# 1 Cover Page


## QUICK START GUIDE

# 2 Table of Contents

# 3 Quick Start Guide

## PURPOSE OF THIS GUIDE

This **Quick Start Guide** will quickly walk you through the basic Installation and Setup Steps for SentryOne. Following this guide you should be able to install SentryOne, complete basic configuration, start managing schedules, monitoring performance, and generating notifications across your enterprise within 10 to 30 minutes.

Please note, this guide is focused on initial installation and configuration steps only, and does not cover such topics as using the visual job calendar, performance monitoring, chaining, queuing, etc. Please refer to the SentryOne User Guide for more information on these and other topics related to everyday use. The User Guide can be accessed at any time from the SentryOne Client Help menu.

# 4 Important Concepts

## SENTRYONE COMPONENTS

SentryOne consists of the **Client** (a thin client application), the **Monitoring Service** (a Windows service), and a SQL Server **Database**. The SentryOne Database stores event metadata and history information collected by the SentryOne Monitoring Service and the SentryOne Client provides a thin client interface for viewing and managing this information.

One SentryOne Monitoring Service is typically required for every 50 to 100 monitored SQL Servers, Analysis Services Instances, SharePoint Servers, Oracle Databases or Windows Task Schedulers. Multiple Monitoring Services can be installed for scalability, redundancy, or to collect information from remote sites. Normally a SentryOne Client will be installed on each DBA's workstation. All SentryOne Monitoring Services and Clients connect to the same database.

It's important to note that SentryOne does not attempt to replace SQL Server Agent, Oracle Scheduler, Windows Task Scheduler or any other scheduling agents. Instead, SentryOne communicates with these schedulers to ascertain event status, and collect history and performance information using a lightweight polling architecture. Thus SentryOne does not require agents installed on each monitored target, dramatically reducing associated setup and maintenance overhead of agent-based systems. SentryOne also does not install a database on every monitored SQL Server.

## ALERTING AND RESPONSE SYSTEM

As part of its Alerting and Response system, SentryOne uses the concept of **Conditions** and **Actions.** Conditions describe the various states of monitored objects in your environment. You configure Actions to take place when a Condition is met.

All Actions work on the principle of inheritance. This means that if you configure an Action in response to a Condition being met at the global level (Shared Groups node in the Navigator pane), it will be automatically passed down to all applicable objects below it. This allows you to define global Actions for the most common issues across your environment once, and have those passed down to every monitored target automatically.

You can further refine Actions at each level as needed. This gives you the ability to determine exactly what happens in response to events occurring in your server environment. Each Instance type supports multiple Conditions and Actions.

Configuring Actions globally provides a powerful way to significantly reduce the setup and configuration time required to implement notifications. For example, by enabling the **Send Email Action** for the global **SQL Server Agent Job: Failure Condition**, you will automatically receive email alerts for any SQL Agent job failures across your enterprise. The only requirement is that the SQL Server Instance and its jobs be "watched" by SentryOne. For a more detailed explanation of how Conditions and Actions work see the Alerting and Response System topic in the *SentryOne User Guide*.

## "WATCHING" INSTANCES AND OBJECTS

Throughout this document you'll also see the term **"watch"** used frequently, in the context of watching Instances or objects. When you have SentryOne "watch" an Instance or object via the context

menu this simply means that SentryOne will begin monitoring it.

Please consider these rules regarding watched Instances and objects:

1. When an Instance is watched, SentryOne will monitor the Instance and fire any applicable conditions for the Instance based on its type.
2. When an object is watched, SentryOne will monitor the object and fire conditions for the object based on its type.
3. An Instance can be watched without watching any of its objects.
4. If any object on an Instance is set to watched, the Instance will also be automatically set to watched.
5. An object and its Instance must be watched to utilize SentryOne's queuing, chaining, and performance monitoring features.

# 5 Installation Recommendations

## WHERE TO INSTALL THE SENTRYONE COMPONENTS

The SentryOne Client, Monitoring Service, and SentryOne Database are typically installed as follows:

- The SentryOne Client is installed on your workstation computer(s)
- The SentryOne Database is installed on a SQL Server instance on your local area network
- The Monitoring Service is installed on the same computer as the SentryOne Database, or any other non-production server in the same LAN.
- For Azure SQL Database or Azure SQL Data Warehouse monitoring, the Monitoring service is installed either on-premises, or on a Virtual Machine in the cloud

The SentryOne Clients and Monitoring Services are each configured to connect to the same SentryOne Database during setup.

> ***NOTE***: The SentryOne Database must be installed on a SQL Server 2008 or higher instance. SQL Server Express Edition or Azure SQL Database is **not** supported for the SentryOne Database. Please see the System Requirements section for more information.

### INSTALL ALL COMPONENTS ON THE SAME LOCAL AREA NETWORK

For performance reasons, it is recommended that the SentryOne Client, Monitoring Service and database be installed on the same LAN. For example, you would not want to connect the SentryOne Client or Monitoring Service to a SentryOne Database over a slow WAN link, as performance will suffer.

When monitoring Azure SQL Database or Azure SQL Data Warehouse targets the monitoring service will contact the Azure endpoints remotely.

### WHERE TO INSTALL THE MONITORING SERVICE(S)

It is not recommended that the Monitoring Service be installed on a production server, as it does incur some memory and CPU overhead. Exactly how much depends on the number of Instances and objects being monitored.

### INSTALLING THE SENTRYONE MONITORING SERVICE AND DATABASE ON THE SAME COMPUTER

Depending on your environment, you may want to install the Monitoring Service on the same SQL Server machine where the SentryOne Database is located, to minimize network overhead for communications between the Monitoring Service and the database. Because both Microsoft SQL Server and the SentryOne Monitoring Service are multi-threaded, **to ensure adequate performance when running both on the same computer it is very important that the computer have at least two CPUs**. See System Requirements for more information.

### INSTALLING MULTIPLE SENTRYONE CLIENTS AND MONITORING SERVICES

Depending on the size of your SQL Server environment, you may need to install multiple SentryOne Clients and Monitoring Services. Typically each DBA will have the SentryOne Client installed on their workstation, and one Monitoring Service will be installed for every 50 to 100 SQL Server, Oracle, or Windows Task Scheduler instances being monitored.

## CLUSTERING SENTRYONE MONITORING SERVICES

Multiple Monitoring Services can be installed to handle more than 100 Instances, and/or to provide automatic redundancy and load balancing. There is no configuration required to implement a basic SentryOne cluster. Simply install more than one Monitoring Service and connect each to the same SentryOne Database during setup, and they will automatically distribute the monitoring load evenly between themselves. If one Monitoring Service fails, the remaining Monitoring Service(s) will pickup the load automatically. See the Load Balancing and Fault Tolerance topic in the **SentryOne User Guide** for more details.

## INCREASED FAULT TOLERANCE FOR THE SENTRYONE DATABASE

If increased fault tolerance is required for the SentryOne Database, we recommend installing the database on a clustered SQL Server instance. Log shipping can also be used with the SentryOne Database, however a separate SentryOne license is required for the standby server. Customers can obtain this standby license by visiting our Customer Portal and modifying the Server name of their current license key to the name of the standby server and applying this license key to the SentryOne Database on the standby server.

# 6 System Requirements

## SENTRYONE COMPONENTS

### SentryOne Client computer

- Windows version from **supported** list below
- Microsoft .NET 4.5 (included in the setup package)
- Minimum Single 1.6 GHz CPU, 1 GB RAM

### SentryOne Monitoring Service computer

- Windows version from **supported** list below
- Microsoft .NET 4.5 (included in the setup package)
- Minimum Dual 1.6 GHz CPUs, or 1.6 GHz multi-core CPU, 1 GB RAM

### SentryOne Database

- SQL Server 2008, Standard and Enterprise
- SQL Server 2008 R2, Standard and Enterprise
- SQL Server 2012, Standard, BI, and Enterprise
- SQL Server 2014, Standard, BI, and Enterprise
- SQL Server 2016, Standard, BI, and Enterprise
- Minimum Dual 1.6 GHz CPUs, or 1.6 GHz multi-core CPU, 4 GB RAM
- Disk Space: 10GB

> ***IMPORTANT***: These system requirements are the **minimum recommended requirements for a standard 5-server installation**. When monitoring more than 5 servers with SentryOne, additional RAM and disk space may be needed for the database server. See the SentryOne Overhead Analysis document and the Performance Analysis Data Capacity Planning topic for more details.

> ***NOTE***: For performance reasons, it is not recommended that the SentryOne Client, Monitoring Service, or SQL Server (including the instance housing the SentryOne Database) be run simultaneously on the same single CPU computer. However, the Monitoring Service may perform satisfactorily on one CPU if there are no other CPU intensive programs or services operating on the same system, such as SQL Server. Additional factors include the number of instances being monitored and the number of objects on those instances.

## SUPPORTED OPERATING SYSTEMS

### Supported Operating Systems (x86)

- Windows Server 2008
- Windows Vista
- Windows 7
- Windows 8

- Windows 10

**Supported Operating Systems (x64)**

- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Vista
- Windows 7
- Windows 8
- Windows 10

## WATCHED TARGETS AND INSTANCES

**Watched (monitored) Windows Targets**

- Windows Server 2003
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows 10

**Watched Data Warehouse Targets**

- Microsoft Analytics Platform System (AU3)
- Azure SQL Data Warehouse

**Watched (monitored) SQL Server instances**

- SQL Server 2005
- SQL Server 2008
- SQL Server 2008 R2
- SQL Server 2012
- SQL Server 2014
- SQL Server 2016

**Watched (monitored) SQL Server Analysis Services**

- SQL Server 2005
- SQL Server 2008
- SQL Server 2008 R2
- SQL Server 2012, Tabular and Multi-Dimensional Mode
- SQL Server 2014, Tabular and Multi-Dimensional Mode

- SQL Server 2016, Tabular and Multi-Dimensional Mode

**Watched (monitored) Azure SQL Server instances**

- Azure SQL Database V12 or higher
- All service tiers

> ***NOTE***: SentryOne does not support monitoring any version of SQL Server Express Edition.

> **Note**: When monitoring a Windows Cluster with Performance Analysis for Windows, it is recommended that cluster nodes are monitored individually.

