Syslog Provider Parse Map Structure and Usage

Technical Reference

December 2009

Contents

The dedicated Security Manager syslog provider uses a custom XML file called a parse map that lets you specify how the provider should parse incoming message data and map parsed parameters into Security Manager events for processing and log archival.

1010 Windows Server 2008 Enhanced Cryptographic Provider

1012 Windows Server 2003 Enhanced Cryptographic Provider (RSAENH)

This product may also claim FIPS compliance by use of one or more of the Open SSL cryptographic components listed below. These components were certified by the Open Source Software Institute and obtained the FIPS certificates as indicated.

918 - OpenSSL FIPS Object Module v1.1.2 - 02/29/2008 140-2 L1

1051 - OpenSSL FIPS Object Module v 1.2 - 11/17/2008 140-2 L1

1111 - OpenSSL FIPS Runtime Module v 1.2 - 4/03/2009 140-2 L1

Note: Windows FIPS algorithms used in this product may have only been tested when the FIPS mode bit was set. While the modules have valid certificates at the time of this product release, it is the user's responsibility to validate the current module status.

Overview

The dedicated Security Manager syslog provider uses a custom set of XML parsing instructions called a **parse map** that lets you specify how the provider should parse incoming message data and map parsed parameters into Security Manager events for processing and log archival.

This document provides an overview of the parse map structure and usage, starting with higher-level elements and then going deeper into the details of each element or sub-element.

The parse map is an XML-based document that follows the XML schema specified in the `ParseMapXMLSchema.xsd` schema definition file installed with Security Manager. The `.xsd` file is located in the OnePoint folder where the Development Console is installed. You should always refer to the schema document for the technical details of schema restrictions.

When you specify the parse map XML in the provider user interface, Security Manager validates the specified XML against the XSD schema before saving the parse map into the OnePoint database. Security Manager displays any errors encountered during the validation as error messages.

Basic XML Elements

This section outlines the high-level elements and child elements necessary for a syslog provider parse map, including both **filter rules** and **parsing rules**.

Parse Map Root Element

The parse map starts with a `parse-map` root element. The `parse-map` root element is the starting point of the parse map XML. For example:

```
<?xml version="1.0"?>
<parse-map name="Parse Map Example" content-
version="6.5.0.0">
   <event-timestamp-format/>
   <filter-rule/>
   <parse-rule/>
</parse-map>
```

The parse map `name` is a unique name that distinguishes the parse map for the content author, while the `content-version` value helps to identify the version of the parse map you are using. The `content-version` value is meant to reduce confusion if multiple versions of the parse map exist.

The parse map defines the matching mechanics of the syslog provider in a series of **filter** and **parsing rules**, specified by `filter-rule` and `parse-rule` elements, respectively. Each of these rules defines a matching pattern using regular expressions and specifies the associated action the provider takes if received data matches an expected pattern.

With a filter rule, the provider filters out message data deemed to be unimportant and does not capture that data. If the message data matches a pattern defined by a parsing rule, the provider captures the data and generates an event based on the format specified within that parsing rule.

The `event-timestamp-format` element is an optional element that defines the set of timestamp formats supported in the parse map. Specifying a list of time stamp formats indicates that the parsed message is expected to conform to one of the listed formats.

If none of the formats match, the provider instead uses the time at which the provider received the message. For more information about event timestamps, see “event-timestamp-format Element” on page 3.

The following table defines the attributes and child elements of the root-level `parse-map` XML element:

Element	Attribute	Child Element	Description
`parse-map`	`name`		Required string attribute
	`content-version`		Optional string attribute
		`event-timestamp-format`	Optional child element. Possible value range: 0 to unbounded. For more information about this element, see “event-timestamp-format Element” on page 3.
		`filter-rule`	Optional child element. Possible value range: 0 to unbounded. For more information about this element, see “Filter Rules” on page 4.
		`parse-rule`	Required child element (at least one). Possible value range: 1 to unbounded. For more information about this element, see “Parsing Rules” on page 6.

When creating a new `parse-map` XML element, keep the following considerations in mind:

- Received data can only match one parsing or filter rule. If the data matches the regular expression of a single parsing or filter rule, the provider does not continue to evaluate the data against other parse or filter expressions.

- The order of the child elements within the root `parse-map` element is important and is enforced in the schema. The `event-timestamp-format` child element needs to be specified first, followed by `filter-rule` elements and then by `parse-rule` elements, as applicable. The syslog provider always evaluates filter rules first, before parsing rule expressions.

event-timestamp-format Element

The first possible parse map element is the `event-timestamp-format` element. This element is optional, and you must specify the element before the filter and parsing rules.

