SANS INSTITUTE

Securosis, L.L.C.

Understanding and Selecting a Database Activity Monitoring Solution

By Rich Mogull

This Report Sponsored By:

Guardium® SAFEGUARDING DATABASES™

IMPERVA®

SECERNO A security brain for your database

sentrigo™

TIZOR

Author’s Note

The content in this report was developed independently of any sponsors. It is based on material originally posted on the Securosis blog but has been enhanced, reviewed by SANS, and professionally edited.

This report is released in cooperation with the SANS Institute.

Special thanks to Chris Pepper for editing and content support.

Thanks to Stephen Northcutt for review.

Copyright

Sponsored by Guardium

Guardium delivers the most widely-used solution for database activity monitoring, security and auditing. The company's enterprise security platform is now installed in more than 350 data centers worldwide, including 3 of the top 4 global banks, one of the world's largest PC manufacturers and a global soft drink brand. Guardium was recognized was "a Leader across the board" in The Forrester Wave: Enterprise Database Auditing and Real-Time Protection, Q4 2007, with the highest overall scores for Architecture, Current Offering, and Product Strategy.

For more information. please visit: http://guardium.com/

Sponsored by Imperva

Imperva SecureSphere award-winning products deliver activity monitoring, real-time protection, and risk management for business applications and databases. SecureSphere products offer proven, automated capabilities for achieving, maintaining and documenting regulatory compliance for over 2500 organizations in over 30 countries on 6 continents.

For more information on Imperva SecureSphere, visit www.imperva.com/dmg

Sponsored by Secerno

Secerno provides the world's most advanced, comprehensive and intelligent database activity monitoring and security solution. Secerno.SQL delivers the highest levels of protection against internal and external threats, optimizes compliance auditing and delivers the ability to improve the security of applications. Powered by proprietary SynoptiQ technology—based on breakthrough research into efficient grammatical clustering—Secerno.SQL automatically fingerprints the true intent of all database interactions, enabling organizations to see, prove and control-—with unprecedented granular analysis—exactly how data is accessed.

For more information on Secerno, visit http://www.secerno.com/?pg=secernoSQL-for-IDS

Sponsored by Sentrigo

Sentrigo Hedgehog is a software-only database activity monitoring and intrusion prevention solution that protects databases in real-time against misuse, prevents data theft and speeds up regulatory compliance, including PCI DSS, SOX, SAS70 and HIPAA. Sentrigo Hedgehog is downloadable and available for free evaluation.

To download a trial version of Hedgehog and for a chance to win an iPhone go to: https://www.sentrigo.com/sans_promo.htm

Sponsored by Tizor

Tizor provides the world's largest companies with the only database monitoring and protection solutions that can monitor and report on all critical data activity across the enterprise – including databases, file servers, and mainframe applications for compliance assurance for SOX, PCI, GLBA HIPPA, data protection and theft detection.

Mantra deploys faster with the lowest total cost of ownership compared to any other solution, without any changes to your database, network or applications.

For more information, please visit: http://www.tizor.com/Solutions/fastest-time-to-deployment

Table of Contents

Introduction to Database Activity Monitoring 6

A Key Technology For Security And Compliance 6

Defining DAM 6

Market Drivers 7

Use Cases 7

Technical Architecture 9

Base Architecture 9

Collection Techniques 9

Central Management 12

Aggregation and Correlation 12

Policy Creation 12

Alerts 13

Workflow 14

Reporting 15

Advanced Features 16

Content Discovery 16

Connection Pooled User Identification 16

Blocking and Enforcement 17

Application Activity Monitoring 17

Pre-Configured Application Policies 17

Pre-Configured Compliance Policies 17

Change Management 18

Vulnerability Assessment 18

The DAM Selection Process 19

Define Needs 19

Formalize Requirements 20

Evaluate Products 20

Internal Testing 20

Conclusion 22

The Foundation of Information-Centric Security 22

About the Author 23

About Securosis 23

About the SANS Institute 23

Introduction to Database Activity Monitoring

A Key Technology For Security And Compliance

Over the past five years we have seen major changes in both the threats we face online, and the regulatory compliance landscape we do business in. Both the bad guys and the regulators are now focused on our data, not just our networks. We see breach disclosures and the regulations meant to protect them growing every year, with no end in sight. But managing this risk is more complicated than simply dropping in a firewall or installing antivirus software. Our applications and databases run in complex environments with numerous dependencies and business requirements. While we want to protect our information, we need to do it in a way that doesn't materially interfere with doing business. To balance these needs we see new technologies arise, one of the most significant of which is Database Activity Monitoring (DAM). With an estimated market size of $40M in 2006, and approximately $60M to $80M in 2007, Database Activity Monitoring rivals Data Loss Prevention in terms of market size. DAM tools provide powerful, immediate, non-intrusive benefits for security and compliance, and a long-term platform for comprehensive protection of databases and applications.