## FEATURES WITH ADDITIONAL REQUIREMENTS

- Deadlock tab and associated data in Performance Analysis for SQL Server is available on SQL 2005 and higher
- Execution plan collection requires SQL 2005 SP2 or higher
- Monitoring Analysis Services with Performance Analysis requires SQL Server 2005 or higher
- Indexes tab and Fragmentation Manager require SQL Server 2005 or higher
- Monitoring the Windows Event Log with Performance Analysis for Windows is only supported for Windows Vista or higher
- Hyper-V Virtualization is supported for Hyper-V v2 (included with Windows Server 2012)
- VMware Virtualization is supported for vCenter 5.5 and vCenter 6
- Monitoring Azure SQL Database targets requires access to the SQL endpoint (over outbound port 1433) with the appropriate Azure SQL Firewall rules created

> ***NOTE***: Windows Vista introduced Task Scheduler 2.0. Task Scheduler 2.0 is backwards compatible with Task Scheduler 1.0, however, Task Scheduler 1.0 is not forwards compatible with Task Scheduler 2.0. For this reason, in order to Watch or Synchronize Task Scheduler 2.0 Instances, you must have a SentryOne Monitoring Service and SentryOne Client running Windows Vista or higher.
>
> Windows 8 and Windows 2012 also introduced changes to Task Scheduler. In order to Watch or Synchronize Windows 8 and Windows Server 2012 Targets, you must have a SentryOne Monitoring Service and SentryOne Client running Windows 8 or Windows 2012.

# 7 Installation and Setup Steps

Once you receive your license and setup file download information, copy the setup executable to the target on which you want to install the SentryOne Monitoring Service and then run it.

If you are upgrading SentryOne from a previous version, it is strongly recommended that you backup your SentryOne Database prior to beginning the process.

Follow these steps:

1. Install SentryOne
2. Complete the Setup Wizard
3. Start Using the Client

## 7.1 Step 1: Install SQL Sentry

A Welcome dialog will be displayed when the SentryOne Setup program is first started, click **Next** to continue or **Cancel** to exit. The License Agreement dialog is displayed next, select the checkbox and click **Next** to continue. For future reference, a copy of the license file is located in the "Client" folder of the installation.

If the setup program detects that SentryOne is already installed it will prompt for removal. The installation process enables you to easily upgrade from previous versions and maintain all of your existing configuration settings, including any Users and Groups, notification settings, etc. Any time SentryOne is upgraded or another component is installed, the existing software is first uninstalled. This is to ensure that all components are of the latest version, and therefore compatible. This only applies to the Client and Server files; the SentryOne Database, where all of your settings and history are kept, is not removed.

> ***NOTE***: .NET Framework 4.5 is required for all installations. A reboot may be required if the .NET Framework files are in use. Temporarily stopping any applications that make use of the .NET Framework can help to avoid a reboot.

### A. CHOOSE THE COMPONENTS

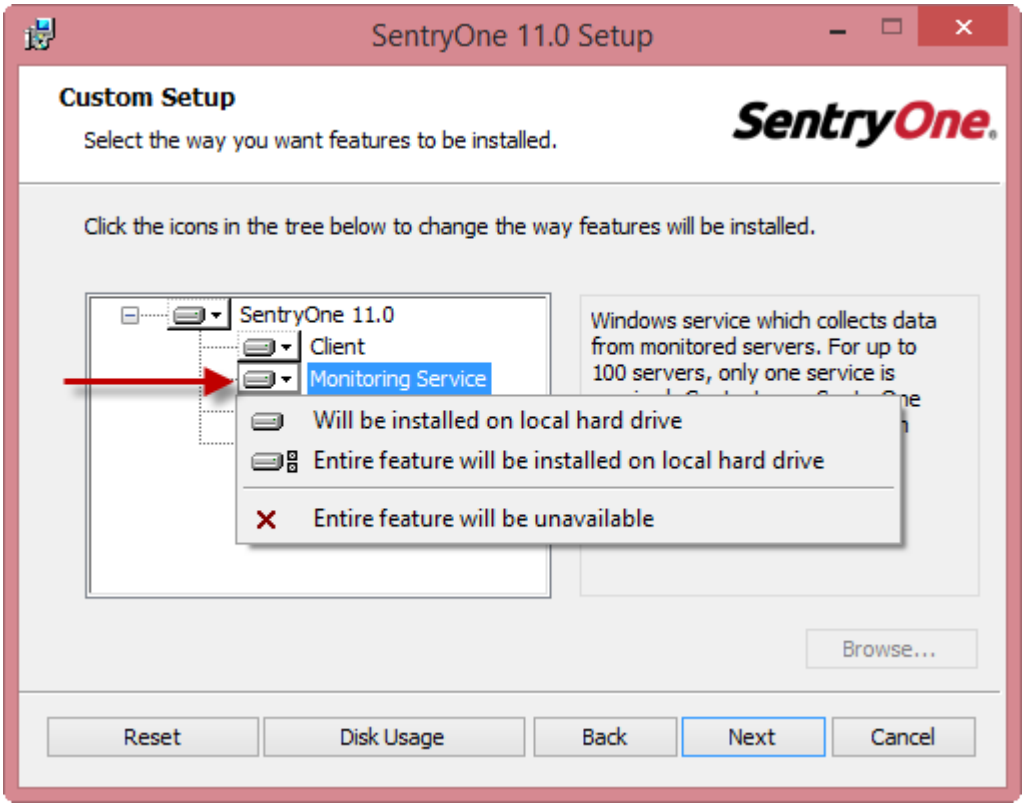
#### DETERMINE THE MONITORING SERVICE COMPUTER

For the initial installation it is recommended that you first **determine which computer the Monitoring Service will be installed on** and then install **both the SentryOne Monitoring Service and Client together** on that computer.

This is analogous to installing the native client tool and SQL Server on the same computer. Even if you don't plan on using the SentryOne Client regularly from this machine, the SentryOne Client is used to enter your license key and will enable you to complete the licensing process during the initial installation. You will be prompted to launch the SentryOne Client and enter your license key at the end of the install.

#### DBA WORKSTATION COMPUTERS

Only one Monitoring Service is required for your SentryOne Enterprise. **Unless desired, there is no need to install a Monitoring Service on any DBA workstation machines.** To install just the SentryOne Client do the following: On the Custom Setup screen, select the drop down arrow next to the Monitoring Service component and choose the *Entire feature will be unavailable* option.




For more information about where components are typically installed see the Installation Recommendations topic.

## B. CHOOSE THE INSTALL LOCATION

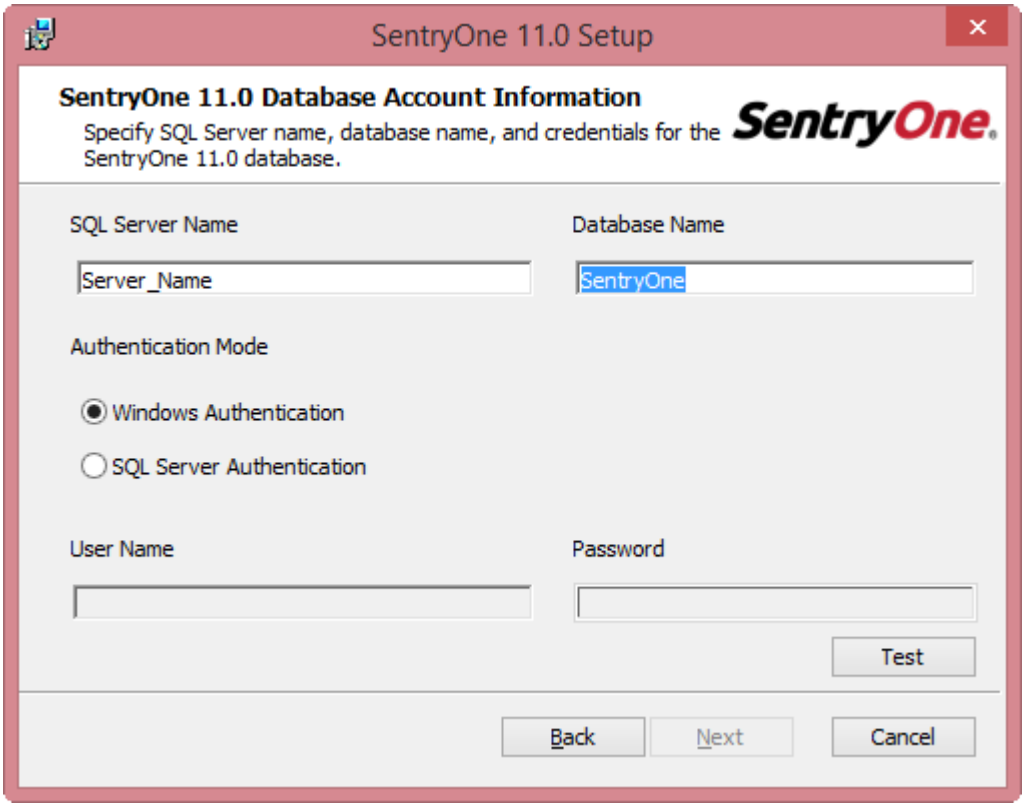
Setup will show the default install location; click **Next** to use the default location. To install to a different location, click **Browse**... then select the appropriate location.

## C. SENTRYONE DATABASE ACCOUNT INFORMATION

On the Database Account Information screen you will chose a location and name for the SentryOne Database. In the *SQL Server Name* field enter the the server instance where you would like to install the SentryOne Database. In the *Database Name* field enter a name for the SentryOne Database. The SentryOne Database will be created as part of the installation process.

If the Windows user account you are using for the installation does not have SysAdmin privileges on the selected SQL Server, deselect the "Windows Authentication" and enter a SQL Server login and password for an account with SysAdmin privileges.

If you are upgrading, specify the existing SentryOne Database. All the necessary schema changes will be applied to the existing database.



