OPTIMIZE

MERCURY BUSINESS AVAILABILITY CENTER™

Advanced Monitor Options

MERCURY™

BUSINESS TECHNOLOGY OPTIMIZATION

Mercury Business Availability Center

Advanced Monitor Options

Version 6.2

Document Release Date: June 20, 2006

MERCURY™

Mercury Business Availability Center, Version 6.2
Advanced Monitor Options

This manual, and the accompanying software and other documentation, is protected by U.S. and international copyright laws, and may be used only in accordance with the accompanying license agreement. Features of the software, and of other products and services of Mercury Interactive Corporation, may be covered by one or more of the following patents: United States: 5,511,185; 5,657,438; 5,701,139; 5,870,559; 5,958,008; 5,974,572; 6,137,782; 6,138,157; 6,144,962; 6,205,122; 6,237,006; 6,341,310; 6,360,332, 6,449,739; 6,470,383; 6,477,483; 6,549,944; 6,560,564; 6,564,342; 6,587,969; 6,631,408; 6,631,411; 6,633,912; 6,694,288; 6,738,813; 6,738,933; 6,754,701; 6,792,460 and 6,810,494. Australia: 763468 and 762554. Other patents pending. All rights reserved.

Mercury, Mercury Interactive, the Mercury logo, the Mercury Interactive logo, LoadRunner, WinRunner, SiteScope and TestDirector are trademarks of Mercury Interactive Corporation and may be registered in certain jurisdictions. The absence of a trademark from this list does not constitute a waiver of Mercury's intellectual property rights concerning that trademark.

All other company, brand and product names may be trademarks or registered trademarks of their respective holders. Mercury disclaims any responsibility for specifying which marks are owned by which companies or which organizations.

Mercury provides links to external third-party Web sites to help you find supplemental information. Site content and availability may change without notice. Mercury makes no representations or warranties whatsoever as to site content or availability.

Mercury Interactive Corporation
379 North Whisman Road
Mountain View, CA 94043
Tel: (650) 603-5200
Toll Free: (800) TEST-911
Customer Support: (877) TEST-HLP
Fax: (650) 603-5300

© 2006 Mercury Interactive Corporation, All rights reserved

If you have any comments or suggestions regarding this document, please send them by e-mail to documentation@mercury.com.

AMLIB_SISADM6.2/01

Table of Contents

Welcome to Advanced Monitor Options ..... **v**
How This Guide is Organized v
Who Should Read This Guide vi
Getting More Information vi

Chapter 1: Using Regular Expressions **1**
Using Regular Expressions in SiteScope 2
Defining a Regular Expression 2
Matching String Literals 4
Matching Patterns with Metacharacters 6
Search Mode Modifiers 10
Retaining Content Match Values 11
SiteScope Date Variables 12
Examples for Log File Monitoring 17
Some Pitfalls in Working with Regular Expressions 20

Chapter 2: Monitoring XML Documents **25**
Understanding Content Matching for XML Documents 25
Using XML Content Match Values in Monitor Configurations 27

Chapter 3: Setting up and Using Rolling Baselines **29**
About Rolling Baselines 29
Enabling Rolling Baselines 31
Disabling Rolling Baselines 32

Chapter 4: Setting Up and Using Static Performance Baselines **33**
About Static Performance Baselines 34
Understanding Performance Baselining 35
Enabling Static Performance Baselining for Monitors 36
Setting Monitor Thresholds Using Baselines 37
Monitor Properties for Static Baselines 39

Chapter 5: SiteScope Monitoring via Secure Shell (SSH)....41
SiteScope and SSH 41
Configuring Remote UNIX Servers for SSH Monitoring....44
Configuring Remote Windows Servers for SSH Monitoring....45
SiteScope SSH Client Connection Options 58

Chapter 6: Working with SSH Clients....59
Using the Integrated Java SSH Client....59
Using an External SSH Client....62

Chapter 7: Creating Custom Properties 71
About Custom SiteScope Properties 71
Working with Custom Monitor Properties 72
Working with Custom Display Columns....76
Displaying Custom Properties in Custom Columns....78

Index....81

Welcome to Advanced Monitor Options

This guide provides advanced information related to SiteScope monitoring.

How This Guide is Organized

The guide contains the following chapters:

Chapter 1 Using Regular Expressions

Describes how to define SiteScope management reports that are generated automatically at regular time intervals.

Chapter 2 Monitoring XML Documents

Describes how to use SiteScope's content matching capabilities to traverse XML documents.

Chapter 3 Setting up and Using Rolling Baselines

Describes how to use SiteScope's rolling performance baselining option to aid in establishing a performance baseline.

Chapter 4 Setting Up and Using Static Performance Baselines

Describes how to use SiteScope's static performance baselining option to aid in establishing a performance baseline.

Chapter 5 SiteScope Monitoring via Secure Shell (SSH)

Describes SiteScope's capability for remote server monitoring using Secure Shell (SSH) connections.

Chapter 6 Working with SSH Clients

Describes some of the client configuration possibilities and issues involved in using SSH for SiteScope monitoring.

Chapter 7 Creating Custom Properties

Describes how to add custom monitor properties and how to add a custom display column to SiteScope real-time dashboard views.

Who Should Read This Guide

This guide is intended for the following users of Mercury Business Availability Center:

- Mercury Business Availability Center administrators
- Mercury Business Availability Center data collector administrators

Readers of this guide should be knowledgeable about enterprise system administration, infrastructure monitoring systems, and SiteScope, and have familiarity with the systems being set up for monitoring.

Getting More Information

For information on using and updating the Mercury Business Availability Center Documentation Library, reference information on additional documentation resources, typographical conventions used in the Documentation Library, and quick reference information on deploying, administering, and using Mercury Business Availability Center, refer to *Getting Started with Mercury Business Availability Center*.

1

Using Regular Expressions

Several SiteScope monitors allow for content matching on the text returned from a monitor's request or action (see the documentation for each monitor for more information about the content returned). This adds an important level of functionality in system monitoring. SiteScope makes use of regular expressions to match text content.

This chapter describes:	On page:
Using Regular Expressions in SiteScope	2
Defining a Regular Expression	2
Matching String Literals	4
Matching Patterns with Metacharacters	6
Search Mode Modifiers	10
Retaining Content Match Values	11
SiteScope Date Variables	12
Examples for Log File Monitoring	17
Some Pitfalls in Working with Regular Expressions	20

Using Regular Expressions in SiteScope

Regular expressions is a name given to a text parsing tool that was developed for use with scripting languages such as Awk and Perl as well as several programming environments such as Emacs, Visual C++, and Java. Regular expressions themselves are not a programming language. They do, however, make use of many special combinations of characters and symbols that often make them more difficult to interpret than some programming languages. The many different combinations of these special characters, known as metacharacters, make regular expressions a very powerful and flexible for parsing and isolating specific text within a larger body of text.

Including a regular expression in the **Match Content** text box of a monitor instructs SiteScope to parse the text returned to the monitor when it is run and look for content that satisfies the pattern defined by the regular expression. This document presents an overview of the syntax and metacharacters used in regular expressions for use in matching content for SiteScope monitors.

Defining a Regular Expression

The element of a match content expression in SiteScope is the forward slash (/) character. Entries in the **Match Content** text box of a SiteScope monitor must start and end with a forward slash to be recognized as regular expressions. For example, entering the expression /website/ into the **Match Content** box of a monitor instructs SiteScope to search the text content received by the monitor for the literal text string: website. If a match is not found, the monitor reports an error status. When a match is found, the monitor reports a good status as long as all other monitor status threshold conditions are also met. If you enter text or other characters into the **Match Content** box without delimiting the entry with forward slashes, the entry will either be ignored or reported as a content match error by SiteScope.

Adding parentheses ( ) within the forward slashes surrounding the regular expression is another very useful feature for regular expressions in SiteScope. The parentheses are used to create a "back reference." As a back reference, SiteScope retains what was matched between the parentheses and displays the text in the Status field of the monitor detail page. This is very useful for troubleshooting match content. This is also a way to pass a matched value from one monitor to another or from one step of a URL Sequence Monitor to the next step of the same transaction. Parentheses are also used to limit alternations, as discussed below.

Generally, it is best to use an iterative approach when building regular expressions for content matching with SiteScope. The following are some general steps and guidelines for developing regular expressions for content matches:

- Create a regular expression using literal characters to match a single sample of the data you want to monitor. For example, /value: 1022.5/
- Iteratively replace literal characters with character classes and metacharacters to generalize the literal into a pattern. For example, the literal in the example above could be changed to: /value:\s\d\d\d\d\.\d/ to match any four digits, a decimal point, and one more digit.
- Consider that the pattern of the data you want to match may vary. Adjust your pattern to match expected or possible variations in the target data. Continuing with the example used above, the expression /value:\s\d\d\d\d\.\d/ might become /value:\s[\d]{1, 8}\.[\d]{1,2}/. This pattern allows for variation in the number of digits to the left of the decimal point and the number of digits to the right of the decimal point. It expects that there will be a decimal point. See the following sections for more information about the character classes used here.
- Consider that the literal string or pattern you want to match may appear more than once in the content. Identify unique content that precedes the content you want to match, and add regular expression patterns to ensure that the expression will match that unique content before it tries to match the content you are trying to monitor. In the example used here, the pattern may match the first of several entries that have a similar /value: numbers/ pattern. By adding a literal to the pattern that matches some static content that delimits the particular data can be used to be sure the match is made for the target data.

For example, if the data you want to match is preceded by the text Open Queries, this literal can be added to the pattern, along with a pattern for any intervening content: /Open Queries[\s\W]{1,5} value:\s[\d]{1, 8}\.[\d]{1,2}/.

The following sections describe many of the different elements and patterns that can be used for creating regular expressions for content matching.

Matching String Literals

Finding and matching an exact or literal string is the simplest form of pattern matching with regular expressions. In matching literals, regular expressions behave much as they do in search/replace in word processing applications. The example above matched the text Web site. The regular expression /Buy Now/ will succeed if the text returned to the monitor contains the characters Buy Now, including the space, in that order.

Note that regular expressions are, by default, **case-sensitive** and **literal**. This means that the content must match the expression in case and order, **including non-alphanumeric** characters. For example, a regular expression of /Website/, without any modifiers, will succeed only if the content contains the string Website exactly but will fail even if the content on the page is website, WEBSITE, or Web site. (In the last case the match fails because there is space between the two words but not in the regular expression.)

There are cases where you may want to literally match certain non-alphanumeric characters which are special "reserved" metacharacters used in regular expressions. Some of these metacharacters may conflict with important literals that you are trying to match with your regular expression. For example, the period or dot symbol (.), the asterisk (*), the dollar sign ($), and back slash (\) have special meanings within regular expressions. Because one of these characters may be a key part of a particular text pattern you are looking for, you must "escape" these characters in your regular expression so that the regular expression processing treats them as literal characters rather than interpreting them as special metacharacters. To force any character to be interpreted as a literal rather than a metacharacter, add a back slash in front of that character.

For example, if you wanted to find the string 4.99 on a Web page you might create a regular expression of /4.99/. While this will match the string 4.99, it would also match strings like 4599 and 4Q99 because of the special meaning of the period character. To have the regular expression interpret the period as a literal, escape the period with a forward slash as follows: /4\.99/. You can add the back slash escape character in front of any character to force the regular expression processing to interpret the character following the back slash as a literal. In general, use this syntax whenever you want to match any punctuation mark or other non-alphanumeric character.

Using Alternation

Alternation allows you to construct either/or matches where you know that one of two or more strings should appear in the content. The alternation character is the vertical pipe symbol: | . The vertical pipe is used to separate the alternate strings in the expression. For example, the regular expression /(e-mail|e-mail|contact us)/ will succeed if the content contains any one of the three strings separated by the vertical pipes. The parentheses are used here to delimit alternations. In this example, there are no patterns outside of the alternation that need to be matched. In contrast, a regular expression might be written as /(e-mail|e-mail|contact) us/. In this case, the match succeeds only when any of the three alternates enclosed in the parentheses is followed immediately by a single white space and the word us. This is more restrictive than the previous example, but also shows how the parentheses limit the alternation to the three words contained inside them. The match fails even if one or more of the alternates are found but the word “us” is not the next word.

Matching Patterns with Metacharacters

Often you will not know the exact text you need to match, or the text pattern may vary from one session or from one day to another. Regular expressions have a number of special metacharacters used to define patterns and match whole categories of characters. While matching literal alphanumeric characters seems trivial, part of the power of regular expressions is the ability to match non-alphanumeric characters as well. Because of this, it is important to keep in mind that your regular expressions need to account for the presence of non-alphanumeric characters in the content you are searching. This means that characters such as periods, commas, hyphens, quotation marks and even white spaces, need to be considered when constructing regular expressions.

