Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2012 Proceedings

Proceedings

Database Intrusion Detection: Defending Against the Insider Threat

Kevin Barton
Computer Information Systems and Security, Our Lady of the Lake University, San Antonio, TX, United States., kabarton@ollusa.edu

Database Intrusion Detection: Protecting Against the Insider Threat

Kevin A. Barton
Our Lady of the Lake University
kabarton@ollusa.edu

Carol J. Jeffries-Horner
Our Lady of the Lake University
cjeffries@ollusa.ed

ABSTRACT

Not only are Databases an integral and critical part of many information systems, they are critical information assets to many business enterprises. However, the network and host intrusion detection systems most enterprises use to detect attacks against their information systems cannot detect transaction-level attacks against databases. Transaction-level attacks often come from authorized users in the form of inference, query flood, or other anomalous query attacks. Insider attacks are not only growing in frequency, but remain significantly more damaging to businesses than external attacks. This paper proposes a database intrusion detection model to detect and respond to transaction-level attacks from authorized database users.

Keywords

Database intrusion detection, insider threat, database security, inference attack, query flood, anomalous query

INTRODUCTION

The 2008 CSI Computer Crime and Security Survey (Richardson, 2008) reported continuing trends in the frequency and severity of insider abuse and financial fraud. From 2004 through 2011, respondents consistently reported insider abuse as the second most frequent type of security incident, trailing only behind malware (Gordon, Loeb, Lucyshyn, and Richardson, 2004, 2005, 2006; Richardson, 2007, 2008, 2011). The specific rate of insider abuse incidents ranged from a low of 42% of all reported incidents (Gordon et al., 2006) to a high of 59% (Gordon et al., 2004; Richardson, 2007). Financial fraud is another form of insider attack, and accounted for 8% to 12% of reported incidents between 2004 and 2011 (Gordon et al., 2004, 2005, 2006; Richardson, 2007, 2008, 2011). However, where the average rate of insider abuse has seen marked decreases in the past 10 years, from nearly 100% in 1999 to 42% in 2008, the average rate of financial fraud has remained relatively constant (Richardson, 2008). Not only are insider attacks frequent, they are also expensive. Financial fraud was reported as the single most expensive type of security incident, with average losses of almost $500,000 per respondent reporting such incidents (Richardson, 2008). Financial fraud has consistently been a source of high losses, being reported as the leading or second leading most expensive category of security incidents (Richardson, 2008). This is particularly noteworthy when compared to the rate of occurrence for malware incidents, which accounts for similar losses. Malware incidents are reported to occur at least five times more frequently than financial fraud, indicating the average loss per incident for financial fraud far exceeds that of any other computer security incident category (Richardson, 2008). The two insider related categories computer security incidents, insider abuse and financial fraud, account for a major portion of computer security losses (Richardson, 2007, 2008).

Databases are a central component of most information systems today, and therefore represent critical assets to many organizations (Liu, 2002). However, securing databases from attacks presents unique challenges (Bertino, Kamra, Terzi, and Vakali, 2005; Chen & Chu, 2008; Kamra, Terzi, and Bertino, 2007; Squicciarini, Paloscia, and Bertino, 2008). Well recognized and accepted technical controls, such as firewalls, antivirus software, access control methods, and file permissions protect information systems at the network or operating system level, but do not protect against database specific attacks (Bertino, Jajodia, and Samarat, 1995; Chen and Chu, 2008; Kamra and Bertino, 2010; Kamra et al., 2007; Liu, 2002). Network and operating system defense mechanisms are largely designed to defend against external attackers (Kamra and Bertino, 2010), but insider attacks are the primary threat in transaction-level database attacks (Liu, 2002). Current database security research is focused on techniques to mitigate insider attacks (Bertino et al., 1995; Hu and Panda, 2003; Kamra et al., 2007; Liu, 2002). Insider attacks against databases come in several forms. The most basic level is from authorized users who do not have database administrator rights (Hu and Panda, 2003; Kamra et al., 2007; Liu, 2002). These users have limited access to the database through a user account, but they do not have administrator privileges. The second category of insider threat is a group of authorized users who do not have database administrator rights, referred to as collaborators (Bertino, Leggieri, and Terzi, 2004; Chen and Chu, 2008). The individual members in the group do not have any more privileges than the single user noted in the first category, but because there are multiple users they can attack the database with greater stealth and potentially compromise more data than they could individually (Chen and Chu, 2008). A countermeasure designed to protect the database from an aggregation of malicious queries may not detect those same queries if they originated from collaborating users (Chen and Chu, 2008). The last insider category is database administrators (Kamra and