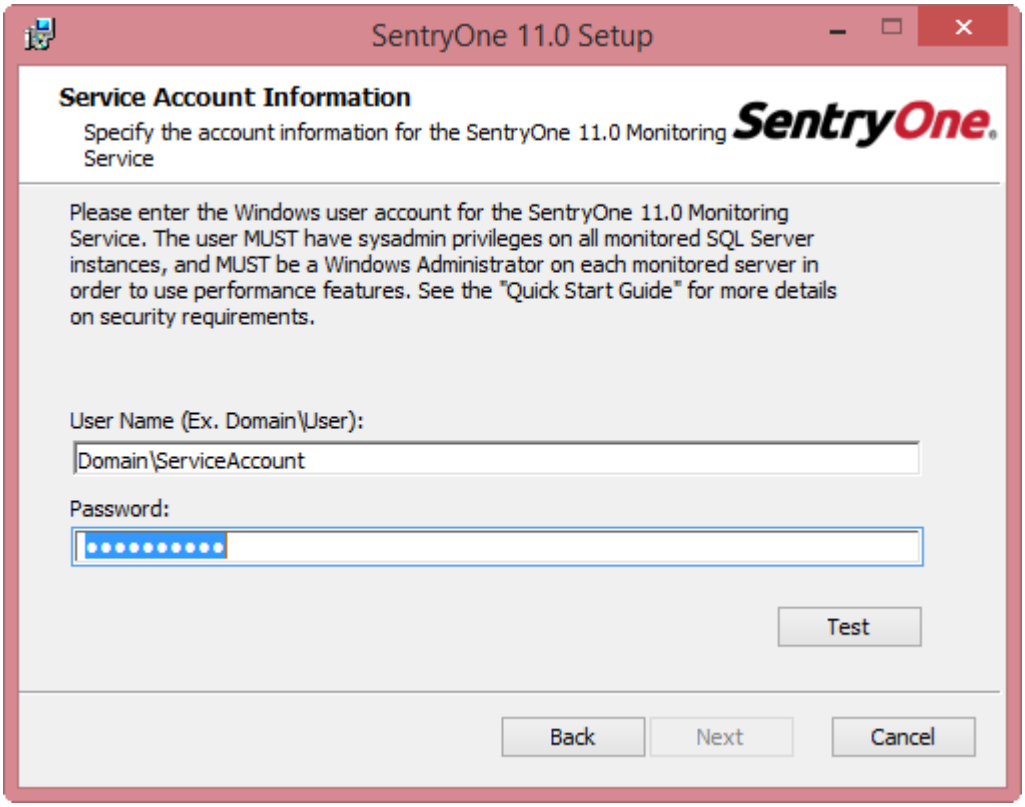

Click the **Test** button to validate the chosen credentials. After a successful test, click the **Next** button to continue the setup.

> ***Note:*** If an existing database has been selected, clicking test will ask you to confirm that you want to upgrade the database.

## D. SERVICE ACCOUNT INFORMATION

At the Service Account Information screen you will enter the Windows account under which the **SentryOne Monitoring Service** will run. This account must have **SysAdmin privileges on each watched SQL Server**. The account must also have **Windows Administrator privileges** on any computer with a watched Windows Task Scheduler Instance, or to collect system level performance metrics with SentryOne Performance Analysis. If the monitoring service does not have Windows Administrator privileges, Instance level metrics can still be collected by using the Limited Access option. For additional information, please see the Monitoring Service Security topic.

> ***Note:*** It is not necessary for this account to be a Domain administrator account. Instead, it is recommended that the service account be a standard user Domain account that has been added to the local Administrators group of each monitored target. For more information please see the Performance Analysis Security Requirements topic.




For information on SentryOne Monitoring Service security settings see the SentryOne Security topics.

Click the **Test** button to validate the chosen credentials. After a successful test, click the **Next** button to continue the setup.

> ***Note***: You can change the service account any time after the initial installation by running the Service Configuration Utility found in the SentryOne program group.

## E. INSTALL

On the **Ready to install SentryOne** screen, click **Install** to begin installation.

## F. COMPLETE SETUP

Click **Finish** to complete setup and launch SentryOne.

> ***Note:***
>
> - If you have not installed the SentryOne Client, you will not see the “Launch” option. You will need to install the SentryOne Client on another machine and run it in order to enter your license key and complete the licensing process.
> - If you have upgraded an existing installation of SentryOne, running the Setup Wizard is not required. All previous settings have been retained.

Click below to proceed:

Step 2: Onboarding

## 7.2 Step 2: Onboarding

### LAUNCH THE SENTRYONE CLIENT

The first time you launch the SentryOne Client, you will be asked a few questions in order to help get you started and provide you with the most accurate information for your monitored environment. Once the onboarding process is complete, you will be able to add your first monitored target, and jump right in to tuning your environment!

During onboarding you will be walked through applying your license, setting up a user for alerting purposes, enabling your email server, configuring email notifications for Advisory Conditions, and setting up your cloud.sentryone.com account. During the process you can easily skip steps that may not be applicable to you, and you can always change any options at a later time from the SentryOne Client.

#### ON ENABLING EMAIL ALERTS

During the onboarding process you will have the ability to enable email alerts. In the **SMTP Server address** field enter the domain name or IP address of the SMTP server to be used for routing SentryOne email notifications. If using localhost, keep in mind this will be the local SMTP server on the machine where the SentryOne Monitoring Service is installed, since it is responsible for sending all notifications. The SentryOne Client does not send any notifications.

The **Email From Address** is the address which will appear on the From line of all email notifications sent by SentryOne. You can also specify a User Name and Password if authentication is required by your SMTP server. This is not usually required in most environments. Use the **Test** button to generate a test email to a specific address.

> ***Note***: You may need to contact your network administrator first to ensure that the IP address of the Monitoring Service computer has been granted both Connect and Relay permissions for the specified SMTP server.

> ***Important***: For the most accurate SMTP test, you should use the Client installed on the SentryOne Monitoring Service computer to send the test message. If you use a Client on a different computer, such as your local workstation, the results may be different. For example, your SMTP server may allow relay from your workstation but not from the SentryOne Monitoring Service computer, in which case the test from your workstation would succeed but the SentryOne Monitoring Service would be unable to deliver notifications.

#### ON ADDING TARGETS AND INSTANCES TO WATCH

Once the onboarding process is complete, you will be able to add your first monitored target.

On the Add Target screen, choose the type of target from the target type drop-down box. Enter the name of the target you would like to monitor in the box below.

For Azure SQL Database and Azure SQL Data Warehouse you will also provide the database or data warehouse name in addition to the Target name (which is the full URL to the endpoint). Both Azure features also require credentials to be provided. These credentials are the SQL logins defined for the

database server or data warehouse, they are not Azure account credentials.

Finally, click **Connect** to test the target for feature availability. This should produce green lights through the list under "Full" if a connection is successful. Otherwise, if there is limited access, you may not be able to collect data for some or all of the given categories (Disk Activity, Disk Space, Performance Counters, Windows Dashboard, or Windows Processes). It is recommended that you try to fix these Limited Access issues, though in some cases Limited Access is your only option, such as using a cloud-hosted SQL Server instance. Because you will not have access to the operating system, this would fall under Limited Access. After reviewing the Feature Availability chart, either run this check again with **Retest** or select **Next** to continue.

> ⚠ ***Important*:**
>
> - If errors occur while adding Targets or Instances, it may be due to problems with security, network connectivity, and/or name resolution. See the topic Security and the SentryOne Server for more details.

Please refer to Additional Tasks for additional configuration options.

Click below to proceed:

Step 3: Start Using the Client!

## 7.3 Step 3: Start Using the Client

Congratulations, you have successfully installed SentryOne, configured global notification settings, and are now ready to start using the SentryOne Client for managing events across your enterprise. Use the different options on the Get Started screen to start exploring the features in SentryOne.

Please refer to the *SentryOne User Guide* available online and through the Client Help menu, for additional information about available features.

### MAINTENANCE

Just as with any other SQL Server database, it is important that regular maintenance activities be performed on the SentryOne database to ensure optimal performance. Please see the SentryOne Database Maintenance topic in the *SentryOne User Guide* for more details and recommendations.

# 8 Additional Tasks Overview

## ADDITIONAL TASKS

Refer to these topics for additional information on configuring SentryOne.

- Add Additional Users and Groups
- Monitor Additional Instances
- Customize Global Settings

# 8.1 Add Users and Groups

The **Contacts** node in the Navigator contains the **Users** and **Groups** sub-nodes. This is where you create and maintain Users and Groups for notification purposes. **At least one user is required for SentryOne to be able to send notifications**.

Click the **Users** node to add a new user. Enter the user's name, email address, optional pager address (SMTP-based), and an optional description. You can add as many users and groups as you want at this point - groups are optional. Click **Save** when you are finished adding each user.

For more information about Users and Groups see the Contact Management topic in the *SentryOne User Guide*.

# 8.2 Monitor Additional Targets and Instances

## TERMINOLOGY

When the word *Target* is used, we are referring to the device that houses your data, whether it's a physical server, cloud installation, or APS appliance.

*Instance* is referring to an instance of SQL Server or SSAS, or an installation of SharePoint that exists on a Target in your environment.

## SUPPORTED TARGETS AND INSTANCES

Currently, SentryOne supports the monitoring of Windows, Azure SQL Database, SQL DW, and APS appliance Targets. Supported Instances include SQL Server, SSAS, and SharePoint. For additional details, including supported versions, please see our Quick Start guide.

## ACCESS LEVEL

When adding a new Target, the first step is a Feature Availability test. The results of this test will determine whether the Target will be added with Full Access or Limited Access. When a Target is added with Full Access, the monitoring service will collect Windows level metrics and you will have full access to the features of Performance Analysis.

If the Target fails the Feature Availability test, you can click the Troubleshooting link and attempt to

resolve the issue. After applying a solution, you can then retest the Target. There are some situations in which Limited Access is the only option. For example, if you are monitoring a cloud-based SQL Server instance, you will likely not have access to the OS. When Limited Access is applied, the monitoring service will not collect Performance Counters, and access to the Windows Dashboard, Disk Activity tab, Disk Space tab, and Windows Processes tab will be restricted. For additional information on adding a Windows Target, please see the Performance Analysis for Windows topic.

## ADDING INSTANCES

You can easily add additional monitored Instances to your SentryOne environment. This is accomplished by right-clicking either the Shared Groups node, a Site node, a Target Group node, or an existing Target node in the Navigator pane and using the **Add Instance** command. You can also add an Instance through the **File** menu.

In the **Add Instance dialog** you may choose the desired **Instance Type** from the drop-down menu (Analysis Services Instance, SharePoint Server Instance, SQL Server Instance, Windows Instance).

## WATCHING INSTANCES

When you add a new Instance to your environment it will be monitored by default unless you explicitly opted not to "watch" the new instance. SentryOne will monitor Instances or objects with a status of “watched”. Instances or objects that are not being watched will be displayed with a grayed-out icon next to their name in the Navigator tree view.