The `event-timestamp-format` element determines the timestamp format expected in the syslog message parsed by the parse map. It is an optional property of the parse map, and if not specified, the provider bases the event timestamp on the time when the provider received the syslog message.

Here is an example of an `event-timestamp-format` element:

```
<event-timestamp-format>
   <timestamp-format>yyyy-MM-dd HH:mm:ss</timestamp-format>
   <timestamp-format>MMM dd yyyy HH:mm:ss Z</timestamp-
   format>
</event-timestamp-format>
```

The provider uses the timestamp formats while parsing the syslog message. The provider applies the timestamp formats on the `detecttime` field of the log archive event formatter and on the `EventTime` of the processing event formatter. For more information about event formatters, see “Log Archive Event and Processing Event Formatters” on page 9.

If there are multiple timestamp formats within an `event-timestamp-format` element, the provider uses the first successful format to parse the event time string. If all formats fail, the provider uses the time at which the provider received the syslog message.

The following table defines the child element of the `event-timestamp-format` element:

Element	Child Element	Description
`event-timestamp-format`	`timestamp-format`	Required child element. You must define at least one instance but can define an unlimited number of instances.

The date/time format is specified by means of a string time pattern. In this pattern, all ASCII letters are reserved as pattern letters, which are defined in the following table:

Symbol	Meaning	Example
G	era designator	`AD`
y	year	`1996`
M	month of year	`July` or `07`
d	day of month	`10`
h	hour in AM/PM format (1-12)	`12`
H	hour of day (0-23)	`0`
m	minute of hour	`30`
s	second of minute	`55`
S	fractional second	`978`

Symbol	Meaning	Example
E	day of week	`Tuesday`
e	day of week (local 1-7)	`2`
D	day of year	`189`
F	day of week in month	`2` (2nd Wednesday in July)
w	week of year	`27`
W	week of month	`2`
a	AM/PM marker	`PM`
k	hour of day (1-24)	`24`
K	hour in AM/PM format (0-11)	`0`
z	time zone	`Pacific Standard Time`
Z	time zone (RFC 822)	`-0800`
v	time zone (generic)	`Pacific Time`

When creating a new parse map with an `event-timestamp-format` element, keep the following considerations in mind:

- If you want to define the timestamp format, you must specify the timestamp format before any filter rules in the parse map. If you specify the timestamp format after any filter rules, Security Manager returns an error.
- Timestamp format parsing is an expensive operation in terms of performance. If you want to create a highly optimized parse map to rapidly parse incoming data, you may not want to specify event timestamp formats.

Rule Elements

A syslog provider parse map can contain both filter rules and parsing rules. The provider uses these two types of rules to evaluate received data and determine whether the provider needs to ignore or format the data for event processing or log archive storage.

Filter Rules

When you create a new parse map, you can specify one or more filter rules, each based on a regular expression, to filter out messages you do not want the syslog provider to collect. The provider does not capture messages that match the filter criteria for event processing or for log archive storage.

Filter rules apply the regular expression specified in the `CDATA` section of the `regex` property to incoming syslog messages. When a message matches the regular expression, the filter rule performs a filter action so the provider disregards the matching data and increments a performance counter that tracks filtered messages (if installed).

Here is an example of a filter rule:

```
<filter-rule name="Filter rule example 1" enabled="true">
   <match-regex>
      <property name="regex"><![CDATA[Regex-Expression]]></
      property>
      <syslog-message/>
   </match-regex>
   <filter-action/>
</ filter-rule>
```

The following table defines the attributes and child elements of the `filter-rule` element:

Element	Attribute	Child Element	Description
`filter-rule`	`name`		Required string attribute that uniquely identifies a filter rule.
	`enabled`		Required boolean attribute that specifies if the filter rule is active or enabled.
		`match-regex`	[1,1] Required child element. You can only define one instance. The child element specifies that the evaluation is based on a regular expression.
		`filter-action`	[1,1] Required child element. You can only define one instance. The child element specifies that the message is filtered out if the regular expression matched.
`match-regex`		`property`	[1,1] Required child element. You can only define one instance. The child element specifies that the regular expression to be matched in order for the filter action to take place
		`syslog-message`	[1,1] Required child element. You can only define one instance. The child element specifies that the filter rule processes the regular expression on a syslog message.