Bertino, 2010). Database administrators have the privileges to largely circumvent security controls designed to protect against authorized, but malicious users (Kamra and Bertino, 2010). Controls specifically designed to defend against privileged users, such as database administrators, are needed to counter this threat (Kamra and Bertino, 2010). This paper synthesizes current research on database intrusion detection and response, and proposes a database intrusion detection model that incorporates the key capabilities developed in earlier research. Database intrusion detection and response has been proposed as a defense against three categories of insider threats: standard users, collaborating standard users, and database administrators (Chen and Chu, 2008; Kamra and Bertino, 2010; Liu, 2002). Relevant studies are reviewed and limitations of those studies identified. The remainder of this paper provides a literature review, proposes a database intrusion detection module, indentifies areas for future research, and summarizes the paper.

LITERATURE REVIEW

Current database intrusion detection and response research can be grouped into two broad categories: architecture (Bertino et al., 2005; Kamra and Bertino, 2010; Liu, 2002) and type of attack (Bertino et al., 2005, 2004; Chen and Chu, 2008; Daswani and Garcia-Molina, 2002; Hu & Panda, 2003; Kamra et al., 2007; Squicciarini et al., 2008). Three types of attack are studied: inference attack (Kamra et al., 2007), anomalous query attack (Bertino et al., 2005; Hu and Panda, 2003; Kamra et al., 2007), and query flood attack (Bertino et al., 2004; Daswani and Garcia-Molina, 2002; Squicciarini et al., 2008). Prior research developed and tested database intrusion detection models (Bertino et al., 2005, 2004; Chen and Chu, 2008; Daswani and Garcia-Molina, 2002; Kamra and Bertino, 2010; Kamra et al., 2007; Squicciarini et al., 2008), while other research developed but did not test database intrusion detection systems (Hu and Panda, 2003; Liu, 2002).

Architectures

Five intrusion tolerant database architectures have been proposed (Liu, 2002). The core components of the proposed architectures consisted of the database management system and database themselves, a repair manager, and a database intrusion detection system (Liu, 2002). Additional components were added to the basic architecture to improve attack isolation and system resiliency, but each proposed architectural scheme retained the intrusion detector (Liu, 2002). Although the proposed architectures did not specify an intrusion detector, two functional requirements were noted (Liu, 2002). First, the intrusion detector must be capable of detecting attacks from multi-layers (Liu, 2002). Second, the intrusion detector must be capable of providing alarms or alerts (Liu, 2002). These database models (Liu, 2002) illustrate how to integrate an intrusion detection system into database systems to create intrusion tolerant database systems, but they do not specify what intrusion detection component to use.

