Configuring Log Parsers Step-by-Step

Quick Start Guide

Version R91

English

May 20, 2015

Agreement

The purchase and use of all Software and Services is subject to the Agreement as defined in Kaseya's "Click-Accept" EULATOS as updated from time to time by Kaseya at http://www.kaseya.com/legal.aspx. If Customer does not agree with the Agreement, please do not install, use or purchase any Software and Services from Kaseya as continued use of the Software or Services indicates Customer's acceptance of the Agreement."

Contents

Introduction

The VSA is capable of monitoring data collected from many standard log files. **Log Monitoring** extends that capability by extracting data from the output of any text-based log file. Examples include application log files and syslog files created for Unix, Linux, and Apple operating systems, and network devices such as Cisco routers. To avoid uploading all the data contained in these logs to the Kaseya Server database, **Log Monitoring** uses parser definitions and parser sets to parse each log file and select only the data you're interested in. Parsed messages are displayed in Log Monitoring, which can be accessed using the Agent Logs tab of Live Connect > Agent Data or the Machine Summary page or by generating a report using the Agent > Logs - Log Monitoring page. Users can optionally trigger alerts when a **Log Monitoring** record is generated, as defined using Assign Parsing Sets or Parser Summary.

Parser Definitions vs. Parser Sets

When configuring Log Monitoring it's helpful to distinguish between two kinds of configuration records: **parser definitions** and **parser sets**.

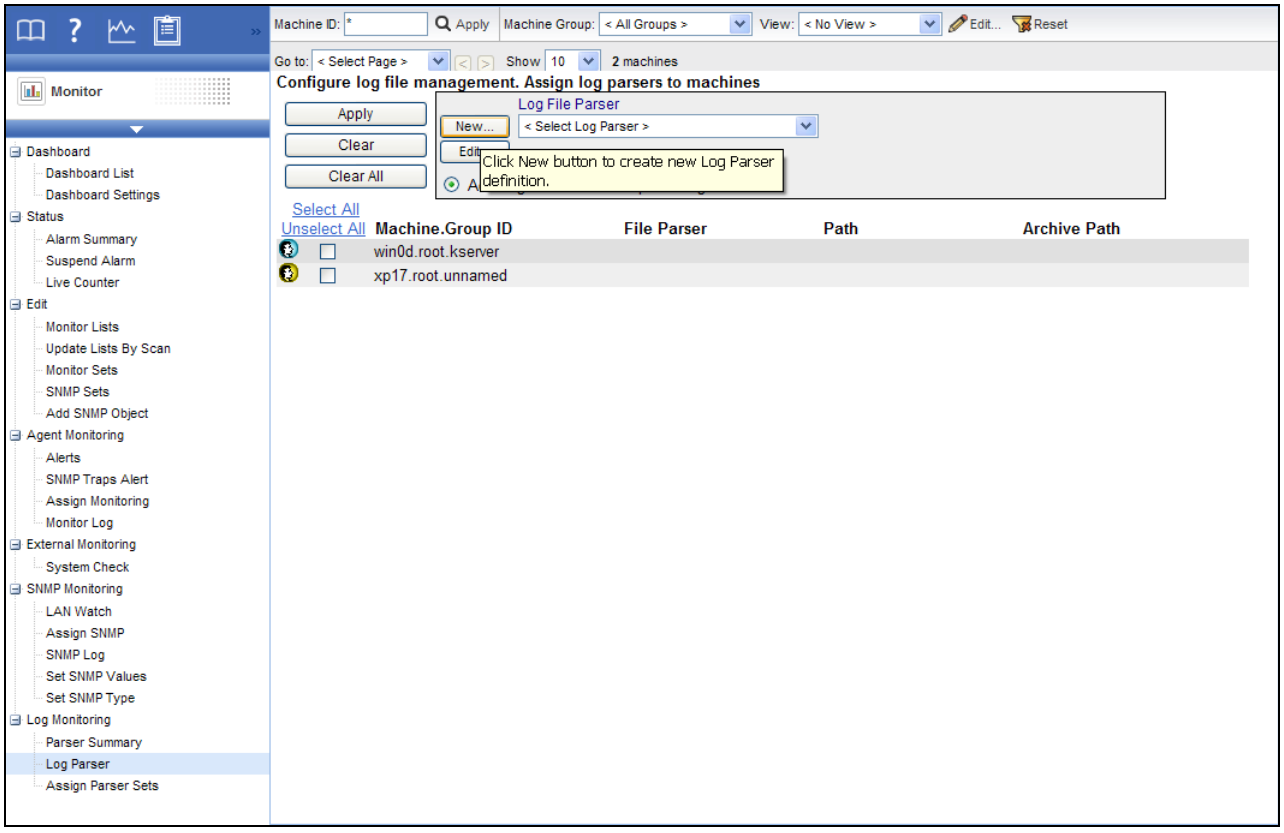
A **parser definition** is used to:

- Locate the log file being parsed.
- Select log data based on the log data's *format*, as specified by a template.
- Populate parameters with log data values.
- Optionally format the log entry in **Log Monitoring**.

A **parser set** subsequently *filters* the selected data. Based on the *values* of populated parameters and the criteria you define, a parser set can generate log monitoring entries and optionally trigger alerts.

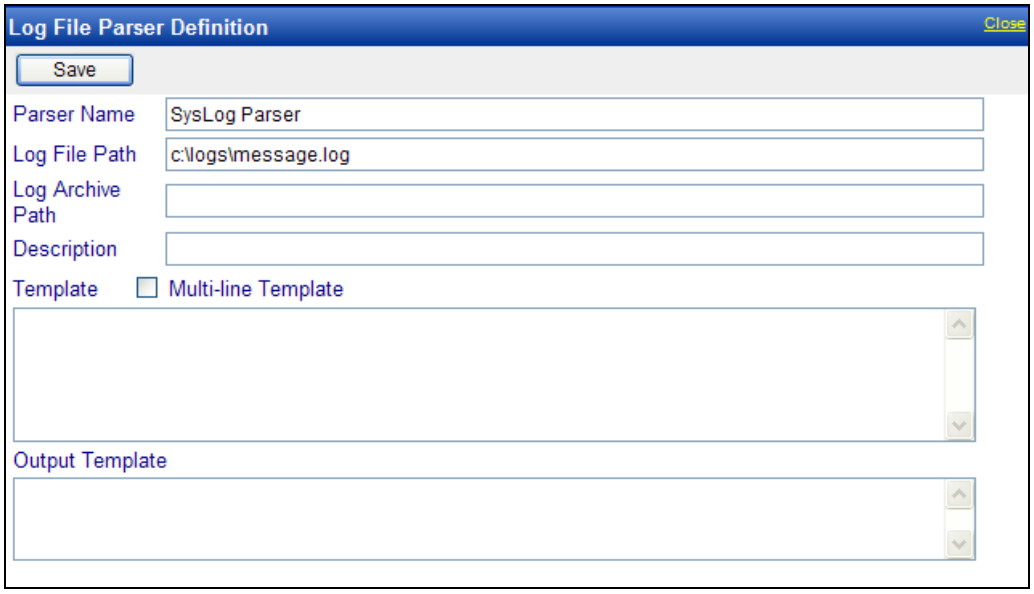
Without the filtering performed by the parser set, the Kaseya Server database would quickly expand. For example a log file parameter called $FileServerCapacity$ might be repeatedly updated with the latest percentage of free space on a file server. Until the free space is less than 20% you may not need to make a record of it in **Log Monitoring**, nor trigger an alert based on this threshold. Each parser set applies only to the parser definition it was created to filter. Multiple parser sets can be created for each parser definition. Each parser set can trigger a separate alert on each machine ID it is assigned to.

