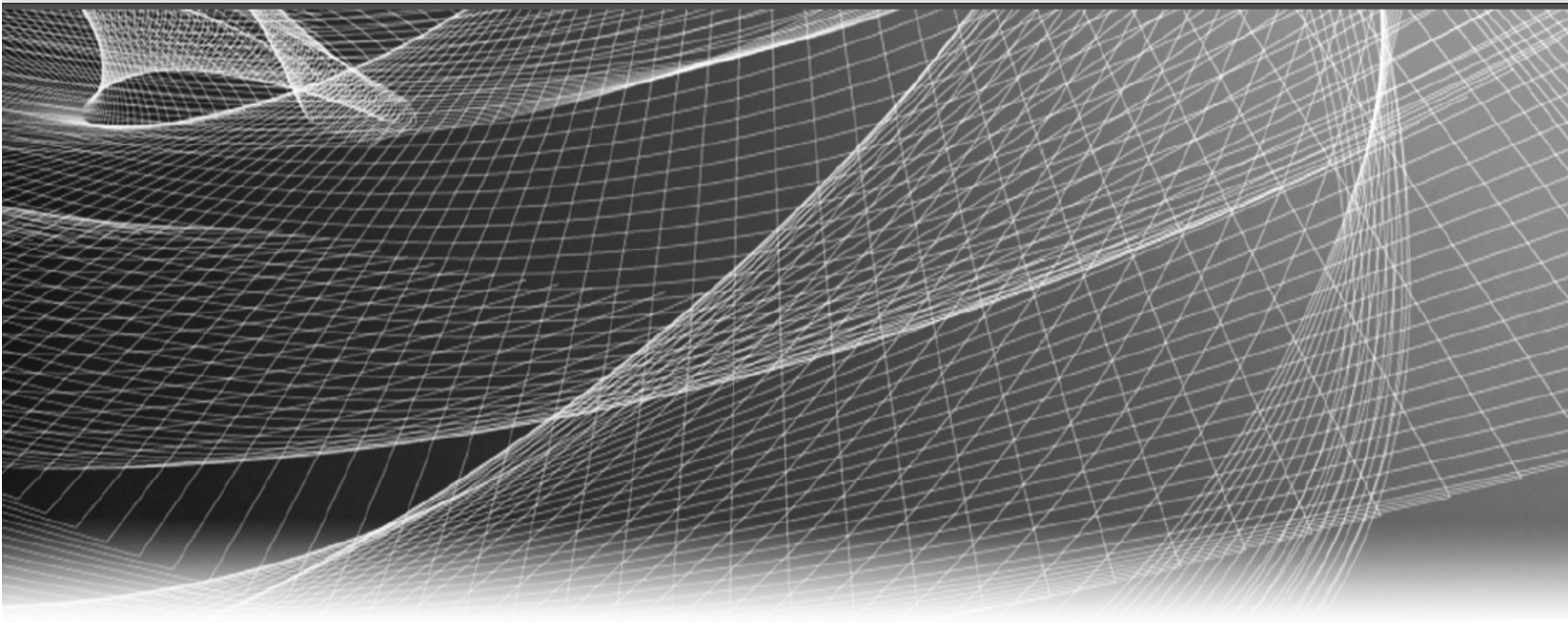


RSA Security Analytics

Decoder and Log Decoder Configuration Guide

for Version 10.6.5

RSA EMC²

Contact Information

RSA Link at https://community.rsa.com contains a knowledgebase that answers common questions and provides solutions to known problems, product documentation, community discussions, and case management.

Trademarks

For a list of RSA trademarks, go to www.emc.com/legal/emc-corporation-trademarks.htm#rsa.

License Agreement

This software and the associated documentation are proprietary and confidential to EMC, are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright notice below. This software and the documentation, and any copies thereof, may not be provided or otherwise made available to any other person.

No title to or ownership of the software or documentation or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software and the documentation may be subject to civil and/or criminal liability.

This software is subject to change without notice and should not be construed as a commitment by EMC.

Third-Party Licenses

This product may include software developed by parties other than RSA. The text of the license agreements applicable to third-party software in this product may be viewed on the product documentation page on RSA Link. By using this product, a user of this product agrees to be fully bound by terms of the license agreements.

Note on Encryption Technologies

This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when using, importing or exporting this product.

Distribution

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

November 2017

Contents

Decoder and Log Decoder Configuration Guide 11

Decoder and Log Decoder Basics 12

Decoder and Log Decoder Required Procedures 13

- Step 1: Verify System Configuration 13
- Step 2: Configure Capture Settings 13
- Step 3: Enable or Disable Parsers 13
- Step 4: Configure Decoder Rules 13
- Step 5: Start and Stop Data Capture 13

Step 1. Verify System Configuration 15

- Procedure 15

Step 2. Configure Capture Settings 17

- Procedure 17
- VLAN Fixup Configuration 19

Configure System-Level (BPF) Packet Filtering 20

- Add System-Level Packet Filters 20
- Examples 22
- Testing 23
- Conversions 23

Step 3. Enable and Disable Log Parsers 24

- Prerequisites 24
- Procedure 24
- Result 25

Step 4. Configure Decoder Rules 26

- Network Layer Rules 26
- Application Layer Rules 26
- Correlation Rules 26
- Common Uses 26
- Rule Sets 27

Rule Processing 27
Rule Configuration 27
Capture Rule Syntax 28
Procedures 29
Configure Capture Rules 29
Add a Rule 31
Remove a Rule 31
Edit a Rule 31
Disable a Rule 31
Enable a Rule 32
Import Rules from a File 32
Export a Rule to a File 33
Push Rules to Other Services 34
Push Selected Rules 34
Push All Rules 35
Change Execution Order of Rules 36
Restore a Rule Snapshot from History 36

Configure Application Rules 38

Sample Application Rules 38
Procedures 38
Navigate to the App Rules Tab 38
Add or Edit an Application Rule 39

Configure Correlation Rules 42

Sample Correlation Rules 42
Explanation 43
Procedures 43
Navigate to the Correlation Rules Tab 43
Add or Edit a Correlation Rule 44

Configure Network Rules 47

Sample Network Rules 47
Procedures 47
Navigate to the Network Rules Tab 47
Add or Edit a Network Rule 48

Step 5. Start and Stop Data Capture **51**
Procedure 51
Decoder and Log Decoder Additional Procedures **53**
Configure Feeds and Parsers **54**
Procedures 54
Configure Parsers 54
Configure Feeds 55
Create and Deploy Custom Feeds Using a Wizard **56**
Sample Feed Definition File 56
Feed Definition Equivalents for Custom Feed Wizard Parameters 57
Create a Custom Feed **61**
Create an Identity Feed **72**
Prerequisites 72
Create an Identity Feed 72
Import the SSL Certificate 81
Cannot Verify Identity Feed URL 82
Edit a Custom Feed **84**
Use Custom Parsers **87**
Procedures 87
Upload Parsers to a Decoder or Log Decoder 87
Manage Upload Jobs 90
Delete Deployed Parsers 90
Configure 10G Capability **92**
Hardware Prerequisites 92
Software Prerequisites 93
10G Decoder Installation 93
Prerequisites 93
BIOS Installation Instructions 93
Update 10G Decoder 94
Install 10G Decoder 94
Configure 10G Decoder 95

Storage Considerations **98**
Using the Series 4S Hardware (With Two or More DAC Units) 98
Using SAN Storage 98
Parsing and Content Consideration for Packet Capture 99
10G Best Practices 99
Aggregation on a 10G Decoder to Other Security Analytics Components 99
Decoder 100
Storage Considerations 100
Using the Series 4S hardware, with two DAC units 100
Other Storage Configurations (SAN, etc.) 100
Parsing at High Speeds 101
Tested Live Content 101
Aggregation on a 10G Decoder 102
Configure Syslog Forwarding to Destination **103**
Prerequisites 103
Procedure 103
Create Custom Meta Keys Using Custom Feed **106**
Procedure 106
Add custom meta key in Log Decoder 106
Deploy feed in Live 106
Add the custom meta entry in Concentrator index file 109
Investigate 109
Additional Procedures 110
Configure Parser Mappings **115**
Update IP to Event Source Mapping 115
Read IP to Event Source Type Mappings 118
Edit an IP to Event Source Type Mapping 119
Delete an IP to Event Source Type Mapping 120
Sort the Hostname or Event Source Type 120
Import IP to Event Source Mapping Entries 120
Export IP to Event Source Mapping Entries 121
Search IP to Event Source Mapping Entries 122
Fix Rules with Deprecated Syntax **124**
Procedure 124

Enable or Disable Lua and Flex Parsing Systems 127
Procedure 127
Map IP Address to Service Type 128
Procedure 128
IPdevice Command 128
Time Zone Support 129
Result 129
Procedure 129
iptmzone Command 130
Examples 130
Event Time Support 131
Result 132
Procedure 132
Missing Year Support 133
Latent log during transition to new year 133
Logs from forward time zones (or skewed clocks) just before new year transition 133
Latent logs received where a leap day cannot be successfully assigned to an appropriate leap year 133
Limitations 134
Examples 134
Upload Log File to a Log Decoder 136
Procedure 136
Upload Packet Capture File 138
Procedure 138
Verify Decoder System Information 140
Procedure 140
Configure a Log Decoder to Accept Protobuf 142
Procedure 142
Decoder and Log Decoder References 145
Services Config View - Data Privacy Tab 146
Features 146

Services Config View - Feeds Tab 148
Features 149
Feeds Tab Toolbar 149
Feed Grid 149
Upload Feeds Dialog 151
File Grid 152
Upload Job Grid 152
Upload Feeds Dialog Buttons 152
Services Config View - Files Tab 154
Feed Definitions File 156
feed-definitions.xml 156
Flex Parser 157
NwFlex.xml 157
Arithmetic Functions 159
Language Definition 159
Common Parser Operations 161
Match Port and Identify Immediately 161
Match Port and Delay Identification 161
Match Token and Identify Immediately 162
Match Multiple Tokens 162
Match Token and Create Metadata 163
General Functions 164
General Functions Language Definition 164
Logging Functions 167
Language Definition 167
Nodes 168
Nodes Language Definition 168
Payload Functions 174
Language Definition 174

Regex **177**
Language Definition 177
String Functions **178**
String Functions Language Definition 178
Geo IP Parser **182**
GeoPrivate.ipl 182
Lua Parsers **183**
List of Lua Parsers 183
Search Parser **184**
search.ini 184
search.ini Search String Syntax **185**
Syntax 185
Parameters 185
Example 186
Wireless LAN Configuration **187**
wlan-config.xml 187
Services Config View - General Tab **188**
Features 189
System Configuration 189
Decoder Configuration 191
Adapter 192
Cache 193
Capture Settings 194
Database Max File Sizes 196
Hash 197
Parsers Configuration 197
Additional Service Parsers Configuration for Log Decoder 199
Services Config View - Parser Mappings Tab **201**
Features 202
Parser Mappings Toolbar 202
Parser Mappings Grid 202

Services Config View - Parsers Tab 203
Features 204
Parsers Tab Toolbar 204
Parser Grid 204
Services Config View - Rules Tabs 205
Rules Tab Toolbar 205
Rules Actions Menu 206
Rules Grid Context Actions 207
App Rules Tab 209
Application Rules Tab Columns 210
Rule Editor Dialog 210
Correlation Rules Tab 214
Network Rules Tab 217
Features 218
Supported Meta Keys in Network Rules 221
Supported Meta Keys in Network Rule Conditions 221
Rule and Query Guidelines 223
Rule Examples 223
Strict Mode Configuration for Security Analytics 10.6 224
Valid Syntax with the Modern Parser 224
Ambiguous Syntax Examples with the Legacy Parser 225
Services System View - Decoders 227
Features 228
Service Info Toolbar 228

Decoder and Log Decoder Configuration Guide

This topic introduces the Decoder and Log Decoder and the methodology for configuring them in Security Analytics.

Topics

- Decoder and Log Decoder Basics
- Decoder and Log Decoder Required Procedures
- Decoder and Log Decoder Additional Procedures
- Decoder and Log Decoder References

Decoder and Log Decoder Basics

This topic introduces the Decoder and Log Decoder in RSA Security Analytics.

Security Analytics supports two types of Decoders:

- The Decoder, which captures network data in packet form.
- The Log Decoder, which captures log data as events.

A Log Decoder is a special type of Decoder, and is configured and managed in a similar way to a Decoder. Therefore, most of the information in this section refers to both types of Decoders. Differences for Log Decoders are noted.

Adding a Decoder makes it visible and available for use with Security Analytics Administration, Live, and Investigation. To add a service in Security Analytics, you select the service type, provide service connection information, and validate that the service can be reached.

Configuring the Decoder to capture data involves selecting a capture adapter and choosing cache and capture settings.

When the Decoder is available in Security Analytics, it is ready to capture traffic. You can configure each Decoder to control the type of traffic captured using rules, feeds, and parsers.

Decoder and Log Decoder Required Procedures

These are the required configuration steps for a new Decoder or Log Decoder, and also for changing the configuration of an existing Decoder. Unless otherwise stated, Decoder refers to both packet and log Decoders. Perform the steps in the section in the sequence they are given.

