Intrusion Detection Technology

Tyrone Grandison and Evimaria Terzi
IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120
{tyroneg,eterzi}@us.ibm.com

September 7, 2007

1 Definition

Intrusion Detection (ID) is the process of monitoring events occurring in a system and signalling responsible parties when interesting (suspicious) activity occurs.

Intrusion Detection Systems (IDSs) consist of 1) an agent that collects the information on the stream of monitored events, 2) an analysis engine that detects signs of intrusion, and 3) a response module that generates responses based on the outcome from the analysis engine.

2 Historical Background

The concept of ID has existed for decades in the domains of personal home security, defense and early-warning systems. However, automated IDSs emerged in the public domain in 1980 [4] and sought to identify possible violations of the system's security policy by a user or a set of users.

One of the basic elements of an IDS is the audit log that captures the system activity. The initial IDSs exposed to the academic community stored operating system actions, i.e., addressed the operating system layer. Over time, other IDSs have emerged that store different artifacts, and try to identify intrusive behaviors at different layers of operation. The following layers of operation can be easily identified:

Operating System: The logs in this layer contain information from the kernel and other operating system components and help determine if an attacker is trying to compromise the OS. Examples include the Audit Analysis Project [12], HayStack [39], USTAT [22], Wisdom and Sense [43], ComputerWatch [9], ISOA (Information Security Officer's Assistant) [45], IDES [32], Hyperview [8], ASAX [11], DPEM [21], IDIOT [24] and NIDES (Next-generation Intrusion Detection Expert System) [1, 2, 17, 31].

Network: At the network layer, communication data is analyzed to determine if an attacker is trying to access one's network. Examples of IDSs that operate on this layer include NADIR (Network Audit Director and Intrusion Reporter) [14], NSM (Network Security Monitor) [13], DIDS (Distributed Intrusion Detection System) [40], GrIDS (Graph Based Intrusion Detection System) [7], JiNao [18], EMERALD [34] and Bro [33].

Application: Application level IDSs examine the operations executed in an application to ascertain if the application is being manipulated to extract behavior that is prohibited. Examples include MIDAS (Multics Intrusion Detection and Alerting System) [37] and Janus [10]. Database-specific IDSs form an important group of application-level IDSs. Examples of such systems include Discovery [42] and RIPPER [27]. Due to the sensitive information stored in database systems, issues related to database-specific IDSs were among the first to be addressed [5, 26, 44].

The above categorization is historical and mostly depends on the type of log data the IDS uses in order to identify abnormal patterns. Irrespective of the operational layer the very basic detection techniques used by different IDSs have some common basis, which we describe in the next section.

3 Scientific Fundamentals and Key applications

3.1 Types of attacks

In this section we give a generic classification of the types of attacks that ID systems have traditionally tried to cope with. The classification is mainly inspired by the one provided at [3].

- **External break ins**: When an unauthorized user tries to gain access to a computer system.
- **Masquerander (internal) attacks**: When an authorized user makes an attempt to assume the identity of another user. This attacks are called also internal because they are caused by already authorized users.
- **Penetration attack**: In this attack a user attempts to directly violate the system's security policy.
- **Leakage**: Moving potentially sensitive data from the system.
- **Denial of Service**: Denying other users the use of system resources, by making these resources unavailable to other users.
- **Malicious use**: In this category fall miscellaneous attacks such as file deletion, viruses, resource hogging etc.

3.2 Detection methodologies

In this section we provide a high-level categorization of IDSs and give an abstract idea of how they work. In the discussion we provide examples of existing IDSs. However, the examples presented here are more indicative rather than complete. For a more complete discussion on IDSs we refer to [3, 28, 41].

Traditionally, there are two basic approaches to intrusion detection; *anomaly detection* and *misuse detection*. In anomaly detection the goal is to define and characterize legitimate behaviors of the users, and then detect anomalous behaviors by quantifying deviations from the former. However, identifying the distance between anomalous and legitimate behaviors is a rather difficult notion to quantify.