Step 1: Create a new log parser definition



Go to the **Monitor** tab in the VSA. Select **Log Parser** under **Log Monitoring**. Click the **New** button to create a new log parser definition.

Step 2: Enter Parser Name, Log File Path



Enter the following:

Parser name - The name of this log parser definition.

Log File Path - The full path of the log file to be processed. This path must be accessible by the agent. The log file should contain formatted log entries. Unicode files are not supported yet. Example: `c:\logs\message.log`.

> **Note:** The asterisk (*) wildcard character can be used in the filename. The most recent file will be processed in this case. Example: `c:\logs\message*.log`.

Click the **Save** button after entering the parser name and log file path. The window expands to include parameter definitions.

Optional Information

Log Archive Path - The log parser checks changes of the target log file periodically. The log entries may be archived into different archive files before the log parser can process those entries. So you can specify the archive file path in the field of Log Archive Path. Example: If `message.log` is archived daily to a file in `messageYYYYMMDD.log` format, then you can specify `c:\logs\message*.log` for the **Log Archive Path**. **Log Parser** is able to locate the file it processed last since it keeps a bookmark for the log file.

Description - The detail description of the log parser.

Step 3: Specify templates and define parameters

Template

The template is used to compare with the log entry in the log file to extract out the required data into parameters. Parameters are enclosed with $ character in template. It is important that you must have texts around the parameters so the parameters can be clearly distinguished. Characters in log entry

are compared case sensitively against the template.

Single line temple to parser single line log entry - The template only contains one line entry and the log file is processed line by line.

Multi-line template to parse multi-line log entries - The template contains multiple line entries and the log file is processed by block of lines delimited by a line boundary.

> **Note:** The character string `{tab}` can be used as a tab character and `{nl}` can be used as a new line break. `{nl}` cannot be used in single line template. `%` can be used as wildcard character.

Hint: It is easier to copy and paste the log entry into the **Template** edit box and replace the needed data with parameter names, instead of trying to create a log entry template by typing it all in.

Output Template

This is an optional field. It can be used to format the message when the log entry is saved into the database, otherwise, the log entry itself is saved as the message in the database.

Log File Parameters

Once the template is created, you need to define the list of parameters used by the template. All the parameters in the template have to be defined, otherwise the parser returns an error. Available parameters are *integer, unsigned integer, long, unsigned long, float, double, datetime, string.* The length of parameter name is limited to 32 characters.

Date Time Format String

A template string can contain a date and time format that is used to parse the date time information from log entries. Example: YYYY-MM-DD hh:mm:ss

Formats:

- `yy, yyyy, YY, YYYY` - two or four digit year
- `M` - single or two digit month
- `MM` - two digit month
- `MMM` - abbreviation of month name, ex." Jan"
- `MMMM` - full month name, ex. "January"
- `D`, `d` - single or two digit day
- `DD`, `dd` - two digit day
- `DDD`, `ddd` - abbreviation name of day of week, Ex." Mon"
- `DDDD`, `dddd` - full name of day of week, ex. "Monday"
- `H`, `h` - single or two digit hour
- `HH`, `hh` - two digit hour
- `m` - single or two digit minute
- `mm` - two digit minute
- `s` - single or two digit second
- `ss` - two digit second
- `f` - one or more digit of fraction of second
- `ff` - fffffffff – two to nine digit
- `t` - one character time mark, ex. "a"
- `tt` - two-character time mark, ex. "am"

> **Note:** Each date time parameter must contain at least the month, day, hour, and second data. The value from the `$Time$` parameter is used as the event time if it is specified. Otherwise, the time when the entry is processed is used as the event time in the database.

Example 1 - Single Line Log Entry

Start with a typical log entry from the log file you want to monitor:

```
<189> 2009 Aug 31 06:57:48 (FVS114-ba-b3-d2) 71.121.128.42 ICMP Packet[Destination Unreachable] - Source:192.168.0.186 - Destination:192.168.0.1 - [Receive]
```

Identify the parts of the log entry you want to populate parameters with:

```
<189> 2009 Aug 31 06:57:48 (FVS114-ba-b3-d2) 71.121.128.42 ICMP Packet[Destination Unreachable] - Source:192.168.0.186 - Destination:192.168.0.1 - [Receive]
```

In the template, replace the underline text with parameters:

```
<$code$> $Time$ ($device$) $HostName$ $PackType$ Packet[$Action$] - Source:$SrcAddr$ - Destination:$DestAddr$ - $Msg$
```