When creating a new parse map with one or more `filter-rule` elements, keep the following considerations in mind:

- The filter expression must be specified before any parsing rules. The XSD schema enforces ordering, so you must specify filters first, before the parsing rules.
- A filter rule `name` attribute must be unique to distinguish the rule from other parsing or filter rules.
- If the `enabled` attribute is false, the provider does not consider the filter rule for evaluation.

- The name of the `property` child element of a filter rule must be `regex`. When you specify this name, the parsing engine knows to identify the `CDATA` section of that `property` element as the regular expression.
- If multiple provider instances have filter expressions, the provider evaluates data using the combined filters first and then the combined parsing rules second in the order of evaluation.

Parsing Rules

A parsing rule works by matching received data against a specified regular expression and executing a specified response. The rule evaluates the syslog message against the regular expression. If the data matches, the rule executes the corresponding formatters.

A `format-log-archive-event` child element generates a log archive record that the central computer then sends to the log archive server. Security Manager does not process that log archive event in the regular workflow designed to process events for alerts or responses.

A `format-processing-event` child element generates a Security Manager event, and the provider submits that event into the Security Manager processing workflow. Security Manager executes event processing rules developed in the Development Console against the generated event.

Here is an example of a parsing rule:

```
<parse-rule name="Parse Rule Example 1" enabled="true">
   <match-regex>
      <property name="regex"><![CDATA[Regex-Expression]]></
      property>
      <syslog-message/>
   </match-regex>
<format-log-archive-event/>
<format-processing-event/>
</parse-rule>
```

Notes

- Note the central computer does not submit events generated for the purpose of event processing to a log archive.
- A parsing rule can specify both formatters, which means that the parsing instructions generate events for purposes of both archival and event processing.

The following table defines the attributes and child elements of the `parse-rule` element:

Element	Attribute	Child Element	Description
`parse-rule`	`name`		Required string attribute that uniquely identifies a parse rule.
	`enabled`		Required boolean attribute that determines if the parse rule is active/enabled.
		`match-regex`	[1,1] Required child element. You can only define one instance. The child element specifies that the evaluation is based on a regular expression.
		`format-log-archive-event`	[0,1] Optional child element. You can only define one instance. The child element specifies that the rule generates a log archive event based on the formatting specified in this XML element. A generated log archive event is saved on the log archive server. For more information about this element, see “Log Archive Event and Processing Event Formatters” on page 9.
		`format-processing-event`	[0,1] Optional child element. You can only define one instance. The child element specifies that the rule generates an event for purposes of event and alert processing. The generated event is processed by the Security Manager workflow. For more information about this element, see “Log Archive Event and Processing Event Formatters” on page 9.

When creating a new parse map with one or more `parse-rule` elements, keep the following considerations in mind:

- A parsing rule `name` attribute must be unique to distinguish the rule from other parsing or filter rules.
- If the `enabled` attribute is `false`, the provider does not consider the parse rule for evaluation.
- The `name` of the `match-regex` child element of the parsing rule must be `regex`. When you specify this name, the parsing engine knows to identify the `CDATA` section of that `match-regex` element as the regular expression.

- A single parsing rule can generate both a log archive event and an event for event processing. However, note that because event processing incurs more processing overhead than archiving an event, large numbers of event processing operations can impact log archive processing operations.
- All parsing rules regular expression evaluations are mutually exclusive. Once a provider evaluates data against a regular expression and finds a match, the provider does not evaluate the rest of the parsing rules.
- Order is important. You should consider putting the parsing rule that is the most common and has the highest chance of matching received data at the top of your chain of the parsing rules.

Enable Catch All Option

When you create a new syslog provider using the Development Console, you can select the **Enable Catch All** option. When you select this option, if a syslog message does not match any existing filter or parsing rules, the provider captures the message as a log archive event. If you enable this option, you can then inspect those messages by running Forensic Analysis queries in the Control Center.

The catch-all option configures the provider to capture the following information:

Detect time
The time at which the provider received the syslog data.

Time zone offset
The time zone offset of the agent hosting the provider.

Analyzer model
The analyzer model name, which will be `Generic Syslog`.

Classification origin
The classification name, which will be `Generic Syslog`.

Severity
The severity, which will always be `low`.

Syslog message
The message contained in the message field.

A provider with the catch-all option enabled captures the syslog message and sends the message data directly to the log archive. The provider does *not* generate a real-time event based on catch-all logic.

In order for the catch-all option to be active, your environment and provider must meet the following conditions:

- You must enable a log archival collection rule on that provider instance or on any provider instance sharing the same port.
- You must have enabled the catch-all option in the Development Console.

