Quest®

Quest® InTrust 11.3

# Auditing Custom Logs with InTrust

If you have any questions regarding your potential use of this material, contact:

Quest Software Inc.

Attn: LEGAL Dept

4 Polaris Way

Aliso Viejo, CA 92656

Refer to our Web site (https://www.quest.com) for regional and international office information.

### Patents

Quest Software is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at https://www.quest.com/legal.

### Trademarks

Quest, the Quest logo, and Join the Innovation are trademarks and registered trademarks of Quest Software Inc. For a complete list of Quest marks, visit https://www.quest.com/legal/trademark-information.aspx. All other trademarks and registered trademarks are property of their respective owners.

### Legend

**CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.**

**IMPORTANT**, **NOTE**, **TIP**, **MOBILE**, or **VIDEO**: An information icon indicates supporting information.

InTrust Auditing Custom Logs with InTrust
Updated - May 2017
Version - 11.3

# Contents

# Using Custom Data Sources

InTrust provides two general-purpose data source types for situations in which you need to gather event logs for which no predefined data source exists. When creating such a data source, you specify how to process the log, what information is stored and how it is ordered.

InTrust supports the following two types of user-defined event logs:

- Text logs: information is stored in ASCII text files. Configure collection of such logs using the Custom Text Log Events data source type.
- Database event logs: information is stored in database tables. Configure collection of such logs using the Database Events data source type.

# Creating a Data Source for a Custom Log

***To create a new database log data source***

1. In InTrust Manager, expand the **Configuration** node.
2. Right-click the Data Sources container and select **New Data Source**.
3. Select the **Database Events** type and proceed with the wizard.

For details about custom database log settings, see the Database Events Data Sources topic.

***To create a new text log data source***

1. In InTrust Manager, expand the **Configuration** node.
2. Right-click the Data Sources container and **select New Data Source**.
3. Select the **Custom Text Log Events** type and proceed with the wizard.

For details about custom database log settings, see the Custom Text Log Data Sources topic.

# Editing a Data Source for a Custom Log

To edit an existing database or text log data source, right-click the data source you need in the right pane and select **Properties**.

# Custom Text Log Data Sources

User-defined text log data sources can be configured in any of three modes:

1. **Basic**
   This is the simplest way to set up gathering from a custom text log. It provides a minimum of configuration options and may not be suitable for some log formats.
2. **Advanced**
   This mode offers much more control than Basic mode and provides sophisticated options. In Advanced mode, most text log formats can be defined.
3. **Raw**
   This mode requires that you edit a script in JScript or VBScript that processes log data and prepares it for storage. InTrust acts as the framework for script execution and data gathering. You cannot create a text log data source in Raw mode, but you can convert a Basic or Advanced data source to a Raw data source.

Processing of log information is powered by regular expressions. In Basic mode, you are not exposed to regular expressions (however, you can use them when specifying the path to the log file). In Advanced mode, you specify them as needed to configure the handling of log data. In Raw mode, you use regular expressions in scripts of your own.

Whichever mode you select, the end result is a script that InTrust runs. You can edit the resulting scripts to meet your specific purposes. For example, you can complete the Basic mode wizard to rough out a data source and later edit the script in Raw mode.

You can do the editing directly in a text editor provided by the wizard. If you run the wizard to edit a Basic data source, you can select to convert it to an Advanced or Raw data source. Note that you cannot convert an Advanced or Raw data source to a Basic one, nor can you convert a Raw data source to an Advanced one.

However, in some cases using an Advanced data source in the first place is preferable. This is true when the structure of the resulting regular expressions in the Basic data source is completely different from the type of expression you need eventually.

In general, recommendations for the choice of mode are as follows:

| Basic | Advanced | Raw |
|---|---|---|
| The log has a number of articulated fields. These fields can be distinguished based on delimiters between them or the fixed width of each field. This kind of log can be represented by a table without rearranging or modifying data. | One or both of the following are true:<br>There are mixed-format entries in the log, so the log does not fit in a simple table without rearranging fields.<br>The log includes comments and other data that could break the simple row-and-column-style representation. | You feel more comfortable with a script editor user interface than with the wizard's Advanced mode settings. |

# Basic Text Log Data Source

In Basic mode, specify the following:

- Path to the text log file and name of the log file. You can use wildcards or regular expressions.

> **NOTE:** If you use regular expressions, you can click the button next to the path and file name text boxes to insert special regular expression characters.
>
> Syntax is not checked, so be careful to follow rules while composing regular expressions. Thus, if you need an expression for "one or two digits" and you use the predefined special character **[:digit:]**, an example of correct usage is **C:\Logs\log[[:digit:]]{1;2}** not **C:\Logs\log[:digit:]{1;2}**.
>
> In situations like this, remember to use the provided button to insert brackets or type them manually.
>
> If you want to gather without agents, specify the path using the %COMPUTER_NAME% variable and a share name. You can supply the name of a special Windows share (such as **\\%COMPUTER_NAME%\C$\Logs**) or a regular native Windows or SMB share (such as **\\%COMPUTER_NAME%\logs**). However, if you want to gather text logs from an SMB share on a Unix host without an agent, make sure that this host is a member of an InTrust site in the Microsoft Windows Environment container. The gathering job must be configured for this site, so you also need to use a Windows-specific gathering policy. InTrust currently supports gathering from network shares only in Microsoft Windows Environment sites; this workaround makes InTrust aware of the share even though the processed computer is not actually running Windows.