DAM is an adolescent technology with significant security and compliance benefits. The market is currently dominated by startups, but we've seen large vendors starting to enter this space, although these products are generally not as competitive as those from smaller vendors. Database Activity Monitoring tools are also sometimes called Database Auditing and Compliance, or variations on Database Security.

Defining DAM

Database Activity Monitors capture and record, at a minimum, all Structured Query Language (SQL) activity in real time or near real time, including database administrator activity, across multiple database platforms; and can generate alerts on policy violations. While a number of tools can monitor various level of database activity, Database Activity Monitors are distinguished by five features:

1. The ability to independently monitor and audit all database activity, including administrator activity and SELECT transactions. Tools can record all SQL transactions: DML, DDL, DCL, (and sometimes TCL) activity.
2. The ability to store this activity securely outside the database.
3. The ability to aggregate and correlate activity from multiple heterogeneous Database Management Systems (DBMSs). Tools can work with multiple DBMSs (*e.g.,* Oracle, Microsoft, IBM) and normalize transactions from different DBMSs despite differences between SQL flavors.
4. The ability to enforce separation of duties on database administrators.

Author's Note: Although I call this product category Database Activity Monitoring, I don't believe that name sufficiently describes where the market is headed. Over time we will migrate towards Application and Database Monitoring and Protection, as products combine application and database monitoring with more options for active blocking, but it's still too early to use the more inclusive definition for the market as a whole. Some tools do already offer those options, but both the product and customer maturity still need to advance before we can justify the more comprehensive name.

Auditing must include monitoring of DBA activity, and solutions should prevent DBA manipulation or tampering with logs or recorded activity.

5. The ability to generate alerts on policy violations. Tools don't just record activity, they provide real-time monitoring and rule-based alerting. For example, you might create a rule that generates an alert every time a DBA performs a select query on a credit card column which returns more than 5 results.

Other tools provide some level of database monitoring, including Security Information and Event Management (SIEM), log management, and database management, but DAM products are distinguished by their ability to capture and parse all SQL in real time or near real time and monitor DBA activity.

Depending on the underlying platform, a key benefit of most DAM tools is the ability to perform this auditing without relying on local database logging, which often entails a substantial performance cost. All the major tools also offer other features beyond simple monitoring and alerting, ranging from vulnerability assessment to change management.

Market Drivers

DAM tools are extremely flexible and often deployed for what may appear to be totally unrelated reasons. Deployments are typically prompted by one of three drivers:

- Auditing for compliance. One of the biggest boosts to the DAM market has been increasing auditor requirements to record database activity for SOX (Sarbanes-Oxley) compliance. Some enterprises are required to record all database activity for SOX, and DAM tools can do this with less overhead than alternatives.
- As a compensating control for compliance. We are seeing greater use of DAM tools to address specific compliance requirements, even though database auditing itself isn't the specified control. The most common example is using DAM as an alternative to encrypting credit card numbers for PCI compliance.
- As a security control. DAM tools offer significant security benefits and can sometimes even be deployed in a blocking mode. They are particularly helpful in detecting and preventing data breaches for web facing databases and applications, or to protect sensitive internal databases through detection of unusual activity.

DAM tools are also beginning to expand into other areas of database and application security, as we'll see a bit later. Today, SOX compliance is the single biggest market driver, followed by PCI. Despite impressive capabilities, internally-driven security initiatives motivate a distant third of DAM deployments as most database security projects seem to be driven by compliance needs.

Use Cases

Since Database Activity Monitoring is so versatile, here are a few examples of how it can be used:

- To enforce separation of duties on database administrators for SOX compliance by monitoring all their activity and generating SOX-specific reports for audits.
- If an application typically queries a database for credit card numbers, a DAM tool can generate an alert if the application requests more card numbers than a defined threshold (often a threshold of "1"). This can indicate that the application has been compromised via SQL injection or some other attack.
- To ensure that a service account only accesses a database from a defined source IP, and only runs a narrow group of authorized queries. This can alert on compromise of a service account either from the system that normally uses it, or if the account credentials show up in a connection from an unexpected system.

- For PCI compliance some organizations encrypt the database files or media where they're stored, and also use DAM to audit and alert on access to the credit card field. The encryption protects against physical theft, while the DAM protects against insider abuse and certain forms of external attack.
- As a change and configuration management tool. Some DAM tools offer closed-loop integration with external change management tools to track approved database changes implemented in SQL. Other tools can then track administrator activity and provide change management reports for manual reconciliation.