Log File Parameters

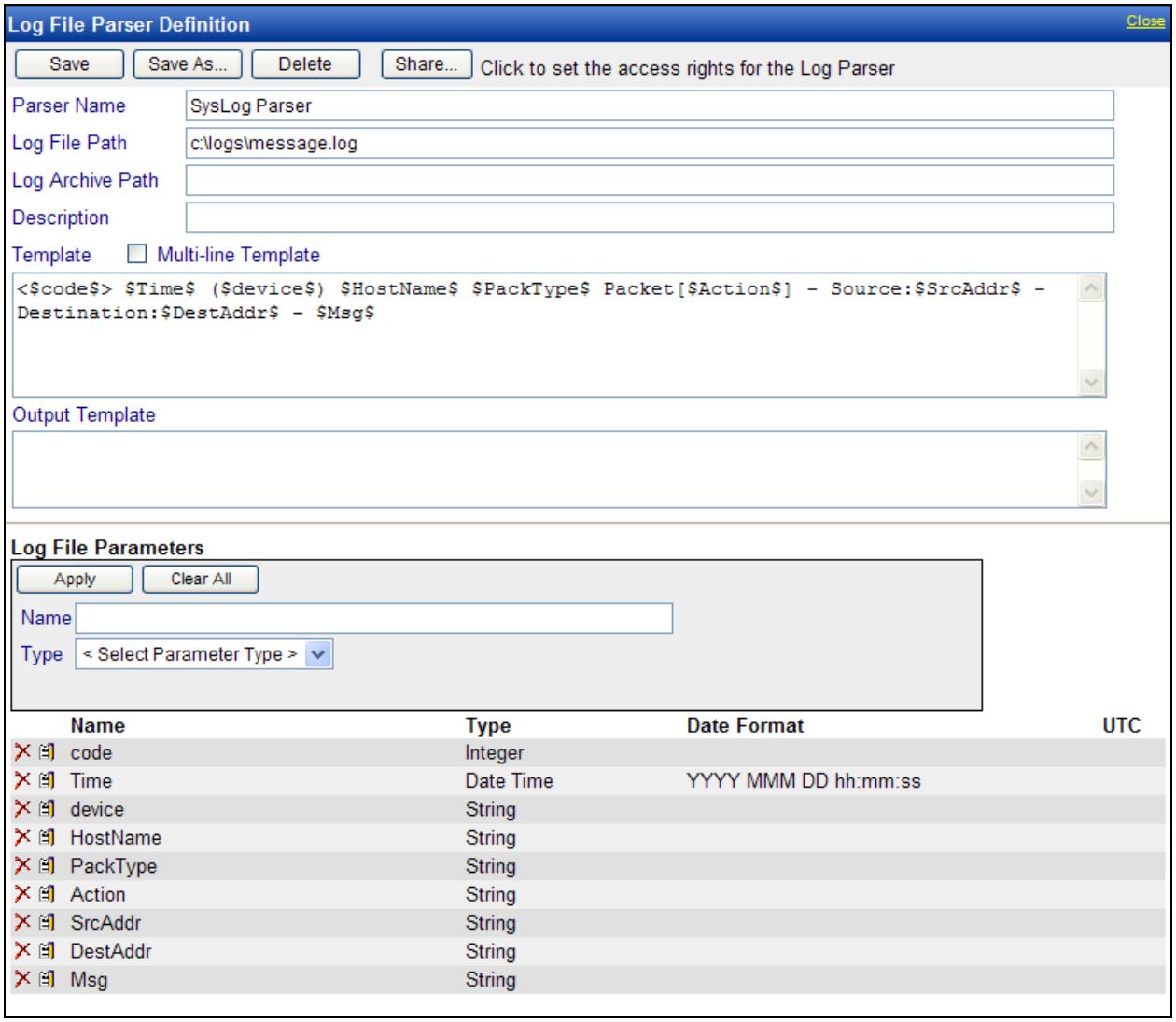
> **Note:** Click the **Save** button at least once to display the **Log File Parameters** section of the dialog box.

Text not used to populate parameters must match text in the log entry. For example: the string '`] - Source:`' must match the text in the log entry, including the space character just before the hyphen.

Define the parameters:

Parameter name	Parameter Type	ParsedResult
code	Integer	189
Time	datetime in "YYYY MMM DD hh:mm:ss" format, not UTC	2006-11-08 11:57:48
device	String	FVS114-ba-b3-d2
HostName	String	71.121.128.42
PackType	String	ICMP
Action	String	Destination Unreachable
SrcAddr	String	192.168.0.186
DestAddr	String	192.168.0.1
Msg	String	[Receive]

Step 3: Specify templates and define parameters



Example 2 – Including the % Symbol (wildcard)

Start with a typical log entry from the log file you want to monitor:

```
<189> 2009 Aug 31 06:57:48 (FVS114-ba-b3-d2) 71.121.128.42 ICMP Packet[Destination Unreachable] - Source:192.168.0.186 - Destination:192.168.0.1 - [Receive]
```

Identify unneeded text in the log file you want to monitor:

<189> 2009 Aug 31 06:57:48 ~~(FVS114-ba-b3-d2)~~ 71.121.128.42 ICMP Packet~~[Destination Unreachable] -~~ Source:192.168.0.186 - Destination:192.168.0.1 - ~~[Receive]~~

In the template, replace the unneeded strikethrough text above with a percent sign (%) wildcard character. Replace other text with parameters:

```
<$code$> $Time$ % $HostName$ $PackType$ Packet% Source:$SrcAddr$ - Destination:$DestAddr$ -
```

Define the parameters:

Parameter name	Parameter Type	ParsedResult
code	Integer	`189`
Time	datetime in `YYYY MMM DD hh:mm:ss` format	`2006-11-08 11:57:48`

HostName	String	`71.121.128.42`
PackType	String	`ICMP`
SrcAddr	String	`192.168.0.186`
DestAddr	String	`192.168.0.1`

Example 3 - Multiple Line Log Entries

Start with a typical multiple line log entry from the log file you want to monitor:

```
Summary Of This Scan
»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»
Total scanning time:00:02:32.765
Objects scanned:91445
Objects identified:0
Objects ignored:0
New critical objects:0
```

Identify text that should be ignored and text that should be populated by parameters.

```
Summary Of This Scan
»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»
Total scanning time:00:02:32.765
Objects scanned:91445
Objects identified:0
Objects ignored:0
New critical objects:0
```