Anomaly detection can be *static* or *dynamic*. A static anomaly detection system is based on the assumption that there is a static portion of the system being monitored. Static portions of the system can be represented as a binary string or a set of binary strings (like files). If the static portion of the system ever deviates from its original form, either an error has occurred or an intruder has altered the static portion of the system. Examples of static anomaly detectors are Tripwire [19, 20] and virus-specific checkers [38].

Dynamic anomaly detectors are harder to build since building them requires a definition of behavior, which is often defined as a sequence (or partially ordered sequence) of distinct events. Differentiating between normal and anomalous activity in dynamic anomaly detection systems is much harder than the problem of distinguishing changes in static elements. Dynamic anomaly detection systems usually create a *base profile* to characterize normal, acceptable behavior. A profile usually consists of a set of observed measures of behavior for a selected set of dimensions. After initializing the base profile the dynamic anomaly detection systems are similar to the static ones; they monitor the behavior by comparing the current behavior with that implied by the base profile. Typically, there is a wide variation of acceptable behaviors and statistical methods are employed to measure deviation from the base profile. The main challenge in dynamic anomaly detection systems is that they must build accurate base profiles and then recognize behaviors that significantly deviate from the profile. An example of dynamic anomaly detection systems that uses statistical approaches to measure deviation from the base profile is NIDES (Next-generation Intrusion Detection Expert System) [1, 2, 17, 31] developed by SRI.

The main advantage of dynamic anomaly detection systems is that they do not require any configuration since they automatically learn the behavior of large number of subjects. Lacking prior knowledge of how an intrusion would manifest itself anomaly detection systems are capable of identifying novel intrusions of variations of known intrusions. However, building base profiles and defining measures of deviations from them is not an easy computational task. For that reason it has been an active area of research, in which several machine learning, time-series analysis and other data-analysis techniques have been employed [6, 5, 15, 23, 25, 29, 35].

Misuse detection is concerned with identifying intruders who are attempting to break into a system using some known technique. If a a system security administrator were aware of all the known vulnerabilities then a misuse detection system would be able to identify their occurrences and eliminate them. A fairly precisely known kind of intrusion is known as *intrusion scenario.*A misuse detection system compares current system activity to a set of intrusion scenarios in an attempt to identify a scenario in progress.

The differentiating factor between the various misuse detection techniques is the model used for describing bad behaviors that constitute intrusions. *Rules* have been primarily used to model the system-administrator's knowledge about the system. MIDAS [36] and IDES [30] are some examples of rule-based systems. Rule-based systems accumulate large numbers of rules which usually prove difficult to interpret and modify. In order to overcome these problems model-based rule organizations and state-transition representations were proposed. These modelling approaches are more intuitive particularly in misuse detection systems where users need to express and understand scenarios. An example of such system is USTAT (Unix State Transition Analysis Tool) [16].

The main advantage of a misuse detection systems is that the system knows for a fact how normal behavior should manifest itself. This leads to a simple and efficient processing of the audit data. The obvious disadvantage of such systems is that the specification of the signatures to be detected is a time-consuming task that requires lots of domain knowledge. At the same time, misuse detection systems lack the ability to identify novel intrusion profiles.

4 Future Directions

One of the major concerns associated with IDSs and their utility is their runtime efficiency. More often than not, IDSs consume too many system resources in order to be effective. Developing resource-aware IDSs systems raises some interesting challenges. One possible way of addressing this concern is via building *Mega Intrusion Detection Systems*. These would be systems that simultaneously monitor all operational layers. That is, the system administrator will not have to run a different ID software for operating system and application specific attacks, but just a single system that will simultaneously be able to detect intrusions in all the desired operational layers. Such systems are expected to be less resource demanding, however their development will certainly create several new design challenges.

In this chapter we have mainly focused on IDSs and described them as mechanisms that guarantee other systems' security. However, IDSs are themselves systems and as such they have their own security risks. Therefore, they also require some protection to prevent an intruder from manipulating the intrusion detection system itself.