- Log file encoding.
- Whether the log uses delimiter characters or fixed width for formatting.
- Which row to start parsing at.
- Path to a sample log file, which you can use to test settings.
- Additional format-specific settings.
- Format for date and time information.

> **CAUTION: If you select to treat event times as local time, and you collect events for a period during which time was set back due to daylight saving time adjustments, the event times will be incorrect after the adjustment for the duration of the difference.**

- Mapping of original log fields to InTrust-stored fields.

# Advanced Text Log Data Source

In Advanced mode, specify the following:

- Path to the text log file and name of the log file. You can use wildcards or regular expressions.

**NOTE:** If you use regular expressions, you can click the button next to the path and file name text boxes to insert special regular expression characters.

Syntax is not checked, so be careful to follow rules while composing regular expressions. Thus, if you need an expression for "one or two digits" and you use the predefined special character **[:digit:]**, an example of correct usage is **C:\Logs\log[[:digit:]]{1;2}** not **C:\Logs\log[:digit:]{1;2}**.

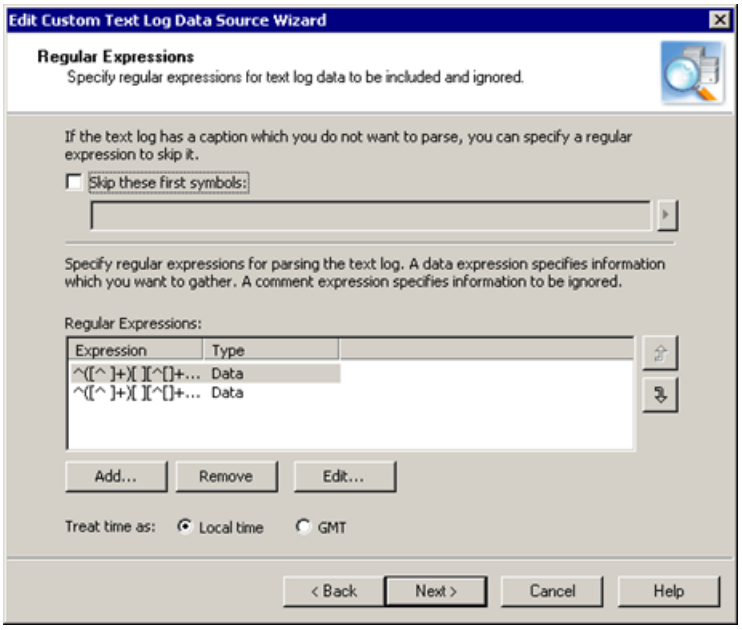
In situations like this, remember to use the provided button to insert brackets or type them manually.

If you want to gather without agents, specify the path using the %COMPUTER_NAME% variable and a share name. You can supply the name of a special Windows share (such as **\\%COMPUTER_NAME%\C$\Logs**) or a regular Windows or SMB share (such as **\\%COMPUTER_NAME%\logs**). However, if you want to gather text logs from an SMB share on a Unix host without an agent, make sure that this host is a member of an InTrust site in the Microsoft Windows Environment container. The gathering job must be configured for this site, so you also need to use a Windows-specific gathering policy. InTrust currently supports gathering from network shares only in Microsoft Windows Environment sites; this workaround makes InTrust aware of the share even though the processed computer is not actually running Windows.

- Regular expressions that match portions of data from the log.
- Log file encoding.

If the log is a mixture of two or more log entry conventions, you must provide two or more regular expressions. Each expression must match only one type of string format (usually a single line) in the log. Be careful not to supply expressions that can match strings with different formats.

If you cannot completely eliminate the possibility of several formats matching one expression, use an appropriate evaluation order for expressions. If an expression matches a string, that string is not parsed again. You set the order using the arrow buttons on the Regular Expressions step of the wizard.




For the entire data source, you can select whether to skip log captions, if any.

For each string format you want to match with a regular expression, specify the following:

- The expression itself.
- Whether to treat the matched string as useful data or as comments

If the string contains data, it is processed for storing in the data store. If it contains comments, it is parsed, but the result does not go into the data store.

**NOTE:** Comments are important for identifying the portions of the log that have no meaning for auditing. Therefore, data and comment regular expressions work like including and excluding filters, respectively. A good set of regular expressions defines both data patterns and comment patterns for a complete definition of the log structure.

- Date and time formatting conventions (only for data regular expressions). For more information about date and time data mapping, see the Understanding Fields topic.

**CAUTION: If you select to treat event times as local time, and you collect events for a period during which time was set back due to daylight saving time adjustments, the event times will be incorrect after the adjustment for the duration of the difference.**

- Field mapping (only for data regular expressions). For more information about log data mapping, see the Understanding Fields topic.