Unwatched Instances or objects can have their status set to watched through their respective context menus with the **Watch** command. Once you have watched a new Instance the SentryOne Monitoring Service will start actively monitoring the Instance and its objects, and begin honoring any associated configured Conditions and Actions.

> ***Note***: Immediately after adding a Instance or setting a Instance to watched status SentryOne will begin to synchronize with that Instance. Exactly how long the synchronization process takes depends on the number of objects associated with the Instance, the amount of historical data available, and how many Instances are being watched at the same time. The *Watch Status Window* will keep you informed of the process and alert you about any errors.

## MODIFYING INSTANCE PROPERTIES

After you’ve added an Instance, you may need to change how SentryOne connects to the target. When right clicking on an Instance, you’ll see two options for connection properties: User and Monitoring Service.

**User Connection Properties** define how your SentryOne Client will connect to a monitored server for the current user. These properties can vary for each Client in your environment. The SentryOne Client only connects directly to a monitored server under specific scenarios. More details regarding those scenarios and specific security requirements can be found in the Client Security section of our Quick Start Guide

**Monitoring Service Connection Properties** define how the SentryOne Monitoring Service will connect to the selected server. The setting can be applied from any SentryOne Client. The Monitoring

Service Connection can be configured by right clicking on the Instance and selecting Monitoring Service Connection Properties from the context menu. Please see our Quick Start Guide for additional information on Monitoring Service Security.

Within the Connection Properties window, there are several properties that can be changed.

- Enable Integrated Authentication. This setting tells the Instance to use the integrated Windows account information.
- Credentials. This is where you enter SQL Server credentials if you are not using Integrated Authentication.
- Alias. By default, you will see the server name that you initially entered when adding the Instance.
- Port. This setting is used to connect to SQL Server if it has been configured to a non-standard port.
- Access Level. This setting is used to assign the level of access that SentryOne has to the selected target. A Target with limited access will not be able to access Windows based features, such as the Windows Dashboard, Windows Processes tab, Disk Space tab, or Disk Activity tab. Limited access targets will also not have access to PerfLib Performance Counters for that target.

All of these settings are available for SQL Server, Oracle, and SharePoint Instances.
SSAS Instances only offer the Port setting.
Targets offer the Access Level setting.

> If you are monitoring the Instance with Performance Analysis, changing the Monitoring Service Connection Properties to SQL Server credentials is not supported. Please see the Quick Start Guide for additional information regarding Performance Analysis Security Requirements.

## 8.3 Introduction to Actions and Settings

When you run the Setup Wizard a number of global Settings are configured for your installation. If you entered your SMTP Settings and added a User, a number of default Conditions and Actions were also added to help you get up and running quickly. As a reminder, the Wizard can be accessed through the **Help** menu at any time.

Before proceeding you should introduce yourself with these basic SentryOne Alerting and Response System concepts.

| | |
|---|---|
| **Conditions** | Conditions describe the various states of any monitored objects. |
| **Actions** | Actions determine what happens when a Condition is met. |
| **Settings** | Settings define criteria for when a Condition is considered to be met. Certain Settings known as Source Settings are used to define what events are collected by SentryOne. |

There are a couple of ways to see how Settings and Conditions/Actions are configured for your SentryOne installation. You can use the **Reports** menu to run the *Active Settings List* and the *Configured Actions List* reports. (Reports menu > General) Or you can view configured Actions and Settings directly in the **Actions and Settings Pane**. By default the **Actions and Settings pane** is displayed on the right side of the SentryOne Client. If you do not see the **Actions and Settings pane**, it can be restored with the **View** menu.

If you would like to configure global Actions or Settings, be sure that the Shared Groups node is selected in the Navigator pane. The Shared Groups node is the global or root node of your SentryOne installation. The SentryOne Alerting and Response System uses the principle of inheritance, so any Action or Setting you configure at the Shared Groups node will be passed down to all applicable objects below it.

For example, if you configure a *Send Email Action* for the *SQL Server Agent Job: Failure Condition* at the global level (Shared Groups), you will receive an email anytime a SQL Server Agent job fails across your entire monitored enterprise.

You can further refine Actions or Settings at each level, as needed. For instance, if you have a development server in your environment that you don't wish to be alerted about, you can easily disable the *Send Email Action* at the Instance level. This configuration would only apply to that Instance and it would not affect any other server in your environment. This level of control gives you the ability to determine exactly what happens in response to events occurring on your monitored targets.

There are several levels within the SentryOne Hierarchy where you can configure applicable Actions and Settings. These are outlined below.

```
Shared Groups
 |
 -Sites
  |
 -Target Groups
   |
   -Target
    |
    -Instance
     |
      -Object
```

For a more in-depth look at the SentryOne Hierarchy and other alerting related features see the Alerting and Response System topic in the *SentryOne User Guide*.

## 8.3.1 How to Configure Actions

As a reminder **Conditions** describe the various states of any monitored objects, and **Actions** determine what happens when a **Condition** is met.

The Conditions displayed in the Actions pane will change depending on which node or object is selected in the Navigator pane. If you do not see the Actions pane once you have selected your desired node in the Navigator pane, use the View Menu (**View →General Actions**).

If you select the Shared Groups node you will see globally applied Conditions in the Actions pane. When you select any applicable object level below the Shared Groups node, you will see two specific sets of applied Conditions in the Actions pane.

The top section is the *Inherited Section*, which shows you any applied Conditions that are being passed down to the current level. Beneath that, is the *Explicit Section*, which shows you applied Conditions that have been set at the current level. Each Action that you set up in your environment will have an associated behavior. This behavior controls how the Action will be carried out relative to any inherited Actions. Please see the table below for an introduction to Action behaviors.

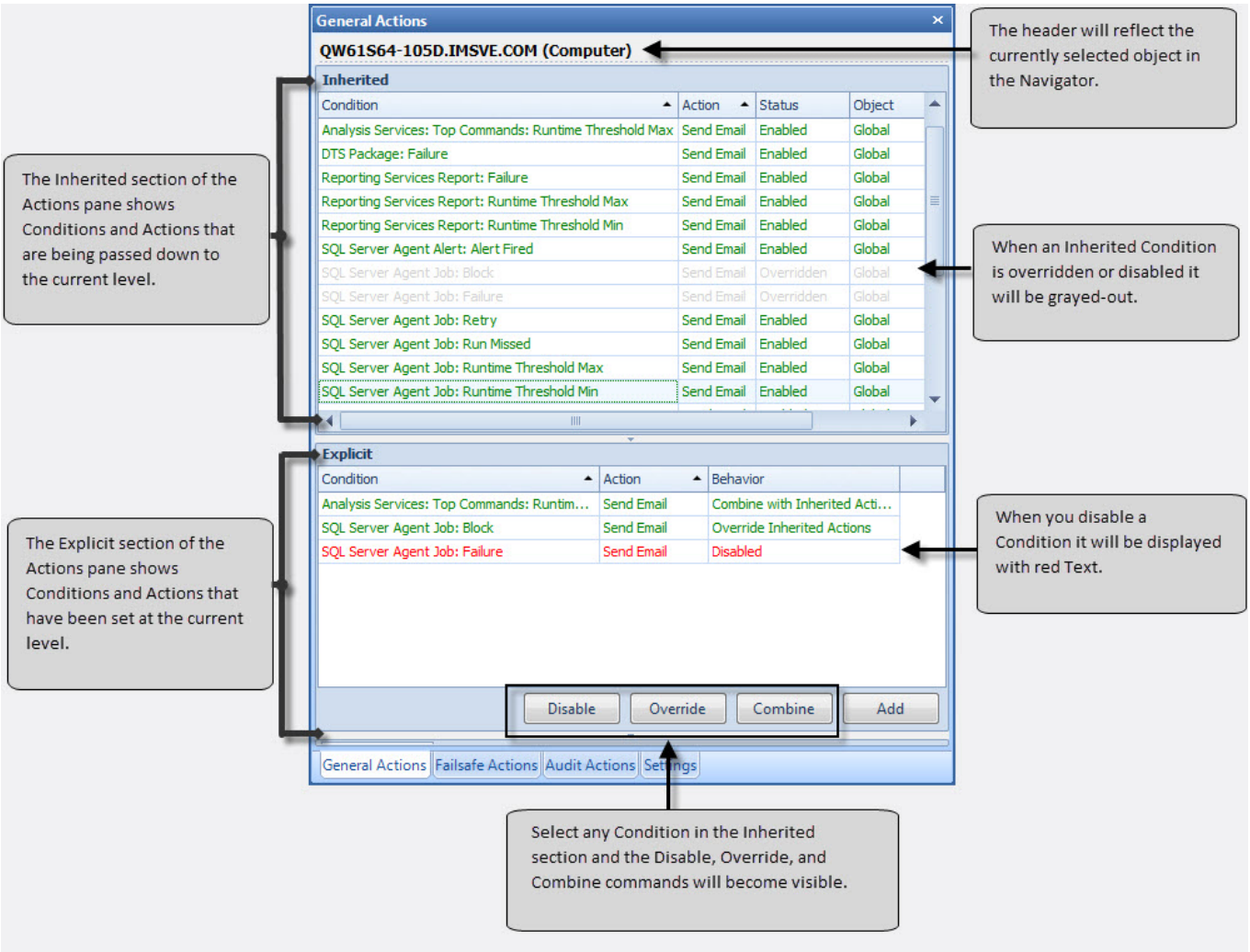
**Action Behaviors**

| | |
|---|---|
| **Override** | This behavior can be thought of as a special set of instructions which are followed instead of the passed down(inherited) instructions. |
| **Combine** | This behavior can be thought of as a set of instructions which are followed in addition to the passed down(inherited) instructions. |
| **Disable** | This behavior can be thought of as a special set of instructions which simply disallow the passed down(inherited) set of instructions. |

To add a new Action, select the desired node in the Navigator pane. Next you will want to click the **Add button** found in the Actions Pane. This will open the Action Selector window. Expand the applicable Object and Condition. Use the check box(s) to select which Actions should be taken in response to this Condition being met. Click the **OK button**.

You may also choose to quickly *Disable*, *Override*, or *Combine* any inherited Actions. To do so, select the Condition in the Inherited Section of the Actions pane and choose the desired command (**Disable button, Override button, or Combine button***)*. When an Inherited Condition is overridden or disabled, it will still show up in the Inherited Section, but its text will be grayed-out and its status will say Overridden.