Later research extended the intrusion tolerant database architecture by examining a potential architecture for a database intrusion detection system (Bertino et al., 2004). The new model recommended a three module intrusion detection system, including modules for training, comparison, and alarms (Bertino et al., 2004). The training module uses knowledge learning logic to develop profiles for each user role in the database system (Bertino et al., 2004). Role profiles are developed by analyzing the database log history to learn normal query patterns for each user profile (Bertino et al., 2004). As highlighted by the development of role profiles, the intrusion detection system is intended for database systems that employ role based access control (Bertino et al., 2004). This database intrusion detection system does not need to build, store, and maintain a profile for each individual user, but instead creates profiles for each user role (Bertino et al., 2004). This significantly reduces the knowledge learning workload, and makes the database intrusion detection system suitable for databases with many users (Bertino et al., 2004). This architecture could be used to detect multiple types of attacks, although the designers anticipated it would be used to counter anomalous queries (Bertino et al., 2004), and assumed anomalous query attacks would come from standard users (Bertino et al., 2004). As a result, the model proposed does not protect databases from malicious database administrators, or others malicious users who are able to gain administrator rights (Bertino et al., 2004; Kamra and Bertino, 2010).

A method to defend against tampering from database administrators strengthens the integrity of the database intrusion detection system (Kamra & Bertino, 2010). Tamper resistance can be achieved through an encryption-based system to sign the detection policies (Kamra & Bertino, 2010). The intrusion detection system validates the authenticity of the detection policies by verifying the attached signatures (Kamra & Bertino, 2010). The Joint Threshold Administration Model (JTAM) implements the concept of separation of duties using Shoup's (Shoup, 2000) RSA threshold signature scheme (Kamra & Bertino, 2010). Three important deficiencies with threshold signatures have been noted (Shoup, 2000). First, previous threshold signature schemes lacked rigorous security proof (Shoup, 2000). Second, signature generation and verification required synchronous communications (Shoup, 2000). Not only are synchronous communications not necessary, they are undesirable for threshold signature schemes (Shoup, 2000). Third, the signature size increased linearly with the number of signers, resulting in potentially large signatures that were difficult to manage (Shoup, 2000). Shoup (2000) addressed these deficiencies with a new and relatively simple encryption algorithm. This new encryption scheme was utilized to have

multiple database administrators sign a policy, requiring collusion between database administrators to tamper with policies (Kamra and Bertino, 2010).

Type of Attack

The second category of database intrusion detection research examines the type of attack, and proposes solutions (Bertino et al., 2005, 2004; Chen and Chu, 2008; Daswani and Garcia-Molina, 2002; Hu and Panda, 2003; Kamra et al., 2007; Squicciarini et al., 2008). The remainder of this section discusses research on three types of database transaction-level specific attacks: inference attacks, query flood attacks, and anomalous query attacks.

Inference Detection

Farkas and Jajodia (Farkas and Jajodia, 2002) reviewed early work on inference detection systems by examining studies on three broad database categories: statistical databases, multilevel secure databases, and general purpose databases. Early developments in database inference detection systems used schema level knowledge to detect potential inference risks during system design (Delugach and Hinke, 1996; Garvey, Lunt, Quain, and Stickel, 1992). However, schema level detection is insufficient (Yip and Levitt, 1998).

Five inference rules have been recommended for data level inference detection (Yip and Levitt, 1998). Although the list may not be comprehensive, the five proposed inference rules are: subsume inference rule, unique inference characteristics rule, overlapping inference rule, complementary inference rule, and functional dependency rule (Yip and Levitt, 1998). A prototype that implements the data level detection rules has been proposed (Chen and Chu, 2006). The prototype maintains a history of user requests and compares new requests to their history, with the goal of preventing users from accumulating enough data over multiple requests to infer data or knowledge they would otherwise be unauthorized to access (Chen and Chu, 2006). Requests that exceed a preset threshold are denied (Chen and Chu, 2006). The prototype also analyzes social relations to reduce inference risks from multiple collaborating users (Chen and Chu, 2006). Further development and subsequent testing of the prototype demonstrated its utility in mitigating inference attacks from both individual and collaborating users (Chen and Chu, 2008).