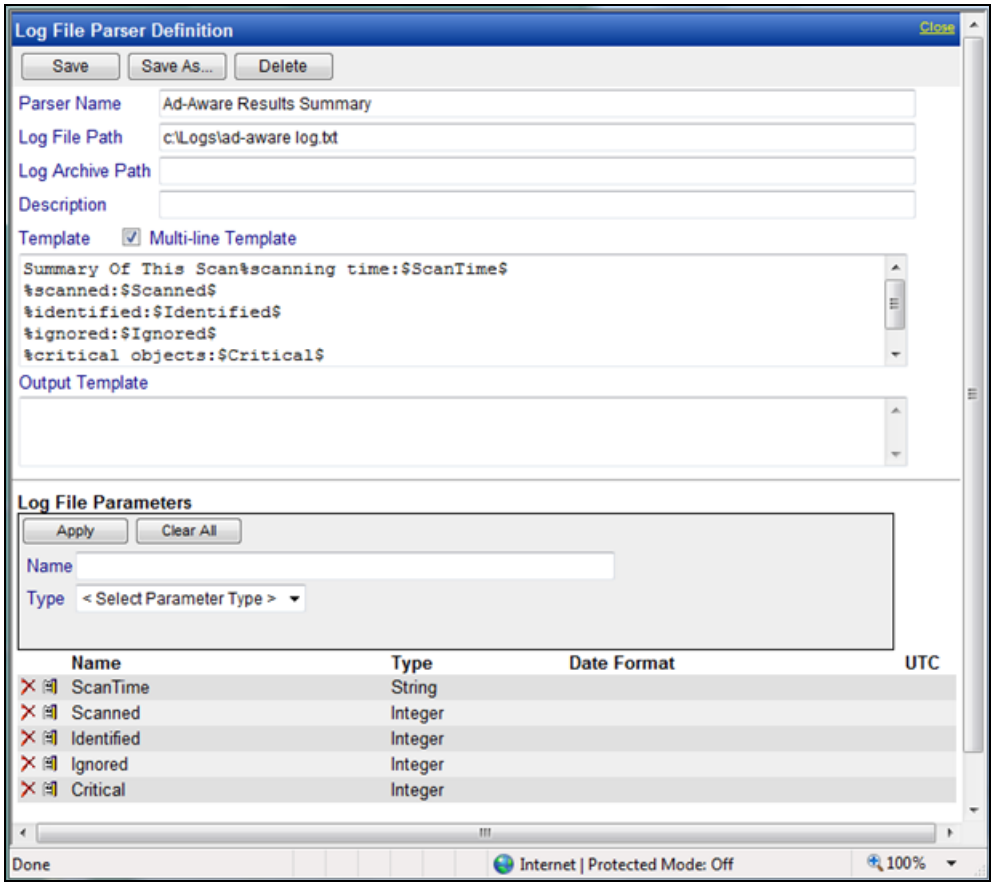
In the template, replace the strikethrough text with a percent sign (%) wildcard. Replace the underlined text with parameters.

```
Summary Of This Scan
%scanning time:$ScanTime$
%scanned:$Scanned$
%identified:$Identified$
%ignored:$Ignored$
%critical objects:$Critical$
```

Define the parameters:

Parameter name	Parameter Type	ParsedResult
ScanTime	String	00:02:32.765
Scanned	Integer	91445
Identified	Integer	0
Ignored	Integer	0
Critical	Integer	0

Step 3: Specify templates and define parameters

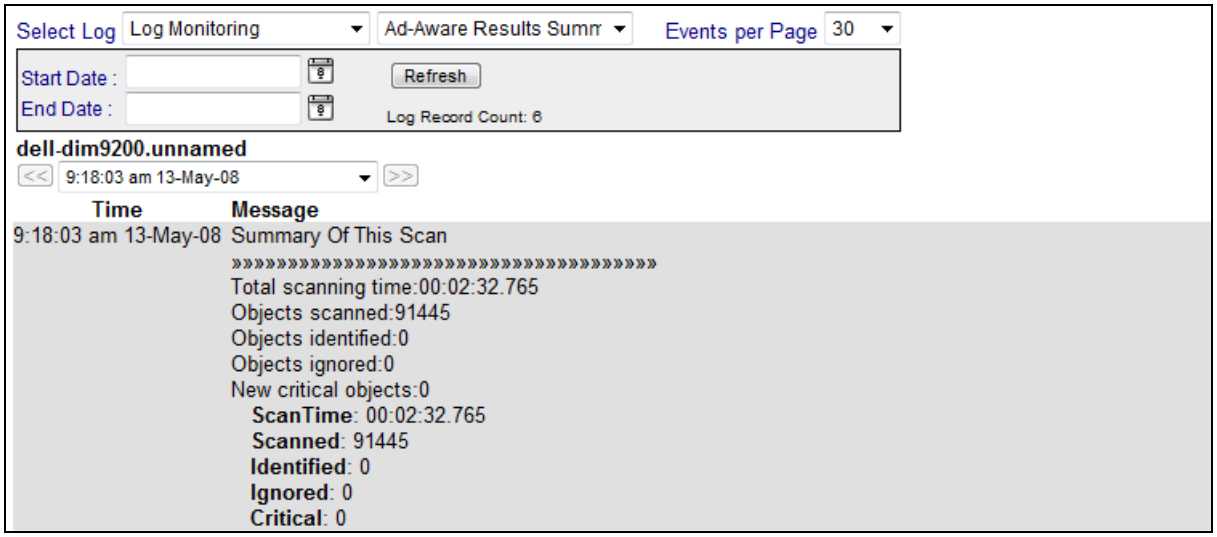


Example 4 – Output template

Start with a typical multiple line log entry from the log file you want to retrieve:

```
Summary Of This Scan
»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»»
Total scanning time:00:02:32.765
Objects scanned:91445
Objects identified:0
Objects ignored:0
New critical objects:0
```

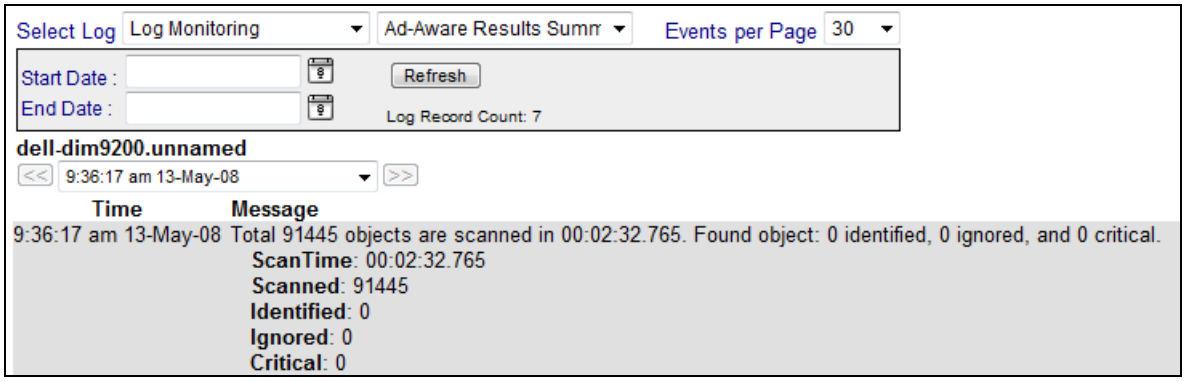
All the above data will be logged as the body of the message in the monitor log if an output template is not specified. Here is example of the output in Log Monitoring without specifying an output template:



In the output template, specify a template by using defined parameters:

