InfoSphere Guardium data security and protection for MongoDB Part 2: Configuration and policies

Protect and secure a new generation of databases

Indrani Ghatare (indrani@us.ibm.com)
Software Engineer, InfoSphere Guardium
IBM

Matt Kalan (matt.kalan@10gen.com)
Senior Solutions Architect
10gen

Sundari Voruganti
QA Specialist, InfoSphere Guardium
IBM

Kathryn Zeidenstein
InfoSphere Guardium Evangelist
IBM

This article series describes how to monitor and protect MongoDB data using IBM InfoSphere® Guardium®, including the configuration of the solution, sample monitoring use cases, and additional capabilities such as quick search of audit data and building a compliance workflow using an audit process. Part 2 describes how to configure InfoSphere Guardium to collect MongoDB traffic and describes how to create security policy rules for a variety of typical data protection use cases, such as alerting on excessive failed logins, monitoring privileged users, and alerting on unauthorized access to sensitive data. Many organizations are just getting started with MongoDB, and now is the time to build security into the environment to save time, prevent breaches, and avoid compliance violations.

View more content in this series

Note: In October 2013, this article was updated to reflect a new recommended S-TAP configuration.
Introduction

The first part of this article series introduced MongoDB and InfoSphere Guardium. It provided some background on how InfoSphere Guardium can help you feel secure about using MongoDB in more applications because of its ability to thoroughly monitor and audit data activity. You learned the architecture of the solution and some of the recommendations on securing MongoDB using its native capabilities as well as the benefit that InfoSphere Guardium brings.

This second article goes into detail on two main topics:

- Configuring the S-TAPs on the MongoDB nodes
- Creating security policies for a variety of typical use cases

Part 3 will delve into some additional capabilities such as blocking users, how to write reports and access audit data, and how to automate compliance reviews.

Installation and configuration

This section describes the steps required to get up and running and start logging activity from MongoDB into the InfoSphere Guardium collector.

1. Install the S-TAP agent on the MongoDB nodes.
2. Configure the S-TAP inspection engine with the correct MongoDB ports.
3. Validate that you are seeing MongoDB traffic.

Before you begin

This article assumes that you have an InfoSphere Guardium collector installed and configured on the network. InfoSphere Guardium activity monitoring for MongoDB requires V9 GPU 50 or later releases. If you are an InfoSphere Guardium customer and are entitled to V9.0, you can download Guardium from Passport Advantage first and then install the GPU, which you can get from Fix Central.

Supported releases of MongoDB are 2.0, 2.2, and 2.4. From a data security perspective, it is recommended that you move to MongoDB 2.4 or later, which provides the security enhancements described in the introduction. (Kerberos requires Enterprise Edition.)

Make note of the following information, which you will need to complete the installation and configuration of the solution:

- IP address of the InfoSphere Guardium collector and the port used to connect to it (16016)
- The ports used by the mongod on the shard servers (default is 27018) and the IP addresses
- Port used by the routing servers (mongos) (default is 27017) and the IP addresses

Install the S-TAP agent on the MongoDB nodes

As shown in Figure 1, we recommend installing S-TAPs on the mongod shards as well as the routing server in order to monitor any administrator activity that may occur on the mongod shards.
S-TAPs are operating system-specific, so you'll need to install the Linux® S-TAP for each of the appropriate nodes. There are a couple of different ways to do this:

- Use Guardium Installation Manager (GIM). With GIM, you are actually installing a GIM agent along with the S-TAP. By using GIM, S-TAP upgrades and future installations can all be controlled from the web console without requiring access to the server again. This is what most enterprises use because of the simplicity of management and updates. See the InfoSphere Guardium information center for more information about GIM. See Resources for a link.
- Use the S-TAP shell installer that you download from Fix Central. This can be done non-interactively, which lets you install on many nodes with the same command.

The details for this process are outside the scope of this article, but refer to the InfoSphere Guardium information center for more information.

If your S-TAPs are configured properly to connect to the InfoSphere Guardium collector, the System View in the administration console will show green, as shown in Figure 2.