# Raw Text Log Data Source

In Raw mode, edit a predefined script using a code editor.

# Understanding Fields

In both Basic and Advanced mode, the wizard provides controls for translating log data into the storage format that InTrust uses. Regular expressions break up log data into strings, which are mapped to particular event fields that InTrust stores.

## Date and Time Fields

Date and time are handled differently from other fields due to their specifics.

Both data source creation modes provide two controls for date and time formatting configuration: the **Log fields** text box and the **Date/Time format** combo box.

In the **Log fields** text box, specify numbers of fields that contain date and time information. Enclose the numbers in angle brackets and use appropriate separators.

These log fields result from the matches that the regular expression returns. A field is made up of data matched by a regular expression fragment enclosed in a pair of parentheses.

Supply field numbers so that they result in properly ordered information after InTrust parses the log. In the **Date/Time format** text box, supply special formatting characters (such as H, Y, m, s, d) and compose appropriate strings with them. These strings tell InTrust how to interpret the text so that the result is date and time information. For more information about using the special characters, see Pattern Letters for Date and Time Designation.

For example, suppose the log contains date and time information in the first five fields. The first, second, third and fourth field each contains fragments of the date, and the fifth contains the time. If you specify “<5> <1> <2> <3> <4>” in the **Log fields** text box, a correct entry for the **Date/Time format** text box might be “HH:mm:ss EEE MMM dd yyyy”. When this is used to process an actual event, InTrust stores something like “15:13:30 Mon Apr 22 2005” as a result.

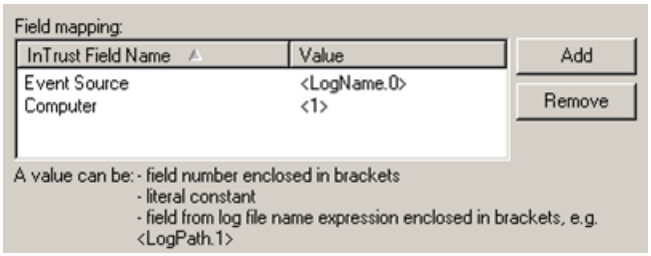
If date and time information in the log is incomplete (for example, it does not specify the year) and the log file name does not complement it, InTrust's gathering engine may order data from the log incorrectly. To avoid such problems, specify missing information explicitly in the **Log fields** text box and supply respective pattern letters in the **Date/Time format** text box below.

For example, if you know that events in the log occurred on April 3 2003, you can configure the date and time fields as shown in the following screenshot:


> **CAUTION: If you select to treat event times as local time, and you collect events for a period during which time was set back due to daylight saving time adjustments, the event times will be incorrect after the adjustment for the duration of the difference.**

# Other Fields

For mapping all other log data, use the controls shown in the following screenshot:




Configure the correspondence between the fields InTrust stores and strings matched by the regular expression. Specify the name of the InTrust field and a corresponding value for the field.

For InTrust fields, you can use only predefined names. However, you can supply as many insertion strings as you like, such as "Insertion String #1" or "Insertion String #15". Edit the insertion string numbers manually.

For values, you can specify field numbers (as parsed by the regular expression), log file name expressions and literal constants.

Field numbers are enclosed in brackets. Log file name expressions put the name of the current log in InTrust fields. Literal constants are useful when you want the same string to always appear in particular InTrust fields.

The following rules apply to log file name expressions:

- Use regular expressions rather than wildcards when specifying the log file name and path. By default, wildcard notation is used.
- Use parentheses to signify parts of file name and file path strings that are matched. For example, from the file name expression **(r2).log.(3)** two strings are returned: r2 and 3. For actual parentheses, use **\(** and **\)**.
- **<LogName.0>** returns the full name of the log file.
- If a number other than 0 is specified with LogName, that number stands for the number of the string match in the log file name expression you specified. So, in the previous example, **<LogName.2>** returns 3.
- **<LogPath.0>** returns the full path to the log file.

- If a number other than 0 is specified with LogPath, that number stands for the number of the string match in the log path expression you specified.

# Handling Log Rotation

You may want to gather logs for which rotation is configured. This section explains how to specify the names of such logs for successful gathering.

The typical configuration for log rotation is as follows:

- There is not one log file but an array of log files. The log file names are numbered so that the oldest events are in the log with the highest index. The current log file, which is constantly updated, usually has no index. Other log files are complete, and their contents remain unchanged.
- On schedule or as soon as the current log is full, the oldest log file with the highest index is deleted.
- The other log files are renamed so that their numbering increases by one. For example, **eventlog.2** becomes **eventlog.3**. If the current file had no index, the index is appended. Thus, for example, the current file **eventlog** is renamed **eventlog.1**.
- A new log file is created and becomes the current log file. When the **eventlog** file in the previous example is renamed, a new **eventlog** is created in its place.

In such situations, you do not want to gather the current log. On the Path to Custom Text Log step of the wizard, explicitly specify one of the backup logs that does not get updated. For example, if rotation is set up for log files **log.1** to **log.5**, supply **log.2** as the log file name.