**Action pane**




For more information about Actions and Conditions please see the Alerting and Response System topics in the *SentryOne User Guide*.

## 8.3.2 How to Configure Settings

As a reminder, **Settings** define criteria for when a **Condition** is considered to be met. Certain Settings known as Source Settings are used to define what events are collected by SentryOne.

To configure Settings first select the desired node in the Navigator pane. For instance, select the Shared Groups node if you want to configure Settings globally, or select an individual Instance node if you would like to configure Settings specific to just that Instance. If you do not see the Settings pane once you have selected your desired node in the Navigator pane, use the View Menu (**View →Settings**). Next, you will want to use the drop-down lists found in the Settings pane to select the Settings which you would like to configure. See below for examples.

For instance, if you wanted to configure the **Top SQL Minimum Duration Collection Setting** globally:

1. Select the Shared Groups node in the Navigator Pane
2. In the Settings pane, use the top drop-down list and select **SQL Server Settings**.

3. Use the second drop-down list to select **Top SQL Source**. You should now see the Top SQL Source Settings that are being applied Globally.
4. Change the **Minimum Duration** to the desired value, it will be saved automatically.

If you wanted to configure the **Top SQL Minimum Duration Collection Setting** for an individual Instance:

1. Select the desired Instance node in the Navigator pane.
2. In the Settings pane, use the drop-down list to select **Top SQL Source**. You should now see the Top SQL Source Settings that are being applied for that Instance.
3. Change the **Inherit From Parent Setting** to False.
4. Change the **Minimum Duration** to the desired value, it will be saved automatically.

## ADJUSTING GLOBAL RUNTIME THRESHOLD SETTINGS

By default, the global Runtime Threshold Settings for SQL Server Agent jobs are set at a **Minimum Runtime Threshold Percent** of 10% and **Maximum Runtime Threshold** Percent of 250%. This means that anytime a job runs for less than 10% of its average runtime or longer than 250% of its average runtime you will be notified. If you find you are receiving too many notifications, these settings can be adjusted. See example below:

If you wanted to configure the **SQL Server Agent Job Maximum Runtime Threshold Percent** globally:

1. Select the Shared Groups node in the Navigator Pane
2. In the Settings pane, use the top drop-down list and select **SQL Server Settings**.
3. Use the second drop-down list to select **SQL Server Agent Job**. You should now see the SQL Server Agent Job Settings that are configured globally.
4. Change the **Maximum and Minimum Runtime Threshold Percents** to the desired values.

You can also specify explicit *time-based thresholds* here. Time-based thresholds are usually less valuable at the global level, particularly the **Minimum Runtime Threshold** which doesn't have much value at all globally. Explicit *time-based thresholds* tend to be more applicable at the actual Instance or object level for overriding the global *percentage thresholds* on a case-by-case basis.

> ***Note:*** Anytime an explicit time-based threshold is specified it will override the percentage based thresholds for that object.

For example, consider a job that has a great deal of volatility in runtime such as a transaction log backup, and can run for anywhere between 30 seconds and 30 minutes, with an average runtime of 5 minutes. To avoid unnecessary percentage-based threshold notifications for the job, you might want to set its **Maximum Runtime Threshold** to "35 Minutes" and **Minimum Runtime Threshold** to "20 Seconds". This can be done by selecting either the job's node in the Navigator or an instance of the

job on the calendar, then following the same steps as above to access and change the job's runtime threshold settings.

For more information about Settings please see the Alerting and Response System topics in the *SentryOne User Guide*.

# 9 SentryOne Security Overview

The **Quick Start Guide** covers the following topics related to SentryOne Security, including required permissions for the various SentryOne components.

| Security Topic | Description |
|---|---|
| Monitoring Service Security | This topic discusses the permissions required by the **SentryOne Monitoring Service** account when watching (monitoring) Instances. |
| Client Security | This topic discusses the permissions required when running the **SentryOne Client**, including scenarios in which the Client connects directly to a monitored target. |
| Watching Targets Across Domains | This topic is a brief overview of the options available for watching (monitoring) targets across domains, including information about **pass-through authentication** and configuring **SentryOne Sites** within your environment. |
| Non-Windows Environment | This topic discusses the options for watching (monitoring) Instances in a non-Windows environment, including **pass-through authentication**. |
| SentryOne Performance Analysis | See this section for advanced information about the Performance Analysis Security Requirements, including Port Requirements for monitored targets. |
| Azure SQL Database and Data Warehouse | This topic covers security aspects specific to Azure SQL Database and SQL Data Warehouse. |

The User Guide covers the following topics related to restricting user access within SentryOne.

| Security Topic | Description |
|---|---|
| Rights Based Security | This topic discusses restricting user access within the **SentryOne Client** based on Windows and SQL Server Authentication accounts. |
| Role Based Security | This topic discusses restricting user access within the **SentryOne Client** based on **SentryOne Database** roles. |

## 9.1 Monitoring Service Security

The **SentryOne Monitoring Service** is a Windows service which runs in the context of a Domain account.

- **This account must have SysAdmin privileges on each watched SQL Server.**
- **The account must also have Windows Administrator privileges** on any computer with a

watched Windows Task Scheduler Instance, or to collect system level performance metrics with SentryOne Performance Analysis.

It is not necessary for this account to be a Domain Administrator account. Instead, it is recommended that the service account be a standard user Domain account that has been added to the local Administrators group of each monitored target. For more information about security and SentryOne Performance Analysis, please see the Performance Analysis Security Requirements topic.

> ***Note*:** As of SQL Server 2008 the local Administrators group of a Windows server is no longer automatically given access to a SQL Server instance installed on that Windows server. Keep this in mind when installing SentryOne for use with SQL Server 2008 and above.
>
> Adding the service account to the local Windows Administrators group for the SentryOne Database server does not automatically grant the service user access to the SentryOne Database.

## CHANGING THE MONITORING SERVICE CREDENTIALS

After the initial installation, the **Service Configuration Utility** is used to update or change the credentials of the **SentryOne Monitoring Service** account. The **Service Configuration Utility** can be accessed within the SentryOne program group in the Windows Start Menu.

Using the **Service Configuration Utility** is the only supported way of changing the **SentryOne Monitoring Service** credentials. For more information please see the Service Configuration Utility topic in the SentryOne User Guide.

## MONITORING SERVICE CONNECTION PROPERTIES

If the Monitor Performance setting is set top False for a target, and you do not have a need to utilize General Performance Monitoring features, you may configure the Monitoring Service to use SQL Server Authentication. This is done through an Instance's **Monitoring Service Connection Properties**.

To access the **Monitoring Service Connection Properties** for an Instance, right-click the Instance and choose the **Monitoring Service Connection Properties** command. From the Connection Properties dialog, uncheck **Use Integrated Authentication**, and enter the SQL Server Authentication account you would like the Monitoring Service to use for the Instance.

## ADJUSTING TARGET ACCESS LEVEL

There may be times when you wish to monitor an Instance where OS level metrics through WMI and/or the Windows Performance Library are inaccessible. This is occasionally the case for cloud based or hosted servers. In these circumstances, a Target may be added with Limited Access. This suspends attempts to access those resources which are required for some functionality such as the Disk Space and Activity tabs, and Windows metrics on the Performance Analysis Dashboard. If access to those resources has been resolved, the Access Level can be set to “Full” in **Monitoring Service Connection Properties** at the Target level in the Navigator Pane. Similarly, if a watched Target starts generating errors due to connectivity issues with the OS level resources that cannot be resolved, changing the Access Level to “Limited” will allow you to continue monitoring non-OS metrics without triggering

connectivity errors for the Target.

> ⚠ **Important:** If you configure SQL Authentication for an Instance which is being monitored with SentryOne **Performance Analysis**, Performance Analysis will not be able to collect Windows level metrics for that Instance. This is because Performance Analysis collects various performance and configuration data directly from Windows, and requires a higher level of access to the operating system than does Event Calendar. See the Performance Analysis Security Requirements topic for more information.

#### STARTING THE MONITORING SERVICE

The **SentryOne Monitoring Service** will start automatically after installation. It will become active upon detecting a valid license on the SentryOne Database. If for some reason the Service fails to start, you may follow these directions to start the service manually.

1. Select the **Services** icon from **Control Panel -> Administrative Tools**.
2. From the list of services select **SentryOne Monitoring Service**, then right-click and select "Start", or click the "Play" button on the toolbar.

## 9.2 Client Security

Although the **SentryOne Client** receives the majority of its information from the **SentryOne Database**, there are times when the Client must connect directly to a monitored server in order to receive information.

#### WHEN DOES THE SENTRYONE CLIENT CONNECT DIRECTLY TO A MONITORED SERVER?

The SentryOne Client connects directly to a monitored server when:

- an Instance is Watched
- a real-time action is initiated
  - a job is manually started or stopped
  - a job is rescheduled
- a QuickTrace is run

The SentryOne Client will also connect directly with the monitored target when a forced metadata and history synch is performed. Selecting **CTRL + Refresh** on the toolbar will perform this action. This is different than just selecting the **Refresh** button alone, which would only retrieve information from the SentryOne Database.

#### AUTHENTICATION METHOD USED WHEN THE CLIENT CONNECTS TO A MONITORED TARGET

In those cases where the Client does need to connect directly to a monitored Instance, the authentication method used varies depending on the specified **User Connection Properties** of that Instance. By default, the Client will use the credentials of the interactive user, whenever it needs to connect directly to an Instance.

As an alternative to integrated authentication, you may specify database specific *credentials* in the **User Connection Properties.** The **User Connection Properties** for an Instance can be accessed through the right-click context menu of the Instance. First unselect the **Use Integrated Authentication** check box and then enter your desired account information. For example, for a SQL Server Instance you would want to enter a SQL Server Authentication Account with the desired Server Role.

---

#### SHARED GROUPS NODE VS SQL SERVER REGISTRATIONS NODE

There are a few differences regarding how authentication works depending on whether you are accessing the Instance from the context of the **Shared Groups** node or the context of the **SQL Server Registrations** node in the Navigator pane.

For SQL Server Instances accessed within the context of the **Shared Groups** node, **Windows authentication** is used by default. However, if you have specified SQL Server credentials using the **User Connection Properties** context item, those credentials will be used instead.