Step 1: Verify System Configuration

The first step which needs to be completed when a new service is added to Security Analytics is the verification of system configuration.

Certain default values for the system configuration parameters are already in effect. These values can be edited and fine tuned for optimal performance.

Step 2: Configure Capture Settings

Next, you can configure the adapter for data capture, enable autostart of data capture, select the parsers that are applied to the captured data, and tune data capture by configuring capture settings.

Step 3: Enable or Disable Parsers

See which parsers have been downloaded and deployed from Live, and manage which ones are enabled or disabled.

Step 4: Configure Decoder Rules

Capture rules can add alerts or contextual information to sessions or logs. They can also define which data is filtered out by a Decoder or Log Decoder. Rules are created for specific metadata patterns, which result in predefined actions when matches are found. For example, to keep all traffic that fits certain criteria, but discard all other traffic, you can create a rule to perform the necessary actions. When applied, rules affect both packet capture file importing, as well as live network capture.

By default, no rules are defined when you first install Security Analytics. Until rules are specified, the packets are not filtered. You can deploy the latest rules from Live. You can define three types of rules: Network Layer Rules, Application Layer Rules, and Correlation Rules.

Step 5: Start and Stop Data Capture

When a Decoder starts up, it automatically begins aggregating data if Capture Autostart is enabled. When autostart is not enabled, you can start and stop data capture manually.

Topics

- Step 1. Verify System Configuration
- Step 2. Configure Capture Settings
- Step 3. Enable and Disable Log Parsers
- Step 4. Configure Decoder Rules
- Step 5. Start and Stop Data Capture

Step 1. Verify System Configuration

This topic provides a procedure for verifying the system configuration of a Decoder or a Log Decoder.

When a service is first added to Security Analytics, default values for the system configuration parameters are in effect. You can edit these values to tune performance.

System Configuration

Name	Config Value
Compression	0
Port	50004
SSL FIPS Mode	☐
SSL Port	56004
Stat Update Interval	1000
Threads	20

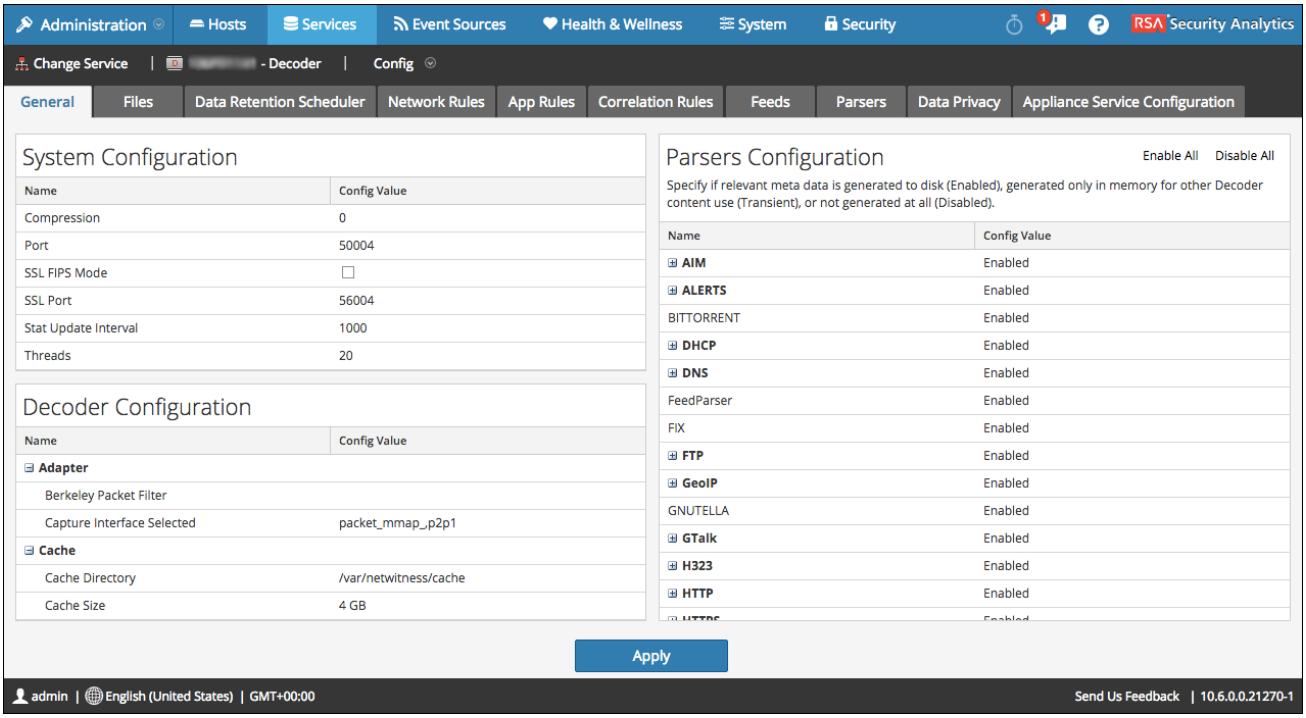
In most cases, the default values for compression, statistics update interval, and number of threads in the thread pool are set at a good point for optimal system performance. One parameter that you may want to change for your environment is the SSL setting, which by default is not enabled. When enabled, the security of data transmission is managed by encrypting information and providing authentication with SSL certificates.

Procedure

To edit system configuration parameters for a Decoder or Log Decoder:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Decoder or Log Decoder service, and select  **> View > Config**.

The Services Config view for the service is displayed with the General tab open.



3. Under **System Configuration**, click in a field that you want to edit, and type a new value.
4. When finished editing, click **Apply.**

Step 2. Configure Capture Settings

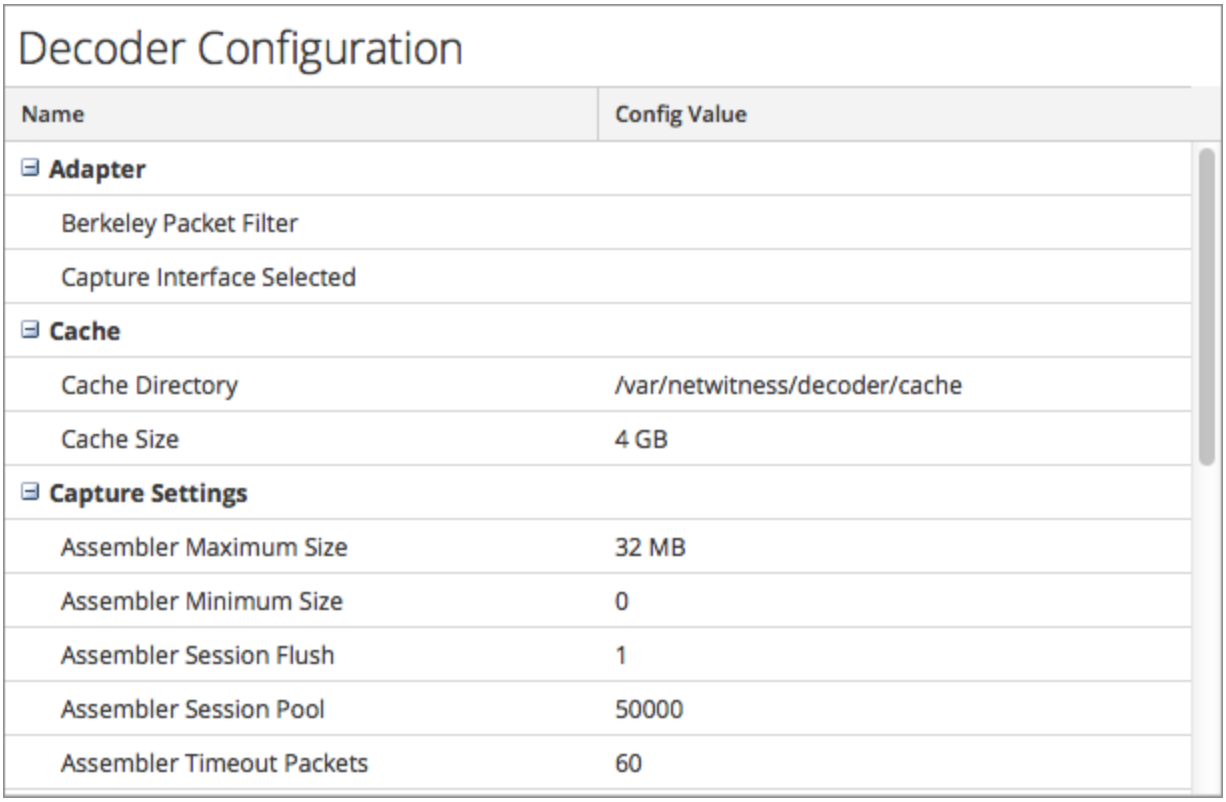
This topic provides a procedure for configuring data capture on Decoders and Log Decoders.

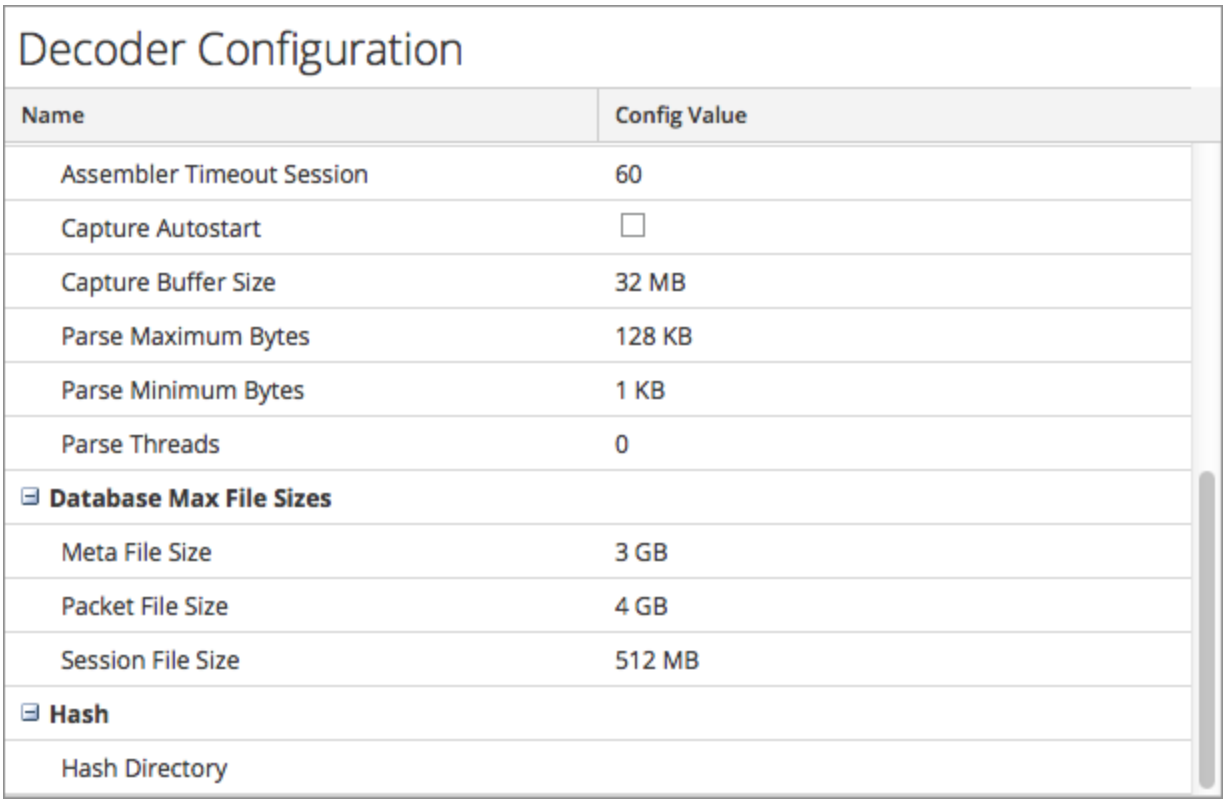
In RSA Security Analytics, you can configure the adapter for data capture, enable autostart of data capture, select the parsers that are applied to the captured data, and tune data capture.

Procedure

To set up a Decoder in preparation for capturing data:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Administration Services** view, select the Decoder service and  **> View > Config.**
   The Services Config view is displayed with the General tab open, and the most commonly used service settings for a Decoder or Log Decoder are available for editing under Decoder configuration.





3. In the **Adapter Settings** section, configure the network interface for capturing data.
4. In the **Cache** section, examine the settings for cache directory and size. If necessary, modify these.

   Note: If you are capturing data on hybrid systems (systems with more than one core service in use), set up separate cache directories for each core service.

5. In the **Capture Settings** sections, review the default values and modify if necessary.
6. If you want the Decoder to begin capturing data automatically when started, select the **Capture Autostart** checkbox.
7. In the **Database Max File Sizes** section, review the default values and modify if necessary.
8. In the **Hash** section, define a directory for hash files if you are using this feature.
9. Do one of the following:
   - In the **Parsers Configuration** panel, review the parsers selected to filter traffic and disable, enable, or mark as transient as necessary.
   - If configuring a Log Decoder, review the parsers selected to filter traffic in the **Service Parsers Configuration** section and disable, enable, or mark as transient as necessary.