**Figure 2. System View shows that the S-TAPs are communicating with the Collector**

Configure the inspection engine

Next, you need to configure the inspection engine for each S-TAP. Inspection engines are how you define to the S-TAP which protocol to use for monitoring (MongoDB) and which ports to monitor.
By default, as shown in Figure 1, the port for mongos is 27017 and the port for mongod (shards) is 27018. Your ports may differ.

To configure the inspection engines, log in to InfoSphere Guardium as admin, and navigate to the Administration Console. From the left menu pane select, Local Taps> S-TAP Control. Locate the S-TAP for the Mongos server, click Modify, and then select the Add Inspection engine pulldown.

Enter the required port information. Your inspection engine configuration for mongos should look something like Figure 3.

**Figure 3. Mongos (query router servers) inspection engine configuration**

As you may know, in a sharded cluster, most query activity is routed through mongos and then routed to the mongods on the shards. If you monitored all traffic on all nodes, the Guardium collector would receive the same message from the mongos and from all the shards to which that command was routed. To avoid this "double counting," and yet still be able to catch traffic that may be coming from local connections on mongos, configure the STAPs on the mongos to capture only local traffic (Figure 3) and on the shards to capture all traffic (Figure 4).
Figure 4. Mongod (shards) inspection engine configuration

Using an API to configure inspection engines

If you have many nodes, you will want to use a Guardium API to add an inspection engine to a specified S-TAP. S-TAP configurations can be modified only from the active Guardium host for that S-TAP, and only when the S-TAP is online (showing green in the System Overview.)

For mongos:

```
grdpi create_stap_inspection_engine client=<ip address of mongos server>/255.255.255.255 protocol=MongoDB ktapDbPort=27017 portMax=27017 portMin=27017 stapHost=<ip of Mongos server where associated STAP is installed>
```

For mongod:

```
grdpi create_stap_inspection_engine protocol=MongoDB ktapDbPort=27018 portMax=27018 portMin=27018 stapHost=<ip of mongod server where STAP is installed>
```

Validate that traffic is being captured

There are several ways to tell if traffic is being sent to the Guardium collector. Experienced Guardium users can ensure a policy is installed that will capture all traffic and look at a report.

- If you log in as a user, on the View tab, you will see a bar chart named **Number of db per type**. You can double click in that report and drill down to see if there is activity.
If you are on a new installation of Guardium 9.0.0.50 or have upgraded and installed the new default policy called Default-Ignore Data Activity for Unknown Connections, you will not see detailed activity. Instead, you need to go to the Connection Profile List report, which will show just the high-level session information for any unknown connections, including those from MongoDB, which should all be unknown at this time. As a user, you will find this report on the View tab under DB Activities, as shown in Figure 6.

**Figure 6. Connection Profile List**

As an admin, you will find this report on the Daily Monitor tab.

The report looks something like Figure 7. It includes the database user name, client IP, and the entire "tuple" of connection information that identifies the connection such as the Client IP, source application, database user name, server IP, and service name.
If you are certain that your policy is configured correctly, and are still not seeing traffic, make sure you have the correct date and time ranges for the reports. If that is not the issue, then you may have some incorrect configuration in your S-TAP or inspection engines, such as incorrect port numbers.

Create groups to be used in policies and reports

An important planning exercise and productivity enhancer is to create groups. For example, you could create a group of admin (privileged) users, a group of sensitive data objects, and groups of certain commands, such as commands that assign users and privileges, and almost anything else. For this article, we’ll look at some monitoring use cases as well as how to create the policy rules to handle those use cases. Almost of all of those rules require the use of groups. Table 1 is a summary of the rules we’ll be creating and the groups that are used in each rule.