Metacharacters Used in Regular Expressions

Metacharacter	Description
\s	Matches generic white space (that is, the Spacebar key). This metacharacter is particularly useful when combined with a quantifier to match varying numbers of white space positions that may occur between words that you are looking to match.
\S	Matches characters that are NOT white space. Note that the \S is capitalized versus the small \s used to match white space.
.	This is the period or dot character. Generally, it matches all characters . SiteScope considers the dot as a form of character class on its own and therefore it should not be included inside the square brackets of a character class.
\n	Matches the linefeed or newline character.
\r	Matches the carriage return character.
\w	Matches non-white space word characters, the same as what is matched by character class [A-Za-z0-9_]. It is important to note that the \w metacharacter matches the underscore character but not other punctuation marks such as hyphens, commas, periods, and so forth.

Metacharacter	Description
\W	Matches characters other than those matched by \w (lower case). This is particularly useful for matching punctuation marks and non-alphabetic characters such as ~!@#$%^&*()+={[}]:;and including the linefeed character, carriage return, and white space. It does not match the underscore character which is considered a word constituent matched by \w.
\d	Matches digits only. This is equivalent to the [0-9] character class.
\D	Matches non-numeric characters (what \d does not match) plus other characters. Similar to \W but also matches on alphabetic characters. In SiteScope this will generally match everything, including multiple lines, until it encounters a digit.
\b	Requires that the match have a word boundary (usually a white space) at the position indicated by the \b.
\B	Requires that the match NOT have a word boundary at the position indicated.

Defining Character Classes

An important and very useful regular expression construct is the character class. Character classes provide a set of characters that may be found in a particular position within a regular expression. Character classes may be used to define a range of characters to match a single position or, with the addition of a quantifier, may be used to universally match multiple characters and even complete lines of text.

Character classes are formed by enclosing any combination of characters and metacharacters in square brackets: [ ]. Character classes create an "any-or-all-of-these" group of characters that may be matched. Unlike literals and metacharacters outside character classes, the physical sequence of characters and metacharacters within a character class has no effect on the search or match sequence. For example, the class [ABC0123abc] matches the same content as [0123abcABC].

The hyphen is used to further streamline character classes to indicate a range of letters or numbers. For example, the class [0-9] includes all digits from zero to nine inclusive. The class [a-z] includes all lower case letters from a to z. You can also create more restrictive classes with the hyphen such as [e-tE-T] to match upper or lower case letters from E to T or [0-5] to match digits from zero to five only.

The caret character (^) can be used within a character class as a negation or to exclude certain characters from a content match.

Example Character Classes

Example	Description
[a-zA-Z]	This matches any alphabetic character, both upper case and lower case, from the letter a to the letter z. To match more than one character, append a quantifier after the character class as described below.
[0-9]	This matches any digit from 0 to 9. To match more than one digit, append a quantifier after the character class as described below.
[\w\s]	This matches any alphanumeric character and/or any white space.
[\w^_]	This matches any alphanumeric character, excluding the underscore.

Using Quantifiers

Another set of metacharacters used in regular expressions provides character counting options. This adds a great deal of power and flexibility in content matching. Quantifiers are appended after the metacharacters and character classes described above to specify against which positions the preceding match character or metacharacter should be matched. For example, in the regular expression /(contact|about)\s+us/, the metacharacter \s matches on a white space. The plus sign quantifier following the *s* means that there must be at least one white space between the words contact (or about) and us.

The following table describes the quantifiers available for use in regular expressions. The Quantifier applies to the single character immediately preceding it. When used with character classes, the quantifier is placed outside the closing square bracket of the character class. For example: [a-z]+ or [0-9]*.

Quantifier	Description
?	The question mark means the preceding character or character class may appear once but is optional and not required to appear in the position indicated.
*	The asterisk requires that any number of the preceding character or character class appear in the designated position. This includes zero or more matches. **Note:** Care must be used in combining this quantifier with the dot (.) metacharacter or a character class including the \W metacharacter, as these will likely "grab" more content than anticipated and cause the regular expression engine to use up all of the available CPU time on the SiteScope server.
+	The plus sign requires that the preceding character or character class appear at least once.
{*min*,*max*}	Using curly braces creates a quantifier range. The range enumerator digits are separated by commas. This construct requires that the preceding character or character class appear at least as many times as specified by the *min* enumerator up to but no more than the value of the *max* enumerator. The match succeeds as long as there are at least as many matches as specified by the *min* enumerator. However, the matching continues up to the number of times specified by the *max* enumerator or until no more matches are found.

Match content in SiteScope is run against the entire HTTP response, including the HTTP header, which is not normally viewable via the browser. The HTTP header usually contains several lines of text including words coupled with sequences of numbers. This may cause failure of some otherwise simple content matching on short sets of numbers and letters. To avoid this, identify a unique sequence of characters near the text you are trying to match and include them as literals, where applicable, in the regular expression.

Search Mode Modifiers

Regular expressions used in SiteScope may include optional modifiers outside of the slashes used to delimit the expression. Modifiers after the ending slash affect the way the matching will be performed. For example, regular expression of /website/i with the i search modifier added will make the match content search insensitive to upper and lower case letters. This would match either website, Website, WEBSite, or even WEBSITE.

With the exception of the i modifier, some metacharacters and character classes can override search mode modifiers. In particular, the dot (.) and the \W metacharacters can override the m and s modifiers, matching content across multiple lines despite the modifier.

Regular Expression Match Mode Modifiers

Mode Modifier	Description
/i	Ignore case mode. This makes the search insensitive to upper case and lower case letters. This is a useful option especially when searching for matches in the text content of Web pages.
/c	The matched pattern must NOT appear anywhere in content that is being searched. This is a "complement" match, returning an error if the pattern IS found, and succeeding if the pattern is NOT found.

Mode Modifier	Description
/m	Match across multiple lines WITHOUT ignoring intervening carriage returns and linefeeds. With this modifier you may still need to account for possible linefeeds and carriage returns with a character class such as [\w\W]* or [\s\S\n\r]*. The .* will NOT match carriage returns or linefeed characters with this modifier.
/s	Consider the content as being on a single line, ignoring intervening carriage returns and linefeed characters. With this modifier both the [\w\W]* character class and the .* pattern will match across linefeeds and carriage returns.

More than one modifier can be added by concatenating them together after the closing slash of the regular expression. For example: /matchpattern/ic combines both the i and c modifiers.

Retaining Content Match Values

Some monitors, like the URL Monitor and URL Sequence Monitor, have a content match value that is logged and can be used to set error status thresholds. Another purpose of the parentheses /(match pattern)/ used in regular expression syntax is to determine which text is retained for the Content Match Value. You use this feature to use content match values directly as thresholds for determining what a URL monitor's or URL Sequence monitor's error threshold will be.

For example, if the content match expression was

/Copyright (\d*)/

and the content returned to the monitor by the URL request included the string:

... Copyright 2003 by Mercury Interactive

then the match is made and the retained content match value would be:

2003

Under the error-if option at the bottom of the monitor set up page, you could then change the error-if condition from the default of status != 200 to content match, then specify the relational operator as !=, and then specify the value 1998. This will set the error threshold for this monitor so that whenever the year in the string Copyright *dddd* is something other than 1998, the monitor will report an error. This mechanism could be used to watch for unauthorized content changes on Web pages.

Checking a Web page for links to other URLs can be an important part of constructing URL Sequence Monitors. The following regular expression can be used to match the URL text of a link on a Web page:

/a href="?([:\/\w\s\d\.]*)"?/i

This expression matches the href="protocol://path/URLname.htm" for many URLs. The question mark modifiers allow the quotation marks around the HREF= attribute to be optional. The i modifier allows the match pattern to be case-insensitive.

Retained or remembered values from content matches can be referenced and used as input for subsequent steps in a URL Sequence Monitor. See the Match Content section of the URL Sequence Monitor for the syntax used for Retaining and Passing Values Between Sequence Steps.

SiteScope Date Variables

SiteScope uses specially defined variables to create expressions that match the current date or time. These variables can be used in content match fields to find date-coded content. The General Date Variables are useful for matching portions of various date formats. The Language/Country Specific Date Variables allow you to automatically extend the language used for month names and weekday names to specific countries, based on ISO codes.

General Date Variables

The following table lists the general variables:

Variable	Range of Values
$hour$	0 - 23
$minute$	0 - 59
$month$	1 - 12
day	1 - 31
$year$	1000 - 9999
$shortYear$	00 - 99
$weekdayName$	Sun - Sat
$fullWeekdayName$	Sunday - Saturday
$0hour$	00 - 23
$0minute$	00 - 59
$0day$	01 - 31 (two-digit day format)
$0month$	01 - 12 (two-digit month format)
$monthName$	Jan - Dec (three-letter month format in English)
$fullMonthName$	January - December
$ticks$	milliseconds since midnight, January 1, 1970

For example, if the content match search expression was defined as:

```
/Updated on $0month$\/$0day$\/$shortYear$/
```

and the content returned by the request includes the string:

```
Updated on 06/01/98
```

then the expression would match when the monitor is run on June 1, 1998. The match fails if the content returned does not contain a string matching the current system date or if the date format is different than the format specified.

If you want the time to be before or after the current time, you can add a $offsetMinutes=*mmmm*$ to the expression, and this will offset the current time by *mmmm* minutes (negative numbers are allowed for going backwards in time) before doing the substitutions.

For example, if the current day is June 1, 1998, and the search expression is:

```
/$offsetMinutes=1440$Updated on $0month$\/$0day$\/$shortYear$/
```

the content string that would match would be:

```
Updated on 06/02/98
```

Note that the date is one day ahead of the system date.

Language/Country Specific Date Variables

The following table lists the SiteScope special variables for use with international day and month name matching. The characters LL and CC are placeholders for two-letter ISO 639 language code characters and two-letter ISO 3166 country code characters (see the notes below the table for more details).

Variable	Range of Values
$weekdayName_LL_CC$	Abbreviated weekday names for the language (LL) and country (CC) specified (see notes below).
$fullWeekdayName_LL_CC$	Full weekday names for the language (LL) and country (CC) specified.
$monthName_LL_CC$	Abbreviated month names for the language (LL) and country (CC) specified.
$fullMonthName_LL_CC$	Full month names for the language (LL) and country (CC) specified.

CC - an uppercase 2-character ISO-3166 country code. Examples are: DE for Germany, FR for France, CN for China, JP for Japan, BR for Brazil. You can find a full list of these codes at a number of Internet sites, such as: http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html

LL - a lowercase 2-character ISO-639 language code. Examples are: de for German, fr for French, zh for Chinese, ja for Japanese, pt for Portuguese. You can find a full list of these codes at a number of Internet sites, such as: http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt or http://www.dsv.su.se/~jpalme/ietf/language-codes.html.

For example, if the content match expression was defined as:

```
/$fullWeekdayName_fr_FR$/i
```

and the content returned by the request includes the string:

```
mercredi
```

then this expression would match when the monitor was run on Wednesday.

If you are not concerned with the country-specific language variations, it is possible to use any of the above variables without including the country code. For example:

```
/$fullWeekdayName_fr$/
```

could be used to match the same content as /$fullWeekdayName_fr_FR$/.

Special Substitution for Monitor URL or File Path

SiteScope Date Variables are useful for matching content as part of a regular expression. The date variables can also be used as a special substitution to dynamically create URLs or file path names for specific monitors. This is useful for monitoring date-coded files and directories where the URL or file path name is updated automatically based on system date information. SiteScope is an example of an application that creates date-coded log files. The log file names include some form of the year, month, and day as part of the file name, such as File2001_05_01.log, where the year, month, and date are included.

Based on this example, a new file is created each day. Monitoring the creation, size, or content of the current days file would normally require the file path name or URL of the monitor to be manually changed each day. Using the SiteScope date variables and special substitution, SiteScope can automatically update the file path to the current day's log file. By knowing the pattern used in naming the files, you can construct a special substitution string similar to a regular expression that will substitute portions of the system date properties into the file path or URL.

For example if the absolute file path to the current day's log file in a file monitor is:

D:/Production/Webapps/Logs/File2001_05_01.log

the log file for the following day would be:

D:/Production/Webapps/Logs/File2001_05_02.log

You can construct a special substitution expression to automatically update the file path used by the monitor, with the following syntax:

s/D:\/Production\/Webapps\/Logs\/File$year$_$0month$_$0date$.log/

The substitution requires that the expression start with a lower-case s and that the expression is enclosed by forward slashes /.../. Forward slashes that are part of the file path must be escaped by adding the back slash (\) character as shown. The SiteScope date variables are separated by the underscore character literals. SiteScope checks the system time properties each time the monitor runs and substitutes with applicable values into the file path or URL before accessing the file.

SiteScope monitor types that support the special substitution are:

- eBusiness chain
- File Monitor
- Log Monitor
- URL Monitor
- URL Sequence Monitor
- Web server monitor

While the special substitution syntax is similar in syntax to the substitution syntax used in regular expressions, they are not the same. While all of the SiteScope date variables can be used in match content regular expressions, the special substitution discussed here can not be used as part of a match content expression.

Examples for Log File Monitoring

SiteScope's Log File Monitor and File Monitor check for entries in files created by other applications. These files may be data files created by a third-party application or they may be logs created by a custom system specially designed for your environment. Where the logs or files are written with a known, predictable format, SiteScope can be configured to regularly check the files for new entries and match on specific content strings. The following are several examples of log file entries and simple regular expression patterns that can be used to check the entries. You can use these examples or modify them to work with a specific case.