For SQL Server Instances accessed within the context of the **SQL Server Registrations** node, the Client uses the authentication method and credentials defined for the corresponding SSMS registration. This is also referred to as the "native registration" and is accessed using the Instance's **Edit Registration Properties** context menu item.

If SQL Server authentication credentials are set using the **User Connection Properties** context item, those credentials will be used instead, and they will effectively override the authentication settings of the native registration. The initial connection to the target will always be made using the native registration credentials, however, so that the Client can ascertain the true identity of the SQL Server, and ensure it isn't already being watched using a different name, as can be the case when an alias has been configured for the target.

---

#### RESTRICTING ACCESS AND SERVER VISIBILITY IN THE SENTRYONE CLIENT

For information about restricting user access within the **SentryOne Client** based on Windows and SQL Server Authentication accounts see the Rights Based Security topic in the **SentryOne User Guide**.

For information about restricting user access within the **SentryOne Client** based on **SentryOne Database** roles see the Role Based Security topic in the **SentryOne User Guide**

## 9.3 Watching Targets Across Domains

It is possible to monitor/watch instances across domains with SentryOne even when there is no trust relationship between them. The best option to achieve this depends on the resources available and number of targets you wish to watch. See below for a short explanation of each option; select the associated link for more information.

#### OPTIONS FOR WATCHING TARGETS ACROSS DOMAINS

| Option | Description |
| --- | --- |
| Pass-through Authentication | Pass-through authentication enables Windows targets in different domains or in non-Windows network environments to communicate with one another by using identical user accounts and passwords on each computer.<br><br>This solution is ideal when you only need to monitor a few targets outside of your primary domain and you do not have the resources available to install another Monitoring Service in the secondary domain. |
| Site Configuration | Sites represent a logical grouping of Targets, Instances, SMTP Servers, and Monitoring Services within your SentryOne environment. With the Site Configuration option, you will install a SentryOne Monitoring Service in each domain/location where you have targets that you wish to monitor.<br><br>Each Monitoring Service will only poll the targets in their own domain. The Monitoring Service located outside of your primary domain will use either Pass-through authentication or SQL Server authentication to communicate with the SentryOne Database server.<br><br>This solution is ideal if you have a need to monitor a large number of targets outside of your primary domain, or have a need to monitor targets which are geographically separated from your main installation.<br><br>This solution also requires that you have the required resources available in the secondary location to install a Monitoring Service. |

## 9.3.1 Pass-through Authentication

**Pass-through authentication** enables Windows computers in different domains or in non-Windows network environments to communicate with one another by using identical user accounts and passwords on each computer.

For example, if user "**JoeDBA**" with password "**SQLrocks!**" is created on **SERVER1** and **SERVER2**, **JoeDBA** will be able to connect and authenticate directly from **SERVER1** to **SERVER2**, and vice versa, without using domain-level authentication.

It is the job of the **SentryOne Monitoring Service** to collect data from monitored targets, then store the data in the SentryOne Database for analysis with the **SentryOne Client**. In the above scenario, **SERVER1** may be the computer where the **SentryOne Monitoring Service** is running, and **SERVER2** either the monitored computer, or the computer where the **SentryOne Database** resides.

***Note***: Additional configuration may be required on machines running Windows Vista and higher with the introduction of User Access Control (UAC). When a remote connection is made using pass-through authentication the machine is unable to resolve elevated permissions under UAC, and for WMI and registry purposes the account is treated as a regular (non-admin) user, even if

> the account exists in the local administrators group.
>
> Please see the Performance Analysis: WMI or Registry Access KB article for more information and configuration details about using pass-through authentication on Windows Vista and higher:

> ***Important:*** SQL Server authentication can be used for any watched SQL Server Instance using an Instance's **"Monitoring Service Connection Properties"** context menu item. This can eliminate the need for pass-through authentication if SentryOne's performance monitoring isn't being utilized to collect Windows performance counters from the targets, and if you aren't monitoring the target with Performance Monitor or Event Manager Windows Task Scheduler.
>
> If performance monitoring is required either via SentryOne Performance Analysis or you need to watch a Windows Task Scheduler, pass-through authentication may still be required.

## 9.4 Non-Windows Network Environment Security

If you are not using Windows Active Directory for domain management, you may need to take additional steps to ensure SentryOne will work properly. The primary means by which this is accomplished is using Windows pass-through authentication.

SQL Server authentication can be used for any watched SQL Server Instance in a non-Windows network using an Instance's "Monitoring Service Connection Properties" context item.

### SENTRYONE CLIENT

In non-Windows networks, in order to connect to watched SQL Servers using the SentryOne Client you must either:

1. Use SQL Server authentication for any SQL Server registrations, or the SQL Server Instance.
2. Use Windows pass-through authentication. This means the Windows user using the SentryOne Client must also exist on the target SQL Server computer. The user name and password on each target must match exactly.

### SENTRYONE MONITORING SERVICE

Pass-through authentication is the only means by which the SentryOne Monitoring Service can collect Windows performance counters or watch Windows Task Scheduler in a non-Windows network environment. Therefore the service user account must exist both on the service target and all monitored targets, and the user name and password must match exactly.

## 9.5 Azure SQL Database and Data Warehouse Security

### AZURE SQL DATABASE AND DATA WAREHOUSE SECURITY

The Microsoft Azure SQL Database and Data Warehouse services are protected by a firewall. As both services are exposed on the internet, the Azure SQL Firewall is in place to help protect access to your

data. When creating a new Azure SQL Database or Data Warehouse Target, the connectivity verifications will check to ensure that an Azure SQL firewall rule is correctly configured and indicate a warning if it is not.

### AZURE SQL DATABASE AND DATA WAREHOUSE FIREWALL CONFIGURATION

The Azure SQL Firewall settings are configured using the Azure Portal or via the command line utilities: PowerShell or the Cross Platform CLI tool. To learn more about the Azure SQL Firewall and how to configure it please read the How to configure an Azure SQL database firewall documentation from Microsoft.

> ⚠ **Important:** Since the Azure SQL Firewall rules can change, the Monitoring Service can lose access. If this occurs, notifications will appear in the System Status and on the dashboard as a warning.
>
> While the firewall is blocking the Monitoring Service, no data will be retrieved from the target.

# 10 Appendix

The Appendix contains the following topics:



## 10.1 SentryOne Performance Analysis

## PERFORMANCE ANALYSIS

This section contains the following topics:



For general information concerning SentryOne Performance Analysis, including an explanation of the Performance Metrics displayed on the Dashboard, please see the Performance Analysis section of the **SentryOne User Guide**.

### 10.1.1 Performance Analysis Security Requirements

Performance Analysis collects various performance and configuration data directly from Windows, and therefore requires a higher level of access to the operating system than does Event Manager. The easiest approach is to either make the SentryOne Monitoring Service account a Domain Administrator level account, or a member of the local Administrators group on any watched targets.

In some scenarios it may be possible to use a non-Administrator service account, although this is not an officially supported approach. This article identifies the steps required to do this:

1. Enable DCOM on the SentryOne Server machine, SentryOne Client machine, and the server to be watched. (See this link for instructions.)
2. Give the SentryOne Monitoring Service account proper permissions to the required WMI

namespaces. You can do this by going to the properties for "WMI Control", found under "Services and Applications" in the Computer Management Client. On the Security tab, ensure that the SentryOne Monitoring Service account has at least "Enable Account" and "Remote Enable" checked for the CIMV2 and WMI nodes.

> **NOTE:** *WMI providers and versions will vary from server to server, and whether or not non-administrative access will function properly for a particular WMI provider is directly dependent on whether or not the provider was designed to support this. Many providers simply are not, including many designed by Microsoft.*

Please consider the following example:

SERVER-A is the exact same make and model as SERVER-B, and both servers are on the same domain. The SentryOne Monitoring Service user account is a Domain User, but does not have Administrator privileges on either server. Performance Analysis can successfully watch SERVER-A, but is unable to watch SERVER-B. The two servers are configured identically, with one exception -- an additional network adapter from Acme Networking was installed in SERVER-B. Unfortunately, Acme Networking didn't design the associated WMI provider to support non-administrative access, therefore Performance Analysis will not be able to successfully watch SERVER-B as a non-Administrator. In this scenario, the only options are to either replace the network adapter with one that is known to support non-administrative access, or to contact Acme Networking to see if they have an updated version of the provider that supports non-administrative access.

## 10.1.2 Performance Analysis Required Ports

In order for Performance Analysis to properly monitor a target on the network, the following ports on the monitored target must be accessible to the SentryOne Server machine(s):

For SQL Server access:

tcp 1433 (or whatever port is used by SQL Server)

For Azure SQL Database and SQL Data Warehouse:

tcp 1433

For Windows Performance Counter access:

tcp 445 (SMB, RPC/NP)

For WMI access:

tcp 135 (RPC)

**-and-**

one of these ranges:

tcp 49152-65535 (RPC dynamic ports -- Vista and Win2008)

**-or-**

tcp 1024-65535 (RPC dynamic ports -- NT4, Win2000, Win2003)

**-or-**

a custom RPC dynamic port range **(see below)**

The only one that may be tricky for firewalls are the RPC dynamic ports. WMI (or any other process that uses DCOM) connects to a  initially using port 135, and the target responds with a dynamic port number for WMI to use for the rest of the session. This port can be in one of the ranges above, which are quite large by default.

To address this, **you can easily specify a custom range for RPC dynamic ports**. You may have already done this in your environment in order to enable networked DCOM access for other applications. It is recommended that you start no lower than port 50000, and allocate no fewer than 255 dynamic ports.

For example, to do this on Server 2008, you can use this command:

```
netsh int ipv4 set dynamicport tcp start=50000 num=255
```

You may need to reboot. More info: http://support.microsoft.com/default.aspx/kb/929851

On other Windows versions, you can use DCOM config in Component Services (http://support.microsoft.com/kb/300083) or the registry (http://support.microsoft.com/kb/154596). You will need to reboot.

You will also need to have your network administrator open up the same port range on the firewall between the SentryOne Server machine and any servers monitored with PA.