Table 1. Rules and groups used to create our sample policy rules

<table>
<thead>
<tr>
<th>Rule order</th>
<th>Rule description</th>
<th>Groups used in the rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alert for excessive number of failed logins per user per database in 3 minutes</td>
<td>None needed. See Real-time alert on excessive authentication failures.</td>
</tr>
<tr>
<td>2</td>
<td>Ignore detailed activity of functional user ID</td>
<td>Use the “tuple” group builder for the ClientIP/Src App./DB User/Server IP/Svc.Name field in the policy rule editor. See Ignore activity for functional users for details.</td>
</tr>
<tr>
<td>3</td>
<td>SkipMongoDB internal commands</td>
<td>MongoDBSkipCommands (prepopulated group, and you can add more as needed based on observed traffic). See Filter out noise commands for details.</td>
</tr>
<tr>
<td>4</td>
<td>Log full details of admin user activity</td>
<td>Admin Users - create your own or modify existing Admin Users group. In this article, we created our own. See Detailed monitoring of privileged users.</td>
</tr>
<tr>
<td>5</td>
<td>Alert when privileged user accesses sensitive data.</td>
<td>Admin Users MongoDB Sensitive Objects – Create your own Optional: Add a group of specific commands that you want to be alerted on. We created one called MongoDB WatchCommands See Real-time alert when privileged user accesses sensitive data.</td>
</tr>
<tr>
<td>6</td>
<td>Alert for MongoDB Access Control commands</td>
<td>No group needed. Log access to the system.users collection. See Real-time alert on Data Control commands.</td>
</tr>
<tr>
<td>7</td>
<td>Log policy violation for collection changes</td>
<td>MongoDB Change Commands - Create your own</td>
</tr>
</tbody>
</table>
In Part 3 of this article series, we'll cover an additional advanced capability in which you can use policy rules to block access in real time. This capability does require a license to advanced activity monitoring.

To create a group, access the Group Builder. If you are logged in as an admin, click on the Tools tab, and from the left menu pane select Config & Control > Group Builder. More details of the Group Builder interface are described in one of our policy rule examples.

**Configure a security policy**

Rule-based security policies are at the heart of how InfoSphere Guardium does its work. It is through these rules that you specify to InfoSphere Guardium what traffic to log, what conditions to alert on, and even what connections to block.

A new InfoSphere Guardium installation of 9.0.0.50 will include a default policy that ignores all traffic. This default helps protect your network from being overloaded when the S-TAP is activated and starts monitoring the database.

We cannot cover all the different types of policy rules and their behavior in this article. What we have chosen to do, instead, is pick some common monitoring use cases and show how to configure policy rules for these use cases. We'll cover these use cases in the next section of the article.

Now, let's create a new policy that you can use to start adding rules.

1. Click on the Tools tab, and from the left menu pane select Config & Control > Policy Builder.
2. From the Policy Finder, click New.
3. Give it a description, then click **Apply**.

**Figure 9. Give the policy a description**

**Optional**: Click **Roles** to associate which roles can use this new policy. For example, if you select admin, anyone with the admin role can work with this policy in the system.

4. Click **Back**.
Now you can edit the policy by adding rules that you need. We'll describe some "typical" rules in the next section. Only when you are ready to validate the behavior of a new rule or set of rules should you install this new policy.

**Monitoring use cases**

In this section, we'll include some additional policy rules that cover additional use cases that may or may not be appropriate for your organization, but which will give you some idea of how to start.

If you have never worked with InfoSphere Guardium before, a key concept is to understand that a policy can contain any number of rules. Each rule has a description, criteria (against which the monitored activity is evaluated), and an action that takes place when the rule is triggered.

There are three types of rules:

- **Access**: Used for the interaction between the database client and the server.
- **Exception**: Used for any exceptions that are returned by the database server to the client. Note that if you use write concern =0 or -1 (not safe) for your MongoDB connections, you will not be able to log and report on error conditions returned by any insert, update, or remove (delete).
- **Extrusion**: Applied on the returned data sets. This is an advanced capability, and we do not discuss it in this article.

**Real-time alert on excessive authentication failures**

A common requirement to guard against hackers who may be algorithmically generating passwords is to alert when the number of failed attempts within a session exceeds a threshold you define, such as over 5 attempts within 3 minutes.

For this rule, you will be defining an exception rule.