Note: All regular expressions must be entered on a single line in SiteScope. Some of the examples below may break across more than one line to fit on this page.

Matching Comma-Separated Values

The following is an example of log file entries that are comma-separated strings of digits and letters:

new,open,changed,12,alerts
new,open,changed,13,alerts
new,open,changed,13,alerts
new,open,changed,14,alerts

A regular expression to match on log file entries that are comma-separated strings of digits and letters.

```
/([\w\d]+,[\w\d]+,[\w\d]+,[\w\d]+)[\n\r]?/
```

Note: If the file entries include punctuation marks such as an underscore or a colon, add that character explicitly to the [\w\d] class pattern. For example, to include a colon character, change each of the [\w\d] patterns to [\w\d:].

Matching Whitespace Separated Values

The following is an example of log file entries that are a sequence of strings and digits separated by spaces:

```
requests 12 succeeded 12 failed
requests 12 succeeded 12 failed
requests 11 succeeded 11 failed
requests 12 succeeded 12 failed
requests 10 succeeded 10 failed
```

The following is a regular expression to match on log file entries that are a sequence of strings and digits separated by spaces.

```
/([\w\d]+\s+[\w\d]+\s+[\w\d]+\s+[\w\d]+\s+[\w\d]+)[\n\r]?/
```

Note: The use of the + character forces the match to include the number of sequences per line included in the match pattern: in this example, five word or number sequences per line of the log file. If the sequences include punctuation marks such as an underscore or colon, add that character explicitly to the [\w\d] class pattern. For example, to include a colon character, change each of the [\w\d] patterns to [\w\d:].

Matching and Retaining the Numbers in a Line of Text and Numbers

The following is an example of log file entries that are comma separated strings that combine digits and letters:

request handle number 12.56, series 17.5, sequence reported 97.45, 15.95 and 19.51
request handle number 15.96, series 27.5, sequence reported 107.45, 25.95 and 19.52
request handle number 11.06, series 36.5, system codes 9.45, 35.95 and 19.53
log reference number 12.30, series 17.5, channel reset values 100.45, 45.95 and 19.54

The following is a regular expression to match on log file entries that are comma-separated strings that combine digits and letters and retain the decimal numeric data:

```
/[,\w\s]+(\d+\.\d+)[,\w\s]+(\d+\.\d+)[,\w\s]+(\d+\.\d+)[,\w\s]+(\d+\.\d+)[,\w\s]+(\d+\.\
d+)[\n\r]?/
```

Note: If the file entries include punctuation marks such as an underscore or colon, add that character explicitly to the [,\w\s] class pattern. For example, to include a colon character that appears embedded in the text sequences, change each of the [,\w\s] patterns to [,:\w\s].

Matching Integers and Floating-point Numbers (Positive or Negative)

The following is an example of log file entries that are a sequence of integers and floating point numbers that may be negative or positive:

12.1987 -71 -199.1 145 -1.00716
13.2987 -72 -199.2 245 -1.00726
14.3987 -73 -199.3 345 -1.00736
15.4987 -74 -199.4 445 -1.00746

The following is a regular expression to match on log file entries that are a sequence of 5 integers and floating point numbers that may be negative or positive. The numbers in each entry must be separated by one or more spaces.

```
/(-?\d+\.?\d{0,})[\s]+(-?\d+\.?\d{0,})[\s]+(-?\d+\.?\d{0,})[\s]+(-?\d+\.?\d{0,})[\s]+(-
?\d+\.?\d{0,})[\n\r]?/
```

Matching Date and Time Coded Log Entries

Many log files include some form of date and time data with each entry. The following is an example of log file entries that include date and time information together with string data separated by commas:

```
20/04/2003 14:29:22,ERROR,request failed
20/04/2003 14:31:09,INFO,system check complete
20/04/2003 14:35:46,INFO,new record created
```

The following is a regular expression to match on log file entries that are date- and time-coded followed by comma-separated strings of letters and digits. This example uses the SiteScope date variables to match only on entries that were created on the same day, month, and year as indicated by the system clock of the server where SiteScope is running.

```
/$0day$\/$0month$\/$year$\s+\d+:\d+:\d+,[\w\d]+,[\w\d]+/
```

The following example uses the SiteScope date variables to match on a more restricted set of entries that were created on the same day, month, year, and within the same hour as indicated by the system clock of the server on which SiteScope is running.

```
/$0day$\/$0month$\/$year$\s+$0hour$:\d+:\d+,[\w\d]+,[\w\d]+)/
```

Some Pitfalls in Working with Regular Expressions

The most significant problem that has been seen with regular expressions in SiteScope is the use of the .* construct. This match construct presents a very large number of possible matches on any page of content. The use of the .* construct is known to cause the regular expression-matching engine used by SiteScope to take over all available CPU cycles on the SiteScope server. If this occurs, SiteScope will not be able to function and will have to be restarted each time the monitor with the offending regular expression is run, until the expression has been corrected.

Note that regular expression matching is run against the entire text content returned to the SiteScope monitor request. As mentioned above, this includes HTTP headers that are normally not viewable in the browser window (for example, not visible using the **View->Source** option). This also means that you need to account for other information that may not be displayed in the browser view. This includes text in META tags used by Internet search engines as well as client side-scripts.

In the case of URLs that contain client side-scripts, such as Javascript, the text matching is done against the code lines of the script and not against the browser's output from the script. This means that if the script dynamically writes or replaces text on the Web page with values calculated by the script, it may not be possible to match this content with regular expressions. If the script is only changing text, you may be able to match the corresponding text strings that appear in the script code. A further pitfall would be that you are trying to check that a certain condition was met in the browser but the matching text string appears in the script content regardless of any user action.

Also note that a regular expression match succeeds as soon as the minimum match requested is satisfied. After a match is made, no further matching is performed. Therefore, regular expressions are not well suited to count the number of occurrences of a repeating text pattern. For example, if you want to check a Web page with a catalog list of items and each item has a link next to it saying Buy Now! and you want to make sure that at least five items are listed, a regular expression of /Buy Now!/ would succeed in matching only the first Buy Now!. Likewise, if your regular expression searches the word catalog on the main browser screen, the match may succeed if the word appears as a META tag in the HTML header section or if it appears as a hyperlink in a site navigation menu that appears in the content before the occurrence you intend to match.

Another pitfall is forgetting to account for non-alphanumeric content. As mentioned above, regular expressions need to be written to account for all of the characters that are and may be present. This includes white space, linefeed, and carriage returns. This is not normally a problem when matching a single-word literal. It can be a challenge when you need to create a match of several words separated by unknown amounts of white space and other non-alphanumeric characters and possibly span more than one line. The [\s\n\r]+ character class can be useful between words used in the expression. Always check the format of the content you are trying to match to look for patterns and special characters, such as periods, commas, and hyphens, that may cause a seemingly simple match to fail.

The use of "greedy" metacharacters can lead to frustration. In some cases, overly generous quantifiers combined with the . or \W metacharacters can grab content that you were intending to match with a literal string elsewhere in your regular expression resulting in a match failure. For example, the following might be used to match the URL content of the hyperlink anchor reference: /a href="([\W\w\s]*)"/. When the monitor performs the check for this regular expression, however, the match grabs the first occurrence of the pattern /a href="... and continues matching multiple lines of text up to the last quotation mark found on the page. Without some other unique ending delimiter, the [\W\w\s]* class and quantifier combination is too "greedy." A more successful syntax that narrows the class of expected characters would be: /a href="?([:\/\w\s\d\.]*)"?/

The following are some examples of syntax for use in regular expressions:

Example Expression	Description
/CUSTID\s?=\s?([A-Z0-9]{20,48})/	This example matches an ID string that is made of 20 or more digits and upper-case letters with no spaces or other non-alphanumeric characters. The \s? construct allows there to be a white space on either side of the equals sign. Using the parentheses around the character class instructs SiteScope to retain this value (up to the maximum of 48 characters) as a content match value and the matched value will be displayed in the monitor detail status column.
/a href="?([:\/\w\s\d\.]*)"?/i	This example matches the URL string in an HTML hyperlink. The "? construct makes a quotation mark on either end of the URL string optional. Using the parentheses instructs SiteScope to retain this value as a content match value and the value will be displayed in the monitor status. The i modifier tells the search to treat upper- and lower-case letters equally.
/"[^"]*"/	This example matches text sequences that are contained between quotation marks. Note the use of the negation caret (^) to define a character class of all characters other than the quotation mark.

As with programming and scripting languages, there is almost always more than one way to construct a regular expression to accomplish a particular match. There is not one right way to build regular expressions. You should plan to test and modify regular expressions as necessary until you get the results you need.

For an in-depth discussion of Perl regular expressions, consult a book about Perl programming, or find a Perl tutorial on the Web.

2

Monitoring XML Documents

SiteScope's content matching capabilities is an important feature in monitoring networked information systems and content. For SiteScope monitors that provide content matching, the basic content matching is available through the use of Perl regular expressions. SiteScope also includes the capability of matching document content by traversing XML documents. For example, you can include an XML match content string using the URL Monitor and Web Services Monitor to match an XML element name, an attribute of an XML element, or the content of an element. You can use this to check for content in XML based Web pages, SOAP or XML-RPC documents, and even WML pages served to WAP-enabled devices.

This chapter describes:	On page:
Understanding Content Matching for XML Documents	25
Using XML Content Match Values in Monitor Configurations	27

Understanding Content Matching for XML Documents

The syntax of XML match content strings reflects the hierarchal structure of the XML document. Match content strings that start with "xml" are recognized as element names within an XML document. The element names are added, separated by periods, in the order of their relationship to the root element. For example, in the document weather.xml the root element is <weather>. This element includes child elements named <area>, <skies>, <wind>, <forecast>, and so forth. To access the content of these XML elements or their attributes, you would use a syntax like xml.weather.area.

To check that specific content or value is present, add an equals sign after the element name whose content you are testing and then add the value of the content. If there are multiple instances of an element name in the document, you can check a particular instance of that element by adding the number indicating the order of the element in the document in square brackets (see the example in the table below). You can also test for multiple elements or values by separating individual search strings with commas. The table below gives several examples of the syntax used to match content in XML documents.

Example Match Content	Description
xml.weather.temperature	Succeeds if any <weather> node in the document contains **one or more** <temperature> elements. The content of the <temperature> element(s) is/are returned by the monitor. If no <temperature> element is found within the <weather> node, an error is returned.
xml.weather.temperature=20	Succeeds if any <weather> node in the document contains **one or more** <temperature> elements where the content of the <temperature> element equals 20. The content of the <temperature> element is **NOT** returned by the monitor if the match is found. An error is returned if no <temperature> element is found within the <weather> node or if no <temperature> element contains the value 20.
xml.weather.forecast.[confidence]	Succeeds if any <weather> node in the document contains a <forecast> element that has an **attribute** called confidence. The value of the confidence attribute is returned by the monitor if the match is found. An error is returned if no <forecast> element is found within the <weather> node or if no confidence attribute is found.

Example Match Content	Description
xml.weather.forecast[3].[confidence]=50	Succeeds if any <weather> node in the document contains three or more <forecast> elements where the third <forecast> element has a confidence **attribute** with a value of 50. An error is returned if the <weather> node has fewer than three <forecast> elements or if the value of the confidence attribute is not equal to 50.
xml.weather.temperature=20, xml.weather.skies=rain	Succeeds if any <weather> node in the document contains **one or more** <temperature> elements where the content of the <temperature> element equals 20 **AND** if any <weather> node contains **one or more** <skies> elements where the content of the <skies> element equals rain. Returns an error if either of the matches fails.
xml.wml.card.p.table.tr.td.anchor=Home Page	Checks the content of <anchor> elements in the designated path of a WML document. Succeeds if any <card> node containing table cells with **one or more** <anchor> elements where the content of any of the <anchor> elements equals "Home Page."

Using XML Content Match Values in Monitor Configurations

Monitors like the URL Monitor have a content match value that is logged to the SiteScope monitor data log and can also be used to set error and warning status thresholds for the monitor. The values of the XML names are saved as the content match values for the monitor.

For example, if the match content expression was xml.weather.temperature and the document was the contents of the file weather.xml, then the content match value would be 46.

You can then set the error, warning, and good status thresholds in the Advanced Options section for the monitor to compare your specific thresholds to the value returned by the content match.

For example, if you were monitoring temperature values and wanted to be alerted when the temperature value dropped below 72 degrees, you could set the monitor status thresholds as follows:

Error if	content match < <= 72
Warning if	content match == <= 72
Good if	content match >= > 72

With this configuration, the monitor would check the content of the temperature element and then compare it to the error and warning thresholds. In the example above, the status of the monitor would be an **error** because the temperature value is 46, which is less than 72.

3

Setting Up and Using Rolling Baselines

SiteScope includes options that allow you to monitor variations in response times and performance in the infrastructure. This allows a monitor to retain previous performance metrics to establish expected performance ranges. The activity involved in determining the expected performance characteristics is called baselining.