The catch-all option is selected by default. You can disable the option using the Development Console.

To disable the catch-all syslog provider option:

1. Log on to the Development Console computer with a user account that is a member of the OnePointOp ConfgAdms group. For more information about groups and permissions, see the *Installation Guide for NetIQ Security Manager.*
2. Start the **Development Console** in the NetIQ Security Manager program group.
3. In the left pane, expand **Security Manager Development Console** > **Advanced** > **Providers**.
4. In the right pane, select the existing syslog provider for which you want to disable the catch-all option.
5. On the Action menu, click **Properties**.
6. Click **Configure XML**.
7. Clear **Enable Catch All**.
8. Click **OK**.
9. Click **OK**.
10. Close the Development Console.

Syslog Event Formatting

After the syslog provider parses received data according to any parsing rules included in your parse map, the parse map uses one or more event formatters to create events based on the parsing rule instructions.

Log Archive Event and Processing Event Formatters

The log archive and processing event formatters share similar XML schema. The main differences are the property names used to describe the event fields for each formatter.

The log archive formatter uses property field names that are described in the log archive `Fieldmap.xml` file included in the `Security Manager Self-monitoring.nqm` file included in the `Modules` folder of your Security Manager installation kit. If you want to view the `Fieldmap.xml` file, unzip the `Security Manager Self-monitoring.nqm` file and open `Fieldmap.xml` in a text editor.

The processing event formatter creates an event with field names known to the Security Manager processing workflow. Those field names are known as **intrinsic properties**, and Security Manager uses those names within a processing event formatter.

The following is the schema for the `format-log-archive-event` elements:

```
<format-log-archive-event>
   <property-list/>
   <mapped-property-list/>
   <lookup-property-list/>
</format-log-archive-event>
```

The following table defines the child elements of the `format-log-archive-event` element:

Element	Child Element	Description
`format-log-archive-event`	`property-list`	[1,unbounded] Required child element. You can define multiple instances. You must define at least one property within a `property-list` child element. A property represents a field within a formatted event. You need to define at least one field for the event to be valid. For more information about event properties, see “Formatting Event Properties” on page 12.
	`mapped-property-list`	[0,unbounded] Optional child element. You can define multiple instances. The child element is an optional mapping expression. For more information about this element, see “Mapped Properties” on page 15.
	`lookup-property-list`	[0,unbounded] Optional child element. You can define multiple instances. The child element is an optional lookup expression. For more information about this element, see “Lookup Properties” on page 17.

The following is the schema for the `format-processing-event` element:

```
<format-processing-event>
   <property-list/>
   <mapped-property-list/>
   <lookup-property-list/>
</format-processing-event>
```

The following table defines the child elements of the `format-processing-event` element:

Element	Child Element	Description
`format-processing-event`	`property-list`	[1,unbounded] Required child element. You can define multiple instances. You must define at least one property within a `property-list` child element. A property represents a field within a formatted event. You need to define at least one field for the event to be valid. For more information about event properties, see “Formatting Event Properties” on page 12.
	`mapped-property-list`	[0,unbounded] Optional child element. You can define multiple instances. The child element is an optional mapping expression. For more information about this element, see “Mapped Properties” on page 15.
	`lookup-property-list`	[0,unbounded] Optional child element. You can define multiple instances. The child element is an optional lookup expression. For more information about this element, see “Lookup Properties” on page 17.

When creating a new parse map with one or more formatter elements, keep the following considerations in mind:

- You can specify one or both types of formatters within a parsing rule. The event processing formatter generates a Security Manager event to be processed by the workflow, while the log archive formatter generates a log archive record to be sent to the log archive.
- The field and property names used by the log archive formatter *must* be defined in the global field map. If the property name is not present, the provider instance cannot load.
- The field and property names used by the event processing formatter must be included in the list of intrinsic properties of a Security Manager event. For more information about intrinsic properties, see “Formatting Event Properties” on page 12.
- Because the log archive formatter generates an event to send to the log archive and does not send the event to be evaluated by the Security Manager workflow, existing log archive filter rule or criteria within a log archive collection rule do not apply to the generated log archive event.

- The agent deactivates the log archive formatters in a parse map if there are no log archive rules associated with the agent's provider instance.
- The agent deactivates the event processing formatters in a parse map if there are no real-time rules of any type associated with the agent's provider instance.

Note
Enabling high-performance log archive mode deactivates all event processing formatters. With high-performance log archive mode enabled, only the log archive formatter is active.