1. Select your new policy from the Policy Finder and click **Edit Rules**.
Figure 10. Edit the rules of your new policy

2. At the bottom of the Policy Rules page, click **Add Exception Rule**.

3. Fill out the policy criteria to specify LOGIN_FAILED from the pull-down menu in the Excpt. Type field. Include the minimum count (in this case 5) and reset interval (in this case 3 minutes).
4. At the bottom of the page, click **Add Action**, and select **ALERT ONCE PER SESSION** from the pull-down menu. This action will generate one alert per session when someone fails to successfully authenticate more than 5 times in 3 minutes.
5. Select a notification type. In our case, we chose SYSLOG and the default message template. Click **Add**, and then click **Apply**.

**Figure 13. Select notification type**

---

Sample alert: Figure 14 shows an example of the alert on the Incident Management tab when you are logged in as an admin.
Ignore activity for functional users or connections

Some organizations have periodic authorized jobs that do such things as batch updating or loading that run in the evenings or in specified batch windows. These applications are usually carefully vetted and run under a functional user ID. To avoid flooding the InfoSphere Guardium collector with activity that is not relevant for auditing, some organizations will use an access rule action called "Ignore S-TAP session."

Note that session start and end information is still logged (that is, timestamp, client IP, Server IP, user name, and so on). This rule just means that detailed command activity will be ignored.

Scheduling policies

InfoSphere Guardium enables you to schedule policy installations, which means you can have a different set of rules activated at night than you do during the day. You could add this rule to another copy of the policy that you automatically schedule to be installed at night when you know a maintenance job is running.

Recommendation: You could create a group of functional users and ignore activity from those users, but if you want to reduce the possibility of missing suspicious activity, you can use connection information to specify the rule. For example, you may want to ignore activity by functional user FUNC1234 coming from client IP 1.22.222.222, but if that user ID is accessing the system from any other IP you want to log the activity.

For that reason, we will create a group called "Functional MongoDB User Connections" and use that in our policy rule. We show both the manual way to populate the group and a way to automate group population by using the Connection Profile List report.

The name of the access rule field in the policy is, aptly enough, Client IP/SrcApp/DB User/Server IP/Svc.Name. This particular field with multiple components is known in Guardium as a "tuple."

You can use wildcards for any part of that connection information. Let's see how it works.

1. Select your new policy from the Policy Finder, and click Edit Rules.
2. At the bottom of the Policy Rules page, click Add Access Rule.

Figure 15. Add Access Rule
3. Give your new rule a name, then click on the group builder icon for the tuple field (Client IP/ SrcApp/DB User/Server IP/Svc.Name), as shown in Figure 16.

**Figure 16. Click on the group builder for the connection tuple field of the policy rule**

4. Fill in the attributes for each of the components of the tuple. You can use wildcards to indicate that everything will qualify. In this case, we have a naming convention for functional user IDs, so we are using that. In addition, we know the work these user IDs do is always coming from a specific client IP, so we are adding that as well.

**Figure 17. Add a tuple as a group member**

5. Click **Add** when you have filled in the attributes for one member. The group should look like Figure 18.
Figure 18. The tuple has been added to the group

When you are done adding members, click Back.

Select this group from the tuple field in the policy rule.

Click Add Action and choose IGNORE S-TAP SESSION from the pull-down menu. Click Apply. The rule now looks like Figure 19.

Figure 19. Ignoring S-TAP sessions for functional user (trusted) connections

6. When you are done adding members, click Back.

7. Select this group from the tuple field in the policy rule.

8. Click Add Action and choose IGNORE S-TAP SESSION from the pull-down menu. Click Apply. The rule now looks like Figure 19.

9. Click Save.
**Hint: Automating the process of populating the Functional User Connections group**

If your MongoDB traffic is already being monitored, you can automate this process using the built-in Connection Profile List report. If you are logged in as an admin, go to the Daily Monitor Tab and click on the **Connection Profiling List** from the left menu pane.

You will see a report similar to Figure 20.