This chapter describes:	On page:
About Rolling Baselines	29
Enabling Rolling Baselines	31
Disabling Rolling Baselines	32

About Rolling Baselines

Consistent response times and service levels have become increasingly important for Web systems. It is the nature of the Internet that performance can vary significantly over short periods of time. These variations may represent random changes in the number of client requests, or in bandwidth usage from file downloads, or they may indicate a systemic problem in the infrastructure. To better document service level and response performance, it is sometimes useful to consider a set of previous performance metrics from a monitor to establish expected performance ranges. When the monitor's performance values vary beyond a certain range, the monitor can signal an error or warning. This monitoring method is called baselining.

Note: SiteScope includes two baselining options. One is the Rolling Baseline discussed here, which is currently applicable only to the URL and NT Performance Counter Monitors. The other option is the Static Performance Baseline applicable to most monitor types.

In contrast to the Static Performance Baseline, which is calculated one time over an interval measured in days, the Rolling Baseline is calculated for an interval measured in monitor runs and the baseline value is updated after each monitor run. The URL Monitor and NT Performance Counter Monitors are distinguished from other monitors by the addition the **Baseline Interval** configuration parameter and rolling baseline-related criteria in the error, warning, and good threshold settings. The steps for enabling the Rolling Baseline are described below.

Baselining a URL Monitor and NT Performance Counter Monitor involves using monitor performance measurements over a given interval to calculate and update a performance average and a standard deviation. The average establishes the baseline and when the performance reported by the monitor exceeds that baseline, either by a multiple of the standard deviation or a percentage difference from the baseline, the monitor may be considered in error.

The acceptable performance range of a monitor is determined by how far the current performance is from the baseline (deviation). This range of the variance is based on percentage deviation from the baseline average. For example, if a baseline for a monitor's round-trip time is currently 2 seconds and the percentage deviation is set to 30%, the monitor will maintain a good performance status as long as it’s round-trip time does not exceed 2.6 seconds.

After a rolling baseline has been enabled and calculated, the text: **(rolling baseline)** is added to the status string of the monitor. This can be viewed in the group detail page to which the monitor belongs. It can also be used as a filter criterion for the **Match Status** box in the Monitor Browser.

Rolling Baseline values are maintained for each monitor across SiteScope restarts.

Enabling Rolling Baselines

There are two main actions that need to be done to setup and use SiteScope Rolling Baselines:

- Enable rolling baseline calculation for selected monitors.
- Edit the configuration settings for the baselined monitors to make use of the baseline data.

After you have configured monitors to use a rolling baseline, the monitors will automatically begin reporting status relative to the baseline data after they have run enough times to accumulate the required amount of data for the baseline calculation. Use the following steps to enable Rolling Baselines:

To enable Rolling Baselines for URL monitor or NT Performance Counter monitors:

1. Create or select a URL Monitor or NT Performance Counter Monitor to be baselined.
2. In the Add Monitor page or the Edit Monitor page, open the Advanced Setting section to find the **Baseline Interval** box.
3. Enter an interval value in the **Baseline Interval** box. The Baseline Interval is the number of monitor runs that should be used to calculate the Rolling Baseline. Valid entries for this setting are positive integers. For example, entering a value of 20 will instruct SiteScope to use an interval of 20 previous monitor runs to calculate the rolling baseline for this monitor.
4. In the Threshold Settings section, edit the error, warning, and good threshold ranges for the monitor to be calculated relative to the baseline data. Use the drop-down selection menu to the right of these settings to select the parameter. This will usually be a Deviation Percentage setting. Select the comparison operator and the comparison value for each threshold.
5. Click **OK** to save the changes and enable the rolling baseline.

   After enabling the rolling baseline, allow the monitor(s) to run for a period at least as long as the Baseline Interval you have entered. For example, if you entered a baseline interval of 10, the monitor needs to run ten times to accumulate enough run data to calculate the baseline.

After accumulating monitoring data for the indicated baseline interval period, the selected monitors will report their status relative to the baseline. The baseline value will be recalculated and updated for each subsequent monitor run.

Disabling Rolling Baselines

You disable the use of Rolling Baselines by editing the applicable monitors. This involves clearing the **Baseline Interval** value and changing the **Error** if, **Warning** if, and **Good** if settings to use non-baseline-related parameters.

To disable Rolling Baseline calculation for a monitor:

1. Click on the group container in the monitor tree to navigate to the group that contains the monitor(s) that have been enabled for rolling baseline calculation.
2. Click to **Edit** the applicable monitor. The Edit Monitor Properties page for that monitor is displayed.
3. Open the **Advanced Settings** section of the monitor setup screen and clear the value in the **Baseline Interval** text box.
4. In the Threshold Settings section, edit the error, warning, and good threshold ranges for the monitor to be set to something other than baseline-related data.
5. Click **OK** button to save the changes.

4

Setting Up and Using Static Performance Baselines

It is often desirable to use past performance metrics from a monitor to establish acceptable or expected performance ranges. When the monitor's performance exceeds that range by some value, the monitor can signal an error or warning. The activity involved in determining the expected performance characteristics is called baselining.

This chapter describes:	On page:
About Static Performance Baselines	34
Understanding Performance Baselining	35
Enabling Static Performance Baselining for Monitors	36
Setting Monitor Thresholds Using Baselines	37
Monitor Properties for Static Baselines	39

Note: Static performance baselines can be configured using only the SiteScope Classic interface in SiteScope 8.0. The steps described in this section apply to the Classic interface.

About Static Performance Baselines

The Static Performance Baseline feature calculates a one-time, or fixed, baseline value based on monitor readings for a time interval you specify. The units of the time interval are expressed in days. After enough monitor data has been accumulated to calculate the baseline, subsequent monitor results are compared to that baseline value. The baseline value remains constant or static.

Note: SiteScope includes two baselining options. One is the Static Performance Baseline discussed here. The other option is the Rolling Baseline available with the URL Monitor and NT Performance Counter Monitor discussed in Chapter 3.

Monitors selected for static performance baselining are distinguished from other monitors of the same type by the addition of new, baseline-related criteria in the error, warning, and good thresholds for the monitors that have been configured for baselining.

After a baseline has been calculated, the text **(static baseline, *refvalue*)** is also added to the status string of the monitor. The refvalue displayed is a reference value of the baseline mean value. The meaning of the refvalue will vary depending on the type of monitor (see the table of monitor types below). The status string can be viewed in the group detail page to which the monitor belongs. It can also be used as a filter criterion for the Match Status box in the Monitor Browser. The specific values used for the baseline are added to the Advanced Options section of the individual monitor setup page as described below. You use the **Edit** link associated with the monitor to view this data.

Understanding Performance Baselining

Baselining a monitor involves using the historical performance measurements to calculate a performance average (that is, a baseline) and a standard deviation. The average establishes the baseline and when the performance the monitor exceeds that baseline, either by a multiple of the standard deviation or a percentage difference from the baseline, the monitor may be considered in error.

The acceptable performance range of a monitor is determined by how far (variance) the current performance is from the baseline. This range of the variance may be based on multiples of the standard deviation or on multiples of a percentage difference. For example, suppose a baseline for a monitor's round-trip time is 3 seconds and the standard deviation is 1.5 seconds. If the range for variance is selected as one times the standard deviation, then the monitor will remain in good status as long as it's round-trip time does not exceed 4.5 seconds. This can be represented as a monitor error condition formula of:

if(round-trip > (baseline + (multiplier ***** stddev)) **then** error = true

Substituting the numbers from the example above gives:

if(4.5 > (3 + (1 ***** 1.5)) **then** error = true

In another example, the baseline for a monitor's round-trip time is 5 seconds. The range for variance is selected as 50 percent of the baseline. The monitor will remain in good status as long as it's round-trip time does not exceed 7.5 seconds. This can be represented as a monitor error condition formula of:

if(round-trip > baseline + (baseline ***** multiplier)) **then** error = true

Substituting the numbers from this second example above gives:

if(7.5 > (5.0 + (5.0 ***** 0.50)) **then** error = true

Enabling Static Performance Baselining for Monitors

The following steps are required to setup and use SiteScope Fixed Performance Baselining:

1. Enable static baselining for selected monitors.
2. Allow the monitors to accumulate data for the baseline period.
3. Edit the configuration settings for the baselined monitors to make use of the baseline data.

   A baseline may be set, reset, or extended whenever necessary to update performance expectations based on increased operational experience or to document the effects of infrastructure changes. Once a static performance baseline has been calculated, it can be updated only by selecting the same monitor(s) and repeating the baseline setup steps.

   Use the following steps to enable baselining of monitors with this feature. These steps assume that you have already created one or more monitors that you want to baseline.

   To enable static performance baselines for monitors:

1. Click the **Manage Monitors/Groups** link on the SiteScope navigation menu. The Manage Monitors and Groups page is displayed.
2. Use the expandable tree menu to select one or more monitors to be baselined. Click the check box to the left of the monitor name in the group and monitor tree menu.
3. Scroll to the bottom section of the Manage Monitors and Groups page and click the **Baseline** button. The Baseline Monitors page is displayed.
4. The monitors you selected for baselining are displayed in the Monitors table list. Enter the baseline data interval you want to use for these monitors in the **Compute a fixed performance baseline using data from the last N days** text box. The interval should be a positive integer representing a number of days.
5. Click the **Baseline** button to enable baselining for the selected monitors.

After you have selected and enabled monitors for baselining, you should allow the monitors to run for a period at least as long as the baseline period you want to use. For example, if you set the baseline interval as 2 days, you can edit the configuration settings for the monitors for which you have enabled baselining to make use of the baseline data.

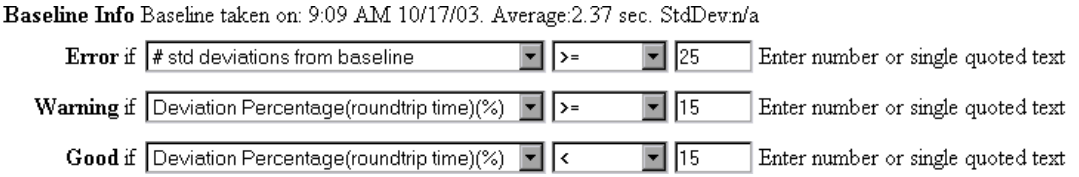
Setting Monitor Thresholds Using Baselines

Once the baseline has been set up for monitors, each monitor must be individually edited to set the error threshold range from the Edit Monitor page. When a baselined monitor is edited, an area on the page named **Baseline Info** is displayed that shows the date the baseline was taken, the baseline value (that is, the average) and the calculated standard deviation. Also, each baselined monitor then includes the following two thresholds in the Error if, Warning if, and Good if drop-down lists:

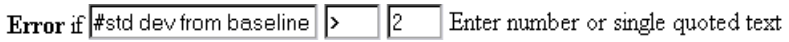
- # std dev from baseline
- % difference from baseline

Examples of Threshold Settings

The following is an example of the threshold setting section on the Edit Monitor page for a URL monitor for which a static baseline has been created. The Baseline Info is displayed above the **Error** if box.

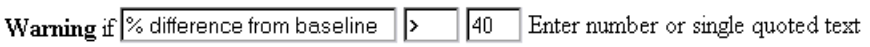


To set a monitor error threshold based on standard deviation from the baseline, select # std dev from the baseline option for the error, warning or good statuses. Then set a comparison operator such as > or >= and the number of standard deviations such 1 or 2. As a result, the threshold would be set as in the following example:



This indicates that an error is any monitor value greater than two standard deviations from the baseline.

When using percent difference from baseline, you select a % difference from baseline option from the error, warning or good if drop-down list. Then set a comparator operator such as < or >= and then the number of standard deviations such a 40. As a result the threshold would be set as in the following example:



which indicates that a warning status is any value greater than 40% of the baseline above the current baseline.

➤ After accumulating monitoring data for the baseline period, edit the monitors that are being baselined to set thresholds to use the baseline data. Use the following steps to edit the configuration settings for the baselined monitors to make use of the baseline data.

To configure monitors to use baseline data for thresholds:

1. Click on the group name in the SiteScope main page or by using the Browse Monitors page, to navigate to the group that contains the monitor(s) that have been enabled for static baselining.
2. Click the **Edit** link for the applicable monitor in the group detail page. The Edit Monitor page for that monitor is displayed.
3. Scroll down to the bottom portion of the page to the Error if and Warning if entries. Set the error and warning threshold ranges to be calculated relative to the baseline data.
4. Click **Update** to save the monitor configuration changes.

Monitor Properties for Static Baselines

As noted above, each baselined monitor will include the number of standard deviations from the baseline and the percent difference from the baseline as two available default threshold settings. Other performance properties available for baselined monitors depend on the monitor type. The following table shows the SiteScope monitor type and the corresponding baseline properties available for that type:

Monitor Type	Properties
Apache	CPU load
ASP	errors per second
Asset	round-trip time **Note:** This monitor appears only if there is an assets directory below the installed SiteScope directory.
Check Point	rejected
Cold Fusion	page hits per second
CPU	% cpu used
Database	round-trip time
Directory	total file sizes
Disk	% full
DNS	round-trip time
File	file age
FTP	round-trip time
IIS	bytes sent per second
IPlanet	bytes transferred
LDAP	round-trip time
Link	number of link errors
Log File	matches per minute

Monitor Type	Properties
Mail	round-trip time
Memory	% full
Network	bytes sent per second
News	round-trip time
Ping	round-trip time
Port	round-trip time
Radius	round-trip time
RTSP	bytes downloaded
Script	round-trip time
Service	status
SilverStream	hits per second
SNMP	OID value
Telnet	round-trip time
URL	round-trip time
URL Sequence	round-trip time
Web Service	hits per minute
Windows Performance Counter	first counter
Windows Dialup	total time
Windows Event Log	match count

5

SiteScope Monitoring via Secure Shell (SSH)

As network security is increasingly important, SiteScope supports a number of security capabilities. One of these is support for remote server monitoring using Secure Shell (SSH) connections. You can use SSH to connect to a server and automatically send a command, so that the server will run that command and then disconnect. This is useful for creating automated processing and scripting.