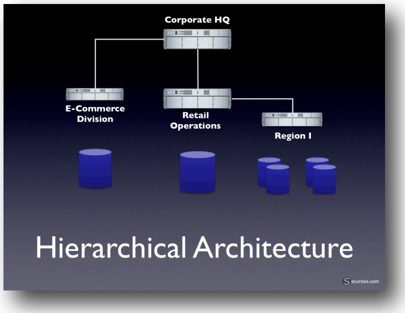
Technical Architecture

Base Architecture

One of the key strengths of DAM is its ability to monitor multiple databases running on multiple database management systems (DBMSs) across multiple platforms (Windows vs. Unix vs. ...). The DAM tool aggregates collected information from multiple collectors to a central, secure server. In some cases the central server/management console also collects information while in other cases it serves merely as a repository for collectors to send data.

This creates three potential options for deployment, depending on organizational requirements:

- Single Server/Appliance: A single server or appliance serves as both the sensor/collection point and management console. This mode is typically used for smaller deployments.
- Two-tier Architecture: This option consists of a central management server and remote collection points/sensors. The central server does no direct monitoring and just aggregates information from remote systems, manages policies, and generates alerts. The remote collectors may use any of the collection techniques, and feed data back to the central server.
- Hierarchical Architecture: Collection points/sensors aggregate to business-level or geographically distributed management servers, which in turn report to an enterprise wide management server. Hierarchical deployments are best suited for large enterprises which may have different business unit or geographic needs. They can also be configured to only pass certain kinds of data between the tiers to manage large volumes of information or maintain unit/geographic privacy and policy needs.



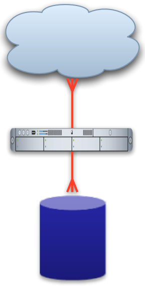
Whatever deployment architecture you choose, the central server aggregates all collected data (except deliberately excluded data), performs policy based alerting, and manages reporting and workflow.

These options focus on typical DAM deployments for database monitoring and alerting; as we delve into the technology we'll see additional deployment options for more advanced features like blocking.

Collection Techniques

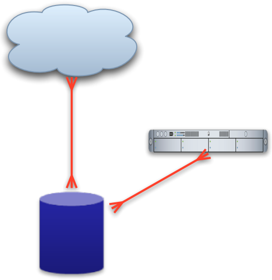
At the core of all DAM solutions are the collectors that monitor database traffic and either store it locally or send it to the central management server, depending on filtering rules and configuration. These collectors are, at a minimum, capable of monitoring SQL traffic. This is one of the defining characteristics of DAM and what differentiates it from log management, Security Information and Event Management, and other tools that also offer some level of database monitoring. There are three major categories of collection techniques.

- Network Monitoring: This technique monitors network traffic for SQL, parses the SQL, and stores it in the collector's internal database. Most tools monitor bidirectionally, but some early tools only monitored inbound SQL requests. The advantages of network monitoring are that it has zero overhead on the monitored databases, can monitor independent of platform, requires no modification to the databases, and can monitor multiple heterogenous database management systems at once. The disadvantages are that it has no knowledge of the internal state of the database and will miss any database activity that doesn't cross the network as SQL, such as logging in locally and remote console connections. For this last reason, network monitoring is only recommended when used in conjunction with another monitoring technique that can capture local activity. Network monitoring can still be used if connections to the databases are encrypted via SSL or IPSec by placing a VPN appliance in front of the databases, and positioning the DAM collector between the VPN appliance and the database, where the traffic is unencrypted.



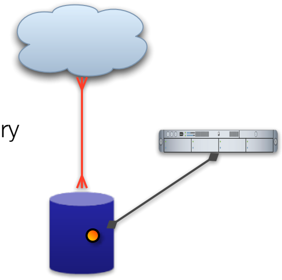
Network Monitoring

- Remote Monitoring: With this technique, the collector is given administrative access to the target database and native database auditing is turned on. The collector externally monitors the DBMS and collects activity recorded by native auditing or other internal database features that output activity data. The overhead on the monitored system is thus the overhead of the native logging/auditing. In some cases this is completely acceptable — in particular, Microsoft SQL Server is designed to provide remote monitoring of the database with little or no overhead. In other cases, particularly Oracle before version 10g, the overhead is material and may not be acceptable for performance reasons. Advantages include the ability (depending on DBMS platform) to monitor all database activity including local activity, performance equal to the performance of native logging/monitoring, and monitoring of all database activity, including internal activity, regardless of client connection method. The major disadvantage is potential performance issues depending on the database platform, especially older versions of Oracle. This also requires opening of an administrative account on the database and possibly configuration changes.