Formatting Event Properties

This section defines the list of properties used as fields within a log archive record or a Security Manager event processed by the workflow.

In log archive formatters, the global field map defines the fields. For a Security Manager processing event, the intrinsic properties of the real-time processing event define the fields.

The following is an example of a property list for a log archive formatter:

```
<property-list>
   <property name ="detecttime" value ="%4%" />
   <property name ="analyzer.model" value ="Device Model
   Name" />
   <property name ="classification.name" value ="%3%" />
   <property name ="message" value ="Message: %0%" />
</property-list>
```

The following is an example of a property list for a processing event formatter:

```
<property-list>
   <property name ="EventNumber" value ="%1%" />
   <property name ="EventType" value ="4" />
   <property name ="UserName" value ="%2%" />
   <property name ="UserDomainName" value ="%3%" />
   <property name ="Category" value ="%4%" />
   <property name ="Message" value ="%0%" />
   <property name ="Parameter 2" value ="%7%" />
</property-list>
```

The values of the properties are defined in the following format:

- %#% represents the regular expression substitution/capture string or value, whose index is #, starting with `1` to indicate the first matching substring index value. For example, `name ="UserDomainName" value ="%3%"` means that the provider uses the third captured string or value of the regular expression evaluation as the user domain name.
- Literals are specified without an enclosing % sign. For example, `name ="Parameter 7" value ="This is a literal value"` results in the formatter using the full string specified as the value for the `Parameter 7` field.
- The special regular expression capture index `0` indicates the original data message against which the regular expression pattern is evaluated. For example, `name ="Message" value ="%0%"` results in the formatter including the full original data message in the message field of the event.

The list of intrinsic properties used by a processing event formatter are as follows:

ProviderInstance
`ProviderInstance` is the GUID of the provider instance. The provider automatically sets the value of this property, but you can overwrite the default property by specifying a new value in the parse map.

Default Value: The provider instance GUID associated with the parse map

EventNumber
`EventNumber` is the event identifier or number of the generated Security Manager event.

Default Value: event number 25256

EventType
`EventType` is the event type of the generated Security Manager event.

Default Value: An informational event type

EventTime
`EventTime` is the timestamp of the generated Security Manager event.

Default Value: The timestamp when the provider received the syslog message

Category
`Category` is the event category of the generated Security Manager event.

ProviderName
`ProviderName` is the provider name associated with the generated Security Manager event. The provider automatically sets the value of this property, but you can overwrite the default property by specifying a new value in the parse map.

Default: The provider instance name associated with the parse map

Message
`Message` is the message or description field of the generated Security Manager event.

Default: The text of the syslog message

UserName
`UserName` is the user name associated with the generated Security Manager event.

UserDomainName
`UserDomainName` is the domain of the user name associated with the generated Security Manager event.

Computer
`Computer` is the name of the computer that is the source of the syslog message.

Default Value: The IP address of the computer that sent the syslog message, discovered using a reverse lookup

Domain
`Domain` is the domain of the source computer that is the source of the syslog message.

SourceName
`SourceName` is the event source field of the generated Security Manager event.

`Parameter #`

Numbered `Parameter` properties where # is between `1` and `99` represent Security Manager event parameters 1 to 99.

The following table defines the child elements and attributes of the `property-list` element:

Element	Child Element	Attribute	Description
`property-list`	`property`		[1,unbounded] Required child element. You can define multiple instances. You must define at least one `property` child element. Each `property` element specifies a field in the generated event.
		`name`	Required string attribute that specifies the name of the field to be added into the formatted event.
		`value`	Required string attribute that specifies the matching substring from the parsed regular expression evaluation.

Optional Event Formatting

In addition to basic event formatting, you can also use the parse map to specify custom formatting for one or more parsed events, including concatenation formatting and mapped or lookup expressions.

Concatenation Formatting

You may need to format a field value within a record or event using concatenated values and strings from the parsed expression. In those cases, you can provide a concatenation format as a value for a field within a parse map.

The concatenation format is only supported for fields defined as being of the `string` type in the field map, including static intrinsic property fields. Using a concatenation format for the value of a field that is not of the `string` type results in an error when the provider loads the parse map.

Place the concatenation format in the value attribute of a property element in the parse map.

For example:

```
<property-list>
   <property name ="Parameter 34" value ="%7% + %8% + %9%" />
</property-list>
```

The property element above instructs the formatter to concatenate the captured values from indexes `7`, `8`, and `9`.