This chapter describes:	On page:
SiteScope and SSH	41
Configuring Remote UNIX Servers for SSH Monitoring	44
Configuring Remote Windows Servers for SSH Monitoring	45
SiteScope SSH Client Connection Options	58

SiteScope and SSH

Secure Shell (SSH), sometimes known as Secure Socket Shell, is a UNIX-based command interface and protocol for securely accessing a remote computer. It is widely used by network administrators to remotely control Web and other kinds of servers. SSH commands are encrypted and secure in several ways. Both ends of the client/server connection are authenticated using a digital certificate, and passwords are protected by encryption. Secure Shell client machines make requests of SSH daemons or servers on remote machines.

Monitoring with SiteScope over SSH has the following basic requirements:

1. The servers that you want to have monitored by SiteScope using SSH need to have a SSH daemon (or server) installed and active.
2. The machine on which SiteScope is running needs to be configured with an SSH client.

   Possibilities and issues involved in using SSH for SiteScope Monitoring include:

   - Beginning with the 7.8.1.1 release of SiteScope, there are two SSH client options for use on the server or machine on which SiteScope is running. SiteScope now includes a SSH client written in Java and native to the SiteScope application code. This client eases the setup of SSH connections and generally uses fewer system resources than external SSH clients.
   - SiteScope for Windows also ships with a copy of the PuTTY SSH client and utilities. The PuTTY SSH client, plink.exe, has been used to enable SSH connectivity for SiteScope for Windows prior to the 7.8.1.1 release. SiteScope for Solaris and Redhat Linux make use of the SSH utilities normally bundled with those operating systems or available for download.

   The following table outlines the SSH connectivity options currently supported with SiteScope. See the notes below the table for important information about configuring and managing SSH connectivity.

SiteScope Platform and Client Options	Monitored Server Platform and Daemon
Windows PuTTY SSH client (included with SiteScope) or SiteScope integrated Java SSH Client	**UNIX / Linux** SSH host daemon (sshd - either proprietary or OpenSSH)

SiteScope Platform and Client Options	Monitored Server Platform and Daemon
UNIX / Linux SSH client (/usr/local/bin/ssh or usr/bin/ssh) or SiteScope integrated Java SSH Client	**UNIX / Linux** SSH host daemon (sshd - either SunSSH, proprietary or OpenSSH)
Windows PuTTY SSH client (included with SiteScope) or SiteScope integrated Java SSH Client	**Windows** 1. SSH server (Cygwin OpenSSH, F-Secure, or OpenSSH for Windows 2. RemoteNTSSH package (included with SiteScope), to be installed into the appropriate directory on the remote server)

Guidelines and Limitations

- There are two different versions of the SSH protocol: version 1 and version 2. Version 1 and version 2 are different protocols and are not compatible with each other. This means that the SSH clients and SSH hosts must be configured to use the same protocol version between them in order to communicate. In many cases, SSH version 1 (SSH1) is the default version used. Some security vulnerabilities have been found in SSH version 1. Also, the SSH1 protocol is not being developed anymore and SSH2 is considered the current standard. We recommend using SSH version 2 (SSH2) for all SSH connections.
- The release version number of the SSH utilities and libraries you have installed must not be confused with the version of the SSH protocol that you want to be using. For example, OpenSSH release 3.5 supports both SSH1 and SSH2 protocols. The release version 3.5 does not mean that the libraries use an SSH version 3.5 protocol. You must configure the OpenSSH software to use either SSH1 or SSH2.
- If you have set up SiteScope remote monitoring using SSH connections and then make configuration changes or upgrades to the SSH daemon or server software deployed on remote servers in the environment, it may be necessary to reconfigure the SSH connectivity between the machine on which SiteScope is running and the remote servers that are being monitored.

➤ The availability of the Integrated Java SSH Client is indicated via the drop-down menu in the **SSH Connection Method** of the Advanced Settings section of the Remote UNIX Server and Remote Windows Server Properties page. If the option for Internal Java Libraries does not appear in the list, you can still use the Plink external SSH Client for SSH connections. You can also contact a Mercury Interactive sales representative to upgrade to a later version of SiteScope that includes the Integrated Java SSH Client.

Configuring Remote UNIX Servers for SSH Monitoring

SiteScope for Solaris or Linux supports remote monitoring via SSH. Setting up the SSH hosts on the remote servers you want to monitor in the UNIX environment can be very complex and is beyond the scope of this document. Some suggested resources on installation of the OpenSSH daemon are http://www.sunfreeware.com/openssh.html (for Solaris) and http://www.redhat.com/docs/manuals/linux/RHL-7.3-Manual/ref-guide/s1-ssh-configfiles.html for Redhat Linux.

The following are requirements for configuring remote UNIX servers for SSH monitoring with SiteScope in a UNIX environment:

1. Secure Shell daemons or servers (sshd) must be installed on each remote server you want to monitor with SiteScope.
2. The SSH daemons on the remote servers must be running and the applicable communication ports must be open. For example, the default for SSH is port number 22.
3. A SSH client must be installed on the server where SiteScope is running. The SiteScope integrated Java SSH client will normally fill this requirement.
4. If you use an external SSH client on Solaris or Linux, the SSH client binaries must be accessible to SiteScope. When SiteScope invokes the SSH client process, it will search in both /usr/bin and /usr/local/bin for the ssh command. The ssh binaries must be in one of these two locations and SiteScope must have permissions to execute the ssh command.

   You should verify SSH client-to-server connectivity from the machine where SiteScope is running to the remote machine you want to monitor. You should check SSH connectivity outside of the SiteScope application before

setting up remote server connections using SSH in SiteScope. For example, if SiteScope is running on Solaris or Linux, use the following command line to request an SSH connection using SSH2 to the server *remotehost*:

ssh -2 *remotehost*

This normally will return text information that indicates the version of SSH protocol that is being used. Also, this will attempt to authenticate as the current user. Use the -l username switch to request a login as a different user.

For SiteScope running on Windows, see the section on Testing SSH connectivity with PuTTY utilities for information about testing SSH connectivity outside of the SiteScope application on Windows NT/2000 machines.

Once you have confirmed SSH connectivity, create or configure UNIX Remote settings in SiteScope to use SSH as the connection method.

Configuring Remote Windows Servers for SSH Monitoring

The default remote connection method used by SiteScope for Windows-to-Windows connectivity and monitoring in Windows NT/2000/2003 networks is NetBIOS. While this has provided ease of connectivity, it does have several disadvantages. One is that NetBIOS is relatively vulnerable in terms of network security. Another is that it does not support remote execution scripts. Running commands on remote servers requires that scripts be executed locally with commands to the remote machine being written using the UNC syntax of remote servers. Even then, some parameters are not returned from the remote server via NetBIOS.

Starting with version 7.8, SiteScope supports monitoring of remote Windows NT/2000 servers using SSH. This technology has been tested with the OpenSSH binaries from Cygwin available at http://www.cygwin.com/ installed as the SSH server on the remote server. It has also been tested with the server available from F-Secure. You may also try OpenSSH for Windows (formerly Network Simplicity "OpenSSH on Windows") which is available on SourceForge. The following is a comparison overview of two of the packages.

OpenSSH Package	Advantages	Disadvantage
Cygwin OpenSSH	1. Provides access to either Windows or UNIX-style scripting on a Windows machine. 2. Provides access to UNIX-style system tools and utilities. 3. SiteScope can access the remote server both as a Windows Remote and /or a UNIX Remote.	Complicated setup procedure.
OpenSSH for Windows	Simple setup procedure.	Only provides access to Windows commands, scripts, and utilities.

Note: OpenSSH for Windows and the Cygwin SSH implementations have been shown to be incompatible with each other and should not both be installed on the same machine.

Note: If there is more than one version of the Cygwin utilities or more than one SSH server installed a machine, there may be conflicts that prevent the SSH connections from working. An error message such as "could not find entry point" is one indication of this kind of conflict. If you suspect this error, search the machine for multiple copies of cygwin1.dll. It may be necessary to remove all versions of the utilities and then reinstall only a single installation to resolve this problem.

There are two main steps for configuring remote Windows Servers for SSH monitoring with SiteScope:

1. Installation and Configuration of a SSH Server

2. Installation of SiteScope Remote NT SSH Files

The following sections describe the steps you use to install the necessary packages:

1. Installation and Configuration of a SSH Server

To enable SiteScope monitoring using SSH, a SSH server must be installed and configured on each remote server to which you want SiteScope to connect. There are two software packages generally available that will enable SSH capability. One is the Cygwin environment available from RedHat at http://www.cygwin.com/. Another package is the OpenSSH for Windows available at OpenSSH for Windows. The following describes the steps to install either of these packages:

Note: These setup steps must be performed for each server that will run the SSH daemon or server.

Installing Cygwin OpenSSH (on Windows)

Perform the following steps to install and configure a Cygwin OpenSSH server on Windows servers:

To install and configure a Cygwin OpenSSH server on Windows NT/2000 servers:

Note: The following instructions assume that no other Cygwin or other ssh utilities are installed on the machine and that the machine has Internet access.

Note: The user login account used to install and run the SSH daemon will need adequate permissions to install the necessary programs, configure several file options, and control Windows services. It does not need to be the account that SiteScope will use to connect to the subject server, although that account must be configured within the Cygwin installation before you can monitor that server with SiteScope.

---

1. Create a new System Environment variable with the following definition: CYGWIN = ntsec tty.
2. Add the string ;C:\cygwin\bin to your PATH variable. Save the changes to the variables.
3. Download the Cygwin setup program into a temporary folder. For example: C:\temp. The setup program is used to select, download, and install different packages and components available with Cygwin.
4. Run the downloaded setup program and choose the **Install from Internet** option when prompted to Choose A Download Source. Click **Next** to continue.
5. If prompted, select a root install directory where the Cygwin package should be installed. This will be where the SSH daemon and related files will be installed. For example, C:\cygwin. Click **Next** to continue.
6. If prompted, select a temporary directory where the Cygwin installation files should be stored. For example, C:\temp. Click **Next** to continue.
7. If prompted, select an Internet Connection option. Normally, **Direct Connection** can be used. Click **Next** to continue.

8 Select a suitable mirror site from which to retrieve the files using the selection list when prompted. Click **Next** to continue.

9 The Setup program queries the mirror site for the packages available and displays a hierarchy tree of package categories. To view and select the packages to download, click on the plus (+) symbol to the left of the category name to expand any of the package trees. Packages that are selected for download and installation will display a version number in the **New** column. If a version number is not displayed for a particular package, it will not be downloaded and installed. Click **Skip** to the left of package name to select the package for download.

Note: Many of the development (Devel) and database (Database) tools that may be selected by default for download are not necessary to run the SSH daemon and can be deselected to reduce download time and installation space.

Select each of the following packages for download and installation:

- cygrunsrv from the Admin branch
- cygwin-doc from the Doc branch
- pdksh from the Shells branch
- openssh and openssl from the Net branch
- your choice of UNIX-style text editor from the Editors branch (for example: vim or emacs)

Then click to download the files as prompted.

10 Depending on your installation options, the Cygwin setup will download and install the selected packages. You may be prompted to choose to have a shortcut to the Cygwin terminal window added to the Desktop or Program Start menu. Click to continue and complete the installation.

11 After the Cygwin setup is complete, open a Cygwin terminal window by clicking on the **Cygwin** desktop shortcut or Program Start menu item.

Note: Depending on the user profile in the Windows system, the default directory that opens in the terminal window may not be within the root Cygwin installation tree. Use the pwd command to display the current directory. Typing in the command string cd / will normally change the directory to the Cygwin root, which by default corresponds to the Windows C:\cygwin directory.

---

12 Update the default Cygwin group file with the group names in use on the machine and on your network. Use the mkgroup utility to update the default Cygwin group file with the groups defined on the server and in your domain. Examples of the commands to use are as follows:

Note: To have Cygwin recognize both domain and local group accounts, run the mkgroup utility twice, once for local users (-l option) and once for domain users (-d option). Remember to use >> syntax and not just >, to append entries to the file.

```
mkgroup -l >> ../etc/group
mkgroup -d >> ../etc/group
```

Note: If you use both the local and domain options, you must manually edit the /etc/group file (using the UNIX style text editor you downloaded) to remove any duplicate group entries. You may also want to remove group entries that are not needed for monitoring or should not have access to this machine.

