Enable Audit Events in MS SQL Server

EventTracker v6.x, v7.x
Abstract

Databases are critical components of the enterprise infrastructure and for this reason they are a prime target for hackers. Microsoft SQL servers have been a favorite target for hackers in the recent past. Worms, such as SQL Spida and Slammer, spreading through the SQL service is one example. One reason for this is the easy access to networked SQL servers. Therefore, security is of utmost concern to database administrators. The potential loss for comprised data can only mean unexpected downtime, lost business, and at times can be devastating to the company's existence. Ideal security practice should be able to track database users and identify malicious activities against the database.

This white paper will help you understand the importance of tracking user activities in Microsoft SQL server and guide you to monitor these audit entries, logged in Windows Application event log, using EventTracker. These are important elements in effective database security management.

Target Audience

EventTracker administrators who wish to track audit events in MS SQL Server.

The information contained in this document represents the current view of Prism Microsystems Inc. on the issues discussed as of the date of publication. Because Prism Microsystems must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Prism Microsystems, and Prism Microsystems cannot guarantee the accuracy of any information presented after the date of publication.

This document is for informational purposes only. Prism Microsystems MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, this paper may be freely distributed without permission from Prism, as long as its content is unaltered, nothing is added to the content and credit to Prism is provided.

Prism Microsystems may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Prism Microsystems, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The example companies, organizations, products, people and events depicted herein are fictitious. No association with any real company, organization, product, person or event is intended or should be inferred.

© 2014 Prism Microsystems Corporation. All rights reserved. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.
# Table of Contents

Introduction ................................................................................................................................................. 3  
Problem Description ................................................................................................................................... 3  
Problem Scope ............................................................................................................................................. 3  
Key Challenges ............................................................................................................................................ 4  
  Log method: tables v/s event log .............................................................................................................. 4  
  Generic Event Id Numbers ....................................................................................................................... 5  
Best Practices .............................................................................................................................................. 6  
MSSQL Server Enterprise ............................................................................................................................... 6  
  Enable Auditing ....................................................................................................................................... 6  
  Track Successful Logon Events ................................................................................................................ 9  
  Setup Authentication ............................................................................................................................... 9  
  Track Logon Failure Events ..................................................................................................................... 9  
  Monitor Backup and Restore Operations ............................................................................................ 10  
  Monitor SQL Server service Startup and Shutdown ........................................................................... 10  
  Track Database and Server Membership and Roles ............................................................................ 10  
MSSQL Server Standard ............................................................................................................................... 11  
  Enable auditing ...................................................................................................................................... 11  
  Automate the process in SQL Server 2005 Enterprise and Standard Edition ...................................... 17  
  Automate the process in SQL Server 2008/2008 R2 Standard Edition ................................................. 19  
  Configure Direct Log Archiver (DLA) to send SQL events to EventTracker ........................................ 32  
  Sample Report ....................................................................................................................................... 35  
EventTracker Solution ................................................................................................................................. 36  
  Built-In Alerts ........................................................................................................................................ 37  
  Service Monitoring ............................................................................................................................... 38  
  Audit Activity Reports ............................................................................................................................ 38  
Summary .................................................................................................................................................... 38  
About Prism Microsystems ......................................................................................................................... 39  
Appendix .................................................................................................................................................... 40
Introduction

In the modern enterprise, valuable information resides in online database (DB) servers, making these prime assets to the company. However, insufficient attention has been paid to safeguarding these servers against intrusions, monitoring database backup, and restoring operations. Monitoring DB transactions, identifying and tracking database access, as well as changes to the database, is vital to secure SQL servers and comply with security policies and regulations.

Problem Description

Monitoring DB activities can be a complex task. While conceptually simple, there are several challenges that must be overcome. Much of the information required for effective security is contained in the native auditing mechanisms built into MS SQL Server. However, all of it is stored locally on each server in the form of application event and error logs, and native no built-in intelligence exists to a) consolidate and centralize logging b) filter irrelevant data and c) trigger alerts in real-time.

Problem Scope

Monitoring, reviewing changes to critical database servers, and accessing security risks periodically, is not only a best practice but also a critical factor in preventing productivity loss and unexpected downtime.