This applies for both log archive and real-time event formatters in the parse map.

The concatenation format consists of:

Operator
The concatenation format defines the + (plus sign) as the concatenation operator between the different string literals and matching substrings.

Operands
Operands can either be:

- String literals enclosed in single quotes. For example, `'abc'` and `'xzy'`.
- Matching substring index numbers enclosed in percentage symbols. For example, %4%.

The following examples below illustrate the usage of the concatenation format:

For a more detailed example, if the concatenated format is:

```
'%3%+'-'+%5%+'-'+%7%'
```

And the parsed captured values are `10`, `High`, and `Connection Accepted`, the formatted value is:

```
10-High- Connection Accepted
```

Notes
- The parser considers any values enclosed in single quotes to be string literals.
- The parser ignores spaces between the plus operators.
- The parser preserves spaces within a string literal.

Mapped Properties

Both the log archive and real-time formatters support mapped properties.

A single `mapped-property` element defines the mapping scheme and has the following required attributes:

`name`
`name` is the name of the field to be inserted into the event after the provider evaluates the mapping expression.

`value`
`value` is the value the provider uses for the lookup. The value of this attribute must be a matching substring that is a result of the regular expression evaluation.

If the mapping does not match any of the mapping keys and the parse map does not provide a default, then the provider does not insert the mapped property into the event, as the mapping did not result in a valid value.

The `mapped-property-list` element is a container element that contains one or more mapped properties.

The following is an example of a `mapped-property-list` element within a formatting node:

```
<mapped-property-list>
   <mapped-property name="classification.name" value="%7%" >
```

```
      <key-value key="tcp" value="reliable"/>
      <key-value key="udp" value="unreliable"/>
      <key-value key="nttp" value="async"/>
      <default-mapping value="unknown"/>
   </mapped-property>
</mapped-property-list>
```

The following table defines the child elements and attributes of the `mapped-property-list` element:

Element	Attribute	Child Element	Description
`mapped-property-list`		`mapped-property`	[1,unbounded] Required child element. You can define multiple instances. You must specify at least one `mapped-property` child element. Each `mapped-property` specifies a field where the provider evaluates the mapping expression.
`mapped-property`	`Name`		Required string attribute that specifies the name of the field that holds the results of the mapping expression.
	`Value`		Required string attribute that specifies the matching substring from the parsed regular expression evaluation used during the mapping conversion.
		`key-value`	[1,unbounded] Required child element. You can define multiple instances. The child element is a key value mapping exercised on the value of the `mapped-property` element. If the `mapped-property` element value matches the key of this `key-value` element, the value specified in this element is used.
		`default-mapping`	[0,1] Optional child element. You can only define one instance. The child element is an optional default value for mapping. If specified and none of the provided `key-value` mapping elements matched, the default value provided in this element is used.

When creating a new parse map with one or more mapped property elements, keep the following considerations in mind:

- The parse map represents key/value pairs using the child element `key-value`.
- The `key-value` element has two attributes. The `key` attribute represents the mapping key, and the `value` attribute is the value of the mapped key.
- The map lookup is case-insensitive.
- The `value` of the `key-value` element can be a literal string or the index of a regular expression evaluation result.

Lookup Properties

Both the log archive and real-time formatters support lookup properties.

A `lookup-property` element has the following required attributes:

`name`
`name` is the name of the field to be inserted as a looked-up field.

`lookup-value`
`lookup-value` is the value to be used during the lookup. It must be the matching substring index.

`lookup-type`
`lookup-type` is the type of the lookup.

The `lookup-property-list` element is a container element that contains one or more lookup properties.

The following is an example of a `lookup-property-list` element within a formatting node:

```
<lookup-property-list>
   <lookup-property name="Parameter 3" lookup-value="%8%"
   lookup-type="IP_LOOKUP" />
</lookup-property-list>
```

The following table defines the child elements and attributes of the `lookup-property-list` element:

Element	Child Element	Attribute	Description
`lookup-property-list`	`lookup-property`		[1,unbounded] Required child element. You can define multiple instances. The mapped property list is a container element of one or more lookup properties.
		`name`	Required string attribute that specifies the name of the field that holds the results of the lookup expression.
		`lookup-value`	Required string attribute that specifies the matching substring index from the parsed regular expression evaluation used during the lookup.
		`lookup-type`	Required string attribute that specifies the type of the lookup operations. Currently supporting IP lookup with the value `IP_LOOKUP`.