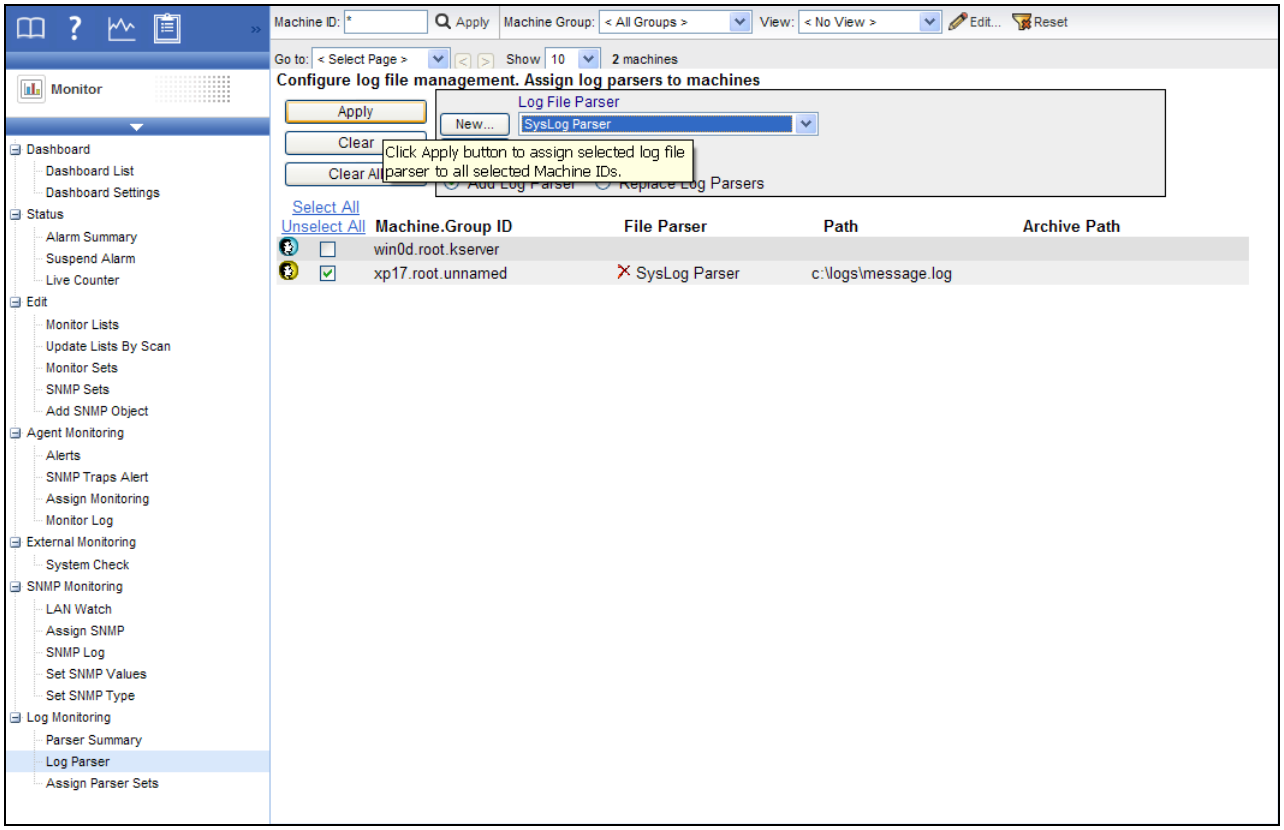
```
Total $Scanned$ objects are scanned in $ScanTime$. Found object: $Identified$
identified, $Ignored$ ignored, and $Critical$ critical.
```

Here is an example of the output in Log Monitoring after specifying an output template:



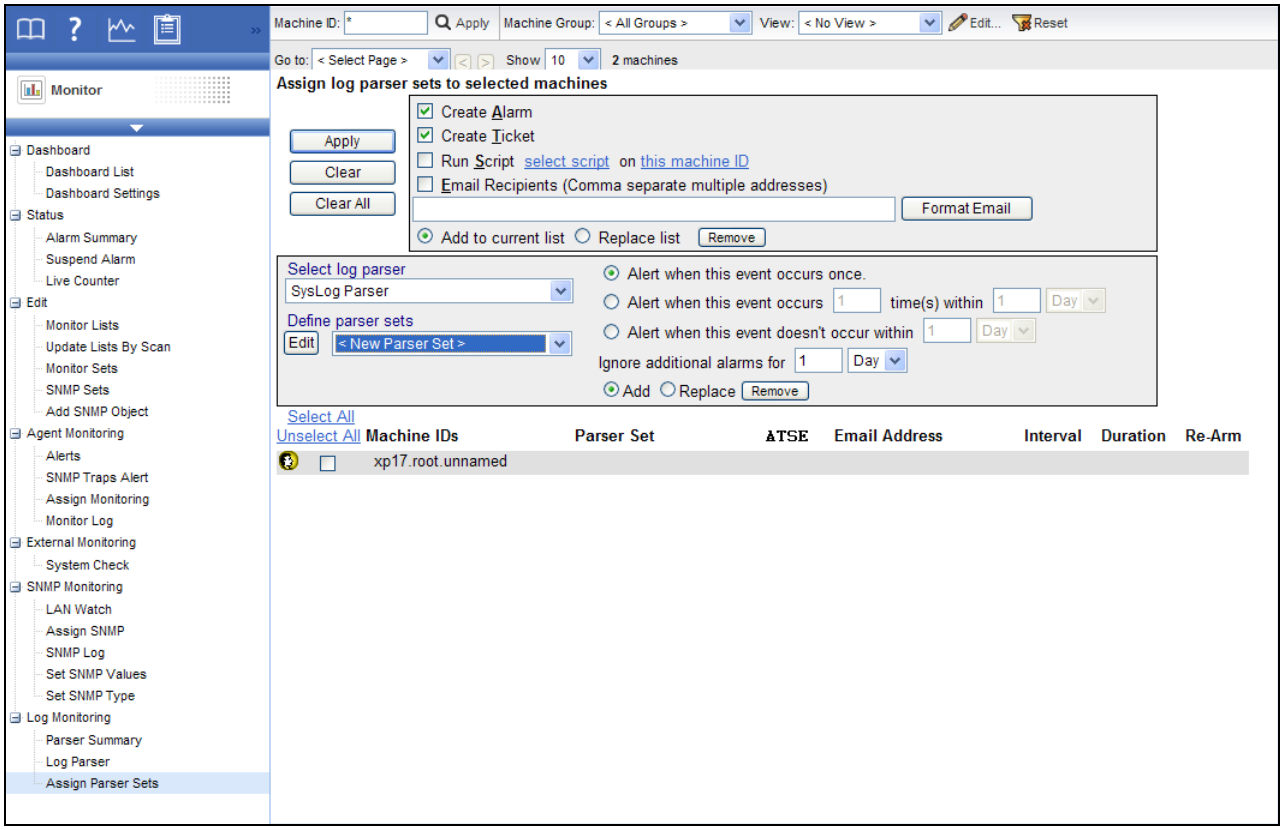
Step 4: Assign the Log Parser Definition

A completed log file parser definition must be assigned to one or machine IDs using the **Log Parser** function. Select the machines IDs to apply the definition to and click the **Apply** button. This means that the parser definition can be used by the selected machines, but parsing does not occur until you select the filter criteria for the log data being collected and assign alert conditions, as described in Steps 5 and 6.