Evolution of government standards and regulations such as Gramm-Leach-Bliley Act (GLBA), the Health Insurance Portability and Accountability Act (HIPAA), the Sarbanes-Oxley Act (SOX), the USA PATRIOT Act, and the management of clinical trial data (FDA Title 21 CFR Part 11) have created an immediate need to proactively monitor database activities. Understanding which user accessed, altered, updated, deleted, or merely viewed critical data is an essential component of regulatory compliance.

Best practices require that organizations monitor, review, and preserve event information including:

- End user activity, including all SQL commands, logout/login, and enabling of application roles
- Database administrator activity and configuration of database or server
• Security events, including GRANT, REVOKE, and DENY; login user/role, and add/remove/configure

• Utility events, including back up, restore, bulk insert, bcp, and database consistency checker (DBCC) commands

• Server events, including shutdown, pause, and start

• Audit events, including add audit, modify audit, and stop audit

Key Challenges

A number of factors complicate effective monitoring of database activities. Although audit mechanisms provided by MS SQL server amass a large amount of audit data in the form of windows application event logs and SQL Errorlogs, there is no facility to a) consolidate these logs from multiple installations, b) filter critical logs, and c) trigger critical alerts in real time. When the total number of servers increases, the complexity is further increased.

Log method: tables v/s event log

In essence, MS SQL Server audit mechanisms rely on storing their output in a local file or table. Since this is common knowledge, the first thing that hackers do is hide the activity in SQL Server. One way to do this is to rollover the error log through DBCC ERRORLOG a total of seven (7) times. This eliminates all the evidence of the intrusion. Thus, storing audit records in this way is inherently unsafe.

This approach also makes it very difficult for the DB Administrators to manually connect to each server at regular intervals and analyze the event and error logs. The preferable approach is to configure MS SQL Server to emit its logs into the Windows Event log. These logs are stored in a binary format and if EventTracker is installed, matching entries are immediately transferred to the central console thereby eliminating the possibility of losing evidence of intrusion.
Generic Event Id Numbers

Another challenge with MS SQL Server events logged into the Windows Application Event log is that most of them are logged with a generic Event ID number of 17055/17052. Hence, by just looking at the log one cannot tell if the event corresponds to a successful logon, a logon failure, or a backup failure. Nor does it tell us what type of login occurred. The actual error code corresponding to the log message is part of the event description.

In order to gain more information one will have to open up each event log record and look at the description. With this logging structure and the fact that Windows event logs are common interface for all application, system, and security events, it makes it almost impossible for the DBAs to sift through the logs, detect malicious activities, and take any proactive measurements before things go wrong.

Therefore, it becomes imperative to have an automated process to consolidate the entire event and Errorlogs into a central database, analyze events, and highlight significant information.

<table>
<thead>
<tr>
<th>Event Viewer (Local)</th>
<th>Application Log</th>
<th>Event Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Type</td>
<td>Date</td>
<td>Time</td>
</tr>
<tr>
<td>Information</td>
<td>2/28/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/26/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/25/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/23/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/21/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/19/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/17/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/15/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/13/2005</td>
<td>2:30:13 AM</td>
</tr>
<tr>
<td>Information</td>
<td>2/11/2005</td>
<td>2:30:13 AM</td>
</tr>
</tbody>
</table>

Figure 1: MS SQL server application event log
Best Practices

Auditing all database activity is impractical, and can lead to information overload. By following the steps outlined in this paper, you can reliably monitor and track critical data access activity.

MSSQL Server Enterprise

Enable Auditing

Default SQL Server auditing

SQL Server stores up to seven subsequently created logs, named Errorlog, ErrorLog.1, ErrorLog.2, etc. in the server installation path, default path being C:\Program Files\Microsoft SQL Server\MSSQL\LOG. Its entries consist of the timestamp, source (this is either server or one of user processes, indicated by its process id), and message.

Custom SQL server messages can also be sent to Windows event log using the raiserror or xp_logevent commands with error numbers starting from 50001, and severity levels 0 through 18.