The Semantic Inference Model (SIM) (Swami and Sawant, 2010) was proposed as a conceptual model of a data level inference detection system similar to that proposed by Chen & Chu (Chen and Chu, 2008). SIM examines data dependency, schema level data, and semantic relationships to create security policy. SIM offered an interesting approach for the use of semantic data (Swami and Sawant, 2010). SIM proposed using domain knowledge to counter data mining threats (Swami and Sawant, 2010). Domain knowledge relates semantic terms that otherwise would not be considered equal, but that relate to a common domain (Swami and Sawant, 2010). For example, although 'window' and 'building' are not equal, they are in the same domain. Because these terms are not equal, separate queries on these terms would not be evaluated to determine the risk of inference. However, the semantic knowledge component of SIM would evaluate these inputs in the same domain (Swami and Sawant, 2010). User inputs are retained in history, and each new user request is evaluated against their history and security policy (Swami and Sawant, 2010). Queries that exceed the threshold are denied (Swami and Sawant, 2010). SIM supported Chen and Chu's (2008) method of detecting threat from collaborating users.

Query Flood Detection

A query flood attack is a database transaction level attack where a single malicious user or a group of collaborating malicious users send a large number of queries or updates to a database with the intent to deny access to other authorized users (Bertino et al., 2004; Squicciarini et al., 2008). Although the threat of query flood attacks had been postulated (Bertino et al., 2004; Daswani and Garcia-Molina, 2002), it had not been shown to be an effective way to disable or degrade a database until research empirically demonstrated query flood attacks can degrade the performance of both small and large databases (Squicciarini et al., 2008).

An early study on query flood attacks examined the threat of a query flood attack in peer-to-peer networks, noting peer-to-peer networks were particularly vulnerable to such attacks (Daswani and Garcia-Molina, 2002). Three types of countermeasures were recommended for query flood attacks in peer-to-peer networks: load balancing, proactive steps, and reactive steps (Daswani and Garcia-Molina, 2002). Peer-to-peer networks are inherently different from databases, and therefore the countermeasures proposed are not sufficient to protect databases (Squicciarini et al., 2008).

The concept of query flood attacks was extended from peer-to-peer networks to databases (Bertino et al., 2004). Network defenses are not capable of protecting databases from query flood attacks because application level attacks do not necessarily involve malicious network traffic (Bertino et al., 2004). The first method proposed to counter database query flood attacks involved the use of database log information to learn patterns of normal behavior (Bertino et al., 2004). Granularity is an important feature of the training function (Bertino et al., 2004). The query flood detection system should be capable of

identifying attacks from misuse and anomalous queries (Bertino et al., 2004). These attacks have different query patterns, therefore the query flood detection system should be tunable to both forms of attack, and capable of responding to either (Bertino et al., 2004). Granularity is a second important characteristic of the query flood detection system (Bertino et al., 2004). Two general approaches have been recommended, schismatic and catholic (Bertino et al., 2004). A schismatic approach allows the database administrator to independently evaluate relations between database tables, where the catholic approach considers the entire database as a whole (Bertino et al., 2004). Incoming queries are evaluated based solely on frequency of queries against the learned profiles of normal user behavior (Bertino et al., 2004).

Three deficiencies have been noted with query flood detection based on learning user behavior patterns from logs (Squicciarini et al., 2008). First, the user behavior learning process requires profiles of normal user behavior, and therefore requires a large amount of logged data (Squicciarini et al., 2008). Second, an attack can only be detected after it has occurred (Squicciarini et al., 2008). Third, because the method is very articulated and not sufficiently tested, it is not clear how practical the method is in an operational environment (Squicciarini et al., 2008). Research on query flood attacks has been extended in several ways (Squicciarini et al., 2008). First, empirical evidence demonstrated for the first time that query flood attacks can severely degrade database performance (Squicciarini et al., 2008). Second, the ability of caching mechanisms to provide an inherent defense against query flood attacks was formally tested (Squicciarini et al., 2008). Third, an algorithm to evaluate queries and detect query flood attacks was developed (Squicciarini et al., 2008). The algorithm is derived from the theorem of the central limit and other experimental studies, but it was not tested (Squicciarini et al., 2008). Although the use of user logs was noted as a deficiency with early query flood detection methods (Squicciarini et al., 2008), more recent research has failed to resolve the deficiency. User logs are still used to learn patterns of normal user behavior (Squicciarini et al., 2008).