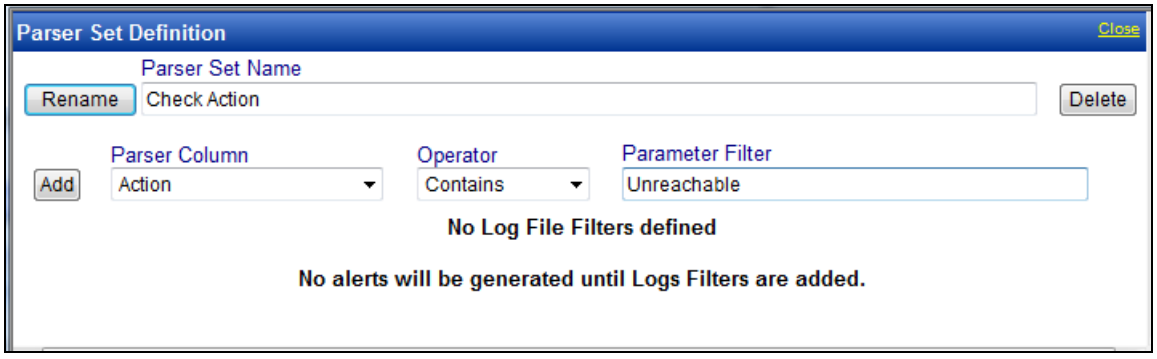


Step 5: Define collection and alerts conditions

Click **Assign Parser Sets** under **Log Monitoring** in the function list. Select the log parser definition from the **Select log parser** drop-down list. Then select `<New Parser Sets>` from the **Define parser sets** drop-down list. *A log parser set is a set of conditions that must be true about the parsing of a log entry to include it in the 'log monitoring' log and optionally create an alert for it.* This ensures that only relevant log entries are posted to the "log monitoring' log. Note that a log parser set is specific to a log parser. You could define multiple log parser sets for the same log parser and trigger a different set of alert for each log parser set.



Define the alert conditions. In the following example, an entry is created in the 'log monitoring' log if a log entry is parsed such that the `Action` parameter contains the text `Unreachable`.



Operators for Parameters

- **String** - `begins with, does not begin with, contains, does not contain, ends with, does not end with, equals, does not equal`

- **Numeric** - `equal, not equal, over, under`
- **Time** - `equal, not equal, over, under`

The **Parameter Filter** for **Time** can be in one of the following formats. A filter string ending with a `Z` indicates an UTC time.

- `YYYY-MM-DDThh:mm.ss`
- `YYYY/MM/DDThh:mm.ss`
- `YYYY-MM-DD hh:mm.ss`
- `YYYY/MM/DD hh:mm.ss`
- `YYYY-MM-DDThh:mm.ssZ`
- `YYYY/MM/DDThh:mm.ssZ`
- `YYYY-MM-DD hh:mm.ssZ`
- `YYYY/MM/DD hh:mm.ssZ`

Example: `2008-04-01 15:30:00.00`

Parser Sets and Conditions

The conditions are defined in a parser set. You can assign multiple conditions to a parser set. You can also assign multiple parser sets to a log parser. A log entry has to meet all the conditions inside a parser set in order to trigger data collection and/or alert. Please note this behavior is different from event log alerts and other monitor sets. For example:

Log contents:

```
05/09/2008 12:21:34 192.168.0.1 error "lookup failed"
05/09/2008 12:21:35 192.168.0.1 error "syslog stopped"
05/09/2008 12:21:37 192.168.0.1 information "syslog starts"
05/09/2008 12:21:38 192.168.0.2 warning "ping failed"
05/09/2008 12:22:04 192.168.0.2 warning "unknown message"
```

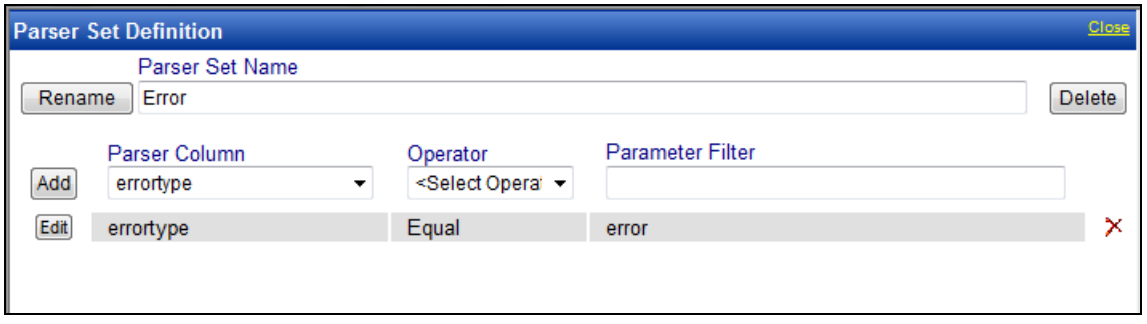
Single line template:

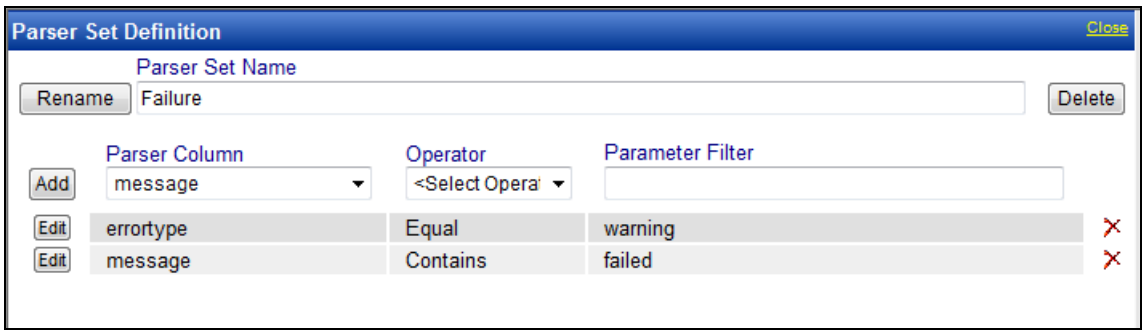
```
$Time$ $hostname$ $errortype$ $message$
```

