Use Guardium outlier detection to detect hidden threats

Advanced mining techniques can help you find a needle in a haystack

Kathryn Zeidenstein
Guardium Evangelist
IBM

Oded Sofer (ODEDSO@il.ibm.com)
Guardium Business Analyst
IBM

Allon Adir (ADIR@il.ibm.com)
Research Scientist
IBM

Rosa Miroshnikov (rosam@ca.ibm.com)
Senior Software Engineer
IBM

Learn how Guardium® database monitoring is extended with enhanced security intelligence—known as outlier detection—to detect potential threats based on relative changes in behavior. Use cases, configuration guidance, and operational considerations are covered.

Too much information

The infamous 2013 data breach at a major retailer revealed that security alarms raised by their monitoring software were often ignored or at least deemed not worth further investigation. This should not be a surprise. Security analysts are bombarded with false positives and without any indication of relative risk, there is no way to prioritize the analysis.

Even worse after the attackers are inside the network, they are often able to steal additional credentials and gain unfettered access to the "crown jewels" resident in an organization's database server. The attackers can then take their time and do this over a long time without detection.
Similarly, SQL injection attacks that occur under application privileges might also be able to access sensitive data under the cloak of normalcy.

This article describes a way to extend traditional database monitoring with increased intelligence to help you understand the risks based on relative changes in behavior.

**How outliers can help: Scenarios**

For example, if Joe the DBA is observed accessing a particular table many more times than he has in the past, it could be that he is slowly downloading small amounts of data over time. If an application generates more SQL errors than it has in the past, maybe there is a SQL injection attack under way.

Consider a valid bank transfer transaction, which can translate to tens or hundreds of SQL statements that ensure proper validation and authorizations, get the account number, calculate balance details, and so on. An attacker or malicious insider will most likely bypass the "normal" way to access data. By bypassing these hundreds of SQL statements, that attacker could be flagged as abnormal, an "outlier."

A disgruntled DBA decides to extract the entire contact list into a CSV file or other format that they can take with them, or dump all the sensitive data into a database. The outlier detection algorithm can detect that access to this particular source is coming from a DBA who is not supposed to extract from operational data. There might also be an increase in exceptions (errors) as the user bypasses access control mechanisms and tries to learn the structure and privileges needed. The volume that is created by the download within the time window is probably exceptional as well. Any or all of these incidents could trigger an outlier indication.

A DBA attempts to put some buggy code into stored procedures that will "blow up" after they are gone to prove how much they were needed. The algorithm can identify that there is an exceptional volume of errors because DBAs are not supposed to access the stored procedure object. It can also detect if the DBA was temporarily granted elevated privileges to hack the stored procedure.

**How outlier detection works**

IBM Guardium Data Activity Monitoring includes an advanced Machine Learning algorithm to aid in the early detection of possible attacks during operation. The algorithm automatically models the normal patterns of a user's activity without requiring any supervision. That is, it is based mainly on unsupervised learning techniques. Clustering users according to behavior captures real effective groups, unlike rule-based security systems that rely on predefined groups.

The process checks not only user activities to see whether they are consistent with that user's previous activities, but also models that user's actions against activity of similar users. For example, a new behavior by a user, when compared with a cluster of similar users, might be consistent and normal for that type of user. This two-pronged approach reduces the number of false positives.
The outlier detection algorithm uses data that is being collected normally for security and compliance reasons. If data is not being audited already by a Guardium security policy, it is not available for Guardium to analyze.

Look at Figure 1 for a quick review of how Guardium collects audit data during its normal operations.

**Figure 1. How Guardium collects and logs database event data**

1. A database user logs in and enters a database command.
2. The Guardium software tap (S-TAP) on the database server captures the activity and sends a copy to the Guardium collector, which is a hardened hardware or software appliance.
3. The analysis engine on the collector parses the command and breaks it down into its component parts for reporting or other analysis. That logged activity includes the user, the time they issued the command, where they logged in from (client IP), what source program they used (JDBC, for example), which database server they accessed, and what data was accessed.

As shown in Figure 2, outlier detection operates on a subset of audit data that is transparently extracted from the collected audit data regularly into a separate data mart. (This extraction occurs hourly, but might be configurable in the future.) There are two phases to outlier detection: training and analysis.

**Figure 2. Outlier detection relies on both a training phase and an analysis phase**

Assuming the model has been trained and analysis is active, new activities are compared against the existing model that represents the normal behavior. Events that fall outside of the established clustering are assigned an anomaly score and a reason for the score.

