

Developing an Effective Policy for Database Activity Monitoring

The Benefits of a Knowledge-based Approach

Maximizing policy development is a critical step toward ensuring an accurate database activity monitoring (DAM) implementation. These systems, so they don't end up as glorified logging systems, require proper configuration based on the needs of the organization, and the unique environmental behavior of the business operation.

There are two fundamental approaches to DAM policy development: Knowledge-based and Learning-based. These approaches can differ in:

- The accuracy of the data collected as defined by the policy created.
- The time and effort that DBA resources need to dedicate to policy development.
- The long term validity of the policy in a dynamic database ecosystem.

Learning-based DAM solutions are hyped for their simplicity: turn on learning and automatically generate policy. In fact, learning-based systems are more complex and collect database activity over a random period of time. All activity collected during this window is deemed acceptable. Any activity not collected is deemed unacceptable. On the surface, learning-based systems are attractive, but in reality, they are complex and require considerable DBA time and effort to create accurate and useful policy.

Application Security's (AppSec) DAM solution employs a knowledge-based approach to policy development. Knowledge-based policy development is a guided, top-down process designed to provide an explicit and accurate policy while minimizing DBA time and effort in a dynamic database ecosystem.

The purpose of this white paper is to provide and understanding of the pros and cons of each approach.

PRECISION MONITORING VERSUS NONSPECIFIC MONITORING

A common mistake while deploying a Database Activity Monitoring solution has been the choice to log ALL activity that occurs at the database level. This is known as Non-Specific Monitoring. At the surface level it seems reasonable to log all activity (with the cost of storage space ever decreasing), however, in practice it presents all sorts of challenges with the most relevant problem being that too much useless information is stored, creating noise that can significantly increase network traffic and potentially introduce latency to critical business systems.

Database Activity Monitoring is best suited to logging relevant activity, and reacting to policy violations. This is known as Precision Monitoring. Setting up your audit policy requires understanding what should be going in and out of your databases and what is authorized versus unauthorized from a business operation perspective. For example, it's good to know that the database supports a financial package that performs some specific month-end and quarter-end activities, and that there are automated processes running after 10 p.m. EDT after the close of business on those dates authorized by a unique account.

To understand your environment, it is best to take a Knowledge-based (Top-Down) approach over a Learning-Based (Bottom-Up) approach—and we'll discuss why on the following pages.

THE LEARNING-BASED APPROACH

What is the learning-based approach? It's the ability to plug a device into the path of the database and allow it to "listen" to the database conversations that occur for a given period of time, such as a month. Over that time, through the powers of observation and mathematical algorithms, it will learn the behaviors that occur during that time and yield some high-probability of what is authorized and what is not. This is simple in concept, but not entirely useful in practice. Essentially, this is an automated means of reverse-engineering. Some common issues with this approach are:

1. It takes a long time
2. Can learn the wrong things
3. Requires highly-technical skills

IT TAKES A LONG TIME. A system must be set to a learning mode for an extended period time so that it can learn and profile an accurate set of behaviors that is a realistic representation of all actions that are valid. The ability to build a proper profile is correlated to the time spent observing. If you have specific processes that are infrequent like month-end or quarterly-activities, you'll need to ensure you are capturing that in your sample model. Imagine that you accept a new position at another company and you need to learn all the ins-and-outs of a specific department's operations, without interacting or asking any questions of anyone, how quickly can you get up to speed to make critical decisions about appropriate behavior and write-down the process for every finite service? What about the dynamic actions? How long do you have to observe before you have seen all the valid possibilities? A system in learning-mode takes time to build a reasonable profile, and yet it may not be accurate.

CAN LEARN THE WRONG THINGS. Building a behavioral model can be difficult if the environment is not the real environment. If you're building a model based on a test system, there is a good chance that it does not capture all the nuances of the production environment. You also have to assume that everything in your model is correct and valid and not implemented wrong. Changes to an environment will require behavior to be unlearned and then re-learned. The greatest concern exists when bad activity is already taking place. This will effectively treat it as valid and good behavior. For example, if an unknown piece of malware already exists in the environment, it may not even be discovered as a bad form of activity. The net result is the policy generated from learning mode is populated with false positives and negatives.