How to configure RPC dynamic port allocation to work with firewalls

http://support.microsoft.com/kb/154596

How To Restrict TCP/IP Ports on Windows 2000 and Windows XP

http://support.microsoft.com/kb/300083

How to troubleshoot WMI-related issues in Windows XP SP2

http://support.microsoft.com/kb/875605

DCOM port range configuration problems

http://support.microsoft.com/default.aspx/kb/217351

The default dynamic port range for TCP/IP has changed in Windows Vista and in Windows Server 2008

http://support.microsoft.com/default.aspx/kb/929851

Service overview and network port requirements for the Windows Server system

http://support.microsoft.com/kb/300083

## SEE ALSO

.

## 10.1.3 Performance Analysis Data Capacity Planning

## PERFORMANCE ANALYSIS DATA CAPACITY PLANNING

Performance Analysis (PA) uses the **SentryOne** database to store all of the performance data it collects, utilizing a high performance storage scheme. **Event Manager (EM) only users should expect their existing database to approximately double in size** if all of the existing SQL Servers watched by EM, are watched by PA. This is a very rough estimate, however, since exactly how much space will be used by PA is directly dependent on:

- The number of databases on the watched SQL Servers, since some of the performance counters collected by PA are database specific.
- The number of physical disks on the watched targets, since related counters are disk specific.
- The Minimum Duration specified for the Top SQL event source. The default global setting is five seconds, meaning that any batches or stored procedures that run for longer than five seconds will be collected. If this threshold is lowered, the amount of Top SQL data collected will increase. Note that a different Minimum Duration can be specified for each SQL Server.
- Whether or not "Collect Statement Events" is set to True for the Top SQL event source. The default is False. If enabled, this may increase the amount of Top SQL data collected by a factor of two or more. This setting is also adjustable for each SQL Server.
- The performance data retention settings. Different settings can be specified for detailed (or raw) performance data, rolled up performance data, and Top SQL/Blocking SQL/Deadlock data.
    - For detailed performance data, retention is specified in hours for each performance counter category in the HistoryDataRetentionHours column of the PerformanceAnalysisCounterCategory table. The default may be either 48 or 72 hours, depending on the category. Raw data is shown by default on the Dashboard and Disk Activity tabs whenever the current date range is <=30 minutes. Over 30 minutes, rolled up data is used.
        - If you have an unusually large number of databases on SQL Servers monitored by PA, you may consider reducing the retention hours for the SQLSERVER:DATABASES and SQLPERF:VIRTUAL_FILESTATS categories. Data for these categories are stored in the PerformanceAnalysisDataDatabaseCounter and PerformanceAnalysisDataDiskCounter tables respectively.
        - Likewise, if you have an unusually large number of physical disks per target monitored by PA, you may consider reducing the retention hours for the PHYSICALDISK category. Data for this category is stored in the PerformanceAnalysisDataDiskCounter table.
        - Data for all other categories is stored in the PerformanceAnalysisData table.
        - Generally, it is a good idea to keep the retention hours the same for categories that are stored in the same table, otherwise page splitting and fragmentation may result during the pruning process which may eventually affect performance.
    - For rolled up performance data, retention is specified in hours for each rollup level in the

HistoryDataRetentionHours column of the PerformanceAnalysisDataRollupLevel table. Rollup data for each break level (specified by the LevelBreakMinutes column) is stored in a separate table, all named PerformanceAnalysisDataRollupXX, where XX represents the ID of the break level. In general, the only rollup table that may get large is the table for the two minute break level, or PerformanceAnalysisDataRollup2. The retention hours for this, or any other break level, can be adjusted as needed.

- Retention for raw Top SQL, Blocking and Deadlock data is controlled by the Purge History Older Than setting on the Performance Monitor tab under SentryOne Server->Settings in the Navigator pane. The default is 15 days.
- If you are using EM with PA, which enables viewing PA data on the EM calendar, the raw Top SQL, Blocking, and Deadlock data is also converted to the native EM storage format and stored in the EventSourceHistory table alongside data for other EM event sources like SQL Agent Jobs and DTS. Retention for all EM sources is controlled by the Purge History Older Than setting on the Event History Monitor tab under SentryOne Server->Settings in the Navigator pane.

Expired performance data is pruned by the SentryOne Monitoring Service every minute or so. The default settings are such that you should always have detailed performance data for the last two or three days. However, if you find that you are frequently navigating to date ranges using the Dashboard or Disk Activity tabs where no data is shown, it may mean that you need to increase the retention hours for the detailed and/or rolled up performance data. You should of course balance any changes with the resulting impact it will have on database size.

When you start using PA, you will likely find that your **SentryOne** database grows quickly at first. After a few days this will level off though, once the pruning of expired data begins and starts keeping pace with the incoming new data. You can get a quick idea of the mix of PA data in your environment by inspecting sizes for the related tables using the script below. Bear in mind that much of the data in EventSourceHistory is likely related to EM sources.

Performance Analysis Data Script

```
SELECT
TableName = OBJECT_SCHEMA_NAME([object_id]) + '.' +
OBJECT_NAME([object_id]),
[RowCount] = SUM(CASE WHEN index_id IN (0,1) THEN row_count ELSE 0 END),
UsedSpaceMB = SUM(used_page_count / 128),
ReservedSpaceMB = SUM(reserved_page_count / 128)
FROM sys.dm_db_partition_stats
WHERE OBJECT_NAME([object_id]) IN
(
'BlockChainDetail',
'EventSourceHistory',
'MetaHistorySqlServerBlockLog',
'MetaHistorySqlServerTraceLog',
'PerformanceAnalysisData',
'PerformanceAnalysisDataDatabaseCounter',
'PerformanceAnalysisDataDiskCounter',
```

```
'PerformanceAnalysisDataRollup11',
'PerformanceAnalysisDataRollup2',
'PerformanceAnalysisDataRollup4',
'PerformanceAnalysisDataRollup6',
'PerformanceAnalysisDataRollup8',
'PerformanceAnalysisTraceData',
'PerformanceAnalysisPlan',
'PerformanceAnalysisPlanOpTotals',
'PerformanceAnalysisTraceCachedPlanItems',
'PerformanceAnalysisTraceDataToCachedPlans',
'PerformanceAnalysisTraceQueryStats',
'MetaHistorySharePointTimerJob',
'PerformanceAnalysisSsasUsageTotals',
'PerformanceAnalysisSsasCubeDimensionAttribute',
'PerformanceAnalysisSsasTraceDataDetail'
)
AND OBJECTPROPERTY([object_id], 'IsUserTable') = 1
GROUP BY [object_id]
ORDER BY TableName;
```

## 10.2 SMTP Settings

Select the **SentryOne Server -> Settings** node in the SentryOne Client Navigator pane. The **SMTP Config** tab will be displayed by default.

1. In the **SMTP Server** field enter the domain name or IP address of the SMTP server to be used for routing SentryOne email notifications. If using *localhost*, keep in mind this will be the local SMTP server on the machine where the Monitoring Service is installed since it is responsible for sending all notifications. The SentryOne Client does not send any notifications.

   > ***NOTE: You may need to contact your network administrator first to ensure that the IP address of the Monitoring Service computer has been granted both Connect and Relay permissions for the specified SMTP server.***

2. Next, enter the **Email From Address**. This is the address which will appear on the From line of all email notifications sent by SentryOne.
3. You can also specify a **Username** and **Password** if authentication is required by your SMTP server. Please note that this is usually not required in most environments.
4. Click the **Test** button and specify an email address, then click **Send**.

   > ***IMPORTANT: For the most accurate SMTP test, you should use the SentryOne Client installed on the Server computer to send the test message. If you use a SentryOne Client on a different computer, such as your local workstation, the results may be different. For example, your SMTP server may allow relay from your workstation but not from the SentryOne Server computer, in which case the test from your workstation would succeed; but the SentryOne Server would be unable to deliver notifications.***

5. Click **Save** on the toolbar when finished.

## 10.3 Uninstalling SentryOne

SentryOne can be uninstalled through the Control Panel in Windows. When you uninstall the SentryOne Client or Monitoring Service, the associated program files will be removed. User preferences stored in the registry, and the SentryOne Database will not be deleted.

Additionally, the .NET Framework files will not be removed when uninstalling SentryOne. This can be accomplished using Add/Remove Programs in the Control Panel.

### REMOVING WATCHED SERVER OBJECTS

If you have stopped watching a SQL Server instance or Azure SQL Database target with SentryOne, and have no plans to watch it again in the near future, scripts are provided to automate the process of removing the objects SentryOne places on a watched target. Click the appropriate link below to view the script.

SQL Server 2000 instances

SQL Server 2005 and above instances

Azure SQL Database

## 10.3.1 Object Removal Script for Watched 2000 Servers

Object Removal Script for Watched 2000 Servers

```
USE msdb
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[sp_sentry_mail]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[sp_sentry_mail_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryEmails_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryEmails_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetBlockInfo_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetBlockInfo_Pre8sp3]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_Pre8sp3]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
```

```
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Queue Monitor')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Queue Monitor'' since there
is already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Queue Monitor'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetJobInfo_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetJobInfo_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetDTSLog_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetDTSLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueHeartbeat_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueHeartbeat_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueJob_Start_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_Start_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueJob_End_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_End_20]
```

```
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueMonitor_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueMonitor_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spReadLogFile_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spReadLogFile_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryQueueLog_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryQueueLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogCache_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCache_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogCacheDTS_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogCacheDTS_20]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Alert Trap')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Alert Trap'' since there is
already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Alert Trap'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
```

```
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spTrapAlert_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spTrapAlert_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spSetupAlertsTrap_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spSetupAlertsTrap_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryAlertLog_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryAlertLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogData_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogData_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryObjectVersion_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[fnGetSQL_20]') and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetSQL_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[fnGetWaittypeDesc_20]') and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetWaittypeDesc_20]
```

```
USE msdb
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[sp_sentry_mail]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[sp_sentry_mail_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryEmails_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryEmails_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetBlockInfo_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetBlockInfo_Pre8sp3]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_Pre8sp3]
```

```
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Queue Monitor')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Queue Monitor'' since there
is already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Queue Monitor'
SELECT @JobID = NULL
END

COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetJobInfo_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetJobInfo_20]
```

```
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spGetDTSLog_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spGetDTSLog_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueHeartbeat_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spQueueHeartbeat_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueJob_Start_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spQueueJob_Start_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueJob_End_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spQueueJob_End_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spQueueMonitor_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spQueueMonitor_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spReadLogFile_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

drop procedure [dbo].[spReadLogFile_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryQueueLog_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)

drop table [dbo].[SQLSentryQueueLog_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogCache_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)

drop table [dbo].[SQLSentryLogCache_20]

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogCacheDTS_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)

drop table [dbo].[SQLSentryLogCacheDTS_20]

GO

BEGIN TRANSACTION

DECLARE @JobID BINARY(16)

DECLARE @ReturnCode INT

SELECT @ReturnCode = 0

-- Delete the job with the same name (if it exists)

SELECT @JobID = job_id

FROM msdb.dbo.sysjobs

WHERE (name = N'SQL Sentry 2.0 Alert Trap')

IF (@JobID IS NOT NULL)

BEGIN
```

```
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Alert Trap'' since there is
already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Alert Trap'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spTrapAlert_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spTrapAlert_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[spSetupAlertsTrap_20]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)
drop procedure [dbo].[spSetupAlertsTrap_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryAlertLog_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryAlertLog_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryLogData_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogData_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryObjectVersion_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[fnGetSQL_20]') and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
```

```
drop function [dbo].[fnGetSQL_20]
if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].
[fnGetWaittypeDesc_20]') and OBJECTPROPERTY(id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetWaittypeDesc_20]
```