A potential flaw exists in the extant research on query flood attacks. Denial of service attacks depend largely on asymmetry. Denial of service attacks try to achieve consuming more resources from the target than are consumed by the attacker, or by distributing the attacking resources across multiple attack sources. Asymmetry in query flood attacks could be achieved two ways: by frequency or complexity of queries. Extant research in query flood attacks only considers frequency (Bertino et al., 2004; Daswani and Garcia-Molina, 2002; Squicciarini et al., 2008), and has not examined the possibility the same effects could be accomplished with low frequency but complex queries, such as queries with large join operations. Failing to consider this possibility could enable an attacker to circumvent a query flood detection system.

Anomalous Detection

Anomalous query attacks represent a broad set of unspecific transaction level database attacks (Bertino et al., 2005; Hu and Panda, 2003). Inference detection and query flood detection systems would be tuned to detect and respond to very specific types of attacks (Kamra and Bertino, 2010; Squicciarini et al., 2008). However, attacks against databases are not limited to these types of attacks, so a more general intrusion detection capability is necessary (Kamra et al., 2007). Two approaches have been proposed: those for databases with role based access control (Bertino et al., 2004; Kamra et al., 2007), and those for databases without role based access control (Kamra et al., 2007).

Role based access control not only greatly simplifies database administration, but it also greatly improves the efficiency of database intrusion detection by reducing the number of user profiles required (Bertino et al., 2004; Kamra et al., 2007). However, not all databases employ role based access control, and therefore a database intrusion detection capability that is not dependent on role based access control is required (Kamra et al., 2007).

Intrusion detection in databases without role based access control presents significant challenges. Researchers have cautioned that even under normal conditions some users will have very few transactions, while other users will have frequent transactions (Kamra et al., 2007). Attempting to build profiles for individual users will create either excessively restrictive rules for frequent users, or excessively loose rules for infrequent users (Kamra et al., 2007). Grouping users with similar behavior patterns, and creating profiles for those "user groups" has been proposed as a solution (Kamra et al., 2007). User groups provide two advantages. First, they greatly reduce the number of profiles (Kamra et al., 2007). Second, they mitigate the potential problem of excessive false positives or false negatives (Kamra et al., 2007). Complementary approaches have been proposed (Hu and Panda, 2003; Kamra et al., 2007). One approach creates profiles based on user access patterns (Kamra et al., 2007), where another create profiles based on data dependency (Hu & Panda, 2003).

An algorithm for user profiling and intrusion detection that consists of five components has been developed (Kamra et al., 2007). Each transaction is characterized by these five components in what is referred to as a quiplet (Kamra et al., 2007). A quiplet consists of (a) the SQL command, (b) projection relation information, (c) projection attribute information, (d) selection relation information, and (e) selection attribute information (Kamra et al., 2007). The specific quiplet content will vary depending on the desired level of representation. Three levels of detail were proposed, coarse, medium, and fine (Kamra

et al., 2007). Initially, anomalous free log data is processed through a classifier used to develop user profiles. The classifier uses a Naïve Bayes Classifier (NBC) for learning and detection (Kamra et al., 2007). NBC was chosen because of low computational requirements and previously demonstrated applicability for this type of function (Kamra et al., 2007). Algorithm accuracy was assessed against test data. The algorithm achieved reasonably satisfactory results (Kamra et al., 2007).