SQL Server connections auditing

During the installation process, MS SQL Server registers itself with the Windows application event log subsystem. When you run instances of SQL Server in your network, a variety of entries that describe SQL Server's startup, shutdown, and day-to-day operations such as backups, server-side traces, etc. are logged in application event logs. SQL Server 2000 also has a built-in ability to audit login activity, including failed login attempts, to the Windows Application log. However, this auditing capability is not enabled by default.

To enable auditing of server and database connections with Enterprise Manager in SQL Server:

1. Expand a server group.
2. Right-click on the database server, and then choose Properties.
3. Select the Security tab. On the security tab, under Audit Level, check the required audit setting. Possible auditing settings are:
   - None. Logs no auditing information.
• Success. Causes only successful logins to be logged. You must stop and restart the server for this setting to take effect.
• Failure. Causes only failed logins to be logged.
• All. Causes successful and failed logins to be logged

4. Stop and restart database server for the audit settings to take effect.

The audit level can also be configured using the sp_loginconfig stored procedure. In environments where there are large numbers of servers, one can quickly change the auditing level through a registry key. To change the auditing level, simply change the AuditLevel value in the HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSSQLServer\MSSQLServer key. Setting the value to 0 means no auditing will be turned on, 1 means successful logins will be audited, 2 corresponds to failed logins, and 3 means all logins will be audited. Following SQL query can be used for modifying the AuditLevel registry key value.

```
xp_instance_regwrite N'HKEY_LOCAL_MACHINE',
N'SOFTWARE\Microsoft\MSSQLServer\MSSQLServer',N'AuditLevel', REG_DWORD,3
```

All the auditing information is also written to the SQL Server Errorlog.
Figure 2: Configure MS SQL Server Auditing
Track Successful Logon Events

Auditing internal data access and use from privileged users who have direct access to data is critical. SQL Server has two basic levels of security: logins and users. Logins allow access to the server and hold server-level permissions. Users are at the database level and hold permissions to individual data objects.

When an attempt is made to log in to the server, SQL Server validates the login, and when an attempt is made to access items in a database, SQL Server validates the database user account.

It is highly recommended to audit all connection attempts to the database. The auditing of connection attempts typically does not result in a significant performance impact on the database and rarely creates an excessive amount of data written to the log.

Setup Authentication

MS SQL Server provides two types of Authentication mechanisms.

- Windows NT Authentication: In this authentication Mode, SQL Server relies on Windows to authenticate users. Microsoft recommends using Windows Authentication method for SQL Server. Integrated SQL server logins are the Windows logins on the local server, which have access to SQL Server. In this mode, by default, any Windows login will have administrative privileges on the databases. Since not all valid Windows users need access to SQL Server, it’s important to monitor and review the roles and privileges of these logins.

- Mixed Mode Authentication: In this Mode, if the user account is not a valid Windows account, SQL Server uses its own database (sysxlogins table) to authenticate users by SQL Server username-password pairs maintained within the SQL Server. The ‘sa’ login is the system administrator’s account for SQL Server. Since the ‘sa’ login is widely known to malicious users, it’s important to audit for logins with null passwords. User logons can be tracked by parsing event logs for error codes 18453, 18454, 18455 and keywords such as ‘login succeeded’.

Track Logon Failure Events

Failed logon attempts are the most important activity to monitor. Most logon failures do not represent threats, merely harmless mistakes by authorized users. A few, however, are the work of hackers, so you need to analyze the data and determine whether unauthorized users are
trying to access critical company resources. Messages relating to login permission denial can be tracked by parsing logs for keywords like 'Login Failed' or 'denied'. It is recommended as security best practice to set up an alert on these messages with severity level 14 to send an email or page to an operator who can quickly respond to the issue.

Monitor Backup and Restore Operations

Database backup settings should be monitored often to ensure that they are in compliance with the security policy. Backups are crucial to data integrity. Backups should be performed at regular intervals and stored in off-site, secured locations. A compromised server or a device failure may need to be restored from a backup. Failing to perform backups within the given time frame exposes the server to the risk of data loss. Parsing event logs for SQL error codes such as 3023, 3036, 3041, 3101, 3143, 3154, 3155, 3206, 3209, 3227, one can monitor critical backup and restore operations.