10. To save the changes, click **Apply.**
11. If necessary to put the changes into effect, navigate to the **Services System** view and restart the service.
    At this point, you can start capture (also in the Services System view).

VLAN Fixup Configuration

When capturing traffic containing VLAN tags, you may need to configure the Packet MMAP capture interface to preserve the VLAN tags in the packets. By default, the network capture hardware removes the tags. Performing the VLAN fixup preserves the tags in the packets, and the tag values are parsed into VLAN meta data for further analysis.

There are two mechanisms for enabling the VLAN fixup.

- Option 1: Set `vlan-fix=true` within `capture.device.params`. This option performs the VLAN fixup on all traffic entering the Decoder. This option is appropriate in most cases, since it is assumed that all the traffic will be VLAN tagged. This mechanism works on either single-interface mode, or on all-interfaces mode.
- Option 2: Use the `interfaces` parameter within `capture.device.params` on a per-device basis. The `interfaces` parameter, as described above, accepts a comma-separated list of interface names on which to capture packets. By adding `:vlan` to an interface name, you can enable the VLAN fixup on individual interfaces. If the interface does not have the `:vlan` suffix added, it will not perform the VLAN fixup.

After editing this parameter, you must restart capture on the Decoder in order for changes to `capture.device.params` to take effect.

These are examples of both options.

Parameter	Value	Effect
capture.device.params	vlan-fix=true	VLAN fixup always performed on all interfaces
capture.device.params	interfaces=eth0:vlan,eth1	VLAN fixup performed on traffic capture on eth0 interface only
capture.device.params	interfaces=eth0:vlan,eth1 vlan-fix=true	VLAN fixup always performed: vlan-fix overrides interfaces setting

Configure System-Level (BPF) Packet Filtering

This topic describes how to use Berkeley Packet Filters to control which packets and logs are processed by a Decoder.

You can use Berkeley Packet Filters to control which packets and logs are processed by a Decoder. The Decoder supports system-level packet filtering defined using tcpdump/libpcap syntax. Specifying a libpcap filter can efficiently reduce packet volume based on Layer 2 - Layer 4 attributes. Berkeley Packet Filters (BPF) are applied to the packet stream before the packets are copied to the Decoder adapter for analysis. This allows unwanted traffic to be efficiently discarded. However, any packets discarded are not accounted for in any Decoder statistics (capture rate, packets dropped, and packets filtered and total packets).

A libpcap filter is appropriate for use when a Decoder is receiving a traffic volume that is placing a load against the physical resources of the platform. In this scenario, the Decoder may consistently drop packets and have a large number of capture pages available (/decoder/stats/capture.pagefree is high).

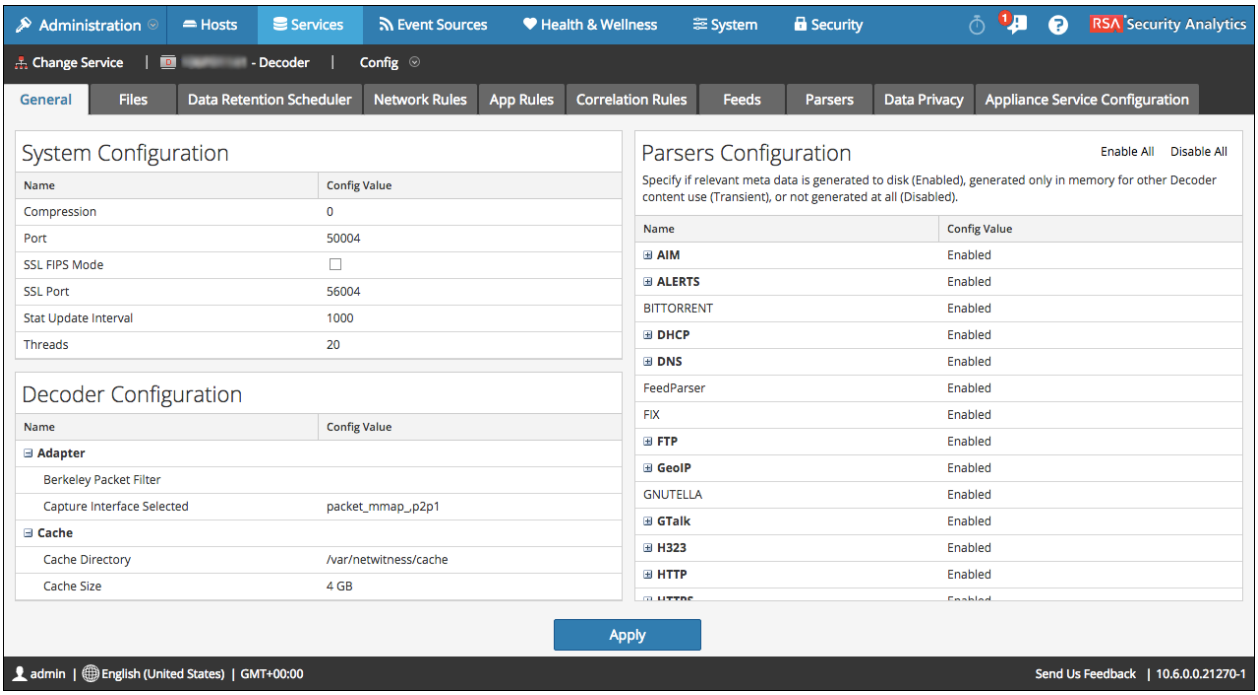
Add System-Level Packet Filters

To add a system-level Berkeley Packet Filter:

1. In the **Security Analytics** menu, select **Administration > Services.**

2. In the **Admin Services** view, select a Decoder service and > **View** > **Config.**

   The Services Config view is displayed with the General tab open.



3. In the **Decoder Configuration Section**, under **Adapter**, click in the field next to **Berkeley Packet Filter**.

4. Type only one filter in the field. If you want to filter multiple items, join multiple expressions using **and**.

   The SA user interface validates input at the time you enter your filter string.

5. To save the filter, click **Apply**.

   If the syntax is correct, a confirmation message is displayed.

   If the syntax is incorrect, a **Packet filter is not valid** message is displayed and a corresponding log message will follow in the log messages on the Decoder:

   ```
   164474800 2015-May-01 19:03:08 warning Decoder Failed to parse filter 'example_badrule': syntax error
   ```

6. To activate the filter, you must stop and start capture on the Decoder:

   a. Change the **Config** view to the **System** view.

   b. Click **Stop Capture**.

   c. Click **Start Capture**.

   The active filter will be displayed in the Decoder logs.

Examples

These are several filter examples:

- Drop packets to or from any address in the 10.21.0.0/16 subnet:
  not (net 10.21.0.0/16)
- Drop packets that have both source and destination addresses in the 10.21.0.0/16 subnet:
  not (src net 10.21.0.0/16 and dst net 10.21.0.0/16)
- Drop packets that are from 10.21.1.2 or are headed to 10.21.1.3.
  not (src host 10.21.1.2 or dst host 10.21.1.3)
- Combine both IP and HOST:
  not (host 192.168.1.10) and not (host api.wxbug.net)
- Drop all port 53 traffic, both TCP & UDP:
  not (port 53)
- Drop only UDP port 53 traffic:
  not (udp port 53)
- Drop all IP protocol 50 (IPSEC) traffic:
  not (ip proto 50)
- Drop all traffic on TCP ports 133 through 135.
  not (tcp portrange 133-135

The following filters combine some of the above to demonstrate how to put multiple directives into one filter:

- Drop any port 53(DNS) traffic sourced from 10.21.1.2 or destined to 10.21.1.3.
  not (port 53) and not (src host 10.21.1.2 or dst host 10.21.1.3)
- Drop any traffic using IP proto 50 or port 53 or any traffic from net 10.21.0.0/16 destined to net 10.21.0.0/16
  not (ip proto 50 or port 53) or not (src net 10.21.0.0/16 and dst net 10.21.0.0/16)

Caution: The use of parentheses can have a large and potentially disruptive effect on the use of Packet Filters. As a best practice, keep "not" operations outside of parentheses and always test your rules before deploying them. Failure to properly format your rules (despite input validation) can cause a packet filter to drop ALL traffic or behave in other unexpected ways. This is due to the way packet Libpcap filters work and is not the result of any logic within NetWitness software.

Testing

BPF filters can and should be tested using either tcpdump or windump to ensure that they will provide the expected behavior before implementing them. This example shows a test of a filter using windump:

windump -nni 2 not (port 53 or port 443) or not (ip proto 50)

Conversions

If for the sake of performance, you have decided that an existing Network Rule filter would be better running as a System-Level Packet Filter, you can convert it. There are a few things to remember when doing conversions.

- **&&** or **and**
- **ip.addr** becomes **host** if a single host or **net** if a network.
- **ip.src** becomes **src host** if a single host or **src net** if a network.
- **ip.dst** becomes **dst host** if a single host or **dst net** if a network.
- Use CIDR notation when listing a network (that is, 10.10.10.0/24).
- || becomes **or**
- **!** becomes **not**
- Multiple rules must be joined with **and**.

The manual for TCPDump also gives examples of filters and strings that can be used: http://www.tcpdump.org/tcpdump_man.html

Additionally, the following site provides an excellent reference for BPF-style packet filters: http://biot.com/capstats/bpf.html

Caution: If you are capturing vlan tagged packets, above standard bpf filter may not work. For example, if you use **not (udp port 123)**to filter vlan tagged NTP traffic on udp port 123, it will not work. This is because the bpf filter machinery is simple and does not account for protocols not referenced in the rule. So the OS executing the bpf filter will look for the udp port values at the byte offset they would occur in a standard Ethernet/udp packet; but the optional vlan tag fields in the Ethernet header pushes those values by 4 bytes, thus the bpf filter rule will fail. To fix it, you need to change the bpf filter to: **not (vlan and udp port 123)**.

Step 3. Enable and Disable Log Parsers

This topic tells administrators how to enable or disable log parsers on a Log Decoder.

This procedure is useful to see which log parsers have been downloaded and deployed from Live, and which of these are enabled.

You should only download and deploy the parsers you need for the following reasons:

- There is an impact on performance as you increase the number of deployed parsers.
- The more parsers you deploy, the more meta created, which impacts data retention
- Not having extra (unnecessary) log parsers deployed reduces the potential for misidentification of messages.

Prerequisites

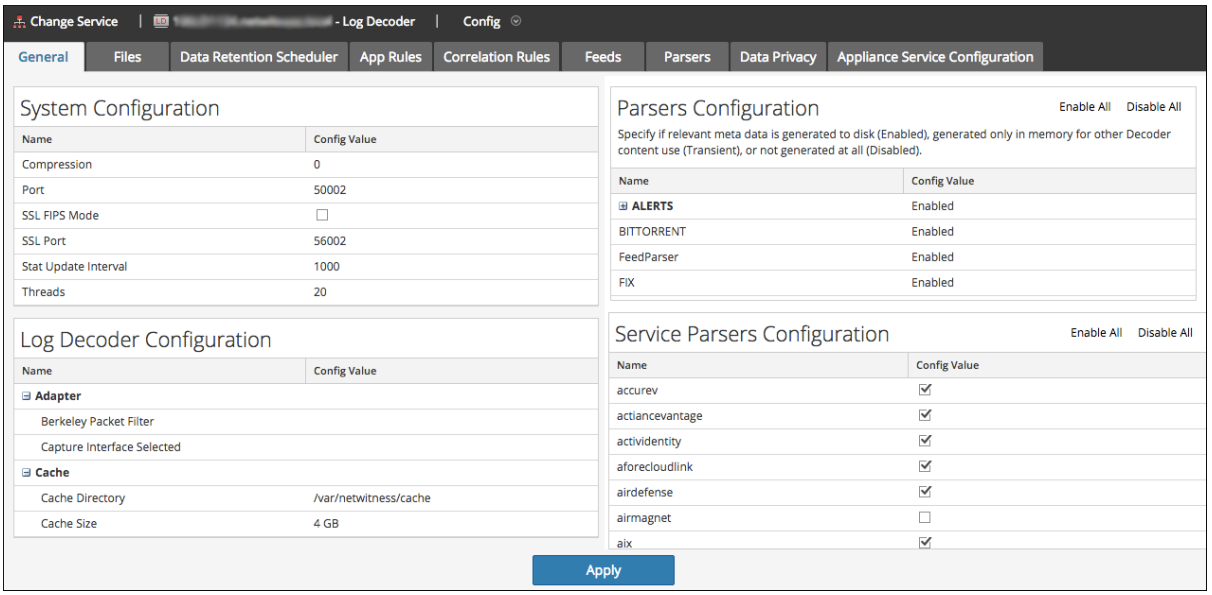
You must have previously deployed log parsers from Live. See the **Find and Deploy Live Resources** topic in *Live Services Management* for details.