**Figure 20. Example Connection Profile List**

![Figure 20. Example Connection Profile List](image)

At the bottom of the report, click on the **Invoke icon** to invoke the API `create_member_to_group_by_desc`. In the pop-up window, change the desc field to be the name of the group you want to add this connection to and then click **Invoke now**, as shown in Figure 21.
Filter out noise commands

This rule will filter out some of the "noise" commands that MongoDB is issuing internally, such as health checks and communication between servers. It uses a built-in group called MongoDB Skip Commands.

1. Select your policy from the Policy Finder and click **Edit Rules**.
2. At the bottom of the Policy Rules page, click **Add Access Rule**.
3. In the section of the policy rule labeled **Command**, choose the MongoDB Skip Commands group from the group pull-down menu, as shown in Figure 22.

**Figure 22. Select the MongoDB Skip Commands from the Group pulldown**

4. Uncheck the Cont. to next rule box if it is not already. (This saves processing because there is no further action that can possibly happen with any commands in this group.)
5. At the bottom of the policy rule, choose **SKIP LOGGING** as your action and click **Apply**.
6. Save your rule.

**Detailed monitoring of privileged users**

In 2.4, MongoDB supports many new roles whose scopes can be roughly divided into server-wide roles and database-level roles. In both cases, there are roles that focus on user administration, cluster administration, and application access.

Because some of these roles are basically equivalent to super users, it is important to ensure that these roles are carefully distributed and monitored.

Some organizations require that any activity by administrative (privileged) users be monitored in detail. The policy rule action to do this is LOG FULL DETAILS. Whenever you need to use LOG FULL DETAILS, you will capture exact timestamps and full details for each action. Be sure that you have appropriately sized your internal InfoSphere Guardium repository and the buffer size on the appliance to handle this workload, especially if your privileged users are reading or writing many documents.

**Prerequisite:** Create a group of MongoDB admin users (including anyone you consider "privileged") as described above.

Access your MongoDB policy and then click **Add Access Rule**.

Add a description and add your admin user group to the DB User field in the rule as shown in Figure 23.

**Figure 23. Excerpt of policy rule focused on DB User criteria**

Because we will be adding an alert on some admin user activity as the next rule, be sure to select the **Cont.to next rule** checkbox and select the action **LOG FULL DETAILS**, as shown in Figure 24.

**Figure 24. Continue to next rule ensures that Guardium will process the next rule even when this one is fired**

If you want to test the policy rules, you must install it. Go to **Tools > Policy Builder > Install and override**.
Real-time alert when privileged user accesses sensitive data

**Sensitive fields**

In MongoDB, you can also alert on activity at the field level. For example, you might want to do this if you knew that your document collection was only sparsely populated with sensitive data, such as driver's license numbers, and you didn't want to alert on all other access to documents in that collection. Note that if a field is nested more than one level deep in the document, the dot notation path to the field will be logged.

```javascript
db.CreditCard.insert(
    {
        "Name" : "Sundari Voruganti",
        "code" : "WM2001_0",
        "product" : "Gold Card",
        "profile" : [
            {
                "CCN" : "11999002"},
            {
                "log" : ["new", "customer", "for", "now"]
            },
            "otherinfo" : "Contact Bob Saget"
        ]
    });
```

In the above example, Guardium would store an object of CreditCard and the following fields:
Name, code, product, profile.CCN, profile.log, otherinfo.

You could set up an alert that contains %CCN% (for a credit card field) and %DLN% for a driver's license field and alert on access to those fields.

Alerts are an excellent way to get near real-time alerts on suspicious or out-of-compliance activity. Alerts are written to the Incident Management tab of the UI (as are other policy violations), but can also be sent to e-mails or written to Syslog. If written to Syslog, you can forward alerts to security intelligence and event management systems such as IBM QRadar or HP Arcsight so that your security team can handle and investigate appropriately.

**Prerequisite:** This policy rule relies on the existence of two groups we have named "MongoDBAdmins" and "MongoDB Sensitive objects." If you want to restrict alerting to certain commands, you can also add a group that contains specific commands such as find and CopyCollection. We will create and use this optional group that we'll call "MongoDB WatchCommands." It contains several commands that we want to watch for such as find, update, insert, delete, cloneCollection, and mapreduce.

