

DATABASE

TRENDS AND APPLICATIONS

Solutions for the Information Project Team • www.dbta.com

Volume 18, Number 1 • January 2004

Alternate Data Management Approaches for Security Applications

With log data management, events cannot be inserted into the database as fast as they are generated.

By Kevin Hanrahan

The term "relational database" is almost superfluous these days. Every major commercial database product -- Oracle, Sybase, DB2--is based on the same underlying relational model. The dominance of the relational model stems from a good reason: the underlying set-theory on which it is based allows relational databases to elegantly represent complex data sets and perform flexible and arbitrary queries.

Application developers reflexively gravitate to relational databases. Security application vendors are no exception; a number of vendors have built and released products which attempt to store millions of firewall, VPN, and IDS records, in addition to server and application logs, in one of these relational database products.

But customers too often find that their security event analysis application fails to manage the large volumes of security-related data needed for proper incident investigation or to meet regulatory compliance. This article examines the weak points of relational databases as they relate to security event management and suggests why companies should investigate alternative solutions to manage and analyze log data.

The Problem

The most immediate issue in log data management is that the events cannot be inserted into the database as fast as they are generated. A number of factors contribute to the database insertion bottleneck. The first issue is indexing. Queries against relational databases perform best when queries access the data used in an index. The natural ten-

dency is for developers to try to optimize performance by constructing an index to handle many, if not most, of the anticipated user queries.

In log management, however, this strategy backfires because the amount of data being inserted is far greater than the number of queries against the data. Each index on a table has a negative impact on the time needed to insert into the table.

Moreover, the efficiency of an index will also degrade over time, as records are added and deleted from the underlying table. A routine task in system administration is the periodic rebuilding of indices to offset this problem. In high volume log management applications, this period may be surprisingly short, often as little as a week. But, the rebuilding of indices against a table containing many millions of records will result in the underlying table being effectively unusable for both insertion and querying for many hours.

Commit and Rollback

The next problem is commit and rollback. The transactional nature of relational databases is indispensable in many applications. In a textbook example, in a financial transfer deducting an amount from one account and adding the amount to a different account involves two linked transactions. Ideally, both transactions should succeed, but if one should fail the other must also. It is unacceptable for one to occur without the other.

In the context of log management, however, no such dependencies exist. The inability to load a log record into the system does not invalidate any other

log record loaded along with it. Security applications that feed into relational databases may choose to minimize the number of transactions by loading fewer batches of many records. This causes a large amount of rollback space to be consumed, as well as an expensive recovery should one record fail in the large batch. Loading many small batches of fewer records has a much smaller recovery overhead, but generally results in lower throughput as the increased number of database commits slows the overall process.

A combination of filtering the amount of data sent to the database and buying sufficiently powerful hardware to handle the insertion of the reduced data, raises other issues, including space issues. In addition to the storage of actual data, relational databases typically require allocated disk space for internal use. If these segments are insufficient, insertions, deletions and queries against the table will fail even when there is ample space remaining in the data partition.

A typical rule of thumb used by some vendors is that 2.5 times more disk will be used than the size of the actual data. In other words, 100GB of disk will permit the storage of only 40GB of logs.

Users may also find that, in order to keep up with their logs, data is almost constantly being inserted into the database. The generally positive transactional nature of relational databases again becomes a liability in this scenario. While the tables are undergoing insertion, portions of the tables and indices will be locked, causing contention with concurrent query access.

Deletion must also be considered.

Deletion of data can be substantially more expensive than insertion of data in a single table. While the number of indices on the table will play a role in the deletion performance, the major issue is that the delete operation is a logged transaction, consuming substantial amounts of rollback space required to "undo" the delete should an error occur.

Finally, relational databases are highly complex applications, requiring specialized and well-compensated administrators who can keep the system running efficiently. Certain routine actions, like periodic deletes and index rebuilds can be scheduled. But a database that han-

dles millions of inserts and deletions per day and stores hundreds of gigabytes of data needs a competent administrator.

Conclusion

An enterprise security solution must fulfill a variety of requirements driven by regulatory, operational and legal demands. The first and foremost priority, however, is to collect and archive all relevant log files and event records generated throughout an organization. It's the only way to achieve complete visibility into business activities.

I have a healthy respect for the power of relational databases, but the problem of security log management is a very

different beast than customer relationship management and requires a different approach to tame it.

This type of solution requires that data is stored in a format optimized for event logs, which enables the storage of huge amounts of data economically, while providing redundancy and scalability. Log management solutions must address all of these issues while simultaneously enabling tens of terabytes of event data to be quickly loaded and queried in a cost-effective manner.

Kevin Hanrahan is the director of security strategy at Addamark Technologies. www.addamark.com