Use Guardium outlier detection to detect hidden threats
The next sections go into more detail on the training and analysis phases.

Training phase

During the training phase, the system learning the analysis uses the following aspects of user activity:

- **Who**: Database user, operating system user, and client host name.
- **What**: The object—table, stored procedure, view, or synonym—and the command, such as select, update, delete, or insert.
- **When**: Statically determined work hours (9 AM to 5 PM) and weekends (Saturday and Sunday) and off hours.
- **Where**: The source program, the database name, and database server.

The normal behavior of the database activity and the database users (roles) are modeled from the perspective of each of the above aspects. The model also does user profiling based on their behavior.

Ideally, the model needs 3-4 weeks of data to create a model based on the dynamic clusters of normal behavior. No outlier indicators are generated until sufficient training has taken place. (You can reduce the initial training period by using a Guardium API command, but expect many more false positives until the models have had time to be fully trained.)

Example

**Example**: Figure 3 is a highly simplified representation of the output of a weekly outlier model for user 'assange' and his use of temporary tables. It provides an average for his use of temporary tables (18) during working hours and a standard deviation (3).

**Figure 3. Weekly training generates enhanced models of normal behavior**

![Image of Figure 3]

**Important**: The past use of temporary tables is just one of many aspects of a user's behavior that is trained and modeled to enable comparison with new observed behavior. Other aspects include, for example, use of temporary applications, volume of activity, time of day/week of activity, rareness of activity, and others. Each of these aspects is handled similarly to what is shown above to create separate dependent or independent scores. The scores are combined and weighted to produce the final score.

**Retraining**: By default, after the model is built, the training phase is re-executed weekly. The frequency of retraining must be balanced. Too frequent training could cause noise in the normal model. If you do get repeated false alarms because of the retraining, you can use the GUI or the Guardium API to tell the system to ignore certain events.
Analysis phase

The analysis phase is the runtime comparison of the learned models of typical historical behavior against new activities that are captured by Guardium. When the models have been adequately trained, analysis occurs on incoming data activity, and outlier data begins appearing in the Guardium interface and reports.

Example: Figure 4 shows that during normal work hours, user 'assange' increased his use of temporary tables beyond what was modeled during weekly training. The increase was significant enough to generate an outlier signal.

Figure 4. Hourly runtime analysis triggers an outlier event

Remember, the data mart is continually being fed new activity data based on your security policies and thus, new outliers can be detected based on the configured detection interval of 1 hour.

You can influence the accuracy of the analysis algorithms by indicating when specific events can be ignored. See Excluding events from outlier detection for more information.

Important: Any sensitive data objects (those in your sensitive object group) or admin users (for example, users in your privileged user group who can access sensitive data) get a higher boost in scoring. These are the groups that, by default, get a higher score. To add additional groups, see Customizing outlier detection.

Table 1. Default groups that get scoring boosts

<table>
<thead>
<tr>
<th>Guardium Group ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Admin Users</td>
</tr>
<tr>
<td>5</td>
<td>Sensitive objects</td>
</tr>
</tbody>
</table>

Configuring outlier detection

Prerequisites

Outlier detection was introduced in Version 9.1 (Version 9, GPU 100). However, this article covers functions included up through V9.5 (Version 9 with GPU 530), which is the recommended release for using outliers. The following prerequisites and recommendations apply:

- Version 9.5 patch 530, available from IBM FixCentral
• It is strongly recommended that you enable outliers only on 64-bit collectors with a minimum of 24GB of RAM. (Note that outlier detection is not available on Aggregators or Central Managers.)

The presentation of the outlier results as described in this article is included with quick search. Therefore, you must ensure that quick search is enabled. Search is enabled by default on new installations of 64-bit systems, or you can use the command `grdapi enable_quick_search`. Another option is to review the Outlier Analytic List report, which does not require that quick search be enabled.

**CLI commands to enable and disable outlier detection**

Log in to the collector as a user or administrator with the CLI role. Use the following GuardAPI command to enable the Outliers Detection function.

```
grdapi enable_outliers_detection
```

If you enter this command with no parameters, the command starts extracting into the data mart on the current date every hour.

If you wanted to delay outlier detection, you could put a scheduled start date on the command:

```
grdapi enable_outliers_detection schedule_start="2015-06-10 00:00:00"
```

To disable outlier detection (which disables data mart extraction, training, and analysis), enter the following command:

```
grdapi disable_outliers_detection
```

**Interpreting the outlier results**

After the analysis phase becomes active, outlier data populates the Guardium system with the results of its analysis of real-time events. You can see this information on the summary chart at the top of the Quick Search user interface.