When creating a new parse map with one or more lookup property elements, keep the following considerations in mind:

- You can use a `lookup-property-list` element to define multiple lookup properties.
- A lookup property currently allows for an IP lookup from a host name.

IP Lookup Property

Currently, the only lookup property implemented for syslog providers is an IP lookup property. If you want to define an IP lookup operation in a parse map, you must specify a lookup-type of IP_LOOKUP. The IP_LOOKUP type indicates that the lookup is an IPv4 lookup.

Note
Other types of IP lookup operation are currently not supported. If you try to define another type of IP lookup, the provider cannot load the parse map and returns an error.

If you want to look up the host name for a dual protocol stack computer, the lookup returns the host name of the IPv4 computer, because the preference is for an IPv4 lookup. If the computer is an IPv6-*only* machine, then the lookup will results to IPv6 since it was the only available IP address.

If the lookup operation fails, the provider does not add the lookup field to the record.

For performance reasons, the provider caches the IP lookup. The caching mechanism works as follows:

- A caching operation caches the host name and the resolved IP address of a computer.
- The caching operation also caches unresolved IP addresses of those computers for which the provider cannot perform a DNS lookup.
- When parsing an IP lookup property, the provider performs a cache lookup on the computer. If the provider finds a record for that host name in the cache, the operation returns the resolved IP address currently stored in the cache. If the cache does not contain a record for that host name, the provider then performs an actual DNS query to resolve the IP address of that computer. The provider adds the newly looked-up address to the cache.
- If a DNS query for a host name fails, the provider caches the result of that failure to prevent future lookup on an unknown host name.
- The provider grooms both type of caches, the one containing resolved addresses and the one containing irresolvable host namehost names, based on a configurable interval. The provider removes older lookups from the cache if the time on which the lookup was performed is older than a certain time.
- When a cache reaches its configured capacity, the provider raises a warning message in the event log to indicate to the user that no more caching can be performed and all future lookups will require an expensive DNS lookup. The provider advises the user to revise the cache maximum value to larger than the default value we ship in the product.

You can control the IP lookup caching mechanics using the registry on the provider computer:

`HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Security Manager\Configurations\<Configuration name>\Operations\Agent\LookupParams`

The following registries are supported:

`IPLookupMaxCacheSize`
: `IPLookupMaxCacheSize` is the maximum number of entries the cache can contain. The default value is `10000`.

`IPLookupCacheCleanupInterval`
: `IPLookupCacheCleanupInterval` is the interval, in minutes, for how often the provider cleans old lookup results from the cache. The default value is `180` (3 hours).

`ResolvedIPCacheExpiry`
: `ResolvedIPCacheExpiry` is the number of minutes the cache can store a resolved IP address before the address is considered aged and removable. The default value is `720` (12 hours).

`IrresolvableIPCacheExpiry`

`IrresolvableIPCacheExpiry` is the number of minutes the cache can store a failed lookup address before the provider tries to resolve the address again. The default value is `360` (6 hours).

Note

You can disable IP lookup capability completely in a provider instance using the parse map user interface. Disabling IP lookup capability means the provider does not attempt an IP lookup regardless of whether or not the parse map contains an IP lookup. This capability is available primarily for purposes of performance tuning, as IP lookup is an expensive operation.

Configuring and Using Syslog Parsing Statistics

You can configure your syslog provider to collect statistics about syslog parsing and write those statistics to a log on the agent computer. You can then use this statistical information to tune the parse map of your syslog provider to better fit incoming syslog data.

For example, you can use parsing statistics to determine which parsing rule the received syslog messages most frequently match and modify your parse map so that parsing rule is first in the execution order. Because the syslog provider parses incoming messages in order, you can optimize the speed at which your provider processes messages.

Enabling Syslog Parsing Statistics

If you want to view syslog parsing statistics on the agent computer, you must first enable parsing statistics in the agent computer registry.

To enable syslog parsing statistics:

1. Log on to the agent computer using an account that has permission to edit the Windows Registry and stop and start services.
2. Update the following registry entry using the Registry Editor:

   ```
   HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Security
   Manager\Configurations\ConfigurationGroupName\Operations\Age
   nt\Event Providers\Syslog\EnableParsingStatistics = true
   ```

   Where *`ConfigurationGroupName`* is the name of your configuration group.

   Warning

   Be careful when editing your Windows Registry. If there is an error in your registry, your computer may become nonfunctional. If an error occurs, you can restore the Registry to its state when you last successfully started your computer. For more information about editing the registry, see the Help for the Windows Registry Editor.