Procedure

To enable or disable an event source parser, or to view the status for each parser:

1. In the **Security Analytics** menu, select **Administration** > **Services**.
2. In the **Services** grid, select a Log Decoder, and from the **Actions** menu (), choose **View** > **Config**.
3. In the **Service Parsers Configuration** panel, search for your event source.
4. In the **Config Value** column, note the current status for your parser.
   - If the parser is already selected, it is enabled.
   - If the parser is not selected, it is currently disabled.

You can toggle the value for any individual log parser. Alternatively, you can select **Enable All** or **Disable All** to update the status for all of your log parsers at once.



5. Click **Apply**.

When you click **Apply**, note that all parsers are reloaded into Security Analytics.

Result

The status for each log parser is updated, based on your selections.

Step 4. Configure Decoder Rules

This topic provides procedures for creating and managing rules for Decoder or Log Decoder traffic capture in the Services Config view > Rules tabs.

Capture rules can add alerts or contextual information to sessions or logs. They can also define which data is filtered out by a Decoder or Log Decoder. Rules are created for specific metadata patterns, which result in predefined actions when matches are found. For example, to keep all traffic that fits certain criteria, but discard all other traffic, you can create a rule to perform the necessary actions. When applied, rules affect both packet capture file importing, as well as live network capture.

Rule and Query Guidelines provides guidelines that all queries and rule conditions in Security Analytics Core Services must follow.

By default, no rules are defined when you first install Security Analytics. Until rules are specified, the packets are not filtered. You can deploy the latest rules from Live. You can define three types of rules: Network Layer Rules, Application Layer Rules, and Correlation Rules.

Network Layer Rules

Network layer rules are applied at the packet level and are made up of rule sets from Layer 2, Layer 3, and Layer 4. Multiple rules can be applied to the Decoder. Rules can be applied to multiple layers (for example, when a network rule filters out specific ports for a specific IP address). Network rules are only available on packet Decoders.

Application Layer Rules

Application layer rules are applied at the session level. If the first rule listed is not a match, the Decoder then attempts to match the next rule listed, until a match is found.

Correlation Rules

Correlation rules are applied over a configurable sliding time window. When a match is found, the service creates a new super session that identifies other sessions that match the rule, then creates a session list for analysis.

Common Uses

The two most common uses of rules are:

- To alert, and thereby create a custom alert meta value, when certain conditions are found
- To filter out certain types of traffic that do not add value to the analysis of the data

Rule Sets

Groups of capture rules form rule sets, which you can import and export. This feature enables use of multiple rule sets for various scenarios. You can import the exported rule set, in the form of an .nwr file, to other Security Analytics services, simplifying the deployment and configuration of multiple services.

Rule Processing

These are the principles governing capture rule processing:

- Multiple rules can be applied to the Decoder.
- Capture rules are executed one after the other, in sequence.
- Rule processing stops when all rules are processed or after a rule configured to stop rule processing is matched.
- A default rule can be used to either include or exclude all traffic not otherwise selected by a rule. A default rule, if used, must always be placed at the bottom of the rule list. Otherwise, rule processing stops as soon as the default rule is evaluated since, by definition, all traffic is selected by the default rule.
- When rule processing stops, the session is saved using the configured session options and debug options.

Rule Configuration

The Decoder and Log Decoder rules are editable in the Services Config view. While each type of rule (network, application, and correlation) has its own tab; the functions are similar for all types of rules. You can:

- Add, edit, and delete rules
- Enable and disable rules
- Change the execution sequence of rules
- Import rules from a file
- Export rules to a file
- Push rules to another service
- Revert or apply rule changes
- Restore one of the last ten rule configurations

Capture Rule Syntax

The syntax for writing capture rules consists of comparing a field to a value using a comparison operator. The supported comparison operators are equals (=) and not equals(!=).

Values can be expressed as discrete values, a range of values, an upper or lower bound, or a combination of these three. Greater than (>) and less than (<) comparisons are accomplished through the use of ranges. You can create a greater than or less than comparison, and test equality or inequality against a range of values or an upper/lower bound.

The following table summarizes the supported comparison operators and the syntax for expressing values.

Syntax	Description
*	Default rule. By using an asterisk (*) as the sole character in a rule, that rule will select all traffic.
=	Equality operator.
!=	Inequality operator.
&&	Logical AND operator.
||	Logical OR operator.
-u	Upper bound. For example, to select all TCP ports above 40000, the syntax would be: `tcp.port = 40000-u`
l -	Lower bound. For example, to select all TCP ports below 40000, the syntax would be: `tcp.port = l-40000`
- (dash)	Denotes a range. This is only applicable to numeric values. Separate the lower and upper bounds of the range with a dash (-) character. For example, to select TCP ports between 25 and 443, the syntax would be: `tcp.port = 25-443`
, (comma)	Denotes a list of values. Single values may be used as well as any combination of ranges and upper or lower bounds. For example, the following is valid syntax: `tcp.port = 1-10,25,110,143-225,40000-u`

Syntax	Description
( )	Grouping operator. An expression can be enclosed in parentheses to create a new logical expression. For example, the following would select traffic on port 80 to/-from 192.168.1.1 OR traffic on port 443 to/from 10.10.10.1: `(ip.addr=192.168.1.1 && tcp.port=80) || (ip.addr=10.10.10.1 && tcp.port=443)`

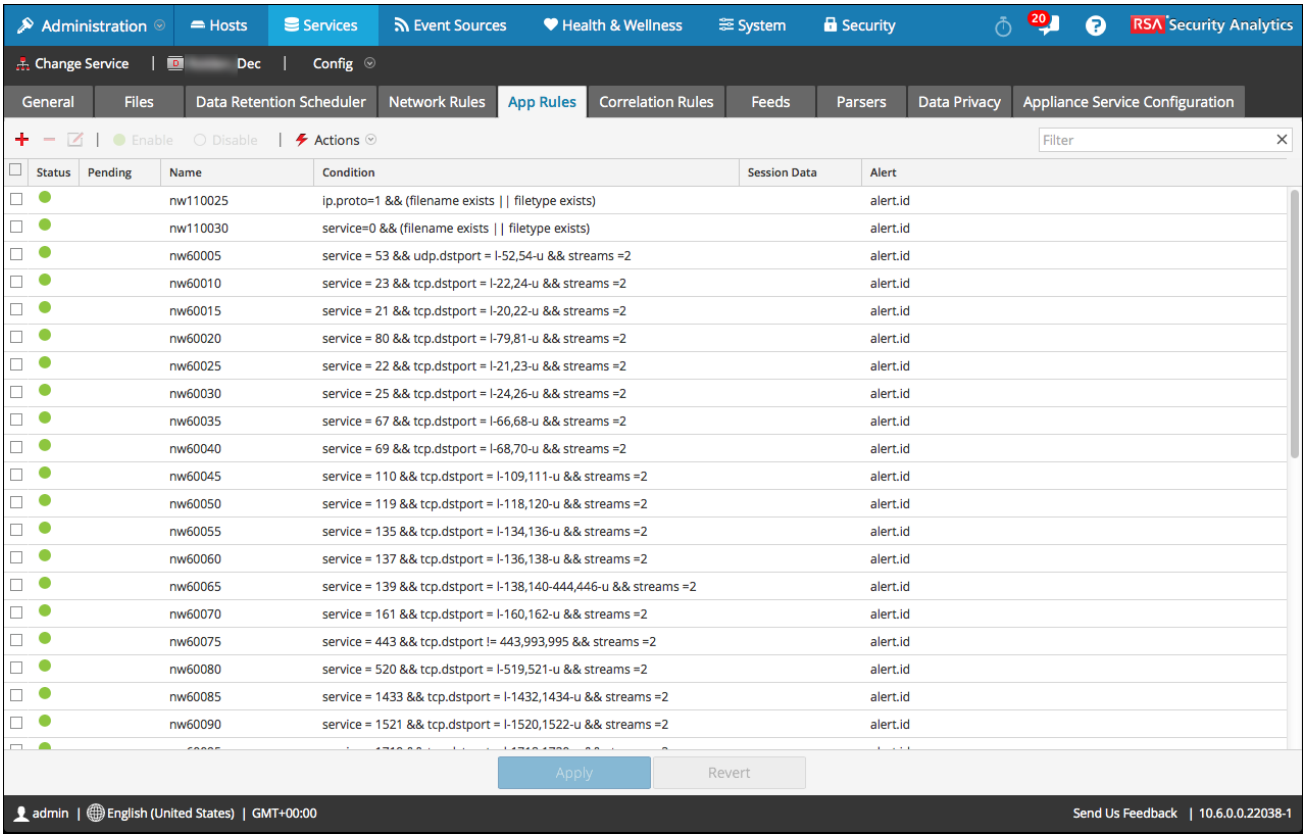
Rule and Query Guidelines provides guidelines that all queries and rule conditions in Security Analytics Core Services must follow.

Procedures

Configure Capture Rules

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Decoder service and  **> View > Config**.
3. In the **Services Config** view, select one of the Rules tabs: Network Rules, App Rules, or Correlation Rules.

The rules grid for the selected rule type is displayed.



Each type of rule has a grid with slightly different columns and different parameters. Several basic guidelines apply to all rule management activities:

- The rules are executed in the sequence they are displayed in the grid. To change the execution sequence of rules, drag and drop rules to the appropriate location in the grid or use the context menu options to arrange the rules in the grid.
- To select a single row, click the row.
- To select a group of adjacent rows, click the first, then shift-click the row at the end of the group.
- To select multiple non-adjacent rows, click the first, then control-click the others.
- When editing rules in the rules tab, you must apply the configuration changes in order to activate.
- Until changes are applied, you can discard edits to the grid and revert to the unedited rules.
- Once rules are applied, you can recover the last ten rules configurations using the **History** option in the **Actions** menu.

Add a Rule

To add a rule in any Rules tab, do one of the following:

- Click  .
- Right-click a rule, and select **Insert Above** or **Insert Below** from the context menu.
  The Rule Editor dialog for that type of rule is displayed.

For more details, see one of the following sections:

- Configure Application Rules
- Configure Network Rules
- Configure Correlation Rules

Remove a Rule

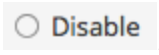
1. From any Rules tab, select the rules to remove from the rules grid.
2. Click .
   The selected rules are removed from the grid, but still exist on the service.

Edit a Rule

1. From any Rules tab, select the rule to edit.
2. Click  or double-click the rule row.
   The Rule Editor dialog for that type of rule is displayed. For more details, see one of the following sections:

- Configure Application Rules
- Configure Network Rules
- Configure Correlation Rules

Disable a Rule

1. From any Rules tab, select the rules to disable.
2. Click 

   .
   The status changes to disabled in the grid, but the rule is still enabled on the service.

Enable a Rule

1. From any Rules tab, select the rules to enable.
2. Click Enable.

   The status changes to enabled in the grid, but the rule is still disabled on the service.

Import Rules from a File

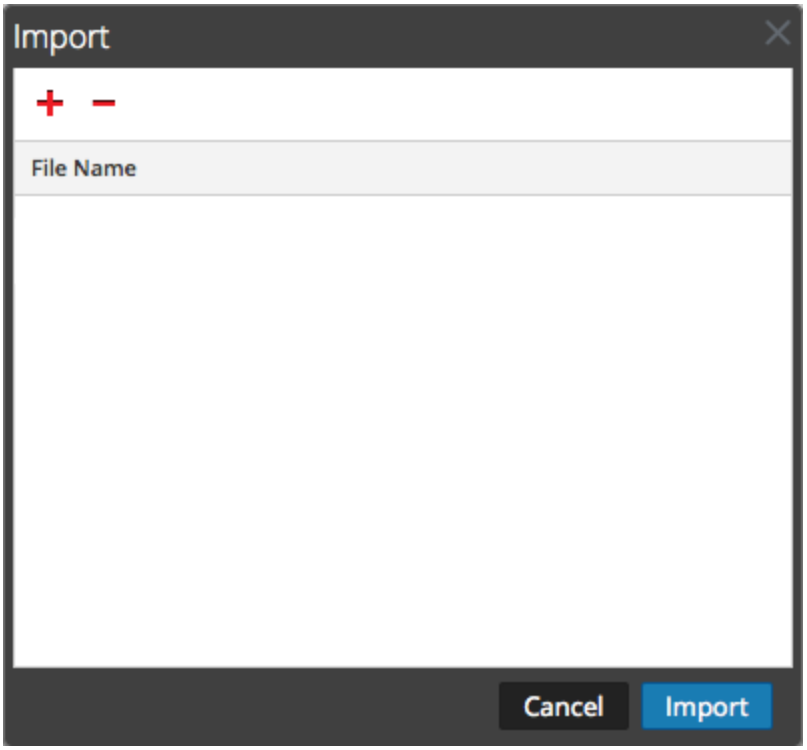
You can import network, application, and correlation rules to a Decoder from a file that contains rules of the same type. After the rules are imported, you can edit and manage them as you would any other rules.

When you attempt to import a group of rules, Security Analytics Administration checks the type of rules imported. If you are successful, a message displays the number of rules imported. If the rule type differs from the active tab type, the rules are not imported. You must re-import the rules under the correct tab or select another file to import.