To get to Quick Search, click the microscope in the Guardium banner.

**Figure 5. Click search to open the Quick Search UI**

The summary chart (Figure 6) includes a blue line (with circles) to indicate the volume of activity for the particular tab selected (activity, errors, or violations).

Outliers show up as red and yellow indicators that reflect the severity or total outliers score for a time interval. Red indicators reflect highly anomalous events that require immediate attention. Yellow indicators represent less extreme anomalies that warrant attention as part of other or related investigations. The outlier score is a calculated aggregate value based on the volume of
outliers as compared to the predicted volume of outliers for a given time of day, the severity of individual outliers, and other factors.

For example, on a system that typically identifies zero outliers at 1 AM and 5-10 outliers at 1 PM during weekdays, the presence of two additional outliers (of two outliers at 1 AM or of 12 outliers at 1 PM) is more significant—and weighted more heavily—than the hourly total itself.

**Figure 6. How outliers appear on the Quick Search UI**

By hovering on one of the outlier icons, you can see the amount of activity in this time period and link directly to the detailed outliers or activities in the related time period.

**Figure 7. Hovering on an outlier alert**

**Why is this activity an outlier?**

The Outlier Reason column lists the reason or reasons that a particular activity was called out as an outlier:

<table>
<thead>
<tr>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rare</td>
<td>A seldom seen condition</td>
</tr>
<tr>
<td>High Volume</td>
<td>An unusually high incidence of a condition</td>
</tr>
<tr>
<td>New</td>
<td>A condition that is seen for the first time</td>
</tr>
<tr>
<td>Error</td>
<td>An unusually high incidence of error conditions</td>
</tr>
</tbody>
</table>
Outlier reasons are assigned in combinations when needed. For example, an outlier might be flagged as both rare and high volume if a seldom-seen condition suddenly occurs many times.

**Customizing outlier detection**

Although the Guardium outlier detection capability is designed to require minimal intervention to operate, there are some things that you can do to optimize the capability for your environment, such as adding additional groups of privileged users or sensitive objects, or by telling the system to ignore certain events.

In addition, although it is a bit more advanced, you can tweak other things that are related to the algorithm such as anomaly score thresholds.

**Boosting scores of users and objects**

As stated in the beginning of this article, there are two default groups that get scoring "boosts:"
Admin Users and Sensitive Objects. However, you might already have additional groups set up as part of your normal operating procedures that could also be useful for outlier detection. For example, you might be maintaining a group of Suspicious Users or you might have several different groups of sensitive objects that are aligned with different applications.

You can use a `grdapi` command to add additional groups to the outlier detection algorithm.

**Prerequisite:** This command requires that you know the Guardium group ID. To get the group ID, you can use the command `grdapi list_group_by_desc`. For example, if you have a group that is named BadGuys, you can enter the following command to get its Guardium group ID:

```
grdapi list_group_by_desc desc="BadGuys"
```

After you have the ID (let's assume it is 1234), you can add it to the outlier detection as follows:

```
grdapi change_outliers_detection_parameter privUsersGroupIds = 1234
```

You can do the same thing with sensitive objects:

```
grdapi change_outliers_detection_parameter sensitiveObjectGroupIds=333,156
```

**Excluding events from outlier detection**

If you want to exclude events from outlier detection, such as activity from test applications, you can right-click on a particular outlier and select **Ignore**.
Use Guardium outlier detection to detect hidden threats

**Figure 8. Right-click on an outlier to invoke actions**

![Outlier Detection Diagram]

You can ignore the entire event as is, as shown in Figure 9:

**Figure 9. All event conditions**

![All Event Conditions Diagram]

Or you can widen the scope by deleting specific event parameters. For example, if you want to ignore the source program MIPGE001 when running against SYBASE on a particular database server host, you would remove all the other parameters and click **OK**.

**Figure 10. Click on red X to remove a criteria**

![Remove Criteria Diagram]

**Note:** The values for date, time, and instances are ignored and do not affect exclusion criteria.

This feedback is recorded and can be reported on in the Analytic User Feedback report, as shown in Figure 11. The first line in the report shows what it looks like if you select an activity with no excluded criteria. The second line shows what it looks like if you select a subset of fields as criteria.