References

[1] D. Anderson, T. Frivold, and A. Valdes. Next-generation intrusion detection expert system (NIDES): A summary. In *SRI International Computer Science Laboratory Technical Report SRI-CSL-95-07*, 1995.

[2] D. Anderson, T. Lunt, H. Javitz, A. Tamaru, and A. Valdes. Detecting unusual program behavior using the statistical component of the next-generation intrusion detection expert system (NIDES). In *SRI International Computer Science Laboratory Technical Report SRI-CSL-95-06*, 1995.

[3] S. Axelsson. Research in intrusion detection systems: A survey. In *Technical Report 98-17 (revised in 1999) Chalmers University of Technology*, 1999.

[4] Rebecca Gurley Bace. *Intrusion Detection*. Macmillan Technical Publishing, 2000.

[5] Elisa Bertino, Ashish Kamra, Evimaria Terzi, and Athena Vakali. Intrusion detection in rbac-administered databases. In *ACSAC*, pages 170–182, 2005.

[6] Elisa Bertino, Teodoro Leggieri, and Evimaria Terzi. Securing dbms: Characterizing and detecting query floods. In *ISC*, pages 195–206, 2004.

[7] S. Stanfiford 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 *19th National Information Systems Security Conference*, 1996.

[8] Herve Debar, Monique Becker, and Didier Siboni. A neural network component for an intrusion detection system. In *IEEE Computer Society Symposium on Research in Security and Privacy*, 1992.

[9] Cheri Dowell and P. Ramstedt. The computerwatch data reduction tool. In *IEEE Symposium on Research in Security and Privacy*, 1989.

[10] Ian Goldberg, David Wagner, Randi Thomans, and Eric Brewer. A secure environment for untrusted helper applications (confining the wily hacker). In *Sixth USENIX Security Symposium*, 1996.

[11] Jani Habra, Baudouin Le Charlier, Abdelaziz Mounji, and Isabelle Mathieu. Asax: Software architecture and rule-based language for universal audit trail analysis. In *ESORICS*, 1992.

[12] T. Lunt Lawrence Halme and J. Van Horne. Automated analysis of computer system audit trails for security purposes. In *Thirteenth National Computer Security Conference*, 1990.

[13] L.T. Heberlein. A network security monitor. In *IEEE Symposium on Research in Security and Privacy*, 1990.

[14] J. Hochberg, J. Jackson, C. Stallings, J.F. McClary, D. Dubois, and J. Ford. Nadir: an automated system for detecting network intrusion and misuse. *Computer Security*, 12(3):235–248, May 1993.

[15] Y. Huang, W. Fan, W. Lee, and P. Yu. Cross-feature analysis for detecting ad-hoc routing anomalies. In *Proc. 23rd Intl. Conf. on Distributed Computing Systems*, 2003.

[16] Koral Ilgun, Richard A. Kemmerer, and Phillip A. Porras. State transition analysis: A rule-based intrusion detection approach. *IEEE Trans. Software Eng.*, 21(3):181–199, 1995.

[17] H. Javitz and A. Valdes. The NIDES statistical component: Description and justification. In *SRI International Computer Science Laboratory Technical Report*, 1993.

[18] Y. Frank Jou, Fengmin Gong, Chandru Sargor, Shytsun Felix Wu, and Cleaveland W Rance. Architecture design of a scalable intrusion detection system for the emerging network infrastructure. In *North Carolina State University Technical Report CDRL A005*, 1997.

[19] G. H. Kim and E. H. Spafford. A design and implementation of tripwire: A file system integrity checker. In *Purdue Technical Report CSD-TR-93-071*, 1993.

[20] G. H. Kim and E. H. Spafford. Experiences with tripwire: Using integrity checkers for intrusion detection. In *Purdue Technical Report CSD-TR-94-012*, 1994.

[21] Calvin Ko, George Fink, and Karl Levitt. Automated detection of vulnerabilities in privileged programs by execution monitoring. In *10th Annual Computer Security Applications Conference*, 1994.