Remote Monitoring

- Local agent: This technique entails the installation of a software agent on the database server to collect activity. Individual agents vary widely in performance and techniques used, even within a product line, due to requirements for DBMS and host platform support. Some early agents relied on locally sniffing a network loopback interface, which misses some types of client connections. Current agents hook into the platform kernel to audit activity without modification to the DBMS and with minimal performance impact, or leverage shared memory monitoring. Leading agents typically impact performance by no more than 3-5%, which seems to be the arbitrary limit database administrators are willing to accept. Advantages include collection of all activity without turning on native auditing, ability to monitor internal database activity such as stored procedures, and potentially low overhead. Disadvantages include limited platform support (a new agent must be coded for every platform) and the requirement to install an agent on every monitored server.



Agent

Moving From Network, To Hybrid, To Agents

Which collection technique is best? The answer is "all of them". Different collection techniques have different advantages and disadvantages depending on circumstances. It often makes sense to use multiple techniques to meet the different needs of different databases. In some cases, network monitoring involves less database and management overhead, but it misses many types of database connections and activity. Audit log monitoring has lower overhead, but can be problematic on some platforms. Not all organizations are willing to install agents on critical database servers and current offerings aren't available on every platform. Thus a combination of techniques can allow for effective monitoring with acceptable performance impact.

Over the long term it won't be surprising if network monitoring fades away as agents and log monitoring improve their performance and platform support. All the leading network monitoring vendors already also provide agent or log monitoring capabilities, which provides deeper insight than network sniffing can provide. The exception may be deploying in-line devices for database intrusion prevention or firewalling.

Central Management

Aggregation and Correlation

The one characteristic Database Activity Monitoring solutions share with log management and even Security Information and Event Management, tools is their ability to collect disparate activity logs from a variety of database management systems. Where they tend to exceed the capabilities of related technologies is their ability to not only aggregate, but to normalize and correlate events. By understanding the Structured Query Language (SQL) of each database platform, they can interpret queries and parse their meaning. While a simple SELECT statement might mean the same thing across different database platforms, each database management system (DBMS) is chock full of its own particular syntax. A DAM solution should understand the SQL for each covered platform and be able to normalize events so the user doesn't need to know the ins and outs of each DBMS. For example, if you want to review all privilege escalations on all covered systems, a DAM solution will recognize those events regardless of platform and present you with a complete report without you having to understand the SQL.

A more advanced feature is to then correlate activity across different transactions and platforms, rather than just looking at single events. For example, some DAM tools can recognize a higher than normal transaction volume from a particular user, or (as we'll consider in policies) connect a privilege escalation event to a large SELECT query on sensitive data, which could indicate an attack. All activity is also centrally collected in a secure repository to prevent tampering or a security breach of the repository itself.

Since they collect a massive amount of data, DAM tools must support automatic archiving. Archiving should support separate backups of system activity, configuration, policies, alerts, and case management.

Policy Creation

One of the distinguishing characteristics of Database Activity Monitoring tools is that they don't just collect and log activity, they analyze it in real or near-real time for policy violations. While still technically a detective control (we'll talk about preventative deployments later), the ability to alert and respond in practically real time offers security capabilities far beyond simple log analysis. Successful loss-bearing database attacks are rarely the result of a single malicious query — they involve a sequence of events leading to the eventual damage. Ideally, policies will be established to detect the activity early enough to prevent the final loss-bearing act. Even when an alert is triggered after the fact, it facilitates immediate incident response, and investigation can begin immediately, instead of after days or weeks of analysis.

Policies fall into two basic categories:

- Rule-based: Specific rules are set up and monitored for violations. They can include specific queries, result counts, administrative functions (new user creation, rights changes), signature-based SQL injection detection, UPDATE or other transactions by users of a certain level on certain tables/fields, or any other activity that can be specifically described. Advanced rules can correlate across different parts of a database or even different databases, accounting for data sensitivity based on DBMS labels or through registration in the DAM tool.

- Heuristic: The DAM solution monitors database activity and builds a profile of "normal" activity (we sometimes call this *behavioral profiling*). Deviations then generate policy alerts. Heuristics are complicated and require tuning to work effectively. They are a good way to build a base policy set, especially with complex systems where manually creating deterministic rules by hand isn't realistic. Policies are then tuned over time to reduce false positives. For well-defined systems where activity is consistent, such as an application talking to a database using a limited set of queries, they are very useful. Heuristics, of course, fail if malicious activity is mis-profiled as good activity.

The more mature a solution, the more likely it is to come with sets of pre-packaged policies. For example, some tools come with pre-defined policies for standard deployments of databases behind major applications, like Oracle Financials and SAP. Yes, you'll have to tune the policies, but they are far better than starting from scratch. Built-in policies for PCI, SOX, and other generic compliance requirements may need even more tuning, but will help you jump-start the process and save many hours of policy building.

Policies should include user/group, source/destination, and other important contextual options. They should also support advanced definitions, such as complex multi-level nesting and combinations. Ideally, the DAM solution will include policy creation tools that limit the need to write everything out in SQL or some other definition language. You can't avoid having to do some things by hand, but basic policies should be as point-and-click easy as possible.