To import rules to a service:

1. From any Rules tab, select Actions > **Import**.

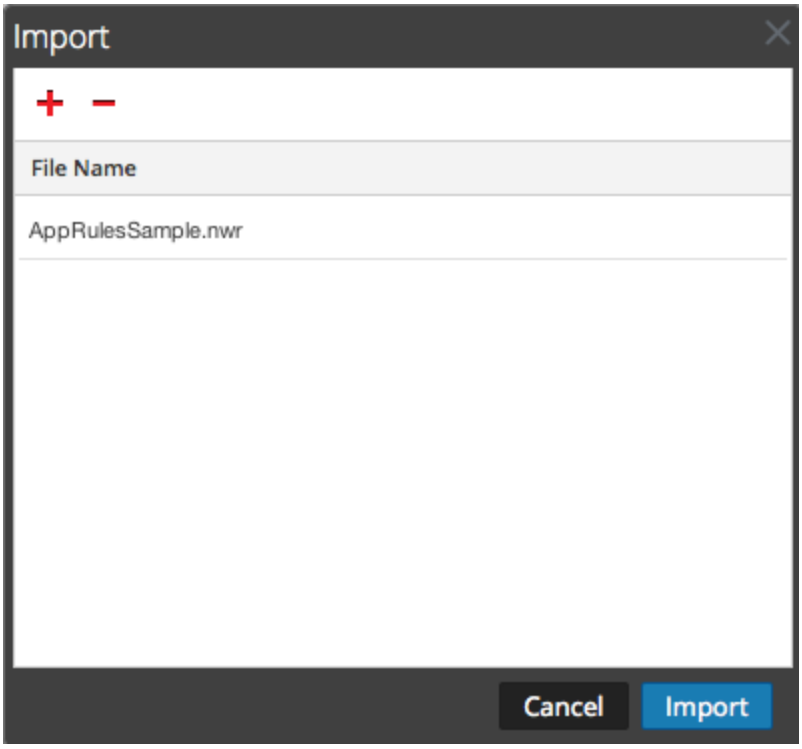
   The Import dialog is displayed.



2. Click +.

   A view of the directory structure is displayed.

3. Choose one or more NetWitness rules (.nwr) files to import, and click **Open**.
   The file is added to the list in the Import dialog.



4. Click **Import**.
   The rules are imported into the user interface. Imported rules have a red corner in each edited column.
5. Edit or reorder the rules if needed.
6. To save the rules to the service, click **Apply**.
   The rules for the service are updated with the changes.

Export a Rule to a File

1. To export a subset of the rules, select the rules to be exported.
2. Do one of the following:
   - In the toolbar, select Actions > **Export** > **Selection**. (**Export** > **All** exports all rules in the grid even if you have a subset selected for export.)
   - Right-click the selected rules and select **Export Selection**.

A prompt for the filename is displayed.

3. Enter the filename and click **Export**.
   The .**nwr** file is downloaded.

Push Rules to Other Services

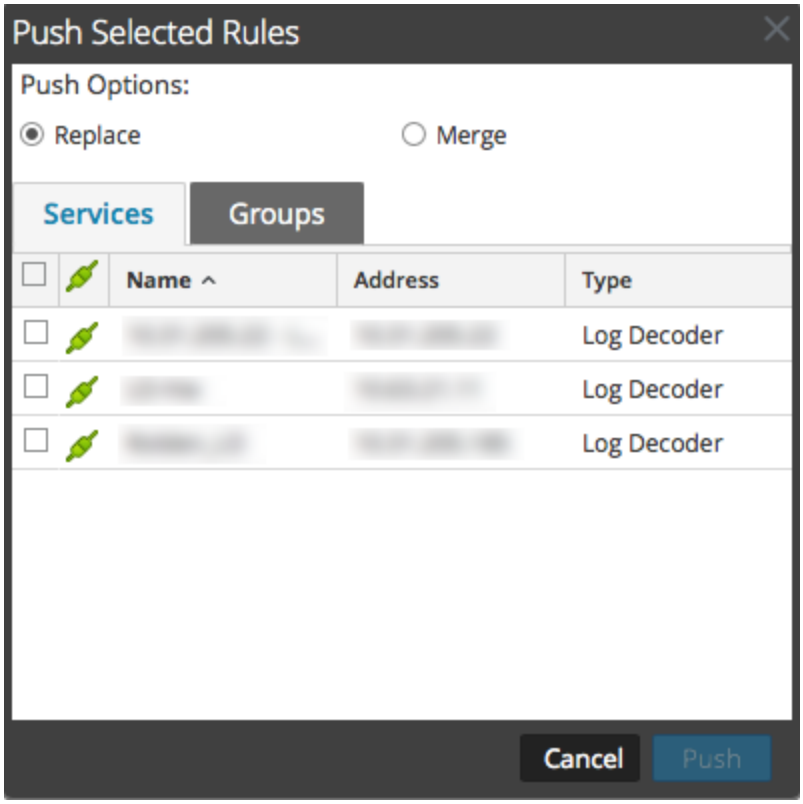
You can apply (push) rules or selected rules to other services (Decoders or Log Decoders) or service groups.

Push Selected Rules

To push selected rules from this Decoder to other Decoders:

1. From any Rules tab, select the rules that you want to push to another Decoder.
2. Do one of the following:
   - Select Actions > **Push** > **Selection**.

- Right-click the selected rules and select **Push Selected Rules**.
  The Push Selected Rules dialog is displayed.



3. Select a Push Option:
   - Select **Replace** to delete all rules on the target services and replace them with the selected rules. This is the default selection.
   - Select **Merge** to merge the selected rules with the existing rules on the target services.
4. On the **Services** tab, select the target services to receive the pushed rules, or select the groups of services from the **Groups** tab.
5. Click **Push**.
   The rules are pushed to the selected services and become effective immediately.

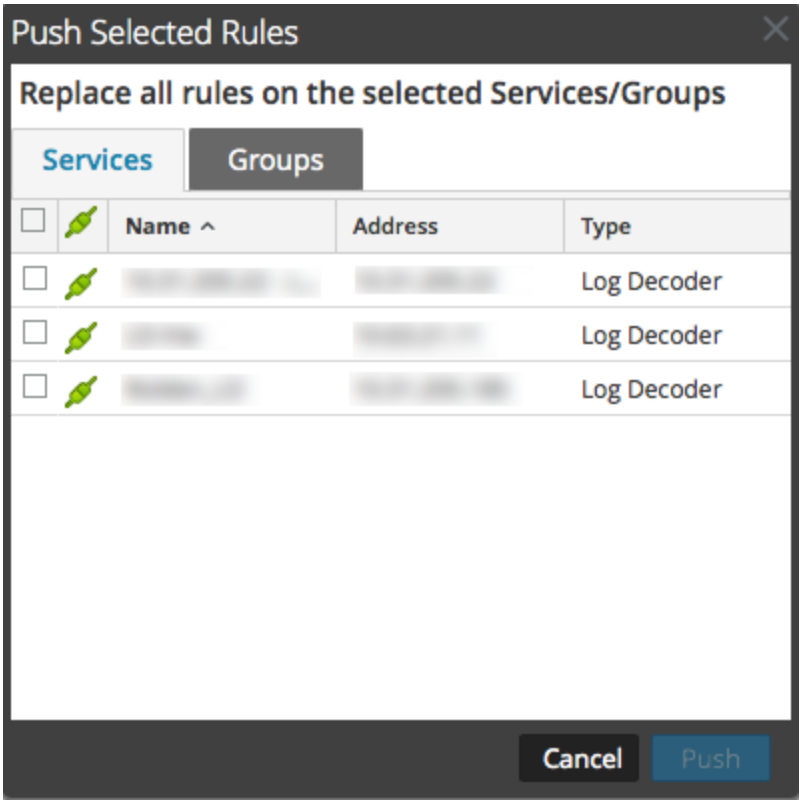
Push All Rules

When you push all rules to other services, all rules on the target services are removed and replaced with all of the rules on the source service.

To push all rules from this Decoder to other Decoders:

1. From any Rules tab, select Actions > **Push** > **All**.
   (**Push** > **All** pushes all rules in the grid even if you have a subset selected to push.) The Push

Selected Rules dialog is displayed.



2. On the **Services** tab, select the target services to receive the pushed rules, or select the groups of services from the **Groups** tab.
3. Click **Push**.
   All rules from the target services are deleted and replaced with all of the rules from source service. The rules become effective immediately.

Change Execution Order of Rules

Capture rules are applied in the order they are displayed in the grid. To reorder rules, use either of these methods:

- Drag and drop the rules in the appropriate location in the grid.
- Right-click a rule to display the context menu, and use the **Cut** and **Paste** options.

Restore a Rule Snapshot from History

Security Analytics keeps the last ten snapshots of rules applied to a service. To restore a rules snapshot from history:

1. Select Actions > **History**.
   A submenu of snapshots is displayed.

2. Select the snapshot time from the submenu.
   The rules from the snapshot are loaded into the grid, replacing the current set. But the current set is still in use on the service.
3. To apply the rules to the service, click **Apply**.
   The rules are applied to the service.

Configure Application Rules

This topic introduces application rules and provides instructions for creating application rules.

Application layer rules are applied at the session level.

Sample Application Rules

To truncate packets carried via Server Message Block protocol (SMB), create a rule as follows:

- Rule Name: Truncate SMB
- Condition: service=139
- Rule Action: Truncate

To retain email to and from a specific e -mail address, create a rule as follows:

- Rule Name: Email Filter Tom Jones
- Condition: email='Tom.Jones@TheShop.com'
- Rule Action: Filter

Procedures

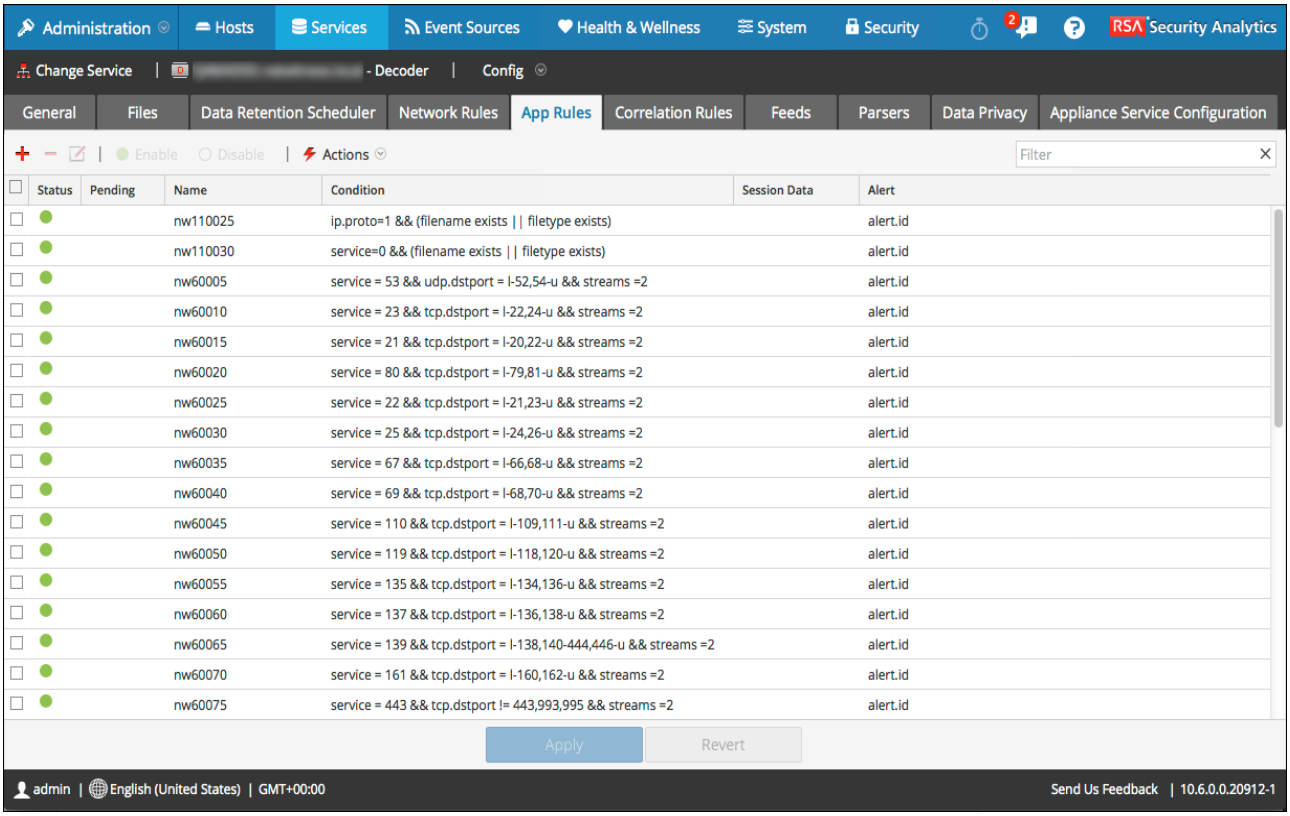
Navigate to the App Rules Tab

Navigating to the App Rules tab is always the first step in defining application rules. To access the App Rules tab:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a Decoder or Log Decoder service and  **> View > Config**.