**Figure 25. Sensitive objects group. For MongoDB, a collection is an object**
Figure 26. A group of specific commands that we want to monitor when used with sensitive data

To create your policy rule, select your policy from the Policy Finder, click **Edit Rules**, and then click **Add Access Rule**.

Ours looks like Figure 27.

Figure 27. This policy alerts when privileged users use specific commands to access sensitive data
To test the new rule, be sure to reinstall the policy.

Figure 28 shows what our alert looks like.

**Figure 28. Alert fires when privileged user uses one of the disallowed commands to access sensitive data (excerpt of the alert)**

<table>
<thead>
<tr>
<th>Access Rule Description</th>
<th>Client IP</th>
<th>Server IP</th>
<th>DB User Name</th>
<th>Full SQL String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert: Privileged User (Access on sensitive data)</td>
<td>76.144.233.8</td>
<td>76.144.233.8</td>
<td>Guardium</td>
<td>User: 'sundari' Password: 'guardium', roles: ['readWrite']</td>
</tr>
<tr>
<td>Alert: Privileged User (Access on sensitive data)</td>
<td>76.144.233.8</td>
<td>76.144.233.8</td>
<td>Guardium</td>
<td>User: 'sundari' Password: 'guardium', roles: ['readWrite']</td>
</tr>
</tbody>
</table>

**Real-time alert on Data Control commands**

A common requirement is to monitor any commands that give users access and privileges. In MongoDB, administrators can create and add users, and in MongoDB 2.4 there are additional roles that can be given to users. See Resources for links to more information about MongoDB security and roles.

The credentials and user privilege information are stored in the collection system.users.

So, for example, assume someone creates a new user as follows:

```javascript
db.addUser({user: 'sundari', pwd: 'guardium', roles: ['readWrite']}).
```

As shown in the report in Figure 29, InfoSphere Guardium will log this activity as an insert on the collection system.users. The activity will have two objects: the name of the new user and the system.users collection.

**Figure 29. Excerpt of an audit report showing access on the system.users collection**

For our policy rule, we may want easy visibility to any activity on the system.users collection. To do this, you can add a new access rule to your policy that logs access to the system.users collection. Figure 30 is our policy rule in which we have simply added system.users as the object and Log Only as the action, which will add it also to the Incident Management tab of the UI.
Figure 30. Policy rule to log changes to system.users so they will be visible on the incident management tab

![Policy rule diagram]

Figure 31 shows a partial output of an incident.

Figure 31. Admin has added Sundari as a user, which shows up on the incident management tab of the Guardium UI

![Incident output]

Note: Logging to incident management is good to get real-time records of events. But if this is an activity that requires periodic review, you may want to create a report of this activity and route it to reviewers.

Log policy violation for collection changes that could impact applications

Administrators and application owners in some organizations may want a record of changes in the database that could impact application logic or performance, such as dropping or renaming a collection, or dropping an index or a database. You can create a group that contains the commands you want to track. Note that helper methods likely flow across the wire differently. The commands we want to track include:

- deleteIndexes
• `drop` (to catch drops of collections)
• `dropDatabase`
• `renameCollection`

You may also need to add a group of ‘frozen’ objects if you want to avoid firing this rule for any test or QA activity that could generate a lot of drops and renames.

**Figure 32. Group of commands that we want to log**

Then you can add an access policy rule that includes the group and choose an action to take when that rule is triggered. In our case, we are logging the policy violation but not generating an alert.

**Figure 33. Excerpt of change command occurrence on Incident Management tab**

**Real-time alert: Read access to sensitive data exceeds threshold**

Many organizations prohibit their employees (and hackers) from retrieving an excessive amount of potentially sensitive data and would like to be alerted if this occurs so they can quickly investigate and determine if a serious breach has occurred.

One way to do this is to create a threshold based on "records affected" in the access rule of your MongoDB policy.