For common kinds of policies, including detecting privileged user activity and count thresholds on sensitive data, policy wizards are extremely useful.

Content-Based Policies

An emerging feature in some tools is support for content-based policies. As with Data Loss Prevention, these tools are able to analyze queries and results for specific content.

Identifying all known locations of sensitive data within multiple heterogenous database management systems is a complex process, even with the support of content discovery which we'll talk about later. Credit card and Social Security Numbers can easily be placed where they shouldn't be, either deliberately or by accident. Content based policies, typically using regular expressions, analyze database activity for unapproved use of sensitive data. For example, a policy could look for credit card numbers in any result set except those previously approved.

Note: All DAM tools evaluate the content of SQL statements. In this case we're talking about content *within* content. For example, a Social Security Number embedded in a query result set.

It's very early days, but we expect to see more and more content and context awareness in DAM tools over time. The most critical data we're usually trying to protect (at least these days) falls into structured formats we can define and detect outside its proper bounds (using data labeling and other registration techniques). Long-term, we'll be able to do really interesting things as we improve our ability to monitor and understand business context with the content, moving us ever closer to the elusive goal of stopping misuse of legitimate access.

Alerts

DAM tools should support both active alerting and an incident handling queue, similar to DLP. These alerts take a few different forms, from email integration, to self-contained events, to communications with outside security tools such as SIEM, using anything from SNMP to syslog to proprietary integration.

Policies should support granular alerting based on conditions, such as thresholds. For example, detection of a single errant query might trigger a low level incident within the included incident handling system, while an incident involving an administrator or high count of credit cards is emailed to a security admin and dropped into the SIEM tool as a high alert.

This is not to say that you should rely on a SIEM or other external tool to manage database incidents; those tools will never contain the full context and investigative abilities of a dedicated DAM workflow. External alerts play a valuable role in escalating incidents and correlating against external events, but the primary handling will tend to be managed within the DAM tool itself. Databases are complex beasts, and full understanding of what's going on internally requires a dedicated tool.

Policy-based alerts tend to fall into two or three interrelated categories which often overlap:

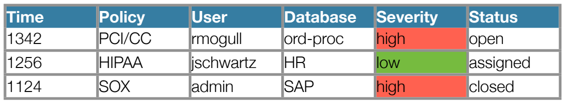
1. User activity: Incidents where a user takes an action that violates policy. It could be a user running a query on sensitive data, or updating an existing financial transaction outside of an application, or an application running a query never seen before.
2. Attack activity/signatures: Some DLP solutions include built-in detection for certain attack activity. This may be linked to vulnerability analysis, signature-based, or heuristic.
3. System and administrative activity: Incidents involving administrative or internal system activity, *e.g.,* new account creation, privilege escalation, DML/DDL changes, system updates, stored procedures, and other configuration changes. These alerts are typically being focused on SQL (and non-SQL) outside of simple SELECT, INSERT, UPDATE, and DELETE queries.

Workflow

Once an incident is created and any external alerts sent out, it should appear in an incident handling queue for management. This is similar to what we see in Data Loss Prevention and many other security tools, but optimized for database activity.

The queue should be visually well designed to make critical information easier to find, and allow customization for different work styles and interests. Unlike DLP, it's less important that the queue appeal to non-technical handlers since it's far less likely that anyone without database or security knowledge will work directly within the system. For DAM, we tend to rely more on reports for the auditors, risk managers, and other non-security types.

Incidents should be easy to sort and include color coding for sensitivity and criticality. When you click on an incident, it should let you drill down into more detail to assist the investigative process. Handlers should be able to assign, share, and route incidents to different users within the system. I'm a big fan of drop-down fields to change incident status right on the incident row. The system should also support role-based administration, allowing you to assign specific handlers/administrators based on the policy violated, database affected, and other factors.



Time	Policy	User	Database	Severity	Status
1342	PCI/CC	rmogull	ord-proc	high	open
1256	HIPAA	jschwartz	HR	low	assigned
1124	SOX	admin	SAP	high	closed

The basic workflow must allow for quick sorting, analysis, and investigation of incidents. Once an incident is detected, the handler can close it, add supporting investigative material, change the priority, assign it to someone else, or escalate it. To support investigations you should be able to correlate the current incident with other activity in that database by that user, violations of that policy across different systems, and other factors, in order to determine what's going on.

Since incident handlers may come from either a database or a security background, look for a tool that appeals to both audiences and supplies each with the information they need to understand incidents and investigate appropriately.

DAM products to date have focused on database-only incidents, but some systems are now expanding into platform activity on the database host and application activity.