To collect entries which meet one of following conditions you need to define two parser sets and assign both to the log parser:

```
$errortype$ is "error"
$errortype$ is "warning" AND $message$ contains "failed"
```

Here are the corresponding screen captures for these two parser sets:



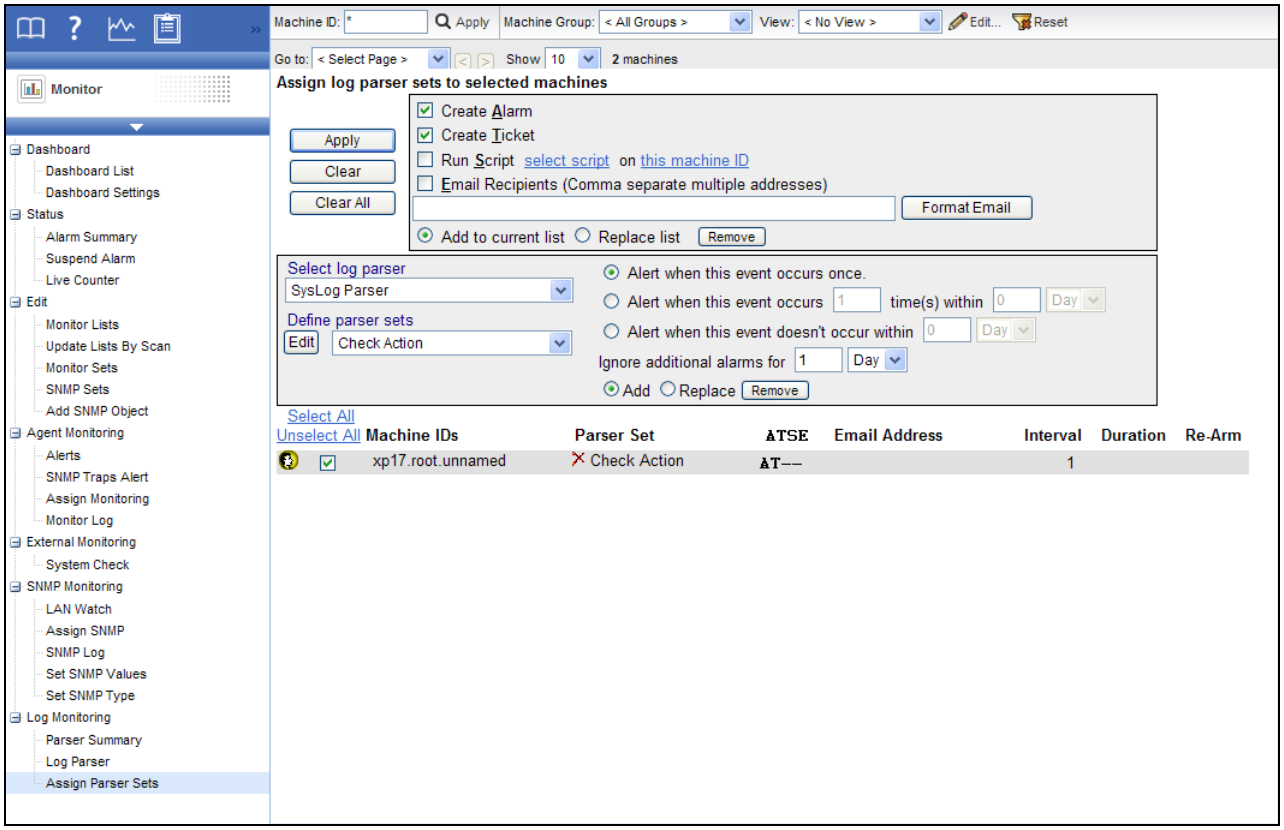


Step 6: Assign Parser Set

Select a machine ID, alarm options, and types of alerts, then click the **Apply** button to assign the log parser set to a machine ID. Once the machine ID receives the log parser configuration, the agent on the managed machine will start parsing the log file *whenever the log file is updated.*

Notification

The agent collects log entries and creates an entry in the 'log monitoring' log based on the criteria defined by the parser set, *whether or not any of the notification methods are checked.* You don't have to be notified each time a new log monitoring entry is created. You can simply review the 'Log Monitoring' log periodically at your convenience.