   The Systems Config view for the selected service is displayed.

3. Select the **App Rules** tab.

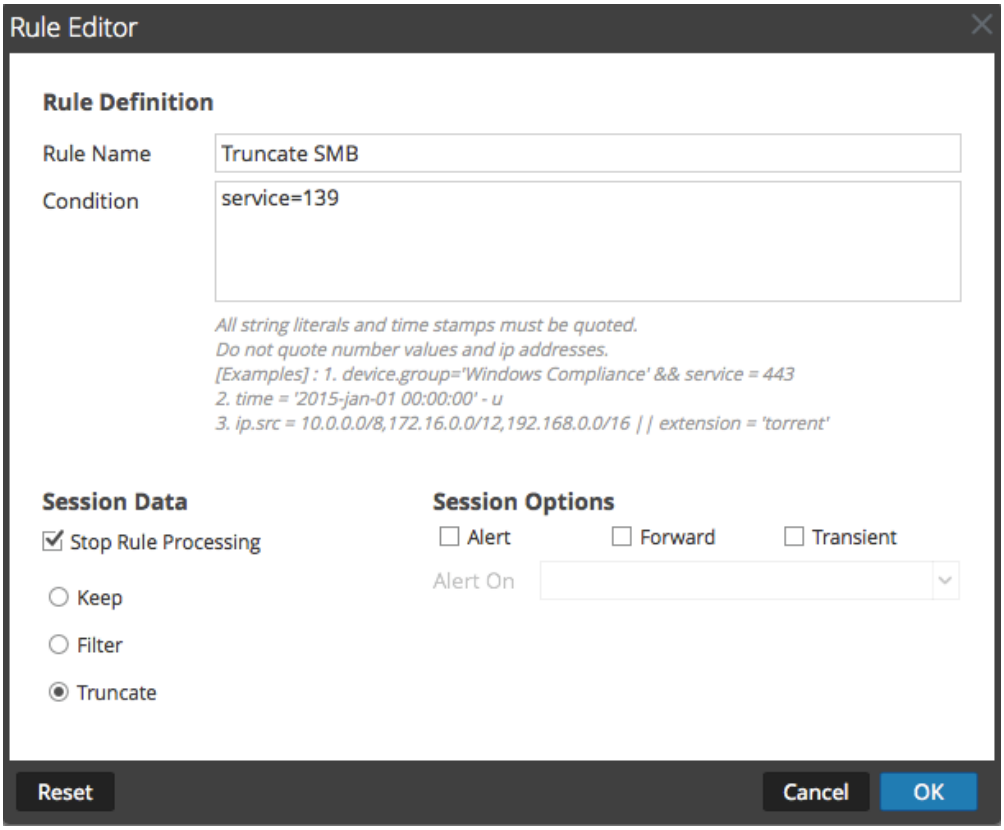


Add or Edit an Application Rule

In the App Rules tab:

1. Do one of the following:
   - If adding a new rule, click  .
   - If editing a rule, select the rule from the rules grid and click  .

2. The Rule Editor Dialog is displayed with application rule parameters.



3. In the **Rule Name** field, type a name for the rule. For example, for a rule that truncates all SMB, type **Truncate SMB**.
4. In the **Condition** field, build the rule condition that triggers an action when matched. You can type directly in the field or build the condition in this field using meta from the window actions. As you build the rule definition, Security Analytics displays syntax errors and warnings. For example, to truncate all SMB, type **service=139**.
   All string literals and time stamps must be quoted. Do not quote number values and IP addresses. The Rule and Query Guidelines topic provides additional details.
5. If you want rule evaluation to end with this rule, check the **Stop Rule Processing** checkbox.
6. In the **Session Data** section, choose one of the following actions to apply when a matching packet is found:
   - **Keep**: The packet payload and associated meta are saved when they match the rule.
   - **Filter**: The packet is not saved when it matches the rule.
   - **Truncate**: The packet payload is not saved when it matches the rule, but packet headers and associated meta are retained.

7. In the **Session Options** section, do any of the following:

- To generate a custom alert when a session metadata matches the rule, enable the Alert flag and select the name of the alert meta from the **Alert On** drop-down list.
- To perform syslog forwarding when the log matches the rule, enable the **Forward** flag.

  Note: Make sure that:
  - You have enabled both the Alert and Forward flags to carry out syslog forwarding.
  - The name of the rule mentioned in the Rule Editor dialog matches the syslog forwarding destination name specified in the Log Decoder > View > Explore > /decoder/config/logs.forwarding.destination parameter.

- To prevent the alert metadata that is created from being written to the disk, enable the **Transient** flag.

8. To save the rule and add it to the grid, click **OK**.
   The rule is added at the end of the grid or inserted where you specified in the context menu. The plus sign is displayed in the **Pending** column.
9. Check that the rule is in the correct execution sequence with other rules in the grid. If necessary, move the rule.
10. To apply the updated rule set to the Decoder or Log Decoder, click **Apply**.
    Security Analytics saves a snapshot of the currently applied rules, then applies the updated set to the Decoder and removes the pending indicator from the rules that were pending.

Configure Correlation Rules

This topic introduces correlation rules and provides procedures for creating correlation rules.

Basic Correlation Rules are applied at the session level and alert the user to specific activities that may be occurring in their environment. Security Analytics applies correlation rules over a configurable sliding time window. When the conditions are met, alert metadata is created for this activity and there is a visible indicator of the suspicious activity.

Sample Correlation Rules

Objective: In sessions where tcp.dstport exists, if there is any combination of ip.src and ip.dst where the count of unique instances of tcp.dstport > 5 within 1 minute, then alert. To achieve this objective, create a rule as follows:

- Rule Name: IPv6 Vertical TCP Port Scan 5
- Rule: tcp.dstport exists
- Instance Key: ip.src,ip.dst
- Threshold: u_count(tcp.dstport)>5
- Time Window: 1 min

Objective: In sessions where action==login and error==fail, if there is any combination of ip.src and ip.dst that appears in more than 10 sessions within 5 minutes, then alert. To achieve this objective, create a rule as follows:

- Rule Name: IPv4 Potential Brute Force 10
- Rule: action='login' && error='fail'
- Instance Key: ip.src,ip.dst
- Threshold: count()>10
- Time window: 5 mins

Explanation

Both sample rules have the same instance key: ip.src and ip.dst. Because we are looking for unique combinations of ip.src and ip.dst that match the correlation condition, **ip.src** and **ip.dst** are **primary keys**.

Threshold can include an **associated key** that identifies the meta type that we are counting to determine if the condition is satisfied. In the first example, the associated key specified in Threshold is **tcp.dstport**. We are counting unique instances of tcp.dstport for every ip.src/ip.dst pair. In the second example, the associated key is not specified in the Threshold because it is merely a count of sessions. It is helpful to think of this scenario as counting unique session IDs and the associated meta is implicitly session.id. We are counting unique session.id for every ip.src/ip.dst pair.

Invalid use case: In sessions where (rule), if there is any combination of ip.src and ip.dst that have a unique count of ipv6.dst > 5 within (time window), then alert. This case does not work because the associated key ipv6.dst is an IPv6 meta type. IPv4 and IPv6 meta types are not permitted to be used as associated keys.

Procedures

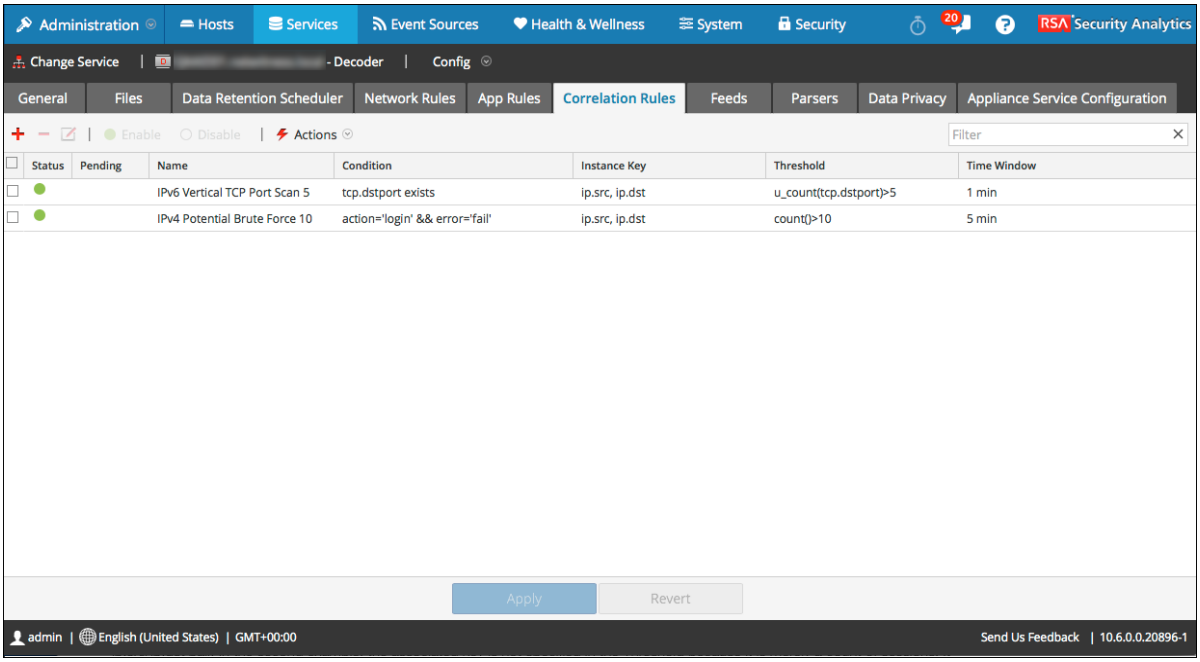
Navigate to the Correlation Rules Tab

The first step in working with correlation rules is to access the Correlation Rules tab:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **> View > Config**.

   The Service Config view for the selected service is displayed.

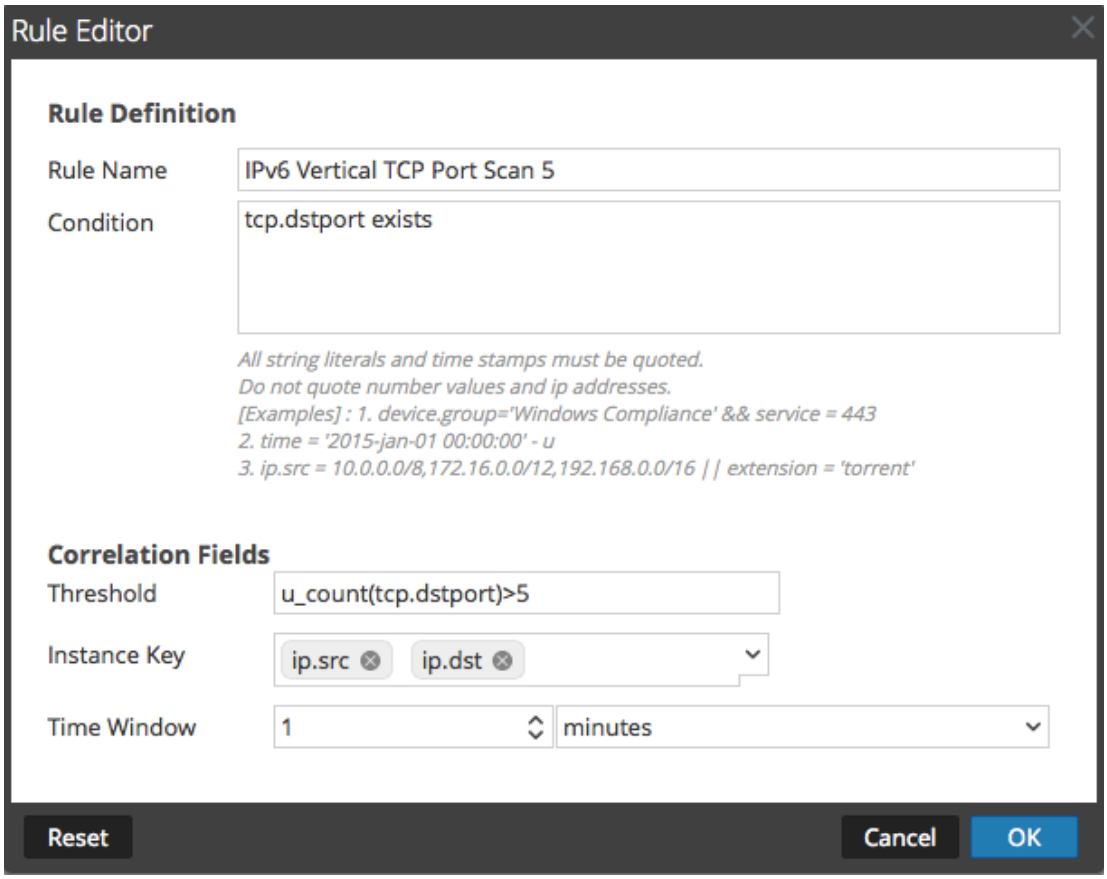
3. Select the **Correlation Rules** tab.