Reporting

As with nearly any security tool, you'll want flexible reporting options, but pay particular attention to compliance and auditing reports to support compliance needs. Aside from all the security advantages we've been talking about, many organizations initially deploy DAM to meet database audit and compliance requirements. Built-in report templates can save valuable time, and some vendors have worked with auditors from the major firms to help design reports for specific regulations, like SOX.

Reports should fall into at least three broad categories: compliance and non-technical reports, security reports (incidents), and general technical reports.

Advanced Features

Although covering every possible feature is beyond the scope of this report, there are a few major features we see appearing frequently across products.

Content Discovery

As much as we like to think we know our databases, the reality is we really don't always know what's inside them. Many systems grow organically over years, some are managed by external consultants or application vendors, and others find sensitive data stored in unusual locations. To counter these problems, some Database Activity Monitoring solutions are adding content discovery features. These tools allow you to set content-based policies to identify the use of things like credit card numbers in the database, even if they aren't located where you expect them. Discovery tools crawl through registered databases, looking for content that matches policies, and generate alerts for sensitive content in unapproved locations. For example, you could create a policy to identify any credit card number in any database, and generate a report for PCI compliance. The tools can run on a scheduled basis so you can perform ongoing assessments, rather than combing through everything by hand every time an auditor comes knocking.

Some tools allow you to then build policies based on the discovery results. Instead of manually identifying every field with Social Security Numbers and building a different protection policy for each, you create a single policy that generates alert every time an administrator runs a SELECT query on any field discovered to contain one or more SSNs. As the system grows and changes over time, the discovery component identifies the fields matching the protected content, and automatically applies the policy.

We're also starting to see DAM tools that monitor live queries for sensitive data. Policies are then freed from being tied to specific fields, and can generate alerts or perform enforcement actions based on the result set. For example, a policy could generate an alert any time a query result contained a credit card number, no matter what columns were referenced in the query.

Connection Pooled User Identification

One of the more difficult problems we face in database security is the sometimes arbitrary distinction between databases and applications. Rather than looking at them as a single system, we break out database and application design and administration, and try to apply controls to each without understanding the state of the other. This is easy to see in the connection pooling problem. Connection pooling is a technique where we connect large applications to large databases using a single shared connection (or a small number) running under a single database user account. Unless the application was carefully designed, all queries come from that single user account (*e.g.,* APP_USR) and we have no way, at the database level, to identify the user behind the transaction. This level of abstraction makes it difficult, if not impossible, to monitor user activity and apply user policies at the database level.

An advanced feature of some Database Activity Monitoring solutions allows them to track and correlate individual query activity back to the application user. This typically involves integration or monitoring at the application level. You now know which database transactions were performed by which application users, which is extremely valuable for both audit and security reasons.

Blocking and Enforcement

Today, most users just deploy Database Activity Monitoring to audit and alert on user activity, but many of the tools are also perfectly capable of enforcing preventative policies. Enforcement happens at either the network layer or on the database server itself, depending on product architecture.

Enforcement policies tend to fall into two categories. The first, similar to many of the monitoring policies we've described, focus on user behaviors like viewing and changing sensitive records. Rather than just alerting when a DBA pulls every account number out of the system, you can block the query. The second is focused on database exploits; similar to an intrusion prevention solution, the system blocks queries matching signatures for known attacks like SQL injection.

The nature and level of blocking vary based on the architecture of the DAM tool. Integrated agent solutions may offer features like transaction rollback, while network tools might block the traffic from hitting the DBMS in the first place. Digging into specific architectures and benefits is beyond the scope of this report.

Application Activity Monitoring

Databases rarely exist in a vacuum; more often than not they are extensions of applications, yet we tend to look at them as isolated components. Application Activity Monitoring adds the ability to watch application activity, not just the database queries that result from it. This information can be correlated between the application and the database to gain a clear picture of just how data is being used at both levels, and identify anomalies which may indicate a security or compliance failure.

Since application design and platforms vary even more than databases, off-the-shelf products cannot cover every custom application in your environment. We see vendors focusing on major custom application platforms, such as SAP and Oracle, and monitoring web-based application activity.

As Database Activity Monitoring evolves into Application and Database Monitoring and Protection (ADMP), the combination of application integration, content awareness, and enforcement options will be critical.

Pre-Configured Application Policies

With or without application monitoring, some DAM solutions come with pre-configured policies for common applications (*e.g.,* PeopleSoft). Although they must be tuned to account for application customization, they can jump start the policy building process and save you from manually building all your compliance and security policies.

Pre-Configured Compliance Policies

Although no tool will make you compliant out of the box, pre-configured compliance policies for common platforms facilitate the process, especially when you don't know where to start. Most vendors hire or partner with auditors to help them build their compliance policy and reporting packages for common regulations, such as SOX and PCI-DSS, which frequently impact databases.