This is required so that gathered events are not duplicated and no events are missed due to the rotation cycle.

If rotation takes place on schedule, set gathering to occur as frequently as the rotation sessions or more frequently.

If rotation happens whenever the current log is at capacity, configure the gathering to take place as often as possible. This way, you will not skip a file due to a sudden increase in the number of events. Performance will not suffer if you gather often, because InTrust does not collect the same events from the same log a second time. In such a configuration, a log file is collected only once in its entirety. All other InTrust gathering sessions simply check whether the file has been replaced.

# Gathering with Custom Text Log Data Sources

***To use a text log data source in a new gathering policy***

1. Make sure the data source you need exists and is configured properly.
2. Start creating a new gathering policy.
3. At the Data Sources step of the New Policy Wizard, click **Add**. The Add Data Source Wizard opens.
4. Select the custom text log data source.
5. Finish the wizard.

# Examples of Custom Text Log Data Source



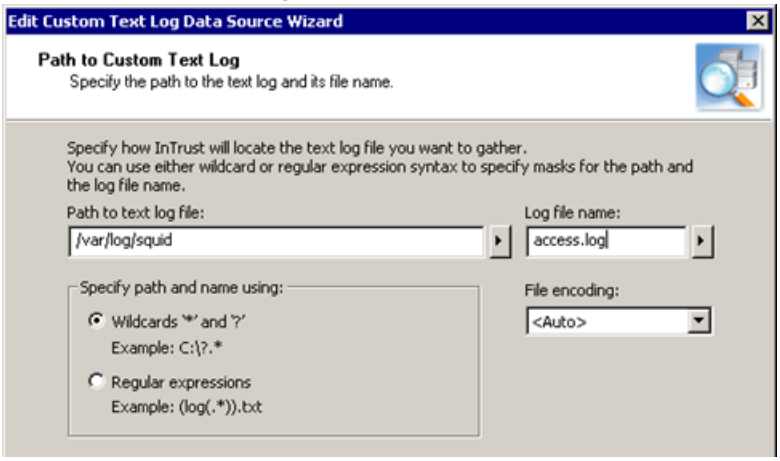
# Basic Text Log Data Source

In this example, you will create a working data source that handles gathering of the Squid proxy server's access log. The Squid access log can have three formats. This example uses the access log native format which is the default format.

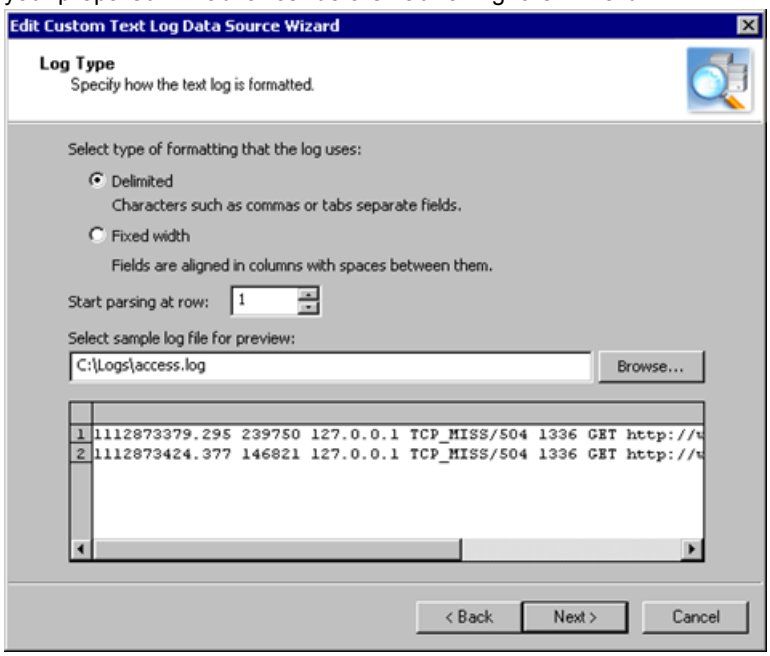
Before running the New Data Source Wizard, make sure there is sample log file in a location that is accessible from the wizard. The directory **/var/log/squid/** where the file **access.log** resides cannot be accessed directly from InTrust Manager. Copy the file to a local folder on the computer where InTrust is running, for example **C:\Logs**.