---

13 Update the default Cygwin user (passwd) file with the users defined on the local machine plus any individual domain users you want to grant access to Cygwin on this machine. Use the mkpasswd utility to update the default Cygwin user file. Examples of the commands to use are as follows:

Note: By default, Cygwin is set to run the OpenSSH daemon as the local user called SYSTEM. To have Cygwin recognize both domain and local machine user accounts, run the mkpasswd using the -l option to add all local users, and run it with the -d and -u options to add individual domain users. Remember to use >> syntax and not just >, to append entries to the file.

```
mkpasswd -l >> ..\etc\passwd
mkpasswd -d -u username >> ..\etc\passwd (domain users)
```

Note: If you use both the local and domain options, you must manually edit the /etc/passwd file (using the UNIX style text editor you downloaded) to remove any duplicate user entries. You may also change the default /home path and default shell for individual users. This may be necessary to install the RemoteNTSSH package in the /home/sitescopeaccount/ directory of the user account to be used by SiteScope.

---

14 Change the active directory to the /bin directory by typing cd /bin.

15 Create a symbolic link in the /bin directory that points to the Windows Command (CMD) shell by entering the following command line (be sure to include the trailing space and period):

```
ln -s /cygdrive/c/winnt/system32/cmd.exe .
```

16 It is recommended that you change permissions and ownership of several Cygwin files and directories. Also create a log file for the SSH daemon. Type the following command lines in the Cygwin terminal command line:

Note: Exact syntax is required, including spaces.

Press **Enter** after each command line entered:

```
cd /
chmod -R og-w .
chmod og+w /tmp
touch /var/log/sshd.log
```

Note: Inconsistent and incorrectly assigned file and directory permissions can be one reason that the SSH daemon can not be started or that SiteScope is unable to connect to and execute commands or scripts on the remote server.

---

17 Configure the SSH daemon to run as a Windows service by entering the following command:

```
ssh-host-config -y
```

When presented with the CYGWIN= prompt, enter ntsec tty to match the environment variable you set at the beginning of this procedure. Normally, this will configure the SSH daemon or service to restart automatically if the server needs to be restarted.

18 Configure the encryption keys and files for the SSH daemon using the following command:

```
ssh-user-config -y.
```

You will be asked to enter passphrases for several keystore files. Enter appropriate passphrases when prompted. The program asks you to re-enter the passphrase for confirmation.

19 You must change the ownership of several files and folders for use by the SSH daemon. The program will normally not run if the permissions on these files allow them to be changed or executed by group or "world" level users. Enter the following command strings to restrict access to these files:

```
chown SYSTEM:Users /var/log/sshd.log /var/empty /etc/ssh_h*
chmod 755 /var/empty
```

20 Check the installation by starting and then stopping the CYGWIN sshd service using the **Programs** -> **Administrative Tools** -> **Services** panel.

Note: Cygwin includes a server utility to start the SSH daemon. However, there have been a number of situations where this method failed to start the server, whereas using the Windows Services panel was able to start the server.

---

21 Configure the default shell or command environment for the user account you will use for monitoring with SiteScope. The shell you select will effect what types of scripts or commands can be run remotely using the SSH connection. Use the UNIX-style text editor and edit the /etc/passwd file. Find the entry for the SiteScope login account you intend to use and change the shell from /bin/bash to the shell you want to use as described below. This will normally be the last entry in the line for that account entry.

1. If you chose to have SiteScope interact with the remote server using the Windows Command shell, change the default shell entry to /bin/cmd. Use this option when you plan to use Windows-style batch files and scripts You must also include the symbolic link to the Windows cmd.exe kernel in the /bin directory as described in a previous step of this procedure.

2. If you chose to have SiteScope interact with the remote Windows server using a Cygwin UNIX shell, change the default shell entry to be /bin/pdksh. The SiteScope SSH client may not accurately parse Cygwin's default bash shell. You must also configure a Remote UNIX server connection to this (Windows) server that connects to the Cygwin SSH daemon.

Save the changes to the file.

22 Edit the PATH and the default prompt commands in the /etc/profile file to ensure that Cygwin can find certain files and that SiteScope can parse the output from the remote shell. Use the UNIX-style text editor and edit the /etc/profile file. Find the PATH definition entry near the top of the file. For example:

```
PATH=/usr/local/bin:/usr/bin:/bin:$PATH
```

Change this to include the following:

```
PATH=.:/usr/local/bin:/usr/bin:/bin:$PATH
```

23 To change the default prompt commands, edit the /etc/profile file, and find the section similar to the following:

```
;;
sh  | -sh   | */sh  |\
sh.exe  | -sh.exe   | */sh.exe )
#Set a simple prompt
PS1='$ '
;;
```

Immediately under this entry, add the following:

```
;;
pdksh  | -pdksh   | */pdksh  |\
pdksh.exe  | -pdksh.exe   | */pdksh.exe )
#Set a simple prompt
PS1='> '
;;
```

24 Save the changes to the file.

25 Change the active directory to the home directory of the user you have created for SiteScope monitoring.

After making these changes and starting the SSH daemon, you should be able to connect to the server using an SSH client. For information about testing SSH connectivity outside of the SiteScope application on Windows NT/2000 machines, see the section on Testing SSH connectivity with PuTTY utilities.

Note: Any time you run the mkpasswd -l /etc/passwd command (for example, when adding a new user), edit the /etc/passwd file again to make sure that the default shell for that user is set to the appropriate value for any account being used by SiteScope.

Installing OpenSSH for Windows

The OpenSSH for Windows package is an alternative to the Cygwin SSH package and can be easier to install. Like most products, the Cygwin product and the Open SSH for Windows are subject to change. There are cases where some versions of the Cygwin SSH server have not returned the data needed for SiteScope monitoring. If the OpenSSH for Windows package can solve this problem, you should use this package in place of the Cygwin package.

To install and configure an OpenSSH for Windows server on Windows NT/2000 servers:

1. Download and install the OpenSSH for Windows package.
2. Open a command prompt and change to the installation directory (C:\Program Files\OpenSSH is the default installation path).
3. Change the active directory to the OpenSSH\bin directory.
4. You must update the default group file with the group names in use on the machine and in your network. Use the mkgroup utility to update the default OpenSSH group file with the groups defined on the server and in your domain. Examples of the commands to use are as follows:

Note: To have OpenSSH recognize both domain and local group accounts, run the mkgroup utility twice, once for local users (-l option) and once for domain users (-d option). Remember to use >> syntax and not just >, to append entries to the file.

```
mkgroup -l >> ..\etc\group

mkgroup -d >> ..\etc\group
```

Note: If you use both the local and domain options, you must manually edit the /etc/group file (using the UNIX style text editor you downloaded) to remove any duplicate group entries. You may also want to remove group entries that are not needed or should not have access to this machine.

---

5 You must update the default OpenSSH user (passwd) file with the users defined on the local machine plus any domain user you want to grant access to the SSH server on this machine. Use the mkpasswd utility to update the default user file. Examples of the commands to use are as follows:

Note: To have OpenSSH recognize both domain and local machine user accounts, run the mkpasswd utility using the -l option to add all local users and run it with the -d and -u options to add individual domain users. Remember to use >> syntax and not just >, to append entries to the file.

```
mkpasswd -l >> ..\etc\passwd

mkpasswd -d -u username >> ..\etc\passwd
```

Note: If you use both the local and domain options, you must manually edit the /etc/passwd file (using the UNIX style text editor you downloaded) to remove any duplicate user entries. You may also change the default /home path and shell for individual users (see instructions below).

---

6 Check the installation by starting the **OpenSSH Server** service using the **Programs** > **Administrative Tools** > **Services** panel.

2. Installation of SiteScope Remote NT SSH Files

SiteScope includes a set of files that must be installed on each remote Windows server to enable certain commonly used server monitoring functions. The following sections describe the steps you use to install these files according to the SSH package you are working with.

Use the following steps to install the SiteScope SSH Files on Cygwin OpenSSH installations:

To install the SiteScope SSH Files on Cygwin installations:

1 Verify that a /sitescope_login_account_name directory exists within the install_drive:/cygwin/home directory on each machine that will be monitored

by SiteScope using SSH. Replace sitescope_login_account_name with the user account name you will use to connect to the machine using the SSH server.

2. One of the advantages of using SSH on Windows is that it allows SiteScope to execute scripts on the remote server running the SSH daemon. To be able to use the Script Monitor to run remote scripts, create a scripts subdirectory in the /home/sitescope_login_account_name directory. Scripts you create for execution by the SiteScope Script Monitor must be placed inside this directory.

3. On the machine where SiteScope is installed, find the file called RemoteNTSSH.zip in the <SiteScope install path>\SiteScope\tools directory.

4. Copy this file to the install_drive:\cygwin\home\sitescope_login_account_name directory on each of the remote Windows NT/2000 servers where you have installed the SSH server or daemon software.

5. Unzip the RemoteNTSSH.zip file on the remote server. Place the contents of the zip file into the install_drive:\cygwin\home\sitescope_login_account_name directory. This should create a install_drive:\cygwin\home\sitescope_login_account_name\scripts subfolder. You use this subfolder to hold scripts that can be run by the SiteScope Script Monitor.

6. Start the CYGWIN sshd service on the remote server.

Use the following steps to install the SiteScope SSH Files on OpenSSH for Windows installations.

To install the SiteScope SSH Files on OpenSSH for Windows installations:

1. On the machine where SiteScope is installed, find the file called RemoteNTSSH.zip in the <SiteScope install path>\SiteScope\tools directory.

2. Copy this file to the install_drive:\WINNT directory on each of the remote Windows NT/2000 servers where you have installed the SSH server or daemon software.

3. Unzip the RemoteNTSSH.zip file on the remote server. Extract the contents of the zip file into the install_drive:\WINNT directory. This should create an install_drive:\WINNT\scripts subfolder. You use this subfolder to hold scripts that can be run by the SiteScope Script Monitor.

4. Start the OpenSSH server service on the remote server.

After you have completed the steps above, it is recommended that you test SSH connectivity from your SiteScope server by using plink.exe or PuTTY.exe as described in the section Testing SSH connectivity with PuTTY utilities. After confirming SSH connectivity between SiteScope and the remote server, you can set up Remote Windows configurations as described in the User Guide, and select SSH as the connection method. You can then configure CPU, Disk, Memory Windows Performance Counter, and Script monitors to use the SSH connectivity.

SiteScope SSH Client Connection Options

After you have set up SSH servers or daemons on remote servers, you need to configure the SSH client that SiteScope will use to connect to the remote servers. As noted above, SiteScope includes two client options for SSH connectivity. The following presents an overview of the client options. More information about each option is included in the sections indicated.

1 **Configuring SiteScope to use the integrated Java SSH client**

As of version 7.8.1.1, SiteScope includes an integrated SSH client written in Java. This is the recommended option for SSH connectivity. One advantage of using this client option is that it uses fewer system resources than the external clients would use. Also, configuration of this client is simpler in some cases. To configure your remote using an external client, see the section “Using the Integrated Java SSH Client”.

2 **Configuring SiteScope to use an external SSH client**

SiteScope on Windows ships with a third-party SSH client called plink. Plink is one part of a set of SSH tools called PuTTY. SiteScope on UNIX and Linux require that an external SSH be installed on the machine where SiteScope is running. To configure your remote using an external client, see the section “Using an External SSH Client”.

6

Working with SSH Clients

If you need to use Secure Shell (SSH) to connect to remote Unix or Windows servers, SiteScope must have access to a SSH client to make the connection and transmit data. This section presents some of the client configuration possibilities and issues involved in using SSH for SiteScope monitoring.

This chapter describes:	On page:
Using the Integrated Java SSH Client	59
Using an External SSH Client	62

Using the Integrated Java SSH Client

SiteScope provides a SSH client written in Java that is integrated into the SiteScope application. This client significantly reduces the required system resources used by SiteScope when connecting to servers via SSH. The Java client supports both SSH version 1 and version 2 protocols as well as both password-based and key-based authentication. The SiteScope configuration for the client is identical for UNIX, Linux, and Windows SiteScope. Documentation on using an external SSH client can be found at External SSH Client.

Working with the Integrated SSH Client

As noted previously, there are two different versions of the SSH protocol: version 1 and version 2. While they are both considered to be Secure Shell protocols, version 1 and version 2 are considered to be two different protocols and are not compatible with each other. Some security vulnerabilities have been found in SSH1. This resulted in several changes in SSH2, which is considered the current standard. Most SSH software will

support both protocols. However, to be sure that a request for a SSH connection uses SSH2 instead of SSH1, it is necessary to configure SSH clients and SSH hosts to use the same protocol version between them to communicate. In many cases, SSH version 1 (SSH1) is the default version used for connections, as it is considered the lowest common denominator between a SSH client and a SSH host.

There are two ways to force SSH2 connections. These are:

1. Configure all SSH daemons or servers to accept only SSH2 connection requests.
2. Configure the SSH client on the SiteScope server to only make SSH2 requests.

The first option is perhaps the most secure but may be the most time-consuming unless each server was configured for this option when it was installed and activated. The second option requires changes only to the client on the SiteScope server. For the integrated Java SSH client, this can be controlled by a setting in the SSH Advanced Options section on the remote server setup page.