Change Management

Although many organizations have rigorous change management policies for the database platform and underlying system, far fewer enforce change management at the query level (which is invisible to traditional change management tools). An advanced feature of certain DAM solutions offers integration with external change management tools for closed-looped tracking of query-level changes. The requested change is approved in the change management system and a ticket number issued. The DBA enters that ticket number as part of their session, and all database changes (even to individual field updates) are recorded and correlated back to the original change ticket.

Vulnerability Assessment

Vulnerability assessment in databases is a large topic beyond the scope of this report, but is sometimes offered as a feature (usually at additional cost) of a DAM solution. Database vulnerability assessment tools look deeper than patch levels, down to specific configurations and even an analysis of user entitlements. Some tools integrate the results of VA into DAM policies to generate alerts (or block activity) that may target an identified vulnerability.

This section barely begins to cover all the advanced features offered by different products, and is provided only to give you an idea of a few common ones.

The DAM Selection Process

Define Needs

Before you start looking at any tools, you need to understand why you might need DAM and how you plan on using it, as well as the business processes around management, policy creation, and incident handling.

1. Create a selection committee: Database Activity Monitoring initiatives tend to involve four major technical stakeholders, and one or two non-technical business units. On the technical side it's important to engage the database and application administrators of systems that may be within the scope of the project over time, not just the one database and/or application you plan on starting with. Although many DAM projects start with a limited scope, they can quickly grow into enterprise-wide programs. Security and the database team are typically the main drivers of the project, and the office of the CIO is often involved due to compliance needs or to mediate cross-team issues. The technical staff will also map requirements to the technical platforms. On the non-technical side, you should have representatives from audit, as well as compliance and risk (if they exist in your organization). Once you identify the major stakeholders, you'll want to bring representatives together into a selection committee.
2. Define the systems and platforms to protect: DAM projects are typically driven by a clear audit or security goal tied to particular systems, applications, or databases. In this stage, detail the scope of what will be protected and the technical specifics of the platforms involved. You'll use this list to determine technical requirements and prioritize features and platform support later in the selection process. Remember that needs grow over time, so break the list into a group of high priority systems with immediate needs, and a second group summarizing all major platforms you may need to protect later.
3. Determine protection and compliance requirements: For some systems you might want strict preventative security controls, while for others you may just need comprehensive activity monitoring for a compliance requirement. In this step, map your protection and compliance needs to the platforms and systems from the previous step. This will help you determine everything from technical requirements to process workflow.
4. Outline process workflow and reporting requirements: Database Activity Monitoring workflow tends to vary based on use. When used as an internal control for separation of duties, security will monitor and manage events and have an escalation process should database administrators violate policy. When used as an active security control, the workflow may more actively engage security and database administration as partners in managing incidents. In most cases, audit, legal, or compliance will have at least some sort of reporting role. Policy Management, Operational

Who's in charge? One of the obstacles for successful database security is the split between database and security administration. DBAs have long managed security issues for their systems, but new compliance and security demands are pulling in security professionals who have limited database skills. The key to a successful program is to get these teams working together, along with application administrators and divide up duties based on expertise and security/compliance requirements. Each needs to then share expertise for DAM projects to succeed.

Management, Reports, and Remediation are all distinct roles for most public corporations, and should be reflected in your system processes and roles. Since different DAM tools have different strengths and weaknesses in terms of management interfaces, reporting, and internal workflow, thinking through the process before defining technical requirements can prevent headaches down the road.

By the completion of this phase you should have defined key stakeholders, convened a selection team, prioritized the systems you want to protect, determined protection requirements, and roughed out process workflow.

Formalize Requirements

This phase can be performed by a smaller team working under the mandate of the selection committee. Here, the generic needs determined in phase 1 are translated into specific technical features, while any additional requirements are considered. This is the time to come up with any criteria for directory integration, additional infrastructure integration, data storage, hierarchical deployments, change management integration, and so on. You can always refine these requirements after you proceed to the selection process and get a better feel for how the products work.

At the conclusion of this stage you will have a formal RFI (Request For Information) for vendors, and a rough RFP (Request For Proposals) to clean up and formally issue in the evaluation phase.