### *To create a data source for the Squid access log*

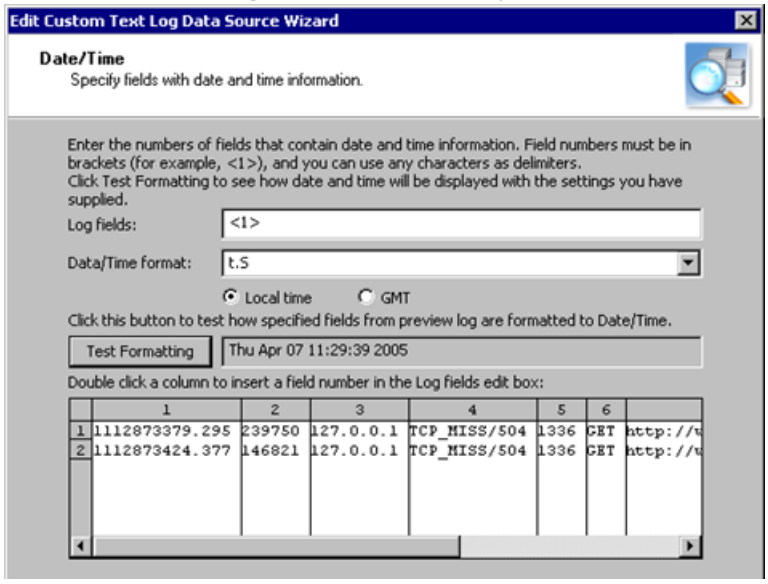
1. Run the New Data Source Wizard and start creating a custom text log data source in Basic mode.
2. Specify the path to the log directory (in this case, **/var/log/squid**) and the file name of the log (in this case, **access.log**).



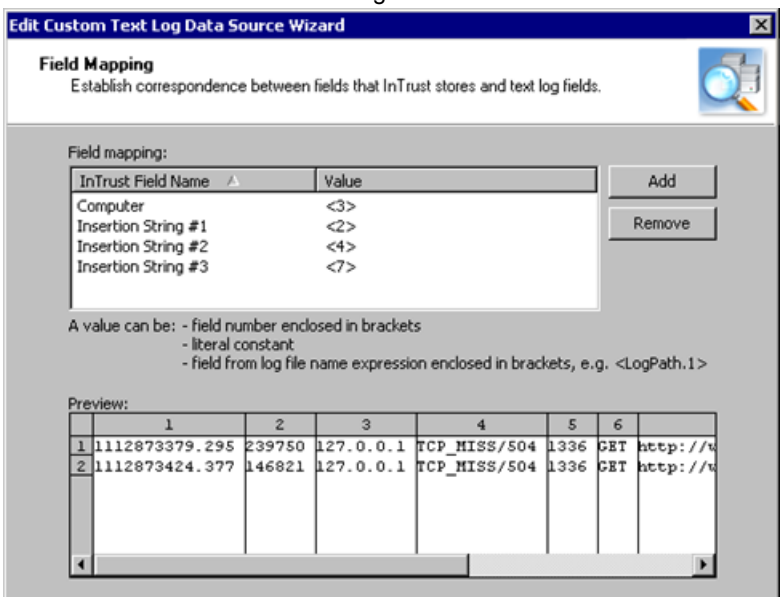
3. Select **Delimited** as the formatting type. Specify the location of the sample log file that you will use to verify your settings. The actual log file is not available for preview, so use the file you prepared in advance before launching the wizard.




4. Specify which log fields contain data related to the date and time. Click **Test Formatting** to make sure that you have supplied proper settings.



5. Configure the correspondence between the names of fields that InTrust recognizes and the numbers of fields in the text log. The screenshot below shows a sample configuration.




6. Provide a name and, optionally, a description for the data source, and finish the wizard.

# Advanced Text Log Data Source

This example shows how to create a data source for the Apache access log. The example assumes that the following settings are specified in the Apache configuration file **httpd.conf**:

- LogFormat "%h %l %u %t \"%r\" %>s %b" common
- CustomLog logs/access_log common

### *To create a data source for the Apache access log*

1. Run the New Data Source Wizard and start creating a custom text log data source in Advanced mode. Specify a valid path to the log directory (for example, /etc/httpd/logs) and the file name of the log (in this case, access_log).
2. On the Regular Expressions step, specify the following expression:
   ```
   ^([^ ]+)[ ][^[]+\[([^ ]+)[ ][-+\d]+\][ ]"([^"]+)"[ ](\d+)[ ]([-\d]+)[\r\n]+
   ```
3. Complete the wizard.

The expression you specified matches lines similar to the following:

- 192.168.10.1 – jane [03/Apr/2005:13:21:46 -0400] "GET /cgi-bin/eshop.pl?seite=;cat%20/etc/passwd| HTTP/1.0" 404 294
- 192.168.10.5 – – [03/Apr/2005:13:21:47 -0400] "HEAD / HTTP/1.0" 200 0
- somehost.somedomain.org – – [03/Apr/2005:15:17:51 -0400] "GET /somepage.html HTTP/1.1" 404 304

The expression matches five fragments in each line, and these fragments are mapped to fields as follows:

| Match | Field | Description |
|---|---|---|
| 1 | Computer | Client IP address or hostname |
| 2 | Date/Time | The time when the server finished processing the request |
| 3 | Insertion String 1 | Request line from the client |
| 4 | Insertion String 2 | Status code that the server sends back to the client |
| 5 | Insertion String 3 | The size of the object returned to the client |

***To see an example of field mapping configuration***

1. Open the properties of a predefined text log data source, such as CheckPoint or Cisco PIX.
2. On the Settings tab, click **Edit** to start the Edit Custom text Log Data Source Wizard
3. On the Regular Expressions step, select a regular expression and click **Edit**.