Another part of SSH security is authentication. The integrated SSH client for SiteScope can be configured to use one of two authentication options. These are:

- Password Authentication
- Key-Based Authentication

Password Authentication is the default method for SSH connections in SiteScope. Key-Based Authentication adds an additional level of security through the use of a passphrase and a public-private key authentication. For more details on how to set up key based authentication for SSH connections, see the following section titled Setting up Key-Based Authentication.

Setting up Key-Based Authentication

To use Key-Based Authentication for SSH remotes, you must first generate a pair of public/private keys. The public key will reside on the remote and the private key will be kept on the SiteScope machine. Both Cygwin OpenSSH and OpenSSH for Windows come with a key generation tool called ssh-keygen. The ssh-keygen tool allows you to create both protocol version 1 and version2 keys. Read the documentation on ssh-keygen to create the type of key that you need. For example, to create an RSA key pair SSHv2:

1. Log on to your UNIX remote server using the same account as will be used by SiteScope.
2. Run the command ssh-keygen -t rsa. The following is an example output from this command:

   Generating public/private rsa key pair.

   Enter file in which to save the key (/home/sitescope/.ssh/id_rsa):

   Enter passphrase (empty for no passphrase):

   Enter same passphrase again:

   Your identification has been saved in /home/sitescope/.ssh/id_rsa.

   Your public key has been saved in /home/sitescope/.ssh/id_rsa.pub.

   The key fingerprint is:

   09:f4:02:3f:7d:00:4a:b4:6d:b9:2f:c1:cb:cf:0e:e1 sitescope@acompany.com
3. Copy the "identification" file, private key, to the SiteScope machine. The default location for this file is SiteScope/groups/identity. This directory must be created by the user.
4. Create or configure the remote server profile in SiteScope to use SSH as the connection **Method** and Key-Based as the **SSH Authentication Method** under the Advanced Settings.

Using SSH Version 2 protocol

By default, the SiteScope Java client will use the SSH1 Protocol if the server it is trying to connect to allows SSH1 connections. If this negotiation fails, then SiteScope will try to connect using version 2 protocol. The SiteScope Java client can be configured to use only SSH2 connections. Making the change on the SiteScope machine may be easier than having to reconfigure a large number of remote SSH servers.

When configuring your remote server profile, select the **SSH Version 2 Only** check box in the Advanced Settings.

Note: When using Key-Based authentication, the Key File supplied must be a version 2 private key.

Using an External SSH Client

SiteScope provides the capability of connecting to remotes using an external SSH client. On Windows platforms, the plink client is shipped with SiteScope. On UNIX and Linux platforms, SiteScope can use a standard SSH client such as SunSSH or OpenSSH.

Working with External SSH Clients

As noted previously, there are two different versions of the SSH protocol: version 1 and version 2. While they are both considered to be Secure Shell protocols, version 1 and version 2 are considered to be two different protocols and are not compatible with each other. Some security vulnerabilities have been found in SSH1. This resulted in several changes in SSH2 which is being considered the current standard. Most SSH software will support both protocols. However, to be sure that a request for a SSH connection uses SSH2 instead of SSH1, it is necessary to configure SSH clients and SSH hosts to use the same protocol version between them to communicate. In many cases, SSH version 1 (SSH1) is the default version used for connections, as it is considered the lowest common denominator between a SSH client and a SSH host.

There are two ways to force SSH2 connections. These are:

1. Configure all SSH daemons or servers to accept only SSH2 connection requests.
2. Configure the SSH client on the SiteScope server to make only SSH2 requests.

The first option is perhaps the most secure but may be the most time-consuming unless each server was configured for this option when it was installed and activated. The second option requires changes only to the client on the SiteScope server. For external SSH client, this is usually controlled via the client settings. For more details on how to set the SiteScope PuTTY client to use SSH2 see the section Setting up SSH2 on SiteScope for Windows below.

Another part of SSH security is authentication. The integrated SSH client for SiteScope can be configured to use one of the following two authentication options:

- Password Authentication
- Key-Based Authentication

Password Authentication is the default method for SSH connections in SiteScope. Key-Based Authentication adds an additional level of security through the use of a passphrase and a public-private key authentication. See the following section for information on how to set up Key-Based authentication for SSH connections.

Monitoring with SSH on Windows Platforms

SiteScope for Windows platforms includes a SSH client to handle connections to remote SSH-enabled servers. SiteScope includes the PuTTY SSH utilities for SSH connectivity to both UNIX and Windows servers. These utilities are found in the <SiteScope install path>/SiteScope/tools directory.

By default, SiteScope SSH connections will use the SSH1 protocol (less secure) unless the server it is connecting to accepts only SSH2 sessions. To force SiteScope to use the SSH2 protocol (more secure), you will need to configure the SSH client on the machine where SiteScope is running and possibly the SSH daemons/hosts on the remote servers to communicate using the SSH2 protocol. For SiteScope on Windows, configure the PuTTY SSH client utility and SiteScope as described below in Setting up SSH2 on SiteScope for Windows.

Note: The PuTTY and plink tools supplied with SiteScope are not the latest release versions of these tools. Starting with version 7.8.1.0, SSH connectivity is handled by the internal Java libraries by default. Consider checking for newer versions and replacing the files supplied with SiteScope with updated versions. More information about the PuTTY SSH client can be found at http://www.chiark.greenend.org.uk/~sgtatham/putty/ or http://www.openssh.org/windows.html.

Instructions for creating Public Keys using the PuTTYGen tool and using them are at http://www.tartarus.org/~owen/putty-docs/Chapter8.html.

Note: SSH uses DES, BLOWFISH, RSA, or other public key cryptography for both connection and authentication. Public Keys are stored on a per-user basis so if you are using key-based logins instead of password-based logins, you should login and run the PuTTYGen tool using the same account as will be used by the SiteScope service.

Testing SSH connectivity with PuTTY utilities

It is recommended that you test SSH connectivity from SiteScope on Windows to remote hosts using either the PuTTY.exe or plink.exe tools. This is also useful for troubleshooting connectivity. You can use utilities to test connectivity with a SSH host. The plink utility is run from the command line. You use the following steps to test connectivity with plink:

To test SSH connectivity with PuTTY:

1. Log on to your Windows machine as the user who runs the SiteScope service.
2. Open a command window to the <SiteScope install path>\SiteScope\tools directory.
3. Run the plink utility with the following syntax:

   ```
   plink -ssh remoteuser@hostname
   ```

   where remoteuser is the login username for a valid user account on the hostname server.
4. Follow the prompts in the terminal window to confirm that the remote login is successful. Log out of the terminal session when you are satisfied that the connection is working correctly.

   If you want to use the SSH2 protocol for connections, you will need to use the PuTTY utility to configure the PuTTY client to use SSH2 instead of the default SSH1. This requires that you save session settings as described in the section "Setting up SSH2 on SiteScope for Windows Platforms" below. After you have done this, you can also use PuTTY to test SSH connectivity. You use the following steps for testing SSH2 connectivity using PuTTY:

To test SSH2 connectivity with PuTTY:

1. Log on to your Windows machine as the user who runs the SiteScope service.
2. Launch the PuTTY utility.
3. From the Session tab or tree, select the Saved Session name of the remote connection you want to test and click the **Load** button to the right of the selection box.
4. Click the **Open** button near the bottom of the dialogue box. This will launch a terminal emulation window.
5. Follow the prompts in the terminal window to confirm that the remote login is successful. Log out of the terminal session when you are satisfied that the connection is working correctly.

Setting up SSH2 on SiteScope for Windows Platforms

SiteScope for the Windows platform uses plink, part of the PuTTY suite of SSH tools, to create its SSH connections for remote monitoring. By default, the plink utility will use the SSH1 Protocol if the server it is trying to connect to allows SSH1 connections. The SiteScope SSH client can be configured to use only the SSH2 protocol for connections. Making the change on the SiteScope machine may be easier than having to reconfigure a large number of remote SSH servers.

Setting up SiteScope for Windows to use only SSH2 to communicate with remote UNIX or remote Windows servers requires two actions:

1. Create settings in the SSH client on the SiteScope server to use only SSH2.
2. Modify SiteScope remote server connection profiles to use the SSH2 connection profile.

The following two sections describe the steps you use to force SiteScope to use SSH2 for connecting to remote servers.

Use the following to steps to setup the PuTTY client on the SiteScope server to use only SSH2 by using the PuTTY utility suite.

To set up PuTTY to use SSH2:

1. Log on to the server where SiteScope is running as the user who runs the SiteScope service. To see which user this is, open your Services control panel, right-click the SiteScope service, select "Properties," and click the "Log On" tab.
2. Find the PuTTY.exe tool in the <SiteScope install path>\SiteScope\tools directory. Alternatively, you can download an updated PuTTY version from the Internet.
3. Launch the PuTTY utility by double-clicking the icon in Windows Explorer (or typing putty in a command window with a path to the <SiteScope install path>\SiteScope\tools directory). No installation steps are needed. The Putty Configuration console opens.
4. With the **Session** tab or tree selected, enter the hostname or IP address of the remote machine to be monitored in the **Host Name** box. Select the SSH radio button below the hostname in the Protocol section.

5 Select the **Connection** tab or tree, and enter the username on the remote machine in the Auto-login username box. This should be a user account with permissions to monitor processes and hardware statistics on the remote server. Optionally, this user account might also have execution privileges to allow SiteScope to run scripts on the remote server.

6 Select the **SSH** tab or tree under the Connection tree, and then choose the **2** radio button in the Preferred SSH Protocol Version section.

7 Return to the Session tab. In the Saved Sessions text box, enter a name for these settings. Any previously saved settings appear in the list box below.

Note: The Saved Session name should not be a resolvable hostname on your network, nor can it contain a white space character. For instance, if these settings are for a machine named "myhost.mydomain.com", the session settings name cannot be "myhost", "myhost.mydomain.com", or "myhost settings" (the latter is not allowed because of the white space between the words). Choose a name, such as "myhost-settings".

---

8 Click the **Save** button. The name of your new settings should appear in the list of saved settings.

Repeat this process to create settings for each remote machine you wish to monitor with SiteScope using SSH2.

Note: Make a note of the Saved Session name for each machine that you configure, to enter this name into the SiteScope configuration file.

Configuring SiteScope to Use SSH2:

Use the following steps to configure SiteScope to use SSH2 for connecting to remote UNIX or remote Windows servers.

To configure SiteScope to use SSH2:

1 Open SiteScope in a Web browser. Click the **Preferences** node.

2. To setup SSH2 for a remote UNIX connection, click the **Unix Remotes Preferences** node. To setup SSH2 for a remote Windows connection, click the **Windows Remotes Preferences** node. The corresponding remote Servers page is displayed.
3. Click the **New Server** button for the type of remote server you are adding. The relevant New Server page is displayed.
4. In the **Host** box, enter the name of the settings you saved. For example, to use the settings for myhost.mydomain.com that were created above, you would enter myhost-settings in the box.
5. Select the applicable operating system of the target remote server in the **OS** drop-down list.
6. Leave the **Login** box blank.
7. Enter the password to log in to the remote machine in the **Password** box.
8. For UNIX Remotes: If the shell prompt for the remote UNIX server is something other than **#**, enter that prompt in the **Prompt** section.
9. For UNIX Remotes: Leave the **Login Prompt** and **Password Prompt** boxes blank.
10. Click **OK** to add the remote server profile.

Note: The remote connection test will FAIL. You may see a message similar to the following error message:

```
Connecting to myhost-settings...

Waiting for prompt(#)...

Unable to open connection:

Host does not exist

Remote command error: unknown host name (-997)
```

---

11. Go to your <SiteScope install path>\SiteScope\groups directory and make a backup copy of the file called master.config. Rename the backup file to master.config.SAV.

12 Open the file master.config file in a text editor, and locate the section of entries or lines beginning with the string _remoteMachine. If you have configured multiple remote server connections, there will be multiple entries that begin with this string. Locate the line that includes the string _host=myhost-settings, where myhost-settings is the name of the host settings you entered in the Server Address box in PuTTY Configuration tool.

13 Add the following string to the end of that line:

Note: This string must be entered on the same line. Do not add any carriage returns, new lines, or extra spaces.

```
_sshCommand=<SiteScope install path>\tools\plink.exe_-ssh_$host$_-pw_$password$
```

Replace <SiteScope install path> with the path to your SiteScope installation. For example, if SiteScope is installed at C:\SiteScope, the string would read:

```
_sshCommand=C:\SiteScope\tools\plink.exe_-ssh_$host$_-pw_$password$
```

After you have finished making modifications, the entire line should look similar to the following example:

Note: This example wraps across multiple lines to fit on this page. When entering this setting into the SiteScope configuration file, be sure that it is entered all a single line.

```
_remoteMachine=_os=Linux _id=11 _trace= _method=ssh
_password=(0x)MGJJKDKLKJNINPNJMJ _login= _host=myhost-settings
_name= _sshCommand=C:\SiteScope\tools\plink.exe_-ssh_$host$_-pw_$password$
```

14 Repeat this step to modify each _remoteMachine entry, using the applicable host name setting created for each host using the PuTTY Configuration tool in the previous section.