DATABASE INTRUSION DETECTION MODEL

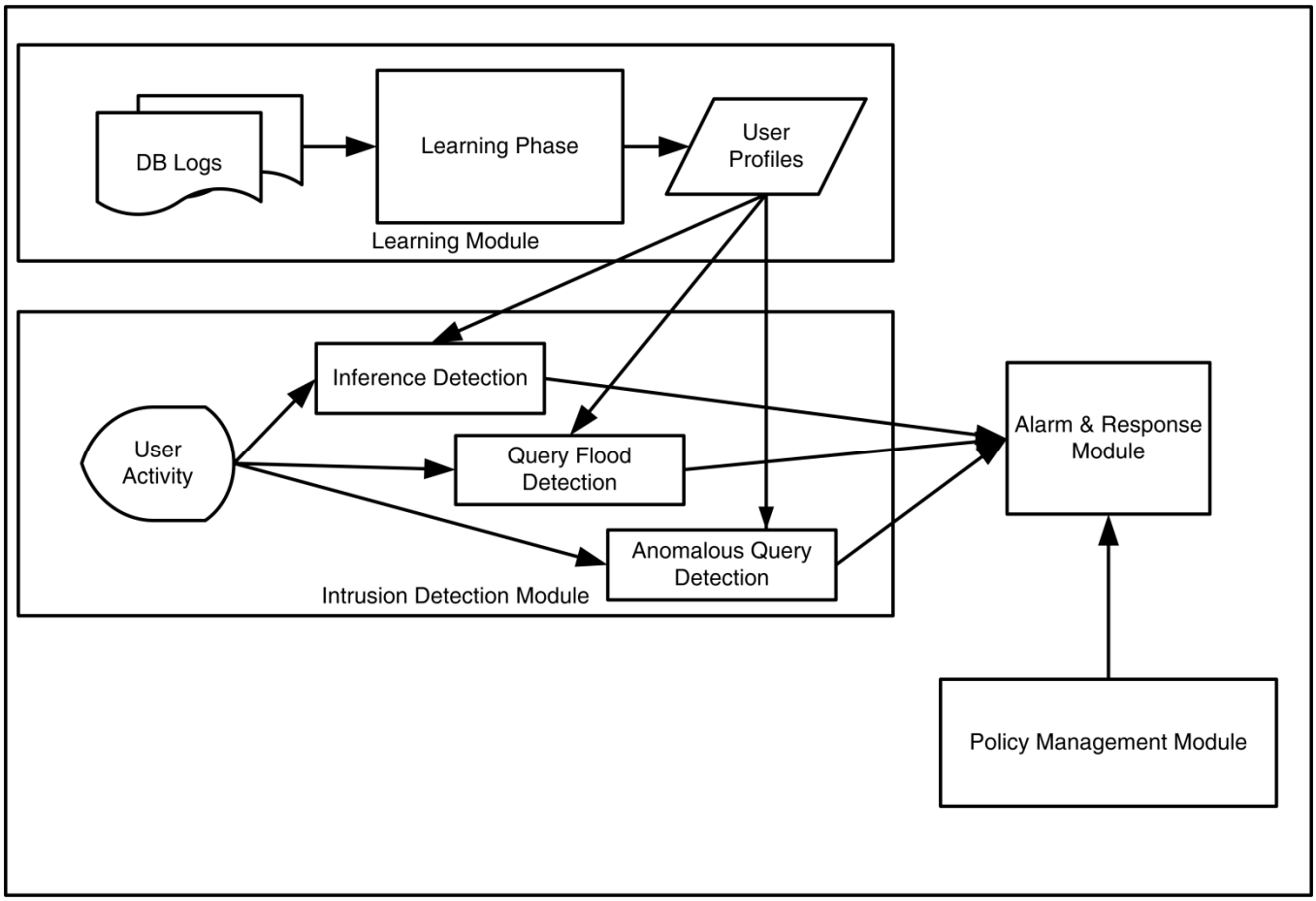

Figure 1. Database intrusion detection model

The proposed database intrusion detection model (DIDM) in Figure 1 includes four modules: (a) a learning module (Bertino et al., 2005), (b) an intrusion detection module (Liu, 2002), (c) an alert module (Liu, 2002), and (d) a policy management module (Kamra & Bertino, 2010). DIDM expands earlier work (Bertino et al., 2005; Kamra & Bertino, 2010) by including specific modules for the three key transaction-level database attacks, as well as a security policy management module.