REQUIRES HIGHLY-TECHNICAL SKILLS. Because learning-based systems generate false positives and negatives, a

model certainly needs to be reviewed to be effective. But if you rely on a learning-based model, it is essentially a reverse-engineering task. The activities collected during the learning-phase will need some human-input to balance the assumptions that otherwise a machine will have to make. The data retrieved will be in the form of low-level SQL structured statements—not some clear objective statement—and there will be lots of them. A technical analyst will need to make sense of what it is trying to do, what it's accessing, what it might be writing or modifying. To do this properly requires someone to have some pretty good contextual knowledge of the applications that are accessing the database, the user-base, and the structure of the database schemas itself; otherwise, broad assumptions will have to be made.

While it's feasible that setting up a policy like this is possible, the room for error is very high. We certainly feel that starting and applying a Top-Down methodology using a Knowledge-based approach is clearer and faster, and involves stakeholders early in the policy-development phase that will yield a much more effective policy.

A SENSIBLE KNOWLEDGE-BASED APPROACH

The knowledge-based approach is the top-down process to describe all of the relevant interactions with the database. By understanding how the database supports the business and what the authorized interactions with the database are, the crafting of a database activity monitoring policy will be a straightforward process that provides accurate audit trails, while providing intrusion protection from various attack vectors.

The knowledge-based approach offers:

1. Clear definition of monitoring objectives
2. Transparency
3. Short implementation timeframe

CLEAR DEFINITION OF MONITORING OBJECTIVES. As opposed to reverse engineering, we start with defining the monitoring objectives by understanding the network and application system architecture. Who or what should have access to the database? And for what purpose? As a general rule, there are a set of prescriptive privileged actions and threat vectors that should always be tracked; we'll cover this in the next section on what to monitor. Then, there are the environmentally specific facts that define authorized activities. These facts will comprise a whitelist. A whitelist is simply a list or entries of valid behavior and conditions. Behavior not found in the whitelist should be scrutinized and audited.

TRANSPARENCY. Understanding what the system does should be straightforward. Systems and applications are really just steps in the business process, and understanding it just makes identifying unauthorized activity that much clearer. You may think as you read this, that this is easier said than done, and that the system underlying the processes may be overly complicated and the institutional knowledge is spread out. Start by involving the stakeholders and the people with the institutional knowledge to define who or what interacts with the database and what purpose. This will foster transparency in the policy process and help build a more accurate list of rules of what should be monitored or not. Typical involvement should include system architects, application owners, DBAs, and auditors. This helps increase awareness, strengthens communication, and reduces opportunity for fraud. Chances are you will find inefficiencies or errors that can be improved upon, but that is out of our scope.

SHORT IMPLEMENTATION TIMEFRAME. There are specific actions to always monitor for regardless of compliance mandates-since most of these actions are fundamentally tied to low-level activities that can be manipulated to open avenues for fraud or theft. These critical concepts are pre-built into DbProtect's SHATTER Knowledgebase, making implementation possible in a matter minutes. Preconfigured compliance templates (SOX, PCI, HIPPA, FISMA...) based on a valid library of rules provide an effective, time-saving starting point for policy development. In DbProtect, whitelist concepts can be defined quickly by using a highly-granular policy and filtering editor that incorporates concepts like origin and target locations, time, object, and content inspection with tools like regular expressions. Database activity monitoring policies can be defined as broadly or as specific as possible.

DbProtect Audit & Threat Management, the database activity monitoring module, leverages the SHATTER Knowledgebase, a library of known database security threats. This knowledgebase is backed by the largest team of dedicated database security researchers in the world that keep it updated for known database vulnerabilities, exploit techniques, security misconfigurations, and privileged actions that can be manipulated to put information at risk. This knowledgebase is kept current to keep abreast of current cyber-threats, and is easily updated.

WHAT TO MONITOR

Here are some definitive activities that should be part of your monitoring policy:

- 1. Intrusion Attempts.** Monitor for all attempts to exploit database vulnerabilities and usage of attack methods like SQL Injection. Regardless of success, failed exploitation or inappropriate probing are still active attacks.
- 2. Logins.** Keep a record of who is connecting to the database and when. This is important for investigative purposes in case an incident occurs.
- 3. Privileged User Activities.** This class of activity refers to administrative actions. These include SQL commands that can modify the system configuration, system availability, security features configuration, and the management of users and access. Authorization commands include GRANT/DENY/REVOKE statements and its derivatives. Many times these are reserved for administrative roles (that is the DBA), but is not limited to.
- 4. Schema Changes.** Monitor for changes to the structures of a schema. A schema typically supports a single application. These include the following class of SQL statements: CREATE/ALTER/DROP and any of its derivatives.
- 5. Access to Sensitive Data.** Identify where your sensitive data is located. Sensitive data can include personally identifiable information, financial data, regulated data such as health information or credit card data. Some examples of these are: Names, Social Security Number, National ID Number, Account Numbers, Date and Location of Birth, Email address, Credit Card Numbers, Phone Numbers, Zip Codes, Passwords, Passport Number, Taxpayer ID, etc.. This list is far from exhaustive and only serves as an example. Your organization should also consider intellectual property or data, if compromised, can damage your business or reputation as sensitive. Monitor both read and write activities to data structures that house the sensitive data. Read activities typically are done in the form of SQL queries (SELECT statements). Write activities are a classification of statements called DML which include INSERT/UPDATE/DELETE statements. Attention should also be paid to stored procedures and functions that access sensitive data.

The next set of activities that should be added to your monitoring policy is best described as is **Abnormal** or **Unauthorized Behavior**. Describing this class of activity is entirely unique to an environment and its processes. The best way to define abnormal, is to define what is normal, and to treat any exceptions as the abnormal.

Gather information about the network and system architecture surrounding the database. Build a list of the privileged users and applications that perform authorized business functions. This will serve as your whitelist. The objective here is to define the trusted activity, and capture the exceptions. This can also be used to help you identify how sensitive data should be handled in the database. This will supplement the privileged activities and threats described earlier in this section.

The chart above illustrates a database activity monitoring whitelist. These are authorized actions.

Using the information above, you can easily create filtered rules to eliminate noise and capture relevant activity that violates the database activity monitoring policy.

EARLY PREVENTION

It would be prudent here to briefly mention that database activity monitoring technology is one of several layers of database protection. Database activity monitoring provides

Database Activity Monitoring Whitelist — Examples of Authorized Actions

Who/Role	Client/ Application	Action	Access to What	When	Connect From Where	Target Database
Payables	Browser-based (iexplore.exe)	Data entry on approved invoice	Bill, Vendor, Address, Approvals	M-F, 9a to 5p EDT. Week before 15th and week before the 1st	192.168.0.6	192.168.1.10, port 1521 SID: FINPACK
Receivables	Browser-based (iexplore.exe)	Notified to invoice	Opportunity, Account, Products, Addresses	M-F, Every day, 9a-5p EDT	192.168.0.6	192.168.1.13, port 1521 SID: CRM

the compensating control to balance the powers provided to database administrators, so that proper governance can occur. But it should not be solely relied upon to compensate for weak security configurations of databases. By properly configuring a database and implementing security controls, most threats can be short-circuited with early prevention. Take advantage of the SHATTER Knowledgebase to securely configure databases and to set up authorizations with the appropriate restricted access.

MORE INFORMATION

DbProtect Audit & Threat Management is a database activity monitoring solution that offers the best security policy for monitoring your database environment with shortest time to deployment, for a dynamic database environment.

For more information about DbProtect, please visit our website at www.appsecinc.com.

Application Security, Inc. offers product training to assist you in successfully implementing and maximizing DbProtect. To find out more about how our training services can help you, please visit us online at:

<http://www.appsecinc.com/support/services/index.shtml>

ABOUT APPLICATION SECURITY, INC. (APPSECINC)

AppSec is a pioneer and leading provider of database security, risk and compliance (SRC) solutions for the enterprise. By providing strategic and scalable software-only solutions – AppDetectivePro for auditors and IT advisors, and DbProtect for the enterprise – AppSec supports the database lifecycle for some of the most complex and demanding environments in the world across more than 2,000 commercial and government customers.

Leveraging the world's most comprehensive database security knowledgebase from the company's renowned team of threat researchers, TeamSHATTER, AppSec products help customers achieve unprecedented levels of data security from nefarious or accidental activities, while reducing overall risk and helping to ensure continuous regulatory and industry compliance.

For more information, please visit: www.appsecinc.com | www.teamshatter.com

For a free database vulnerability assessment visit: <http://www.appsecinc.com/downloads/appdetectivepro/>

Follow us on Twitter: www.twitter.com/appsecinc | www.twitter.com/teamshatter

**APPLICATION
SECURITY, INC.**
www.appsecinc.com