# Database Events Data Sources

If you use software that stores its audit trails in databases, you can set up InTrust to gather such logs and store them. This can be required by regulations, or you may want to create backup storage for those logs using a different platform.

### *To create a custom database events data source*

1. In InTrust Manager, expand the **Configuration** node.
2. Right-click the **Data Sources** node and select **New Data Source**.
3. Select the **Database Events** type and follow the steps of the wizard.

In the wizard, configure the following settings:

- Source of data
  You can manually specify an ODBC connection string, or you can click **Create**, select the ODBC driver from a list and let the wizard generate the connection string for you. The connection string can be created automatically if the ODBC driver is installed on the same computer as InTrust Manager. If you decide to write the query, you can use the **Keyword** button to insert predefined keywords.
    - Include the **%PASSWORD%** keyword in connection strings that you compose. This keyword is there for security reasons and stands for the password to be used for connection. Supply the password in the text box on the same step of the wizard.
    - You can also use the keyword **%COMPUTER_NAME%**. This keyword is resolved as the name of the database server from which data is gathered. Use it if you need a uniform query for several database servers and use the same credentials to access them. There are as many values to **%COMPUTER_NAME%** as there are computers in the site you are gathering from.

  If you use agents for gathering, the keyword is resolved on the agent side, and gathering is performed from all of the site's computers simultaneously, which gives better performance. If you do not use agents, the keyword is resolved by the InTrust server, and each computer in the site is processed in turn.
- SQL query
  Write the SQL query that retrieves necessary data from the database.
- Log name
  Specify the name that InTrust must give to the log with gathered events.
- Database field mapping
  Configure the matching between the original database fields and those that InTrust stores. This governs how the retrieved data is arranged for storage.

- Cleanup query
  Supply an SQL query to be executed after gathering. This query should clear gathered events from the database.
  The query is not run by default. To make it run, enable the **Clear log after gathering** option for policy that uses the data source.
- Name and optional description of the data source.

To edit an existing database events data source, right-click the data source and select **Properties**.

# Marking Where to Start Gathering

The SQL query you specify must include the variable **%LAST_GATHERED_EVENT%**. This keyword defines where the gathering starts from. If you ignore this keyword, the entire contents of the database are gathered. Doing so multiple times severely impedes performance and, if you gather to a repository, results in duplicate data.

It is recommended that you use **%LAST_GATHERED_EVENT%** as part of a "where" statement, such as the following:

```
select Time, ID, TestString from TestDatabase where Time >= %LAST_GATHERED_EVENT%
order by Time
```

It makes sense to associate **%LAST_GATHERED_EVENT%** with event time. This helps avoid duplication of data even if different selection parameters are used in the query for different sessions.

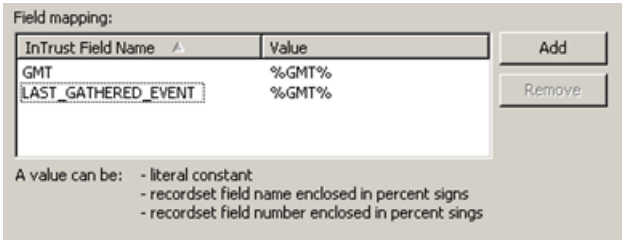
It is also a good idea to make the query order events by time. Otherwise, the value of **%LAST_GATHERED_EVENT%** may remain the same from session to session. If you order by time, which is unique for most events, the value of **%LAST_GATHERED_EVENT%** is updated after each gathering, so you do not have to specify it for subsequent gathering sessions.

When you first create the data source, specify a value for **%LAST_GATHERED_EVENT%**. If you use the keyword as recommended, supply the date and time from which you want gathering to start. If you want to gather everything, supply a time earlier than the earliest in the log.

Be careful to use a time formatting and conversion convention that is appropriate for your RDBMS. If you associate **%LAST_GATHERED_EVENT%** with time, it is best to use the 24-hour format rather than the 12-hour format to avoid confusion.

# Mapping Database Fields

On the Database Fields Mapping step, establish a correspondence between the fields in the original database and the InTrust representation of that database. The following controls are used:




On the left, specify the InTrust event fields. On the right, supply the database fields that match them.

InTrust field names are predefined, and you cannot specify custom names. However, you can supply as many insertion strings as you like, such as “Insertion String #1” or “Insertion String #99”. These are provided for fields that are unrelated to any of the existing InTrust fields.

The GMT and **LAST_GATHERED_EVENT** fields are mandatory. GMT should be mapped to event date and time. **LAST_GATHERED_EVENT** should be mapped to the field whose content InTrust will look at to determine where to start gathering. It is best to map **LAST_GATHERED_EVENT** to event date and time as well.

### *To use the data source in a new gathering policy*

1. Start creating a new gathering policy.
2. On the Data Sources step of the New Policy Wizard, click **Add** and select the data source you created earlier.
3. Optionally, as you proceed with the wizard, you can enable database cleanup and apply a filter to include or ignore specific data.
4. Finish the wizards.

# Example of Database Events Data Source

This example describes how you can transfer data from an InTrust audit database to an InTrust repository. InTrust does not provide a job type that does this. However, you may want to move data from a database to a repository in some situations.