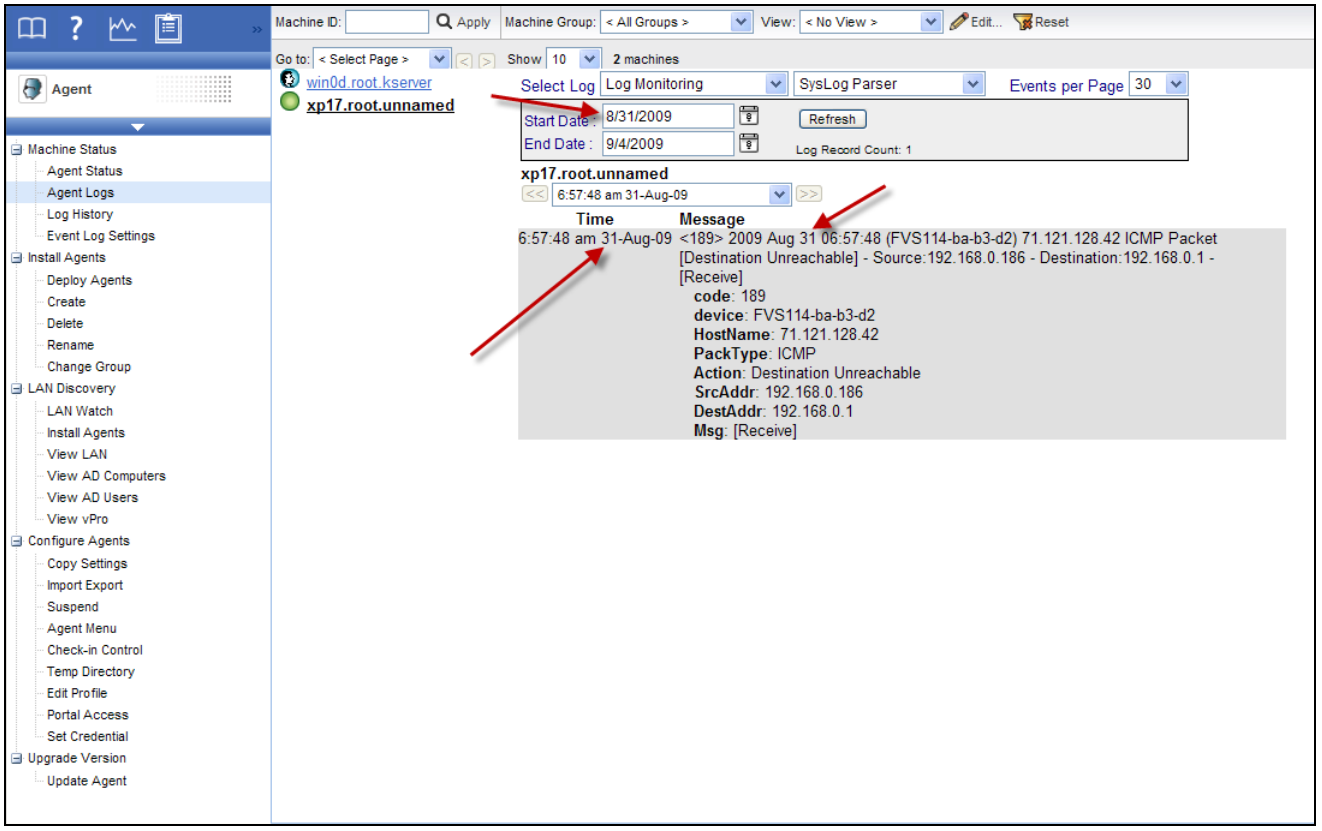


Step 7: Review the 'Log Monitoring' Log

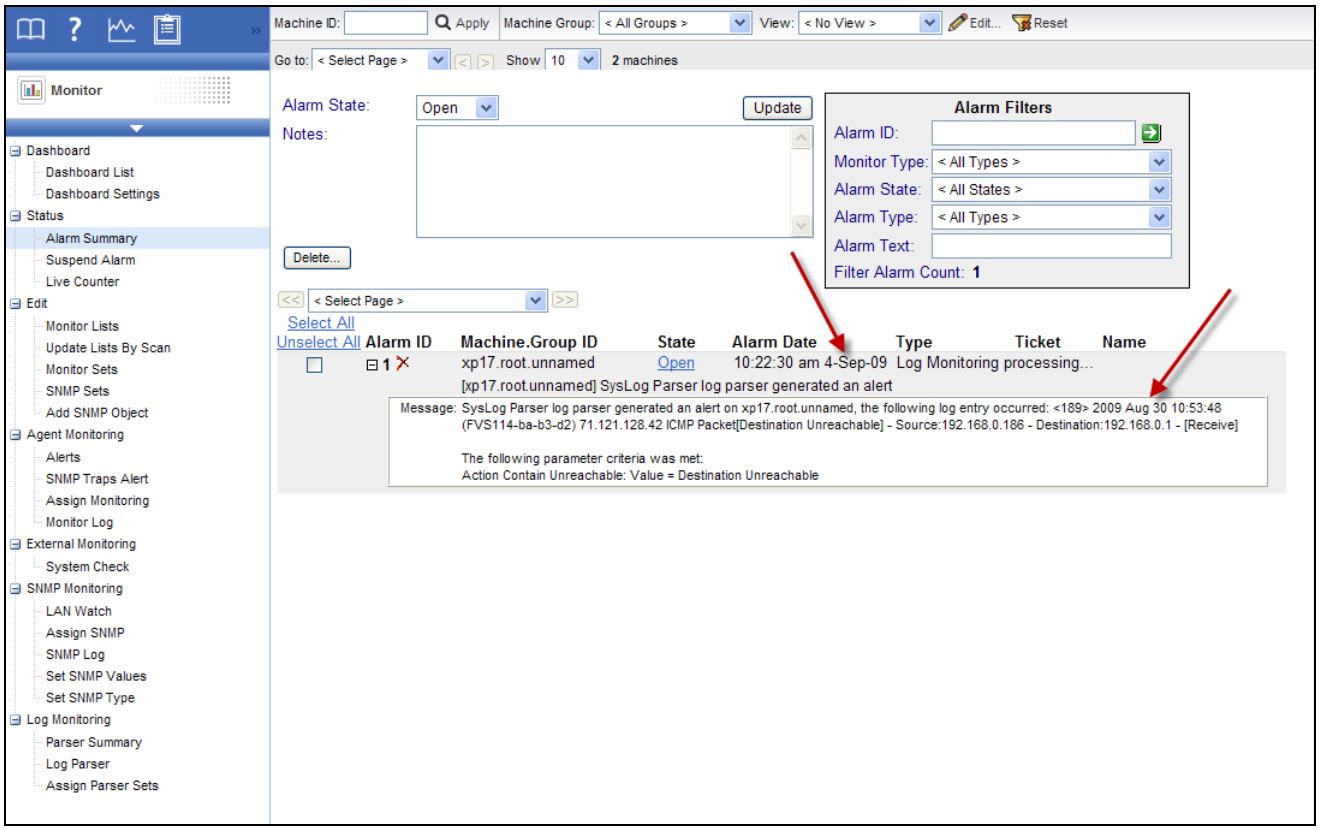
Log Monitoring entries are displayed in **Log Monitoring**, which can be accessed using:

- Agents > Agent Logs > Log Monitoring > (parser definition)
- Live Connect > Agent Data > Agent Logs > Log Monitoring > (parser definition). Live Connect is displayed by clicking the check-in status icon of a selected machine ID.
- Audit > Machine Summary > Agent Logs tab > Log Monitoring > (parser definition). The Machine Summary page can also be displayed by *alt-clicking* the check-in status icon of a selected machine ID.
- The Info Center > Reporting > Reports > Monitor - Logs > Log Monitoring report.

These sample images show the $Time$ parameter being used for Log Monitoring entries. *Date and time filtering in views and reports are based on the log entry time.* If you include a `$Time$` parameter using the `Date Time` data type in your template, **Log Monitoring** uses the time stored in the `$Time$` parameter as the log entry time. If a `$Time$` parameter is *not* included in your template, then the time the entry was added to **Log Monitoring** serves as the log entry time. Be sure to select a date range that displays the log entry dates.



In contrast, alarms dates are based on the date the alarm was created, not the date of entries in the 'Log Monitoring' log.



Index

I

S