**Prerequisites:**

1. Create a group of sensitive data objects that you want to be alerted on.
2. Ensure that your system configuration has **Inspect Returned Data** and **Log Records Affected** enabled for all inspection engines. To do this, go to the Administration Console tab, then select **Configuration > Inspection Engines** and check the appropriate checkboxes, as shown in Figure 34.
Figure 34. Configure Guardium to report on the number of documents that are read

Figure 35 shows the policy rule we created that will alert when the number of records read by any database user on a sensitive data object is greater than 200. Be sure to put a period (dot) in the DB User field to count those records affected per database user rather than across all database users.

Figure 35. Excessive finds alert rule definition
Note: This rule will generate an alert whenever a particular user accesses more than 200 documents cumulatively across all collections in the group for this session. If you want to set specific limits for each collection, use a different rule for each.

The alert in Figure 36 shows that an unidentified user has downloaded more than 200 documents from the credit card collection.

**Figure 36. Excessive finds alert**

<table>
<thead>
<tr>
<th>Policy Violations / Incident Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Violation Log Id</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>551</td>
</tr>
</tbody>
</table>

**Summary and what's next**

In this second part of our three-part article series, we provided some details about how to configure the solution, including how to configure specific inspection engines for both the query routers and the shards. We provided some advice on how to validate that the S-TAP is catching MongoDB traffic and sending it to the collector. Finally, we provided some sample policy rules that you can consider for use in your own environment, depending on the security and regulatory requirements you have in your organization.

In Part 3, we describe an advanced capability, blocking, as well as reporting, audit browsing, and how to automate signoff and review of audit reports.
Resources

Learn

- For information about using Guardium Installation Manager, see the Guardium Installation Manager user guide in the Information Center.
- Read this e-book for more information about security for NoSQL systems.
- The InfoSphere Guardium website includes links to white papers, demos, and more.
- Visit the InfoSphere Guardium Tech Talk page to find links to recordings of previous tech talks and information about upcoming talks.
- The InfoSphere Guardium Information Center includes many "how tos" to help you make the most of InfoSphere Guardium data activity monitoring solution.
- Learn best practices in securing and protecting MongoDB data in this Webinar.
- Watch demos and educational videos on the InfoSphere Guardium YouTube channel.
- Take a look at free online MongoDB training.
- Stay current with information, events, and industry news related to data security and privacy by registering for the InfoSphere Guardium newsletter.
- Follow InfoSphere Guardium on Twitter.

Get products and technologies

- Download MongoDB here.

Discuss

- Participate in the discussion forum for this content.
- A new developerWorks community for InfoSphere Guardium is evolving to include links to relevant technical content, industry-specific information, and FAQs. Join the community and help it grow.
- Get involved in the Guardium users group on LinkedIn to ask questions and get advice from other users.
- Get involved in the developerWorks Community. Connect with other developerWorks users while exploring the developer-driven blogs, forums, groups, and wikis.
About the authors

Indrani Ghatare

Indrani Ghatare has been a software developer at IBM for more than 12 years. Currently, Indrani is a member of the Research and Development team for InfoSphere Guardium Collector. Indrani has worked on the development of the MongoDB parser and the logger component of Guardium Collector.

Matt Kalan

Matt Kalan is a senior solutions architect at 10gen, based in New York, with extensive experience helping more than 200 customers in financial services and other industries solve business problems with technology. Before 10gen, Matt grew Progress Software's Apama Algorithmic Trading and Complex Event Processing (CEP) Platform business in North America and later sold broader operational intelligence solutions to financial services firms. He previously worked for Caplin Systems selling solutions to stream real-time market data over the web to FX and FI portals, and for Sapient providing consulting services to global 2000 clients.

Sundari Voruganti

Sundari Voruganti is a member of the InfoSphere Guardium QA team at IBM Silicon Valley Lab. Sundari has been with IBM over a decade and has a diverse background in both engineering as well as customer enablement roles. As a passionate technologist, she loves the challenge of learning and working with new technologies as well as helping customers understand and implement IM solutions. Sundari has a double Masters in Computer Science from Bangalore University and University of Alberta.

Kathryn Zeidenstein

Kathy Zeidenstein has worked at IBM for a bazillion years. Currently, she is working as a technology evangelist for InfoSphere 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