Add or Edit a Correlation Rule

1. In the **Correlation Rules** tab, do one of the following:
   - If adding a new rule, click  .

- If editing a rule, select the rule from the rules grid and click  .
  The Rule Editor dialog is displayed with correlation rule parameters.



2. In the **Rule Name** field, type a name for the rule. For example, to create the sample rule, **IPv6 Vertical TCP Port Scan 5**.
3. In the **Condition** field, build the rule condition that triggers an action when matched. You can type directly in the field or build the condition in this field using meta from the window actions. As you build the rule definition, syntax errors and warnings are displayed by Security Analytics. For example, to create the sample rule, type **tcp.dstport exists**. When this condition is matched, the session data action is performed.
   All string literals and time stamps must be quoted. Do not quote number values and IP addresses. The Rule and Query Guidelines topic provides additional details.
4. In the **Threshold** field, use one of the threshold parameters to specify the minimum number of occurrences required to create a correlation session and an associated key if required. The associated key cannot be an IPv4 or IPv6 meta type.
   - u_count(associated_key) = the count of unique values of the specified key
   - sum(associated_key) = the values of the specified key

- count = number of sessions (no associated key is specified)

5. In the **Instance Key** field, select the target indicator to base the event upon. This can be a single key or a compound key (two primary keys, separated by a comma).
6. In the **Time Window**, set the duration during which the threshold must be reached to create a correlation session.
7. To save the rule and add it to the grid, click **OK**.
   The rule is added at the end of the grid or inserted where you specified in the context menu. The plus sign is displayed in the **Pending** column.
8. Check that the rule is in the correct execution sequence with other rules in the grid. If necessary, move the rule.
9. To apply the updated rule set to the service, click **Apply**.

   Security Analytics saves a snapshot of the currently applied rules, then applies the updated set to the Decoder or Log Decoder.

Configure Network Rules

This topic introduces network rules and procedures for configuring network rules.

Network layer rules are applied at the packet level on a Decoder and are made up of rule sets from Layer 2 - Layer 4. Network rules can apply to multiple network layers (for example, when a network rule filters out specific ports for a specific IP address). Network rules do not apply to Log Decoders, they apply only to packet Decoders.

Sample Network Rules

To truncate all SSL from the source port, create a rule as follows:

- Rule Name: Truncate SSL
- Condition: tcp.srcport=443
- Rule Action: Truncate

To filter subnet traffic, create a rule as follows:

- Rule Name: Subnet Filter
- Condition: ip.addr=192.168.2.0/24
- Rule Action: Filter

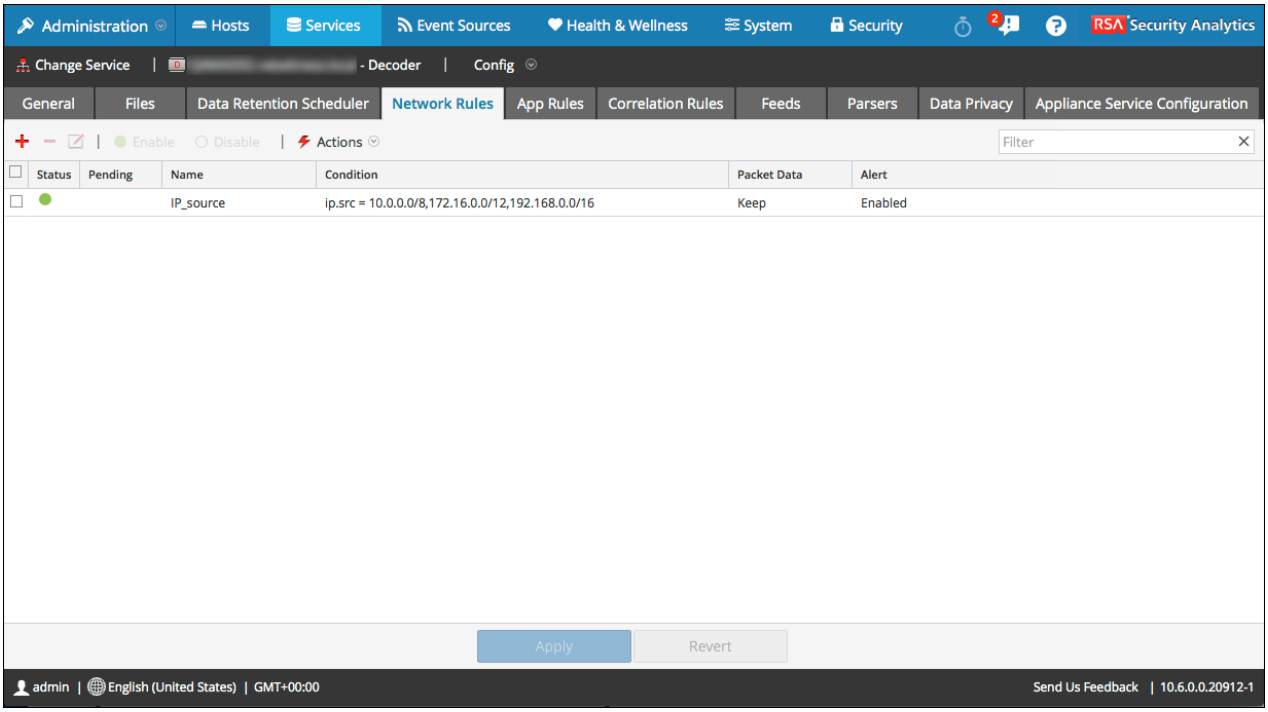
Procedures

Navigate to the Network Rules Tab

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a Decoder service and **> View > Config**.

   The Services Config view for the selected service is displayed.

3. Select the **Network Rules tab**.
   The Network Rules tab is displayed.

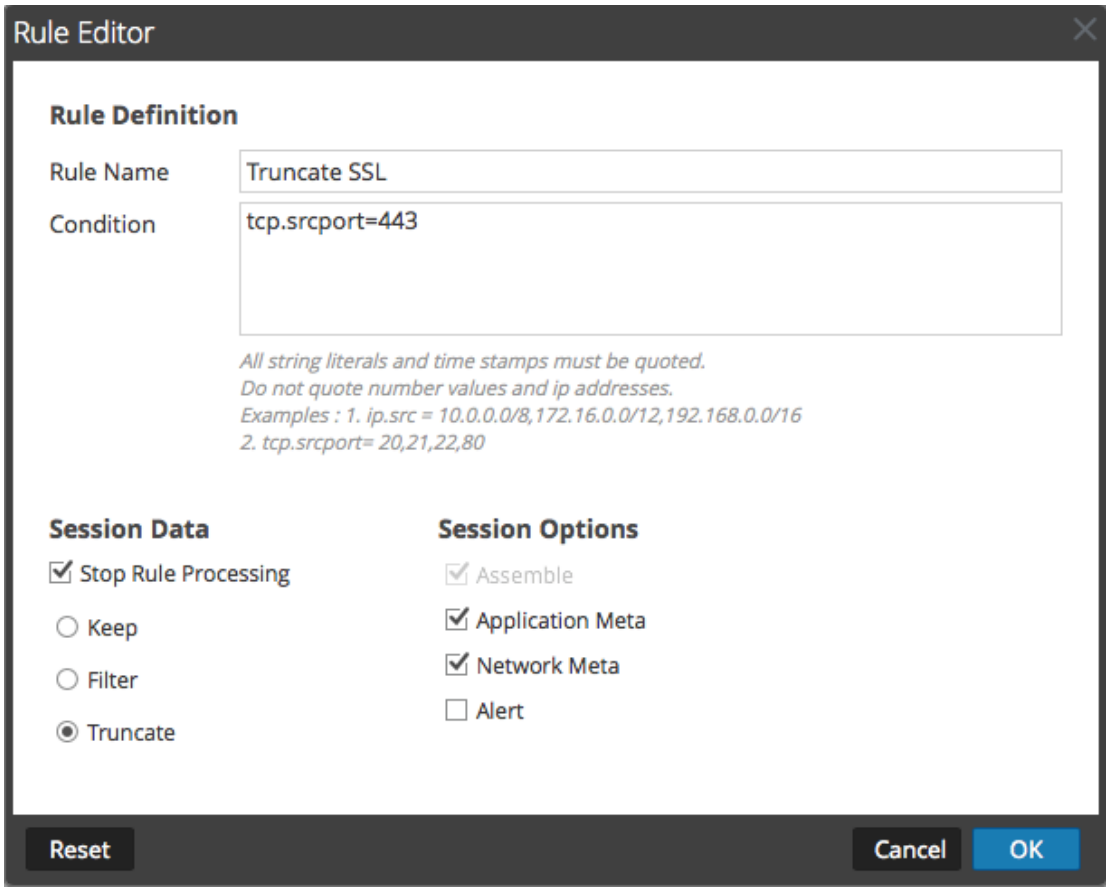


Add or Edit a Network Rule

1. In the **Network Rules** tab, do one of the following:
   - If adding a new rule, click  .

- If editing a rule, select the rule from the rules grid and click .

  The Rule Editor dialog is displayed.



2. In the **Rule Name** field, provide a name for the rule. For example, for a rule that truncates all SSL from the source port, type **SSL Truncate**.
3. In the **Condition** field, build the rule condition that triggers an action when matched. You can type directly in the field or build the condition in this field using meta from the window actions. As you build the rule definition, Security Analytics displays syntax errors and warnings. For example, to truncate all SSL from the source port, **tcp.srcport=443**.

   All string literals and time stamps must be quoted. Do not quote number values and IP addresses. The Rule and Query Guidelines topic provides additional details. Supported Meta Keys in Network Rules describes the meta keys that Security Analytics supports for use in network rule conditions.
4. If you want rule evaluation to end with this rule, select the **Stop Rule Processing** checkbox.
5. In the **Session Data** section, choose one of the following actions to apply when a matching packet is found:

- **Keep**: The packet payload and associated meta are saved when they match the rule.
- **Filter**: The packet is not saved when it matches the rule.
- **Truncate**: The packet payload is not saved when it matches the rule, but packet headers and associated meta are retained.

6. In the **Session Options** section, select all options that apply of these four.
   - **Assemble**: The assembler assembles the packet chain when it matches the rule.
   - **Network Meta:** The packet generates network metadata when it matches the rule.
   - **Application Meta**: The packet generates application metadata when it matches the rule.
   - **Alert**: The packet generates a custom alert when metadata matches the rule.
7. To save the rule and add it to the grid, click **OK**.
   The rule is added at the end of the grid or inserted where you specified in the context menu.
8. Check that the rule is in the correct execution sequence with other rules in the grid. If necessary, move the rule.
9. To apply the updated rule set to the Decoder, click **Apply**.

   Security Analytics saves a snapshot of the currently applied rules, then applies the updated set to the Decoder and removes the pending indicator from the rules that were pending.

Step 5. Start and Stop Data Capture

This topic provides a procedure for starting and stopping data capture on Decoders.

When a Decoder starts up, it automatically begins aggregating data if **Capture Autostart** is enabled. When autostart is not enabled, you can start and stop data capture manually.

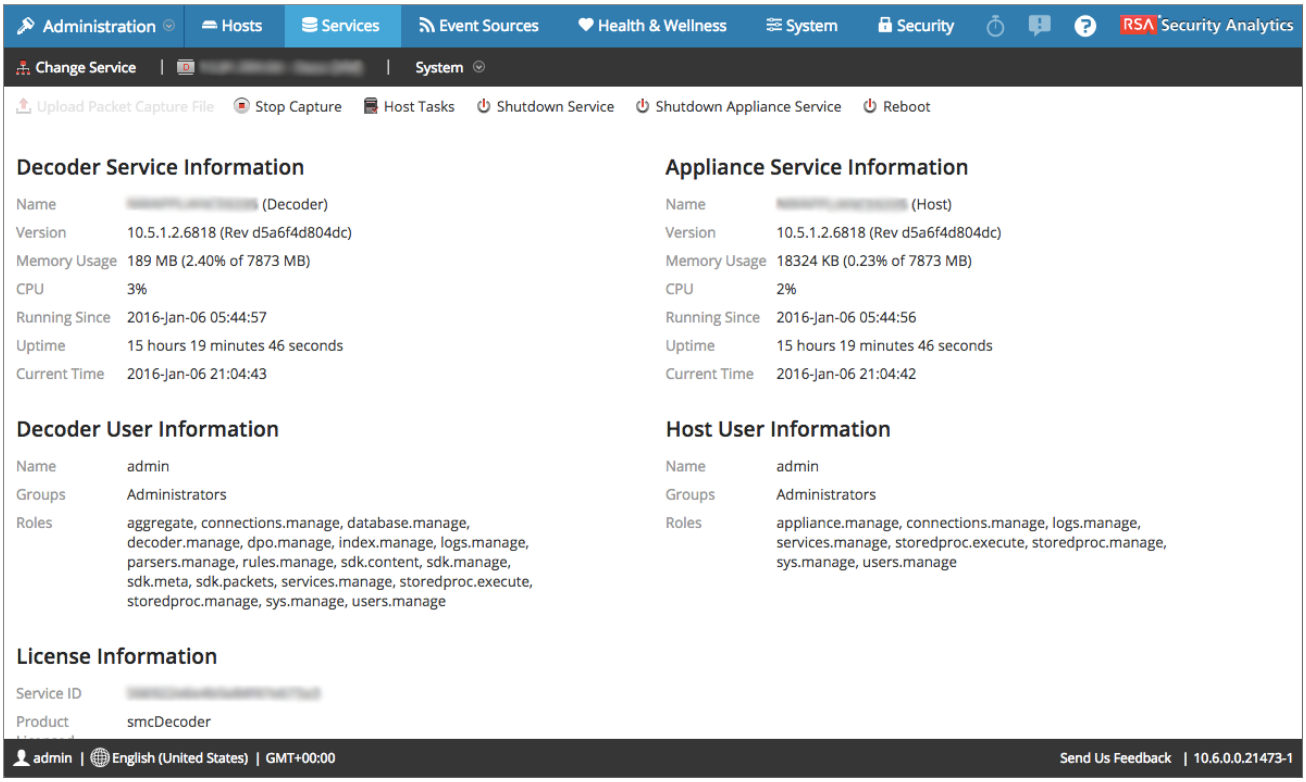
Note: The Capture Configuration Settings in the Service Config view for a Decoder determine whether Capture Autostart is enabled, as well as adapter, cache, data base, and hash settings.