Monitor SQL Server service Startup and Shutdown

When monitoring the availability of critical SQL server services, such as the SQL server service, the SQL server Agent service is vital for remote server diagnosis and problem resolution. Service startup and shutdown during unusual hours can also mean warning signs for intrusions.

Track Database and Server Membership and Roles

Regulatory statutes such as HIPAA mandate tracking permissions changes to and from database users, database roles, and application roles. T-SQL commands such as GRANT, REVOKE, and DENY can be used to manage permissions at the user and role level. While the latest versions of SQL server security model has many enhancements, it also adds the extra layer of permissions that one must monitor to make sure no one has been given more access than they need or that they’ve already circumvented security to elevate themselves.

It is therefore recommended to periodically scan server and database roles to ensure that membership is only granted to trusted individuals. Make sure to assign permissions by group in order to simplify the auditing task.
MSSQL Server Standard

Enable auditing

Audit specifications cannot be used in the standard version of SQL 2008 and 2008 R2. SQL Server Audit can only be done on Enterprise and Datacenter editions of SQL Server 2008 R2.

With SQL Server Audit, SQL Server 2008 introduces an important new feature that provides a true auditing solution for enterprise customers. While SQL Trace can be used to satisfy many auditing needs, SQL Server Audit offers a number of attractive advantages that may help DBAs more easily achieve their goals such as meeting regulatory compliance requirements. Use SQL Server Profiler to monitor only the events in which you are interested.

1. Login to SQL Server Management Studio.
2. Select the Tools menu and then select SQL Server Profiler.
3. Enter valid credentials and then select the Connect button.

Figure 3

Trace Properties window displays.
4. Enter **Trace Name**, and then select **Save to table**: option.

Connect to Server window displays.

5. Enter valid credentials, and then select the **Connect** button.
Destination Table window displays.

6. To save the trace table information in **Table**: drop down, select the destination table for the trace, and then select the **OK** button.
7. Select the **Events Selection** tab.

8. Select the respective events required to monitor.
9. Select the Run button.
10. To save the trace file, select the **File** menu, select **Save as** and then select **Trace file**....

![Image of EventTracker interface](image.png)

**Figure 9**
Automate the process in SQL Server 2005 Enterprise and Standard Edition

1. Create a ‘TestTrace’ text file in the path mentioned below. In our example, C:\.

2. Copy the below and paste in MS SQL Query and execute.

```sql
-- Declare variables
DECLARE @rc INT
DECLARE @TraceID INT
DECLARE @maxFileSize bigint
DECLARE @fileName NVARCHAR(128)
DECLARE @on bit

-- Set values
SET @maxFileSize = 5
SET @fileName = N'C:\TestTrace'
SET @on = 1

-- Create trace
EXEC @rc = sp_trace_create @TraceID output, 0, @fileName, @maxFileSize, NULL

-- If error end process
IF (@rc != 0) GOTO error

-- Set the events and data to collect
EXEC sp_trace_setevent @TraceID, 41, 1, @on
EXEC sp_trace_setevent @TraceID, 41, 12, @on
EXEC sp_trace_setevent @TraceID, 41, 13, @on
EXEC sp_trace_setevent @TraceID, 41, 14, @on
EXEC sp_trace_setevent @TraceID, 41, 15, @on
EXEC sp_trace_setevent @TraceID, 41, 16, @on
EXEC sp_trace_setevent @TraceID, 41, 17, @on

-- Set Filters
-- filter1 include databaseId = 6
EXEC sp_trace_setfilter @TraceID, 3, 1, 0, 6

-- filter2 exclude application SQL Profiler
EXEC sp_trace_setfilter @TraceID, 10, 0, 7, N'SQL Profiler'

-- Start the trace
EXEC sp_trace_setstatus @TraceID, 1

-- display trace id for future references
SELECT TraceID=@TraceID

GOTO finish

-- error trap
error:
SELECT ErrorCode=@rc

-- exit
finish:
GO
```
3. Login MS SQL Server Management Studio.