**Figure 11. Analytic User Feedback report**

![Analytic User Feedback Report]
If your user feedback includes a single criteria (user, server IP, database, and so on) it automatically populates one of the existing analytic exclusion groups as well:

- Analytic Exclude DB User
- Analytic Exclude OS User
- Analytic Exclude Server IP
- Analytic Exclude Service Name
- Analytic Exclude Source Program

For example, if you delete all criteria except for Application Source Program from the Outlier Response window, you could go to the Group Builder, edit the Analytic Exclude Source Program group, and see the item that you entered right there, as shown in Figure 12.

**Figure 12. Guardium Group Builder**

![Guardium Group Builder](image)

Of course, you can also use all the power of the Group Builder to populate the group in bulk, including populating from a query.

You can also use Guardium APIs to populate groups with single exclusion criterion:

```bash
grdapi create create_member_to_group_by_desc desc="Analytic Exclude Source Program" member="MIPGE001"
```

To include previously ignored events, view the Analytic User Feedback report, double-click the previously ignored event, and select **Invoke > delete_analytic_user_feedback**.

**Figure 13. Deleting an event from the ignored events**

![Delete Event](image)

You have the choice of invoking the deletion now or adding the generated command to a script to run later.

**Additional configurations and customizations**

We've already suggested the user of the Guardium API `change_outliers_detection_parameter` for a few different scenarios, such as adding additional user groups or sensitive objects for outlier
Detection consideration. Other aspects of outlier detection can be modified with this API, including increasing or decreasing the amount of time for training, alert thresholds, and more. You can see a list of parameters by entering:

```
grdapi change_outliers_detection_parameter --help=true
```

The parameters include:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cleanupKeepDays</td>
<td>This is how many days to retain model data on the collector. The default is 90 days. This is not training data, which is gathered weekly and then removed. This is the model data that is created from previous trainings.</td>
</tr>
<tr>
<td>SensitiveObjectGroupIds</td>
<td>The Guardium group IDs for objects (tables, views, and so on) to receive scoring boosts.</td>
</tr>
<tr>
<td>PrivUsersGroupIds</td>
<td>The Guardium group IDs for database users to receive scoring boosts.</td>
</tr>
<tr>
<td>minimalRequiredTrainPeriod</td>
<td>This is the default training period to build the initial model. The default is 10 days, but you will get more accurate models and fewer false positives if you do a longer time period, such as 30 days.</td>
</tr>
</tbody>
</table>

### Customizing scoring behavior (advanced users only)

The following API parameters are related to scoring. We recommend that you modify only the behavior of outlier scoring under the direction of someone knowledgeable in mining.

- **IntervalThreshold**
- **alertThreshold**
- **alertRarityThreshold**
- **alertVolumeThreshold**
- **minNumOfIntervalsForAlerts**

Before going into detail on each of these parameters, let's go into a little more detail on how scoring works.

*Intervals* represent an hour of activity of some user or database. Let's call these *user-intervals* and *database-intervals*. Each interval gets an anomaly score according to how anomalous the user or database's behavior was during the interval.

The GUI for outlier detection shows only user- and database-intervals that get scores higher than **intervalThreshold**. The reason the interval is deemed anomalous is the activity of the user or database during that hour. The specific actions that make up this hourly activity are also scored. When the GUI indicates that an interval is anomalous (this is called an "alert," which is not the same as Guardium real-time or threshold alerts), it also lists the highest scoring activities during the interval. Only activities that received a score higher than **alertThreshold** are displayed.

Usually, if an interval has a high enough score that it causes an alert, then there would be some high scoring activities in the interval that would also be displayed. But this is not always the case. For example, an interval could be anomalous due to an abnormally high volume of activities, even if each activity itself is normal.
As described in **Why is this activity an outlier?**, the GUI also shows the non-mutually exclusive reasons why an activity is deemed anomalous: rare, high-volume, new, and error. The final activity score is affected by several scores, including scores that relate to these four causes. The GUI indicates "rare" as a reason if the corresponding rarity score is higher than `alertRarityThreshold`. The GUI indicates a "high volume" reason if the corresponding volume score is higher than `alertVolumeThreshold`.

The anomaly detection is based on comparing current activity against historic activity of the user/database. For this detection to have significant statistical value it is necessary to learn from enough historic behavior. `minNumOfIntervalsForAlerts` defines how much is enough in terms of intervals (that is, hours) of activity of a user/database before corresponding alerts are given.