The learning module uses an NBC to learn user behavior patterns based on historical log data (Bertino et al., 2005; Kamra et al., 2007) to create profiles of normal user behavior. Users could be individual users, user groups, or user roles depending on the database. The intrusion detection module has sub-modules to detect inference, query flood, and anomalous query attacks. The inference detection sub-module is based on SIM. SIM is the most developed inference detection model reviewed, but did not include the capability to detect inference attacks from collaborating users. Therefore, a capability to detect attacks from collaborating users will be included (Chen and Chu, 2006, 2008). The query flood and anomalous query attack modules will also be based on earlier research (Kamra et al., 2007; Squicciarini et al., 2008). Each sub-module compares current user activity against the user profile. The intrusion detection module reports user activities that are anomalous to normal user behavior to the alarm and response module. The alarm and response module compares reported activities to security policy and responds with alerts according to security policy. The security management module updates security policies. JTAM (Kamra and Bertino, 2010), an RSA threshold signature scheme (Shoup, 2000), is employed to ensure security policies are not tampered with by users or administrators.

AREAS FOR FUTURE RESEARCH

DIDM is conceptual and has not been tested. Future research could examine the efficiency and effectiveness of DIDM, as well as the tradeoffs between efficiency and effectiveness. Much of the extant research on database intrusion detection examines either efficiency or effectiveness, either of which can be achieved at the expense of the other (Bertino et al., 2005). Future research should evaluate an optimal balance between accuracy and efficiency for different intrusion detection capabilities.

Database intrusion detection systems are not intended to meet an information system's full security requirements, or intrusion detection capabilities (Bertino et al., 2005; Kamra et al., 2007; Squicciarini et al., 2008), so a critical information system with a database should have not only a database intrusion detection system, but also network and host intrusion detection systems. Individual intrusion detection systems can create extensive amounts of information and audit log data. Administrators could be overwhelmed with the data generated by multiple intrusion detection systems. Future research might examine how to best integrate these various intrusion detection capabilities into an overall enterprise log management system.

Extant research on query flood attacks focuses on the frequency of queries. However, it's possible a low volume of queries that require extensive system resources could degrade or disable a database for legitimate users. Future research could examine how query complexity could be used in a database denial of service attack, and integrate that concept into query flood attack detection and response.

CONCLUSION

The research discussed in this paper states the need for database specific intrusion detection and response systems. Network and host intrusion detection systems are not capable of analyzing transaction level data, and therefore are not capable of detecting or responding to transaction level attacks (Bertino et al., 2005; Chen and Chu, 2006, 2008). Current database intrusion detection and response research does not provide complete and comprehensive systems, but does lay the foundation for such systems.