4. Run the script.

By default @maxFileSize = 5MB. The file size can be modified as per the requirement.

```sql
/* Server Side Trace */

-- Declare variables
DECLARE @rc INT
DECLARE @TraceID INT
DECLARE @maxFileSize bigint
DECLARE @fileName NVARCHAR(150)
DECLARE @on bit

-- Set values
SET @maxFileSize = 5
SET @fileName = N'C:\Trace'
SET @on = 1

-- Create trace
EXEC @rc = sp_trace_create @TraceID output, 0, @fileName, @maxFileSize, NULL

-- If error end process
IF (@rc != 0) GOTO error

-- Set the events and data to collect
EXEC sp_trace_setevent @TraceID, 41, 1, @on
EXEC sp_trace_setevent @TraceID, 41, 12, @on
EXEC sp_trace_setevent @TraceID, 41, 13, @on
EXEC sp_trace_setevent @TraceID, 41, 14, @on
EXEC sp_trace_setevent @TraceID, 41, 15, @on
```

The output can be viewed in the Trace file.

![Picture of SQL Server Management Studio with trace output](image)

Figure 10
Automate the process in SQL Server 2008/2008 R2 Standard Edition

1. Login to MS SQL Management Studio.
2. Select the Tools menu and then select SQL Server Profiler.
3. Enter valid credentials and then select the Connect button.
   
   Trace Properties window displays.
4. In **General** tab, enter **Trace name**.

5. Select **Save to file** option, and then select **Enable file rollover** option if not selected.

6. Select **Events Selection** tab.
7. Select required options, and then select **Run**.

8. In **SQL Server Profiler**, select the **File** menu, select **Export**, and then select **Script Trace Definition**.

9. Select **For SQL Server 2005 – 2008 R2**....

Refer Figure 14.
Save As window displays.
10. Enter the **File name**: and then select **Save**.

   SQL Server Profiler displays successful message.

   ![Figure 16](image)

11. Select **OK**.

12. To create **SQL Agent Job** in **SQL Server Management Studio**, expand **SQL Server Agent** node, select **New**, and then select **Job**...

   ![Figure 17](image)

   New Job window displays.
13. Enter **Name:** of the new job.

![Figure 18](image)

14. In **Select a page** pane, select **Steps**, and then select **New...**
New Job Step window displays.

15. Enter **Step name**:

16. In **Command** pane, copy the script, change the value from ‘0’ to ‘2’, and enter correct path of the trace file.

   Refer Figure 20.
17. In Select a page pane, select Advanced.

18. In Output file: select browse button.

19. Select appropriate path of the file and enter the File name:
20. Select **OK**.
21. Select **Schedules** and then select **New**.

   New Job Schedule window displays.

22. Enter **Name** of the new job schedule.

23. In **Schedule type** drop down, select **Start automatically when SQL Server Agent starts**.
24. Select **OK**.

The new job schedule displays in schedule list.
25. In SQL Server Management Studio, expand SQL Server Agent node, select the trace file created, select Start Job at Step....
Figure 25
Configure Direct Log Archiver (DLA) to send SQL events to EventTracker

1. To configure DLA, logon to EventTracker Enterprise.
2. Select the Admin menu, select Manager, and then select Direct Log Archiver/Netflow Receiver tab.
3. Select Direct log file archiving from external sources option, and then select the Add button.
   Direct Archiver Configuration window displays.
4. In Type drop down, select Others.
5. In Logfile Extension box, enter .trc.
6. Enter Configuration Name in the box.
7. Enter the path of the Log File Folder.
   (OR)
   Click the Browse… button and select any folder.
8. In Field Separator drop down, select Comma, and then select the Configure button.
9. Enter the **Log Source, Computer Name, Computer IP, System Type**.

10. Select **Entire Row as Description** option, and then select the **Save & Close** button.

![Direct Log Archiver/Netflow Receiver tab displays.](image)

**Figure 27**

Direct Log Archiver/Netflow Receiver tab displays.
11. Select the **Save** button.
Sample Report

The details of sample report are given below.