You can change the default values dynamically using the API. The defaults are:

<table>
<thead>
<tr>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>intervalThreshold</code></td>
<td>This is how many days to retain model data on the collector. The default is 90 days. This is not training data, which is gathered weekly and then removed. This is the model data that is created from previous trainings.</td>
</tr>
<tr>
<td><code>alertThreshold</code></td>
<td>Anomaly score threshold for activities in anomalous intervals. The default is 0.99.</td>
</tr>
<tr>
<td><code>alertRarityThreshold</code></td>
<td>Threshold determining when an anomalous activity is shown as &quot;rare.&quot; The default is 0.98.</td>
</tr>
<tr>
<td><code>alertVolumeThreshold</code></td>
<td>Threshold determining when an anomalous activity is shown as &quot;high volume.&quot; The default is 0.9998.</td>
</tr>
<tr>
<td><code>minNumOfIntervalsForAlerts</code></td>
<td>If the number of intervals in training is below this number, no alerts are generated. The default is 100 intervals (hours).</td>
</tr>
</tbody>
</table>

**Operational considerations**

This section describes more about how you can incorporate Guardium capabilities to integrate outlier detection with your operational procedures.

**Use distributed reporting to view outliers from multiple collectors**

To view consolidated outliers data from all collectors or from a group of collectors, you can create a distributed report based on the existing Analytic Outliers List report that is shown in Figure 14.

Distributed reporting is illustrated in Figure 14. Basically, each collector sends its data to the Central Manager on a scheduled basis. (There is also an option to create an online version that allows for adhoc viewing of the centralized report data.)

As input, you need the group of collectors that include outlier data. For details on creating distributed reporting, see the product documentation on Central Management. A direct link is included in Resources.

**Use distributed reports to view outliers from multiple collectors**

As with any report in Guardium, you can set up an automated process for distributing and reviewing outlier report data. This is sometimes known as **compliance workflow automation**.
Use the Audit Process Builder in the Guardium UI to create this process, including appropriate receivers, and add the Outlier report as a task. For more details on creating an audit process, see the link in Resources.

Set up correlation (threshold) alerts

Because outlier detection is a separate process from security policy rules and enforcement, you cannot set up real-time alerts on them. However, because outlier data is included in reports, you can create a correlation alert. A correlation alert is triggered by a query that looks back over a specified time period to determine whether the alert threshold has been met. A prerequisite is to ensure that you have enabled Anomaly Detection in the Guardium Administration Console.

For example, you can create an alert based on the query that is used in the Analytic Outliers Summary by Date report.

Figure 14. Analytic Outliers List Report

Assume that you want alerts that are written to syslog or sent using email. You can create an alert that runs this report query periodically and sets up the alert to be fired whenever there are one or more lines in the report that have an Anomaly Score greater than or equal to 99 over the past 4 hours. Instructions for creating correlation alerts are in the product documentation (see Resources).

Conclusion

We hope that you find this new use of data mining in Guardium as exciting as we do. The product and research teams continue to work closely together to enhance the algorithms as we get more feedback from customers, and enhancements will continue to roll out over time.
Resources

- Participate in the discussion forum for this content.
- "Target Ignored Data Breach Alarms" explains how Target's security team reviewed—and ignored—urgent warnings from threat-detection tool.
- Watch Tech Talk: A real-world case study identifying risk with InfoSphere Guardium to learn more about outlier detection in a real-world environment.
- Watch a Demo of quick search and outliers.
- Get involved in the Guardium Community on developerWorks. Connect with other Guardium users and find a host of technical material.
- Get more information in the Outlier detection topic in the Guardium Knowledge Center.
- Learn how to create correlation alerts in the Guardium Knowledge Center.
- Follow Guardium on Twitter.
About the authors

Kathryn Zeidenstein

Kathy Zeidenstein has worked at IBM for a bazillion years. Currently, she is working as a technology evangelist for Guardium data activity monitoring, based out of the Silicon Valley Lab. Previously, she was an Information Development Manager for InfoSphere Optim data lifecycle tools. She has had roles in technical enablement, product management and product marketing within the Information Management and ECM organizations at IBM.

Oded Sofer

Oded Sofer was the product manager at Guardium, PS director at DB-Motion Interoperability solution for Healthcare, and chief designer and VP of eProduct at Viryanet. He led and designed diverse, innovative, and pioneering products for integration and interoperability.

Allon Adir

Allon Adir is a researcher at the IBM research Labs at Haifa with an MSc in Computer-Science from the Technion University. Allon worked for several years on test and verification technologies for hardware systems. Currently, Allon focuses on data analytics technology in the hardware verification and the data security domains.

Rosa Miroshnikov

Rosa Miroshnikov is a software developer in IBM Analytics.

© Copyright IBM Corporation 2015
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)