The following information is not gathered in this example:

- Information about computers that the data originally came from
  This data will be replaced with information about the SQL Server computer and the database.
- The **RecordNumber** field
  This is an auxiliary field that has meaning only for Quest development and support.
- The **GatheringComputer** field
  InTrust always fills in this field automatically with the name of the computer that is running the current gathering session. Any original values would be lost anyway, so they are not gathered.
- The **EventLog** field
  Information for this field comes from the name of the log in the database events data source. In the example, only security events are gathered, so specify **Security** as the log name.

### *To transfer data from an InTrust audit database to an InTrust repository*

1. Start creating a new database events data source.
2. On the SQL Query step, supply the following query:

```
select

e.ID,e.SessionID,e.VersionMajor,e.VersionMinor,

e.Computer,e.UserName,e.UserDomain,e.EventType,e.Source,e.EventID,e.Category,e.GMT,e.L
ocalTime,
```

```
s.S1,s.S2,s.S3,s.S4,s.S5,s.S6,s.S7,s.S8,s.S9,s.S10,s.S11,s.S12,s.S13,s.S14,s.S15,s.S16
,s.S17,s.S18,s.S19,s.S20,s.S21,s.S22,s.S23,s.S24,s.S25,s.S26,s.S27,s.S28,s.S29,s.S30,s
.S31,s.S32,s.S33,s.S34,s.S35,s.S36,s.S37,s.S38,s.S39,s.S40,s.S41,s.S42,s.S43,s.S44,s.S
45,s.S46,s.S47,s.S48,s.S49,s.S50,

isnull(d.Description,'') Description

from

Events e

inner join

(

select

e.ID,

e.SessionID,

max(case s.StringIndex when 1 then s.StringValue else null end) S1,

max(case s.StringIndex when 2 then s.StringValue else null end) S2,

max(case s.StringIndex when 3 then s.StringValue else null end) S3,

max(case s.StringIndex when 4 then s.StringValue else null end) S4,

max(case s.StringIndex when 5 then s.StringValue else null end) S5,

max(case s.StringIndex when 6 then s.StringValue else null end) S6,

max(case s.StringIndex when 7 then s.StringValue else null end) S7,

max(case s.StringIndex when 8 then s.StringValue else null end) S8,

max(case s.StringIndex when 9 then s.StringValue else null end) S9,

max(case s.StringIndex when 10 then s.StringValue else null end) S10,

max(case s.StringIndex when 11 then s.StringValue else null end) S11,

max(case s.StringIndex when 12 then s.StringValue else null end) S12,

max(case s.StringIndex when 13 then s.StringValue else null end) S13,

max(case s.StringIndex when 14 then s.StringValue else null end) S14,

max(case s.StringIndex when 15 then s.StringValue else null end) S15,

max(case s.StringIndex when 16 then s.StringValue else null end) S16,

max(case s.StringIndex when 17 then s.StringValue else null end) S17,

max(case s.StringIndex when 18 then s.StringValue else null end) S18,

max(case s.StringIndex when 19 then s.StringValue else null end) S19,

max(case s.StringIndex when 20 then s.StringValue else null end) S20,

max(case s.StringIndex when 21 then s.StringValue else null end) S21,

max(case s.StringIndex when 22 then s.StringValue else null end) S22,

max(case s.StringIndex when 23 then s.StringValue else null end) S23,
```

```
max(case s.StringIndex when 24 then s.StringValue else null end) S24,
max(case s.StringIndex when 25 then s.StringValue else null end) S25,
max(case s.StringIndex when 26 then s.StringValue else null end) S26,
max(case s.StringIndex when 27 then s.StringValue else null end) S27,
max(case s.StringIndex when 28 then s.StringValue else null end) S28,
max(case s.StringIndex when 29 then s.StringValue else null end) S29,
max(case s.StringIndex when 30 then s.StringValue else null end) S30,
max(case s.StringIndex when 31 then s.StringValue else null end) S31,
max(case s.StringIndex when 32 then s.StringValue else null end) S32,
max(case s.StringIndex when 33 then s.StringValue else null end) S33,
max(case s.StringIndex when 34 then s.StringValue else null end) S34,
max(case s.StringIndex when 35 then s.StringValue else null end) S35,
max(case s.StringIndex when 36 then s.StringValue else null end) S36,
max(case s.StringIndex when 37 then s.StringValue else null end) S37,
max(case s.StringIndex when 38 then s.StringValue else null end) S38,
max(case s.StringIndex when 39 then s.StringValue else null end) S39,
max(case s.StringIndex when 40 then s.StringValue else null end) S40,
max(case s.StringIndex when 41 then s.StringValue else null end) S41,
max(case s.StringIndex when 42 then s.StringValue else null end) S42,
max(case s.StringIndex when 43 then s.StringValue else null end) S43,
max(case s.StringIndex when 44 then s.StringValue else null end) S44,
max(case s.StringIndex when 45 then s.StringValue else null end) S45,
max(case s.StringIndex when 46 then s.StringValue else null end) S46,
max(case s.StringIndex when 47 then s.StringValue else null end) S47,
max(case s.StringIndex when 48 then s.StringValue else null end) S48,
max(case s.StringIndex when 49 then s.StringValue else null end) S49,
max(case s.StringIndex when 50 then s.StringValue else null end) S50
from
Events e
left join EventsStrings s on s.SessionID=e.SessionID and s.EventID=e.ID and
s.StringIndex<=50
group by
e.ID,
e.SessionID
```

```
) s
on s.SessionID=e.SessionID and s.ID=e.ID
left join EventsDescriptions d on d.SessionID=e.SessionID and d.EventID=e.ID
WHERE EVENTLOG = 'Security' and GMT >= %LAST_GATHERED_EVENT% ORDER BY GMT
```