![Figure 29](image-url)

<table>
<thead>
<tr>
<th>Application used</th>
<th>Loggedin Database</th>
<th>System name</th>
<th>User Logged</th>
<th>Objectname</th>
<th>ObjectType</th>
<th>Servername</th>
<th>Datetime</th>
<th>TextData</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>master</td>
<td>ALICE</td>
<td>Administrator</td>
<td>Information technology</td>
<td>16964</td>
<td>(local)</td>
<td>10:57:22</td>
<td>CREATE DATABASE [Informatica technology] ON PRIMARY ( NAME = Informatica technology)]</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>Research</td>
<td>ALICE</td>
<td>Administrator</td>
<td>Research</td>
<td>16964</td>
<td>(local)</td>
<td>10:53:37</td>
<td>DROP DATABASE [Research]</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>Research</td>
<td>ALICE</td>
<td>Administrator</td>
<td>Research</td>
<td>16964</td>
<td>(local)</td>
<td>10:52:29</td>
<td>DROP DATABASE [Research]</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>Research</td>
<td>ALICE</td>
<td>Administrator</td>
<td>Research</td>
<td>16964</td>
<td>(local)</td>
<td>10:34:49</td>
<td>CREATE DATABASE [Microsoft SQL Server Management Studio] ON PRIMARY</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>master</td>
<td>ALICE</td>
<td>Administrator</td>
<td>National</td>
<td>16964</td>
<td>(local)</td>
<td>10:58:97</td>
<td>CREATE DATABASE [National] ON PRIMARY</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>ultron</td>
<td>ALICE</td>
<td>Administrator</td>
<td>ultron</td>
<td>16964</td>
<td>(local)</td>
<td>10:22:22</td>
<td>DROP DATABASE [ultron]</td>
</tr>
<tr>
<td>Microsoft SQL Server Management Studio</td>
<td>default</td>
<td>ALICE</td>
<td>Administrator</td>
<td>default</td>
<td>16964</td>
<td>(local)</td>
<td>10:22:22</td>
<td>DROP DATABASE [default]</td>
</tr>
</tbody>
</table>
EventTracker Solution

EventTracker is a reliable, proactive, and practical enterprise class solution to centrally monitor, analyze, and manage events being emitted by Windows NT/2K/XP/2003/VISTA, UNIX systems, and SNMP enabled devices.

EventTracker is a cost-effective solution to proactive management of the database server environment. Its built-in intelligence addresses each of the challenges described in this white paper. Its features make it possible to reliably process and respond to SQL Server events raised by the server and posted to the Windows application event log:

- A centralized log consolidates all critical, error, and warning messages from applications events and Errorlog files from critical servers.
- All audit activities are archived in an encrypted vault for auditing.
- Real-time monitoring of critical events such as logon failures, backup failures, service restarts etc., is performed.
- Event correlation modules constantly monitor for malicious hacking activity and inform network security officers and security administrators in real time, enabling them to prevent any attack before hackers can compromise any sensitive data.
- A variety of reports can be generated and reports can be scheduled to occur automatically.

Figure 30: Monitor SQL Server audit logs with EventTracker
Built-In Alerts

EventTracker monitors and analyzes MS SQL server audit events to notify system administrators with crucial alerts. EventTracker’s rich notification allows you to configure alert actions such as beeps and multimedia sound files, network pop-up messages, email, SNMP traps, and user written batch files, scripts, command files or applications. The following set of pre-defined alerts can be used as a template to define various critical SQL alerts:

- Backup Failed: Scanning eventlogs for error codes 18204, 18210, 3443, 3441 and 4301, generates this alert
- SQL server stopped: Scanning event id 17052 and 17055 for error codes 566, 3417, 1619, 3441, 4301, and 17148, generates this alert
- Transaction log full: Parsing event logs for transaction log error 9002 generates this alert.

These messages indicate that SQL Server cannot allocate additional free space, needed for expanding the database.
Service Monitoring

EventTracker monitors all the SQL server services and generates an alert if SQL Server or related services such as SQL Agent fail. EventTracker agents can also be configured to automatically restart these services.

Audit Activity Reports

EventTracker simplifies your day-to-day management tasks. It hides the complex monitoring processes and details of events, while generating all the necessary data and meaningful reports regarding network security. EventTracker provides a variety of pre-packaged report templates designed for forensic analysis and audit compliances. EventTracker also allows you to create custom reports to meet individual needs. Daily, weekly, and yearly reports can be scheduled and run automatically.