REFERENCES

1. Bertino, E., Jajodia, S., & Samarati, P. (1995). Database security: Research and practice. *Information Systems*, 20, 7, 537-556.
2. Bertino, E., Kamra, A., Terzi, E., & Vakali, A. (2005). Intrusion Detection in RBAC-administered Databases. *Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC'05)*, Tucson, AZ, USA, IEEE, 170-182. doi:10.1109/CSAC.2005.33
3. Bertino, E., Leggieri, T., & Terzi, E. (2004). Securing DBMS□: Characterizing and detecting query floods. *Proceedings of the 7th International Security Conference,* Palo Alto, CA, USA, Springer, 195-206. doi:10.1007/978-3-540-30144-8_17
4. Chen, Y., & Chu, W. W. (2006). Database security protection via inference detection. *Proceedings of the Third IEEE International Conference on Intelligence and Security Informatics,* San Diego, CA, USA, Springer, 452-458.
5. Chen, Y., & Chu, W. W. (2008). Protection of Database Security via Collaborative Inference Detection. *IEEE Transactions on Knowledge and Data Engineering*, 20, 8, 1013-1027. doi:10.1109/TKDE.2007.190642
6. Daswani, N., & Garcia-Molina, H. (2002). Query-flood DoS attacks in Gnutella. *Proceedings of the 9th ACM Conference on Computer and Communications Security - CCS '02*, New York, New York, USA, ACM Press, 181-192. doi:10.1145/586135.586136
7. Delugach, H. S., & Hinke, T. H. (1996). Wizard: A database inference analysis and detection system. *IEEE Transactions on Knowledge and Data Engineering*, 8, 1, 56-66.
8. Farkas, C., & Jajodia, S. (2002). The inference problem: A survey. *SIGKDD Exploration*, 4, 2, 6-11.
9. Garvey, T. D., Lunt, T. F., Quain, X., & Stickel, M. (1992). Toward a tool to detect and eliminate inference problems in the design of multilevel databases. *Proceedings of the 6th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, 149-162.
10. Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richardson, R. (2004). *Computer crime and security survey (2004 CSI/FBI)*. Computer Security Institute. Retrieved from http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2004.pdf
11. Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richardson, R. (2005). *Computer crime and security survey (2005 CSI/FBI)*. Computer Security Institute. Retrieved from http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2005.pdf

12. Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richardson, R. (2006). *Computer crime and security survey (2006 CSI/FBI)*. Computer Security Institute. Retrieved from http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2006.pdf
13. Hu, Y., & Panda, B. (2003). Identification of malicious transactions in database systems. *Proceedings of the Seventh International Database Engineering and Applications Symposium (IDEAS'03,)* 1-7.
14. Kamra, A., & Bertino, E. (2010). Design and Implementation of an Intrusion Response System for Relational Databases. *IEEE Transactions on Knowledge and Data Engineering*, 22, 12, 1-15. doi:10.1109/TKDE.2010.151
15. Kamra, A., Terzi, E., & Bertino, E. (2007). Detecting anomalous access patterns in relational databases. *The VLDB Journal*, 17, 5, 1063-1077. doi:10.1007/s00778-007-0051-4
16. Liu, P. (2002). Architectures for intrusion tolerant database systems. *Proceedings of the 18th Annual Computer Security Applications Conference,* Las Vegas, NV, USA, IEEE, 311-320. doi:10.1109/CSAC.2002.1176303
17. Richardson, R. (2007). *Computer crime and security survey (2007 CSI)*. Computer Security Institute. Retrieved from http://gocsi.com/sites/default/files/uploads/2007_CSI_Survey_full-color_no marks.indd_.pdf
18. Richardson, R. (2008). *Computer crime and security survey (2008 CSI)*. Computer Security Institute. Retrieved from http://gocsi.com/sites/default/files/uploads/CSIsurvey2008.pdf
19. Richardson, R. (2011). *2010/2011 computer crime and security survey*. Retrieved from http://gocsi.com/members/reports
20. Shoup, V. (2000). Practical Threshold Signatures. *Proceedings of EUROCRYPT,* 209-222. Bruges, Belgium.
21. Squicciarini, A. C., Paloscia, I., & Bertino, E. (2008). Protecting Databases from Query Flood Attacks. *Proceedings of the 24th International Conference on Data Engineering*, Cancun, Mexico, IEEE, 1358-1360. doi:10.1109/ICDE.2008.4497555
22. Swami, K. S., & Sawant, A. A. (2010). Securing database by using collaborative inference detection. *Proceedings of the Second International Conference on Computer Research and Development,* Kuala Lumpur, Malaysia, IEEE, 260-264. doi:10.1109/ICCRD.2010.47
23. Yip, R. W., & Levitt, K. N. (1998). Data level inference detection in database systems. *Proceedings of the 11th IEEE Computer Security Foundations Workshop,* IEEE, 179-189. doi:10.1109/CSFW.1998.683168