## 10.3.2 Object Removal Script for Watched 2005 and Above Servers

Object Removal Script for Watched 2005 and above

```
USE msdb
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[sp_sentry_mail]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[sp_sentry_mail_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_mail_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[sp_sentry_dbmail_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[sp_sentry_dbmail_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryEmails_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryEmails_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryDBEmails_Attachments_20]') and OBJECTPROPERTY(object_id,
N'IsUserTable') = 1)
drop table [dbo].[SQLSentryDBEmails_Attachments_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryDBEmails_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') =
1)
drop table [dbo].[SQLSentryDBEmails_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spGetBlockInfo_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetBlockInfo_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spGetBlockInfo_Pre8sp3]') and OBJECTPROPERTY(object_id, N'IsProcedure') =
1)
drop procedure [dbo].[spGetBlockInfo_Pre8sp3]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spGetQueryStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetQueryStatsData]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
```

```
[spGetProcedureStatsData]') and OBJECTPROPERTY(object_id, N'IsProcedure')
= 1)
drop procedure [dbo].[spGetProcedureStatsData]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Queue Monitor')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Queue Monitor'' since there
is already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Queue Monitor'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spGetJobInfo_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetJobInfo_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spGetDTSLog_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spGetDTSLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spQueueHeartbeat_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueHeartbeat_20]
```

```
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spQueueJob_Start_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_Start_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spQueueJob_End_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueJob_End_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spQueueMonitor_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spQueueMonitor_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spReadLogFile_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spReadLogFile_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryQueueLog_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') =
1)
drop table [dbo].[SQLSentryQueueLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryLogCache_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') =
1)
drop table [dbo].[SQLSentryLogCache_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryLogCacheDTS_20]') and OBJECTPROPERTY(object_id, N'IsUserTable')
= 1)
drop table [dbo].[SQLSentryLogCacheDTS_20]
GO
BEGIN TRANSACTION
DECLARE @JobID BINARY(16)
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
-- Delete the job with the same name (if it exists)
SELECT @JobID = job_id
FROM msdb.dbo.sysjobs
WHERE (name = N'SQL Sentry 2.0 Alert Trap')
IF (@JobID IS NOT NULL)
BEGIN
-- Check if the job is a multi-server job
IF (EXISTS (SELECT *
FROM msdb.dbo.sysjobservers
WHERE (job_id = @JobID) AND (server_id <> 0)))
BEGIN
-- There is, so abort the script
RAISERROR (N'Unable to import job ''SQL Sentry Alert Trap'' since there is
already a multi-server job with this name.', 16, 1)
GOTO QuitWithRollback
```

```
END
ELSE
-- Delete the [local] job
EXECUTE msdb.dbo.sp_delete_job @job_name = N'SQL Sentry 2.0 Alert Trap'
SELECT @JobID = NULL
END
COMMIT TRANSACTION
GOTO EndSave
QuitWithRollback:
IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
GO
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spTrapAlert_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [dbo].[spTrapAlert_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[spSetupAlertsTrap_20]') and OBJECTPROPERTY(object_id, N'IsProcedure') =
1)
drop procedure [dbo].[spSetupAlertsTrap_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryAlertLog_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') =
1)
drop table [dbo].[SQLSentryAlertLog_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryLogData_20]') and OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [dbo].[SQLSentryLogData_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[SQLSentryObjectVersion_20]') and OBJECTPROPERTY(object_id,
N'IsUserTable') = 1)
drop table [dbo].[SQLSentryObjectVersion_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[fnGetSQL_20]') and OBJECTPROPERTY(object_id, N'IsScalarFunction') = 1)
drop function [dbo].[fnGetSQL_20]
if exists (select * from sys.objects where object_id = object_id(N'[dbo].
[fnGetWaittypeDesc_20]') and OBJECTPROPERTY(object_id,
N'IsScalarFunction') = 1)
drop function [dbo].[fnGetWaittypeDesc_20]
```

### 10.3.3 Object Removal Script for Watched Azure SQL DB instances

Object Removal Script for Watched Azure SQL DB instances

```
if exists (select * from sys.objects where object_id =
object_id(N'[SQLSentry].[spGetProcedureStatsData]') and
OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [SQLSentry].[spGetProcedureStatsData]
if exists (select * from sys.objects where object_id =
object_id(N'[SQLSentry].[spGetQueryStatsData]') and
OBJECTPROPERTY(object_id, N'IsProcedure') = 1)
drop procedure [SQLSentry].[spGetQueryStatsData]
if exists (select * from sys.objects where object_id =
object_id(N'[SQLSentry].[SQLSentryObjectVersion_20]') and
OBJECTPROPERTY(object_id, N'IsUserTable') = 1)
drop table [SQLSentry].[SQLSentryObjectVersion_20]
DECLARE @listOfSqlSentryTables VARCHAR(MAX)
DECLARE @sqlStatement VARCHAR(MAX)
select @listOfSqlSentryTables = COALESCE(@listOfSqlSentryTables+',' ,'') +
'SQLSentry.' +name from sys.objects where name like N'ProcedureStats%' and
OBJECTPROPERTY(object_id, N'IsUserTable') = 1 and schema_id =
schema_id('SQLSentry')
set @sqlStatement = 'drop table ' + @listOfSqlSentryTables
exec(@sqlStatement)
SET @listOfSqlSentryTables = NULL;
select @listOfSqlSentryTables = COALESCE(@listOfSqlSentryTables+',' ,'') +
'SQLSentry.' +name from sys.objects where name like N'QueryStats%' and
OBJECTPROPERTY(object_id, N'IsUserTable') = 1 and schema_id =
schema_id('SQLSentry')
set @sqlStatement = 'drop table ' + @listOfSqlSentryTables
exec(@sqlStatement)
if schema_id('SQLSentry') is not null
drop schema SQLSentry
```

## 10.4 Watched Target Objects

### PERFORMANCE ANALYSIS WATCHED TARGET OBJECTS

When a SQL Server instance is watched by **SentryOne Performance Analysis**, the below objects are placed on the target. To remove these objects, see the Uninstalling SentryOne topic.

#### TABLES (MSDB):

SQLSentryObjectVersion_20

SQLSentryAlertLog_20

SQLSentryDBEmails_20 (SQL Server 2005+)

SQLSentryDBEmail_Attachments_20 (SQL Server 2005+)

SQLSentryEmails_20

SQLSentryLogCache_20

SQLSentryLogData_20

SQLSentryObjectVersion_20

SQLSentryQueueLog_20

### STORED PROCEDURES (MSDB):

spGetBlockInfo_20

spGetBlockInfo_20

spGetJobInfo_20

spQueueHeartbeat_20

spQueueJob_End_20

spQueueJob_Start_20

spQueueMonitor_20

spReadLogFile_20

spSetupAlertsTrap_20

spTrapAlert_20

sp_sentry_mail

sp_sentry_mail_20

sp_sentry_dbmail_20 (SQL Server 2005+)

### SQL AGENT JOBS:

SQL Sentry 2.0 Alert Trap

SQL Sentry 2.0 Queue Monitor

---

## SENTRYONE MONITORING OBJECTS IN AZURE SQL DATABASE TARGETS

When an Azure SQL Database is watched by SentryOne Performance Analysis the user is given the option to allow some monitoring objects to be created in the target database. If these objects are allowed they will be created under a SQLSentry schema. To remove these objects, see the Uninstalling SentryOne topic. Some of the tables will have unique identifiers on them after the table name (indicated below by '...').

### TABLES (TARGET DATABASE)

SQLSentry.SQLSentryObjectVersion_20

SQLSentry.ProcedureStats_...

SQLSentry.ProcedureStats_...

**STORED PROCEDURES (TARGET DATABASE)**

SQLSentry.spGetProcedureStatsData

SQLSentry.spGetQueryStatsData

## 10.5 Standard Vs Enterprise Editions

Please see this link for a Standard Vs. Enterprise feature comparison.

# 11 The SentryOne User Guide

For advanced information on configuring and using SentryOne, please refer to the **SentryOne User Guide**. The User Guide is always accessible from the Help menu of the SentryOne Client.

# 12 Contact Information

## CUSTOMER PORTAL

Access to the SentryOne Customer Portal is available around the clock, allowing you to retrieve a backup license key, expand your enterprise by adding more licenses or even modify an existing license key in the case of hardware changes. In addition, the Customer Portal is where product updates and documentation can be found. The page http://www.sentryone.com/portal can be used to activate and log into your account.

## SUPPORT

If you have any technical questions, or for help with installation or configuration issues, please don't hesitate to contact us:

**Email**: support@sentryone.com

**Phone**: 704-895-6241

**Toll Free**: 855-775-7733

## SUPPORT FORUM

SentryOne support forum: http://support.sentryone.com

## DEMOS

To sign up for one of our regularly scheduled public webinar demos please visit:

http://www.sentryone.com/company/news-events#webinars

## SALES

If you have any pre-sales questions, or would like to place an order, please contact our sales team directly:

**Email**: sales@sentryone.com

**Phone**: 704-895-6241

## FEEDBACK

We always welcome your feedback on this guide and SentryOne in general. Please email any feedback, ideas, or feature requests to support@sentryone.com

# 13 Index