Summary

Addressing company security policies, meeting regulatory requirements, and preventing unexpected downtime are key goals achieved through monitoring MS SQL server audit events. EventTracker (http://www.eventlogmanager.com/) addresses all the issues discussed in the white paper in an elegant and cost effective manner.
About Prism Microsystems

Prism MicroSystems, Inc. delivers business-critical solutions to consolidate, correlate, and detect changes that could impact the performance, availability, and security of your IT infrastructure. With a proven history of innovation and leadership, Prism provides easy-to-deploy products and solutions for integrated Security Management, Change Management, and Intrusion Detection. PMI enables commercial enterprises, educational institutions, and government organizations to increase the security of their environments and reduce risk to their enterprise. Customers span multiple sectors including financial, communications, scientific, healthcare, banking, and consulting.

PMI is a privately-held corporation with corporate headquarters in the Baltimore-Washington high tech corridor. Research and development facilities are located in both Maryland and India. These facilities have been independently appraised in accordance with the Software Engineering Institute’s Appraisal Framework and were deemed to meet the goals of SEI Level 3 for CMM.

We provide the software tools to cost-effectively manage and secure infrastructure investments, including servers, databases, websites, network devices, and security applications. Products include software to provide centralized event management, Window change tracking and management, centralized SNMP management, and Website availability monitoring. For additional information, please visit www.eventtracker.com.
Appendix

<table>
<thead>
<tr>
<th>Category Summary Report Sorted By Computer</th>
<th>EventTracker Report</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prism Microsystems, Inc.</td>
</tr>
</tbody>
</table>

Quick View

This report gives you information on the selected Categories sorted by Computers in your enterprise. The report can be used to track system activity in relation to a Category thereby giving you an insight into the security and other implications.

Category Summary Report sorted by Computer
From 16-Feb-2005 11:07:34 am
To 24-Feb-2005 11:07:34 am

Number of Computers where event has occurred is 1

<table>
<thead>
<tr>
<th>Category</th>
<th>Category Description</th>
<th>Computer</th>
<th>No. Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSSQLServer: All Events server</td>
<td>All events related to SQL</td>
<td>1</td>
<td>283</td>
</tr>
<tr>
<td>MSSQLServer: Logon failures</td>
<td>Failed SQL server login</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Logons</td>
<td>SQL server successful logons</td>
<td>1</td>
<td>35</td>
</tr>
<tr>
<td>MSSQLServer: Backup and Restore</td>
<td>Critical SQL server data backup and restore related events</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>MSSQLServer: Successful Backup</td>
<td>Events related to successful database backup operations</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>MSSQLServer: Backup Failure</td>
<td>Critical database backup</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>events</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The data for this report was collected by EventTracker. For specific report customization, please contact support@eventtracker.com

Figure 32
## Category Summary Report Sorted By Computer

**Category MSSQLServer: All Events** had 1 computers generating 283 events.  
Event IDs included are 17052, 17055  
Computer: *IVORY*  
No. Events: 283

**Category MSSQLServer: Backup and Restore** had 1 computers generating 11 events.  
Event IDs included are 17055  
Computer: *IVORY*  
No. Events: 11

**Category MSSQLServer: Backup Failure** had 1 computers generating 10 events.  
Event IDs included are 17055  
Computer: *IVORY*  
No. Events: 10

**Category MSSQLServer: Logon failures** had 1 computers generating 6 events.  
Event IDs included are 17055, 17055, 17052  
Computer: *IVORY*  
No. Events: 6

**Category MSSQLServer: Successful Backup** had 1 computers generating 7 events.  
Event IDs included are 17055  
Computer: *IVORY*  
No. Events: 7

**Category MSSQLServer: Successful Logons** had 1 computers generating 35 events.  
Event IDs included are 17055, 17052  
Computer: *IVORY*  
No. Events: 35

### Disclaimer

The values in this report are either collected or computed from data collected by EventTracker.

In cases where data is not available, the program makes default assumptions. This can affect the accuracy of the computed values.

Prism Microsystems, Inc. ("Prism") makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose.

This document is for informational purposes only.