[22] Ilgun Koral. Ustat: A real-time intrusion detection system for unix. In *IEEE Symposium on Research in Security and Privacy*, 1993.

[23] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur. Bayesian event classification for intrusion detection. In *ACSAC*, 2003.

[24] Sandeep Kumar and Eugene H. Spafford. An application of pattern matching in intrusion detection. In *Purdue University Technical Report CSD-TR-94-013*, 1994.

[25] Terran Lane and Carla E. Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Trans. Inf. Syst. Secur.*, 2(3):295–331, 1999.

[26] Victor C. S. Lee, John A. Stankovic, and Sang Hyuk Son. Intrusion detection in real-time database systems via time signatures. In *IEEE Real Time Technology and Applications Symposium*, pages 124–133, 2000.

[27] Wenke Lee. A data mining framework for building intrusion detection models. In *IEEE Symposium on Security and Privacy*, 1999.

[28] Wenke Lee and Wei Fan. Mining system audit data: Opportunities and challenges. *SIGMOD Record*, 30(4):35–44, 2001.

[29] Wenke Lee and Dong Xiang. Information-theoretic measures for anomaly detection. In *IEEE Symposium on Security and Privacy*, pages 130–143, 2001.

[30] T. Lunt, R. Jaganathan, R. Lee, A. Whitehurst, and S. Listgarten.

[31] Teresa F. Lunt. A survey of intrusion detection techniques. *Computers & Security*, 12(4):405–418, 1993.

[32] Tersa F. Lunt, R. Jagannathan, Rosanna Lee, Sherry Listgarten, David L. Edwards, Peter G. Neumann, Harold S. Javitz, and Al Valdes. Ides: The enhanced prototype, a real-time intrusion detection system. In *Technical Report SRI Project 4185-010, SRI-CSI-88-12*, 1988.

[33] Vern Paxon. Bro: A system for detecting network intruders in real-time. In *7th USENIX Security Symposium*, 1988.

[34] P.A. Porras and P.G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. In *Nineteenth National Computer Security Conference*, 1997.

[35] Manikantan Ramadas, Shawn Ostermann, and Brett C. Tjaden. Detecting anomalous network traffic with self-organizing maps. In *RAID*, pages 36–54, 2003.

[36] M. Sebring, E. Shellhouse, M. Hanna, and R. Whitehurst.

[37] Michale M. Sebring, E. Shellhouse, M.E. Hanna, and R.A. Whitehurst. Expert systems in intrusion detection: A case study. In *Eleventh National Computer Security Conference*, 1988.

[38] R. Skardhamar. Virus: Detection and elimination. In *AP Professional*, 1996.

[39] Stephen E. Smaha. An intrusion detection system for the air force. In *Fourth Aerospace Computer Security Applications Conference*, 1988.

[40] S.R. Snapp, J. Brentano, G.V. Dias, T.L. Goan, L.T. Heberlein, C. Ho, K.N. Levitta, B. Mukherjee, S.E. Smaha, T. Grance, D.M. Teal, and D. Mansur. Dids (distributed intrusion detection system) motivation, architecture, and an early prototype. *internet Besieged: Countering Cyberspace Scofflaws*, pages 211–227, 1998.

[41] Salvatore J. Stolfo, Wenke Lee, Philip K. Chan, Wei Fan, and Eleazar Eskin. Data mining-based intrusion detectors: An overview of the columbia ids project. *SIGMOD Record*, 30(4):5–14, 2001.

[42] William T. Tener. Discovery: An expert system in the commercial data security environment. In *IFIP Security Conference*, 1986.

[43] H.S. Vaccaro and G.E. Liepins. Detection of anomalous computer session activity. In *IEEE Symposium on Research in Security and Privacy*, 1989.

[44] Shu Wenhui and Daniel Tan. A novel intrusion detection system model for securing web-based database systems. In *COMPSAC*, pages 249–, 2001.

[45] J.R. Winkler. A unix prototype for intrusion and anomaly detection in secure networks. In *Thirteenth National Computer Security Conference*, 1990.