15 Save and close the master.config file.

16 Stop and restart the SiteScope service to force SiteScope to reload the manual changes you made to the master.config file.

17 Open a Web browser to the SiteScope server.

18 Click the **Preferences** node.

19 If you are setting up SSH2 for remote UNIX connections, click the **UNIX Remote Preferences** node. For SSH2 for remote Windows server connections, click the **Windows Remote Preferences node**. The corresponding Servers page is displayed.

20 For UNIX remotes, click the **Detailed Test** button in the Servers table for the UNIX Remote you configured to test the connection and verify that it works. For Windows remotes, click the **Test** button in the Windows Servers Table for the Windows Remote you configured to test the connection and verify that it works.

Note: This test normally takes a few seconds to complete.

7

Creating Custom Properties

The ability to customize SiteScope has been an important feature in extending and adapting it to a wide variety of environments and needs. This section describes how you can add custom property settings to SiteScope monitors and how you can add custom content to real-time Dashboard views in the SiteScope interface.

This chapter describes:	On page:
About Custom SiteScope Properties	71
Working with Custom Monitor Properties	72
Working with Custom Display Columns	76
Displaying Custom Properties in Custom Columns	78

About Custom SiteScope Properties

Each monitor type in SiteScope includes a number of properties. You access these properties using the monitor's Properties view. You can add your own custom properties to SiteScope monitors. For details, see "Working with Custom Monitor Properties" on page 72.

The Dashboard view displays real-time monitor results and related information in a table format. You can add a custom column for display in the Dashboard view in the new interface. You can display a custom column in the group detail view in the classic interface. For details, see "Working with Custom Display Columns" on page 76.

You can link custom properties to custom columns to have the values of your custom properties displayed in real-time SiteScope views. You can also have other SiteScope monitor properties displayed in custom columns. For details, see "Displaying Custom Properties in Custom Columns" on page 78.

Working with Custom Monitor Properties

Each SiteScope monitor instance consists of a set of properties that define what action the monitor is to perform and how it is to perform these actions. Monitor properties also store information that may be passed to the system that is being tested.

The custom monitor property feature allows you to define and store additional information you associate with a monitor. For example, you may want to enable users to define property values that categorize monitors according to their importance, or actions that need to be taken if the monitor reports an error. You may add more than one custom monitor property to SiteScope.

Defining a custom monitor property adds a new form entry item to the add and edit view for monitors. A new form entry item is added for each custom property defined. The custom property definition fields are added to all monitor types.

Note: Custom monitor properties do not have default values and may be blank until you have defined a value for a property. Values are defined only when you explicitly enter or select a value and then update the monitor.

When you add a custom monitor property definition, SiteScope creates a new display panel in the monitor Properties view called Custom Property Settings. In the SiteScope Classic interface, custom monitor properties are listed in the Advanced Options section below the Report Description item.

Note: The Customer Properties area appears in the new interface even if no custom properties were defined for the SiteScope.

Custom Monitor Property Definition Syntax

Custom monitor property definitions are added as single line entries to the master.config file in SiteScope. The default custom property form element is a text box. You can also define a custom property to use selection menu entries. The syntax of the definition string for a custom monitor property that accepts plain text values is as follows:

_monitorEditCustom=*_propertyName*|*display_title*|*display_description*|*default_definition*

The syntax of the definition string for a custom monitor property that an option selection menu is as follows:

_monitorEditCustom=*_propertyName*|*display_title*|*display_description*|*default_definition*|*input_code*

The custom elements of the definition string are separated by the vertical pipe (|) symbol. The following table describes the elements of the custom property definition string:

Element	Description
_monitorEditCustom	The parameter name that signals SiteScope to add a custom monitor property to monitors.
_propertyName	The variable name for the property. Generally, this should begin with an underscore character to signal that it is a property variable. This name can be referenced elsewhere within SiteScope. Do not include spaces or punctuation marks as part of the property name.

Element	Description
display_title	The item title text to be displayed in the interface with the custom property entry or selection field. This string may include HTML markup code.
display_description	An optional text description to be displayed in the interface below the property entry field. Use this to describe the purpose and usage for the property. The string may include spaces and punctuation marks.
default_definition	An optional string to define a default value for the property. The default input object is a text field.
input_code	This field contains the HTML code for an option selection input object for setting the custom property value. This must include the $NAME$ and $VALUE$ special internal reference variables used by SiteScope to work with the list.

Note: The entire definition string must be on a single line of text. Do not insert any linefeeds or carriage returns in the definition.

In SiteScope 8.0, it is not necessary to restart the SiteScope service to activate the custom properties. There may be a delay of several minutes for SiteScope to update its configuration data. Earlier versions of SiteScope require you to restart the SiteScope service to activate the custom properties.

You use the following steps to add a simple custom property:

To add a text or numeric custom property:

1. Make a backup of the master configuration file in a safe location. The file is found at <install_path>\SiteScope\groups\master.config on the machine where SiteScope is running.
2. Open the master.config file using a text editor.
3. Add a _monitorEditCustom= entry as a new line to the file.

4. On the same line, enter the text string to define the *_propertyName*, the *display_title* and the *display_description*. Separate each of the entries with the vertical pipe (|) symbol.
5. If you want to define a default value for the property, add the default value in the place of the *input_definition* entry.
6. Save the changes to the file.

To standardize custom property values, you can define a selection list as the input control. This allows you to define a set of values that can be assigned to a custom property. The following example defines a custom property named _actiongroup with an option selection menu having three options: SysDev, TestDev, and Production.

```
_monitorEditCustom=_actiongroup|Team|Select which support team has
responsibility for this monitor.|SysDev|<select name="$NAME$"><OPTION
selected value="$VALUE$">$VALUE$</option> <option>TestDev</option>
<option>Production</option></select>
```

Use the following steps to add an option selection list input control to a custom property definition.

To define a selection list input for a custom property:

1. Make a backup of the master configuration file in a safe location. The file is found at <install_path>\SiteScope\groups\master.config on the machine where SiteScope is running.
2. Open the master.config file using a text editor.
3. Add a _monitorEditCustom= entry as a new line to the list of parameters.
4. On the same line, enter the string to define the *_propertyName*, the *display_title* and the *display_description.* Separate each of the entries with the vertical pipe (|) symbol.

5 For the *input_definition* item, enter the option selection code using the following syntax:

<select name="$NAME$"><OPTION selected value="$VALUE$">$VALUE$</option><option>*option_value_1* Group</option><option>*option_value_2*</option><option>*option_value_3*</option>...</select>

Where $NAME$ and $VALUE$ are required, internal reference variables and *option_value_1*, *option_value_2*, etc. are the text strings for the options in the selection menu.

6 Save the changes to the file.

Using Custom Properties in Alert Templates

You can reference custom monitor properties in alert templates. This allows you to include custom information about monitors in alerts sent by SiteScope. You use the _propertyName from the custom property definition as a variable reference in alert templates. For details, refer to the section "Template Properties" in the *SiteScope Reference Guide*.

Working with Custom Display Columns

The Custom Column feature of SiteScope provides a flexible tool for customizing the information and functionality available through the SiteScope interface. The custom column can be used to display information as text, provide a hyperlink to a specific URL, call Javascript code, and even include a monitor-specific CGI request to an external resource. At present, only one custom column is support in SiteScope.

Custom columns can be created without creating a custom property. For example, You can create a custom column to display a fixed hyperlink to an external resource in the SiteScope interface. You can display custom monitor properties in a custom column. For details, see "Displaying Custom Properties in Custom Columns" on page 78.

You can also embed short Javascript code into a custom column. A script can call other custom scripts that you can add or reference by using the headerHTML parameter in the master.config file. For example, you can write a script as an external file called sscopecustom.js in the <install_path>\SiteScope\ and then reference this file by adding the following to the headerHTML entry:

_headerHTML=<script language="Javascript" src="sscopecustom.js"></script>

Note: When adding Javascript using the headerHTML parameter, use the existing entry in the master.config. Only one headerHTML parameter is allowed.

Custom Column Definition Syntax

You define a custom column definition using the following syntax:

_monitorTableCustom=*_referenceValue*|*column_title*|*column_content_definition*

The custom elements of the definition string are separated by the vertical pipe (|) symbol. The following table describes the elements of the custom property definition string.

Element	Description
_monitorEditCustom	The parameter name that signals SiteScope to add a custom monitor property to monitors.
_referenceValue	The variable name for the property. Generally, this is a placeholder name although it can be referenced in other places within SiteScope.
column_title	The text to display as the column title.
column_content_definition	The text string or markup code to be displayed in the custom column for each monitor. This can include HTML and Javascript code.

Note: The entire definition string must be on a single line of text. Do not insert any linefeeds or carriage returns in the column definition.

You use the following steps to add a custom column to the SiteScope real-time views.

To add a custom column:

1. Make a backup of the master configuration file in a safe location. The file is found at <install_path>\SiteScope\groups\master.config on the machine where SiteScope is running.
2. Open **master.config** using a text editor.
3. Add a _monitorTableCustom= entry as a new line to the file.
4. On the same line, enter the text string to define the *_referenceValue*, the *column_title* and the *column_content_definition*. Separate each of the entries with the vertical pipe (|) symbol.
5. Save the changes to the file.

Displaying Custom Properties in Custom Columns

By default, custom monitor properties are displayed only in the Properties view of an individual monitor. This means that they are not normally visible in real-time views. In some cases, a custom monitor property is useful only if the value is displayed in the interface. You can display custom properties in Dashboard views by adding a reference to the property in a custom column.

The following is an overview of the steps you use to link monitor properties to custom columns:

1. Create a custom property definition.
2. Create a custom column definition.
3. Reference the custom property variable in the custom column definition.

Referencing Monitor Properties in Custom Columns

When you define a custom monitor property, SiteScope adds the name of the property to its list of properties for the monitor. You can display these properties in custom columns by adding a reference to one or more property variables in the column content definition string. Use the following steps to add references to custom properties to the content definition for custom columns.

To reference property values in custom columns:

1. Make a backup of the master configuration file in a safe location. The file is found at <install_path>\SiteScope\groups\master.config on the machine where SiteScope is running.
2. Open **master.config** using a text editor.
3. Add one or more _monitorEditCustom= entries including the *_propertyName*, the *display_title* and the *display_description* strings. Separate each of the entries with the vertical pipe (|) symbol.
4. Add a _monitorEditCustom= entry including the *_referenceValue*, the *column_title* and the *column_content_definition*. Separate each of the entries with the vertical pipe (|) symbol.
5. Within the string for the *column_content_definition*, include a reference to a custom property by adding the *_propertyName* surrounded with dollar signs ($). For example, if a custom property is defined with the name _NOCAction, add a reference to this property in the *column_content_definition* as $_NOCAction$.
6. Save the changes to the file.

You may include references to some other internal SiteScope property variables in the *column_content_definition* string. This is limited to property variables specific to each monitor type. For information on monitor properties, see "Template Properties" in the *SiteScope Reference Guide*.

Index

B

baseline
    static performance 34

C

content match
    examples for log files 17
    using metacharacters 6
    using regular expressions 1
    using string literals 4
    using system date variables 12
custom columns
    about 71
    creating 71
    syntax 77
    working with 76
custom monitor properties
    syntax 73
    working with 72
custom properties
    about 71
    creating 71
    displaying in columns 78
    in alert templates 76
    referencing 79
    selection list for 75
cygwin OpenSSH, installing on Windows 47

H

headerHTML property 77

I

interface
    customizing SiteScope 77

L

localization
    matching local date formats 14

M

match content
    using regular expressions 1
monitoring via Secure Shell (SSH) 41
monitors
    baselining 34
    custom properties for 73

O

OpenSSH for Windows, installing 55

R

regular expressions 1
    character classes 7
    defining 2
    examples for log files 17
    general date variables 13
    ignoring character case 10
    ignoring line breaks 11
    language and country specific date variables 14
    matching date coded log entries 20
    matching delimited log file entries 18
    matching numbers in log files 19
    matching patterns with metacharacters 6
    matching punctuation marks 7
    matching string literals 4
    pitfalls in working with 20
    preserving line breaks 11

quantifiers 8
retaining match values 11
search mode modifiers 10
SiteScope date variables 12
special substitution for monitor URL or file path 15
the * quantifier 9
using alternation 5
Rolling Baselines
about 29
disabling 32
enabling 31

S

secure monitoring with SSH 41
SiteScope
and SSH 41
customizing display 71
SSH clients 59
SSH clients
external 62
internal Java 59
using SSH version 2 63
SSH monitoring 41
configuration options 42
configuring UNIX servers for 44
configuring Windows servers for 45
cygwin OpenSSH 46
external SSH client 62
installing and configuring SSH server 47
installing Remote NT SSH files 56
internal Java SSH client 59
key based authentication 61
OpenSSH for Windows 46
password authentication for clients 60
SSH client options 58
using SSH2 63
using SSH2 with internal client 62
version compatibility 62
working ith SSH clients 59
Static Performance baselines 34
enabling 36
monitor properties for 39
setting thresholds using 37
understanding 35
system values
accessing in regular expressions 13

X

XML documents
example match content syntax 26
monitoring as URLs 25
using content match values 27