Evaluate Products

As with any products, it's sometimes difficult to cut through the marketing materials and figure out if a product really meets your needs. The following steps should minimize your risk and help you feel confident in your final decision:

1. *Issue the RFI:* Larger organizations should issue an RFI though established channels and contact a few leading DAM vendors directly. If you're a smaller organization, start by sending your RFI to a trusted VAR and email a few of the DAM vendors which seem appropriate for your organization.
2. *Perform a paper evaluation:* Before bringing anyone in, match any materials from the vendor or other sources to your RFI and draft RFP. Your goal is to build a short list of 3 products which match your needs. You should also use outside research sources and product comparisons.
3. *Bring in 3 vendors for on-site presentations and demonstrations:* Instead of a generic demonstration, ask each vendor to walk through your specific use cases. Don’t expect a full response to your draft RFP; these meetings are to help you better understand the different options and eventually finalize your requirements.
4. *Finalize your RFP and issue it to your short list of vendors:* At this point you should completely understand your specific requirements and issue a formal, final RFP.
5. *Assess RFP responses and begin product testing:* Review the RFP results and drop anyone who doesn't meet any of your minimal requirements (such as platform support), as opposed to "nice to have" features. Then bring in any remaining products for in-house testing. You’ll want to replicate your highest volume system and the corresponding traffic, if at all possible. Build a few basic policies that match your use cases, then violate them, so you can get a feel for policy creation and workflow.
6. *Select, negotiate, and buy:* Finish testing, take the results to the full selection committee, and begin negotiating with your top choice.

Internal Testing

In-house testing is the last chance to find problems in your selection process. Make sure you test the products as thoroughly as possible. A few key aspects to test, if you can, are:

- Platform support and installation to determine compatibility with your database/application environment. This is the single most important factor to test, including monitoring coverage for the connection methods used in your organization, since different database platforms support a variety of connection types.
- Performance. Is network or agent performance acceptable for you environment? Are there other operational considerations driving you toward one model or the other? Don't set arbitrary standards; monitor performance on your production systems to ensure your tests represent operational requirements.
- Policy creation and management. Create policies to understand the process and its complexity. Do you need to write everything as SQL? Will built-in policies meet your needs? Are there wizards and less-technical options for non-database experts to create policies? Then violate policies and try to evade or overwhelm the tool to learn its limits.
- Incident workflow. Review the working interface with those employees who will be responsible for enforcement.
- Behavioral profiling, if the product supports it for developing policies.
- Directory integration.
- Change management integration.
- Enforcement, blocking, rollback, and other advanced features.

Conclusion

The Foundation of Information-Centric Security

Database Activity Monitoring is an extremely valuable tool for compliance and security; it is critical to the emerging practice of information-centric security. Database Activity Monitoring gives insight into our most sensitive systems in a non-intrusive way, and can evolve into a proactive security defense. It's one of the few tools that can immediately improve security and reduce compliance overhead without interfering with business processes.

Although deploying a tool that crosses the organizational and technical expertise boundaries between database, security, and application management can be daunting, by understanding your needs and following a structured selection process you can help ensure a successful deployment. Many clients start with a narrowly scoped project which quickly expands throughout the enterprise.

From a business standpoint, the most important factor in a successful deployment is pulling together the key stakeholders across database, security, and application administration to determine initial requirements from a technical and process standpoint. In working with hundreds of clients implementing database security, the most common obstacle I've seen is a failure to manage expectations and responsibilities across these different groups.

After level setting and determining project goals, the most important technical necessity is to understand your platform support and performance requirements. I can't emphasize enough that DAM programs nearly always grow beyond the initial deployment, and you must consider this growth when defining product requirements.

Aside from growing in terms of systems covered, programs also expand into new use cases as administrators become more comfortable with the tools and their capabilities. You may start with auditing only, then add alerting for a few basic security policies, and end with full application integration and security blocking.

Database Activity Monitoring is a a powerful platform for compliance and information-centric security. It is relatively easy to deploy, non-intrusive, scalable, and flexible.

Securosis, L.L.C.

About the Author

Rich Mogull has over 17 years experience in information security, physical security, and risk management. Prior to founding Securosis, Rich spent 7 years as a leading security analyst with Gartner, where he advised thousands of clients, authored dozens of reports, and was consistently rated one of Gartner's top international speakers. He is one of the world's premier authorities on data security technologies, and has covered issues ranging from vulnerabilities and threats, to risk management frameworks, to major application security.

About Securosis

Securosis, L.L.C. is the independent security consulting practice of Rich Mogull.

Securosis provides security consulting services in a variety of areas, including:

- Security Management and Strategy
- Technology Evaluations (for end users and investors)
- Product Selection Assistance
- Security Market Strategies
- Risk Management and Risk Assessment
- Data Security Architecture
- Security Awareness and Education

Securosis partners with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis.

About the SANS Institute

SANS is the most trusted and by far the largest source for information security training and certification in the world. It also develops, maintains, and makes available at no cost, the largest collection of research documents about various aspects of information security, and it operates the Internet's early warning system — the Internet Storm Center.

Many of the valuable SANS resources are free to all who ask. They include the very popular Internet Storm Center, a weekly news digest (NewsBites), a weekly vulnerability digest (@RISK), flash security alerts, and more than 1,200 award-winning original research papers.