3. On the same step, specify a date as the value for the **%LAST_GATHERED_EVENT%** variable. It should be a date that you know precedes the earliest event's date. Use the following format: 2000-01-01 00:00:00.
4. Specify "Security" as the name of the log.
5. When configuring field mapping, map fields to their counterparts. For example, map **Computer** in the left column to **%Computer%** in the right. Map **LAST_GATHERED_EVENT** and **GMT** to **%GMT%**. If you want to map insertion strings, use syntax such as **%S1%** in the right column.
6. Leave the cleanup query blank. Only a subset of data available in the database is gathered, and there is a lot of other useful information in it.
7. Give a descriptive name to the new data source.
8. Specify a valid license.
9. Complete the wizard and commit the changes you have made.

# External Events Data Sources

The External Events data source type is not represented by any predefined data sources. It is different from other data source types in that it generates event records with fields that you define and hands them over to the InTrust agent to process.

Data sources of this type are represented by a command-line utility on the agent side and an InTrust data source object on the InTrust server side.

This command-line utility forces special events on the InTrust agent running on the same computer. The agent stores the events in its backup cache. From there, the events can be captured by the gathering or real-time monitoring engine.

### *To create an External Events data source*

1. Right-click the **Configuration | Data Sources** node and select **New Data Source**.
2. In the New Data Source Wizard, select the **External Events** data source type.
3. Complete the remaining steps.

For details about External Events data source settings, see the Configuring Data Sources topic the InTrust Auditing Guide.

# Script Event Provider Data Sources

InTrust provides an additional option to create a custom data source using the Script Event Provider.

This functionality allows you to create a script that starts with pre-set frequency. Under some conditions that are specified in this script events are generated and then are passed to the InTrust agent. Events are stored in the agent's backup cache. From there, the events can be captured by the gathering or real-time monitoring engine.

You can specify in the certain script: what information is stored and how it is ordered in the certain events, what conditions are required for event generation.

### *To create a custom data source with Script Event Provider*

1. Right-click the **Configuration | Data Sources** node and select **New Data Source**.
2. In the New Data Source Wizard, select the **Script Event Provider** data source type.
3. On the **Script** step select the script language and enter your script text using XML editor.
4. On the same step specify a frequency of the script running.
5. Complete the remaining steps.

# Pattern Letters for Date and Time Designation

| Letter | Date or Time Component | Examples |
|---|---|---|
| G | Era designator | AD |
| y | Year | 1996; 96 |
| M | Month in year | July; Jul; 07 |
| w | Week in year | 27 |
| W | Week in month | 2 |
| D | Day in year | 189 |
| d | Day in month | 10 |
| F | Day of week in month | 2 |
| E | Day in week | Tuesday; Tue |
| a | A.M./P.M. marker | PM |
| H | Hour in day (0-23) | 0 |
| k | Hour in day (1-24) | 24 |
| K | Hour in A.M./P.M. (0-11) | 0 |
| h | Hour in A.M./P.M. (1-12) | 12 |
| m | Minute in hour | 30 |
| s | Second in minute | 55 |
| S | Millisecond | 978 |
| Z | RFC 822 time zone | -0800 |
| t | Second in POSIX time | 1095379198 |

# We are more than just a name

We are on a quest to make your information technology work harder for you. That is why we build community-driven software solutions that help you spend less time on IT administration and more time on business innovation. We help you modernize your data center, get you to the cloud quicker and provide the expertise, security and accessibility you need to grow your data-driven business. Combined with Quest's invitation to the global community to be a part of its innovation, and our firm commitment to ensuring customer satisfaction, we continue to deliver solutions that have a real impact on our customers today and leave a legacy we are proud of. We are challenging the status quo by transforming into a new software company. And as your partner, we work tirelessly to make sure your information technology is designed for you and by you. This is our mission, and we are in this together. Welcome to a new Quest. You are invited to Join the Innovation™.

# Our brand, our vision. Together.

Our logo reflects our story: innovation, community and support. An important part of this story begins with the letter Q. It is a perfect circle, representing our commitment to technological precision and strength. The space in the Q itself symbolizes our need to add the missing piece — you — to the community, to the new Quest.

# Contacting Quest

For sales or other inquiries, visit https://www.quest.com/company/contact-us.aspx or call +1-949-754-8000.

# Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to-videos
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product