single intruder will be required so that one can easily correlate intrusion events. The use of artificial intelligence, neural networks, fuzzy logic and other learning behaviors may need to be incorporated in order to provide the necessary logic to detect intruders. The artificial immune system provides a new mechanism to detect intruders.
The Indra system may prove to be a solid framework on which network intrusion detection can evolve to the next generation by using the infrastructure to not only detection intrusions but to also prevent unauthorized behavior in a network. All of these new methods could be implemented in order to capture and tag offending intruders today as well as in the future.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

References:

[1] *Intrusion Detection FAQ , URL:* http://www.sans.org/newlook/resources/IDFAQ/what_is_ID.htm

[2] Innella, Paul, “The Evolution of Intrusion Detection Systems,” Tetrad Digital Integrity, LLC, last updated November 16, 2001, URL: http://online.securityfocus.com/infocus/1514.

[3] Anderson, James P., “Computer security threat monitoring and surveillance,” Technical Report Contract 79F26400, James P. Anderson Co., Box 42, Fort Washington, PA, 19034, USA, February 26, revised April 15 1980.

[4] Denning, Dorothy E., “An Intrusion-Detection Model,” IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-13, NO. 2, FEBRUARY 1987, 222-232.

[5] Denning, Dorothy E., and Neumann, Peter E, “Requirements and model for IDES-A real-time intrusion detection system,” Technical report, Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 1985.

[6] Sebring, E., Shellhouse, E., Hanna, M., and Whitehurst R., "Expert Systems in Intrusion Detection: A Case Study," Proceedings of the 11th National Computer Security Conference. Washington DC. October 1988.

[7] Halme, L., and Kahn, B., "Building a Security Monitor with Adaptive User Work Profiles," Proceedings of the 11th National Computer Security Conference. Washington DC. October 1988.

[8] Anderson, Debra, Frivold, Thane, Valdes, Alfonso, “Next-generation Intrusion Detection Expert System (NIDES) A Summary,” Computer Science Laboratory, SRI-CSL-95-07, May 1995. URL: http://www.sdl.sri.com/projects/nides/reports/4sri.pdf

[9] Snapp, S., Brentano, J., Dias, G., Goan, T., Heberlein, L., Ho, C, Levitt, K., Mukherjee, B., Smaha, S., Grance, ,T., Teal, D., and Mansur, D., "DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and an Early Prototype," Proceedings of the 14th National Computer Security Conference. October 1991.

[10] Halme, Lawrence R., Bauer, R. Kenneth, “AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques,” Arca Systems, Inc., 2450 North First St., Suite 301, San Jose, CA 95131-1016. URL: http://www.sans.org/newlook/resources/IDFAQ/aint.htm

[11] Goseva-Popstojanova, K., Wang, F., Wang, R., Gong, F., Vaidyanathan, K., Trivedi, K.S., Muthusamy, B., "Characterizing Intrusion Tolerant Systems Using a State Transtion Model," Proc. DARPA Information Survivability Conference and Exposition II (DISCEX-II), Anaheim, California, June 2001.

[12] Want, F., Gong, F., Sargor, C., Goseva-Popstojanova, K., Trivedi, K.S., Jou, F., "SITAR: A Scalable Intrusion-Tolerant Architecture for Distributed Services," Proc. 2nd Annual IEEE Systems, Man, and Cybernetics Informations Assurance Workshop, West Point, New York, June 2001.

[13] Wang, F., “SITAR: A Scalable Intrusion Tolerant Architecture for Distributed Services,” Advanced Networking Research, MCNC, OASIS Winter PI Meeting Presentation, Feb. 13, 2001. URL: http://www.anr.mcnc.org/projects/SITAR/SITAR-norfolk-2001.pdf

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

[14] Eckmann, Steven T., Vigna, Giovanni, and Kemmerer, Richard A. “STATL: An Attack Language for State-based Intrusion Detection,” In Proceedings of WIDS (held in conjunction with ACMCCS 2000), Athens, Greece, November 2000.

[15] Eckmann, Steven T., “Translating Snort rules to STATL scenarios,” Proceedings of Fourth International Symposium on Recent Advances in Intrusion Detection, RAID2001, October 10, 2001, Davis CA, USA.

[16] Staniford-Chen, S., Cheung, S., Crawford, R., Dilger, M., Frank, J., Hoagland, J., Levitt, K., Wee, C., Yip, R. and Zerkle, D., “GrIDS A Graph-Based Intrusion Detection System for Large Networks”, Proceedings of the 19th National Information Systems Security Conference, Mar. 4, 1996.

[17] Jones, Anita, Song, Li, “Temporal Signatures for Intrusion Detection,” 17th Annual Computer Security Applications Conference, Dec. 10-14, 2001, New Orleans, LA.

[18] Sekar, R., Guang, Y., Verma, S., Shanbhag, T., “A High-Performance Network Intrusion Detection System,” Proceedings of the 6th ACM conference on Computer and communications security, Singapore, 1999.

[19] Baylor, Wetzel, “Build a Smarter Search Engine,” JAVAPro Magazine, October 2002.

[20] Nugen, Stephen, M., “Artificial Intelligence in Information Security,” Infotec 2002 ST4, Omaha, NE, April 23, 2002.

[21] Doyle, Jon, Kohane, Isaac, Long, William, Shrobe, Howard and Szolovits, Peter, “Event Recognition Beyond Signature and Anomaly,” Second IEEE-SMC Information Assurance Workshop, West Point, New York, June 5-6, 2001.

[22] Haimowitz, ,Ira, J. and Kohane, S., “An epistemology for clinically significant trends," in Proceedings of the Eleventh Nation Conference on Artificial Intelligence, Washington, DC, 1993, pp. 176-181.

[23] Haimowitz, Ira J. and Kohane, Isaac, S., “Automated trend detection with alternate temporal hypotheses,” in Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, Chambery, France, 1993,,pp. 146-151.

[24] Kohane, I., S. and Haimowitz, I. J., “Encoding patterns of growth to automate detection and diagnosis of abnormal growth patterns,” Pediatric Research, vol. 33, pp. 119A, 1993.

[25] Haimowitz, I. And Kohane, I., “Hypothesis-driven data abstraction,” in Symposium on Computer Applications in Medical Care, Washington, DC, 1993.

[26] Haimowitz, I., J., Facker, J. and Kohane, I., S., “Knowledge-based data display using TrenD$_x$,” in AAAI Spring Symposium: Interpreting Clinical Data, Palo Alto, 1994, AAAI Press.

[27] Doyle, Jon, Kohane, Isaac, Long, William, Shrobe, Howard, and Szolovits, “Agile monitoring for cyber defense,” in Proceedings of the Second DARPA Information Security Conference and Exhibition (DISCEX-II). IEEE, May 2001, IEEE Computer Society.

[28] Kim, J. and Bentley, P. J., "Towards an Artificial Immune System for Network Intrusion Detection: An Investigation of Dynamic Clonal Selection," the Congress on Evolutionary Computation (CEC-2002), Honolulu, pp.1015 - 1020, May 12-17, 2002.

[29] Ramaprabhu Janakiraman, Marcel Waldvogel, Qi Zhang, “Indra: A peer-to-peer approach to network intrusion detection and prevention,” Washington University Technical report #WUCS-01-30, 2001.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

[30] Stallings, William, Pretty Good Privacy. ConneXions, 8(12):2-11, December 1994.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.