   ---

3. Close the Registry Editor.
4. Open the Component Services Administrative Tool located in the Control Panel.
5. In the left pane, click **Services**.

6. In the Services pane, click **NetIQ Security Manager Agent**.

7. On the Action menu, click **Restart**.

8. After the service restarts, close the Services Administrative Tool.

Viewing Syslog Parsing Statistics

After you enable syslog parsing statistics, the syslog provider starts collecting data on parsing operations. When the agent service stops, the provider adds the following set of standard parsing statistics entries to the `NqSmSvc.txt` log file on the agent computer:

`Session start time`
: The date and time at which the provider started processing incoming syslog messages.

`Session end time`
: The date and time at which the provider stopped processing incoming syslog messages.

`Session duration`
: The duration of the particular session during which Security Manager gathered syslog parsing statistics. This value represents the total period of time in which thoe syslog provider remained active, before the service restarted or shut down.

Note
If you make a configuration change to the agent, Security Manager restarts this counter.

`Total Number of messages evaluated`
: The total number of messages processed by the parsing engine, whether or not any of the messages matched one or more rules.

In addition, the syslog provider includes the following statistics for each configured parsing rule:

`Rule Name`
: The name of the parsing rule, as specified in the parse map. Security Manager adds a parsing rule to the statistics in the order in which Security Manager executes or evaluates the rule.

`Num. Evaluated Messages`
: The number of syslog messages the specific parsing rule evaluated, whether or not any of the messages matched the rule.

`Num. Matched Messages`
: The number of messages matching the specific parsing or filter rule.

Note
The syslog provider writes parsing statistics to the `NqSmSvc.txt` log file any time you stop the `NetIQ Security Manager Agent` service.

The provider also writes parsing statistics to the log when a central computer makes a configuration change, because a configuration change causes the `NetIQ Security Manager Agent` service to reload all providers.

To view syslog parsing statistics:

1. Log on to the agent computer using an account that has permission to edit the Windows Registry and stop and start services.
2. Navigate to the `C:\Documents and Settings\All Users\Application Data\NetIQ\Security Manager\Log Files` folder.
3. Open `NqSmSvc.txt`.
4. Navigate to the end of the log and search upwards for `Printing parsing statistics` to find the latest syslog parsing statistics.
5. View statistics for all configured parsing or filter rules.
6. Close the `NqSmSvc.txt` file.

Optimizing a Parse Map

You can use the parsing statistics for a particular syslog provider to optimize the parse map the provider uses.

The provider evaluates syslog messages against parsing or filter rules in order based on the parse map. When a message matches a rule, the provider does not evaluate the message against any subsequent rules.

If you want the provider to evaluate received syslog messages in the most efficient way possible, edit the parse map XML file and move any rules that the parsing statistics highlight as frequently matching syslog messages to the top of the parse map definition.

For example, if the parsing statistics show the `Cisco Firewalls - Login Rule` matched 75 syslog messages but the rule is last in the list, while the `Cisco Firewalls - Enable Password Rule` matched only 12 messages but is first in the list, move the `Login Rule` so the provider evaluates that rule first.

You may need to modify your parse map on a regular basis, depending on the syslog messages your provider receives.

Disabling Syslog Parsing Statistics

Because the syslog provider collects parsing data as it processes incoming messages, enabling syslog parsing statistics can cause a performance impact, particularly in high-volume environments.

For that reason, you may not want the syslog provider to log statistics indefinitely. After using parsing statistics to optimize your parse map, you can disable syslog parsing statistics in the agent computer registry.

To disable syslog parsing statistics:

1. Log on to the agent computer using an account that has permission to edit the Windows Registry and stop and start services.
2. Update the following registry entry using the Registry Editor:

```
HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Security
Manager\Configurations\ConfigurationGroupName\Operations\Age
nt\Event Providers\Syslog\EnableParsingStatistics = false
```

Where *ConfigurationGroupName* is the name of your configuration group.

Warning
Be careful when editing your Windows Registry. If there is an error in your registry, your computer may become nonfunctional. If an error occurs, you can restore the Registry to its state when you last successfully started your computer. For more information about editing the registry, see the Help for the Windows Registry Editor.

---

3. Close the Registry Editor.

4. Open the Component Services Administrative Tool located in the Control Panel.

5. In the left pane, click **Services**.

6. In the Services pane, click **NetIQ Security Manager Agent**.

7. On the Action menu, click **Restart**.

8. After the service restarts, close the Services Administrative Tool.