Procedure

To start and stop capture:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. In the **Admin Services** view, select a Decoder or Log Decoder service, and select  **> View > System**.
3. In the toolbar, click **Start Capture**.
   If the service is a Decoder, it begins capturing packets. If the service is a Log Decoder, it begins capturing logs.
   When packet or log capture is in progress, the option in the toolbar changes to **Stop Capture**, and the option to upload a file is unavailable.

4. Whenever you want to discontinue traffic capture on a Decoder, click **Stop Capture**.
   Packet or log capture ceases, and the option to upload a file to the service is again available.



Decoder and Log Decoder Additional Procedures

This topic explains the additional procedures an administrator could choose to follow which are not essential for the configuration of the Decoder or Log Decoder.

This section can be used to find additional information about Decoders and Log Decoders in Security Analytics.

Topics

- Configure Feeds and Parsers
- Configure 10G Capability
- Configure Syslog Forwarding to Destination
- Create Custom Meta Keys Using Custom Feed
- Configure Parser Mappings
- Fix Rules with Deprecated Syntax
- Enable or Disable Lua and Flex Parsing Systems
- Map IP Address to Service Type
- Upload Log File to a Log Decoder
- Upload Packet Capture File
- Verify Decoder System Information
- Configure a Log Decoder to Accept Protobuf

Configure Feeds and Parsers

This topic introduces feeds and parsers and provides procedures for working Decoder and Log Decoder feeds and parsers.

Feeds and parsers are responsible for analyzing the packets and logs when captured or imported in a Decoder or Log Decoder. Most commonly, they are used for static meta extraction and service identification. The flexible definition allows custom extension of the core defined services to provide extra service type identification and metadata extraction. This is important due to the volume of custom applications that are used on networks.

Note: Unless otherwise stated, any reference to Decoders applies to Log Decoders as well.

Procedures

Configure Parsers

Security Analytics has a set of core parsers that are defined by the system as well as the ability to add additional parsers. Each parser is configurable in the Services Config View - General Tab. The Parser Configuration panel provides a way to enable or disable parsers to use on the Decoder in addition to limiting the metadata that the parser creates.

There are several types of custom configurable parsers:

- GeoIP–This parser associates the IP addresses with actual geographical locations.
- Search–This parser is user -configured to generate metadata by scanning for pre -defined keywords and regular expressions.
- FLEXPARSE–This is a generic parser definition language for extending the existing application protocol support of the Decoder.
- Lua–This parser is defined using the Lua scripting language for extending the existing application protocol support of the Decoder.
- enVision–This application parser supports the Log Decoder and is configured to generate metadata by scanning log files.
- SNORT®–This parser supports the payload detection capabilities of SNORT® IDS rules.

In the Services Config view > Parsers tab, you can view deployed parsers on a Decoder, upload parsers, and delete deployed parsers. The user interface includes an Indicator if the parser originated from Live, installed through Security Analytics, or uploaded manually. Parsers can be added and removed while a Decoder is running without affecting capture.

In addition, you can download parsers using Security Analytics Live.

Configure Feeds

Security Analytics uses feeds to create metadata based on externally defined metadata values. A feed is a list of data that is compared to sessions as they are captured or processed. For each match, additional metadata is created. This data could identify and classify malicious IPs or incorporate additional information such as department and location based on internal network assignments. Some examples of feeds include threat feeds to identify BOTNets, DHCP mappings, or even active directory information such as physical location or logical department.

You can use the Live module in Security Analytics to obtain feeds from outside sources. The **Live Content in Security Analytics**topic in *Live Services Management* provides an overview of the Live content management tool.

Within the Security Analytics user interface, you can view the list of currently deployed feeds, along with an indicator if the feed originated from Security Analytics Live was installed through Security Analytics, or manually. Feeds can be added, removed, and updated while a Decoder is running without affecting capture.

Security Analytics has a Custom Feed wizard, which streamlines the task of creating and managing custom feeds, as well as populating the feeds to selected Decoders and Log Decoders. In addition, you can download existing feed files and edit the files, then edit the feed or create a new feed using the edited file.

Topics

- Create and Deploy Custom Feeds Using a Wizard
- Use Custom Parsers

Create and Deploy Custom Feeds Using a Wizard

This topic provides instructions for using the Custom Feed Wizard in RSA Security Analytics, to quickly populate Decoders with custom feeds.

Security Analytics has a Custom Feed wizard to allow quick creation and deployment of custom Decoder feeds based on deterministic logic that offers the meta keys specific to the selected Decoders and Log Decoders. Although the wizard guides users through the process to create both on-demand and recurring feeds, it is helpful to understand the form and content of a feed file when you create a feed.

Feed filenames in Security Analytics are in the form `<filename>.feed`. To create a feed, Security Analytics requires a feed data file in `.csv` or `.xml` format and a feed definition file in `.xml` format, which describes the structure of a feed data file.

The Custom Feed wizard can create the feed definition file based on a feed data file, or based on a feed data file and the corresponding feed definition file. Security Analytics supports CsvFileFeed and FlatFileFeed types of feed deployment files. The CsvFileFeed file type provides additional parser grammar than the FlatFileFeed file type, which means that more checks will be done on files if you use the CsvFileFeed file type.

The CsvFileFeed files support CSV grammar and escape definitions. The designated delimiter character is , (comma), and the designated escape character is " (double quote). Data values that contain a comma character must be enclosed in double quotes. Data values that contain double quotes are escaped with double quotes, and must be enclosed by double quotes to preserve the data value. Embedding the entire field inside a set of double quotes preserves leading and trailing white space characters.

The files that you use to create an on-demand feed must be stored on your local file system. The files used to create a recurring feed must be stored at an accessible URL, from which Security Analytics can fetch the most current version of the file for each recurrence. After a Security Analytics feed is created, you can download the feed to your local file system, edit the feed files, and then edit the Security Analytics feed to use the updated feed files.

Sample Feed Definition File

This is an example of a FlatFileFeed definition file named **dynamic_dns.xml**, which Security Analytics creates based on your entries in the Custom Feed wizard. It defines the structure of the feed data file named **dynamic_dns.csv**.

Note: The feed file path should be .csv regardless of the Feed Type (Default or STIX).

```
<?xml version="1.0" encoding="utf-8"?>
 <FDF xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="feed-definitions.xsd">

  <FlatFileFeed name="Dynamic DNS Domain Feed"
  path="dynamic_dns.csv"
  separator=","
  comment="#"
  version="1">

  <MetaCallback
  name="alias.host"
  valuetype="Text"
  apptype="0"
  truncdomain="true"/>

  <LanguageKeys>
        <LanguageKey name="threat.source" valuetype="Text" />
        <LanguageKey name="threat.category" valuetype="Text" />
        <LanguageKey name="threat.desc" valuetype="Text" />
  </LanguageKeys>

  <Fields>
  <Field index="1" type="index" key="alias.host" />
  <Field index="4" type="value" key="threat.desc" />
  <Field index="2" type="value" key="threat.source" />
  <Field index="3" type="value" key="threat.category" />
  </Fields>
  </FlatFileFeed>

 </FDF>
```

Note: An example of a CsvFileFeed definition file would be the same as this one, except that the `FlatFileFeed` tag would be `CsvFileFeed`.

Feed Definition Equivalents for Custom Feed Wizard Parameters

The Security Analytics Custom Feed wizard provides options to define the structure of the data feed file. These correspond directly to attributes in the feed definition (.xml) file.

Security Analytics Parameter	Feed Definition File Equivalent
(Define Feed Tab) Feed Type	Select: **Default** - to define a feed based on a .csv formatted feed data file. **STIX** - to define a feed based on STIX formatted.xml file.
(Define Feed Tab) Feed Task Type	Select: **Adhoc** - to create an on-demand feed. **Recurring** - to update the .csv or .xml file persistently and store it in a location accessible by Security Analytics, so Security Analytics downloads a file at regular intervals and pushes it to the downstream devices.
(Define Feed tab) **Name**	The custom feed name in the feed data file. It corresponds to the **`flat-feedfile name`** attribute in the feed definition file. For example, Dynamic DNS Test Feed. **Note:** You can now use special characters to define the name of the custom feed.
(Define Feed tab) **File/ Browse**	This is the name of the feed data file. It corresponds to the **`flatfeedfile path`** attribute in the feed definition file. For example, dynamic_dns.csv.
(Advanced Options tab) **XML Feed File**	The name of the feed definition file. For example, `dynamic_dns.xml`.
(Advanced Options tab) **Separator**	The separator character used to separate attributes in the feed data file. It corresponds to the **`flatfeedfile separator`** separator in the feed definition file. For example, a comma.
(Advanced Options tab) **Comment**	The character used to identify a comment in the feed data file. It corresponds to the **`flatfeedfile comment`** attribute in the feed definition file. For example, #.

Security Analytics Parameter	Feed Definition File Equivalent
(Define Columns tab, Define Index) **Type**	The type of lookup value in the index position of the feed data file. **IP** means that each row in the feed data file contains an IP address in the lookup value position. The IP value is in dotted-decimal format (for example, 10.5.187.42). **IP Range** means that each row in the feed data file contains a range of IP addresses in the lookup value position. The IP range is in CIDR format (for example, 192.168.2.0/24). **Non IP** means that the each row in the feed data file contains a metadata value other than IP address in the lookup value position. The Service Type and Truncate Domain, and Callback Keys fields become active for a Non IP index.
(Define Columns tab, Define Index) **CIDR**	Specifies that the IP value in the lookup position is in CIDR format. The **CIDR** attribute sets the IP address format in the field to Classless Inter-Domain Routing (CIDR) notation.
(Define Columns tab, Define Index) **Service Type**	For a Non IP index, the integer service type to filter meta lookups. It corresponds to the **MetaCallback apptype** attribute in the feed definition file. A value of **0** indicates no filtering by service type.
(Define Columns tab, Define Index) **Truncate Domain**	For a Non IP index, for meta values that contain domain names (for example, hostnames), the system can strip off the host specific element in the data. Truncate Domain corresponds to the **MetaCallback truncdomain** attribute. If the value is www.example.com, it is truncated to example.com. A value of **False** selects no truncation, and **True** selects truncation.

Security Analytics Parameter	Feed Definition File Equivalent
(Define Columns tab, Define Index) **Callback Keys**	For a Non IP index, the available meta keys to match on instead of ip.src/ip.dst (the defaults for IP index type) are selectable from the drop-down list. The Callback Key corresponds to the **MetaCallback name** attribute, and the index column of the csv file must contain data that can match the chosen meta key. For example, if the username meta key is chosen, the index column of the csv file needs to be populated with users to be matched.
(Define Columns tab, Define Index) **Index Column**	Identifies the column in the feed data file that provides the lookup value for the row. Each position in each row of the feed data file is identified by a **Field index** attribute in the feed definition file. A field with an index of **1** is the first entry in a row, the second field has an index of **2**, the third field has an index of **3**, and so on.
(DEFINE VALUES) **Key**	The name of the **LanguageKey**, as defined in the feed definition file, for which meta is created from this row of the feed data file. It corresponds to the **Field key** attribute in the feed definition file. A key applies only to a field whose type is set to **value**. In the feed definition file, there is a list of LanguageKeys from **index.xml**, or a summary name if Source Name and Destination Name are used. For example, **reputation** is a summary name for **reputation.src** and **reputation.dst**. This value is referenced by the Field key attribute.

Create a Custom Feed

You can easily create a custom feed using the Custom Feed wizard. To complete this procedure, you need a feed data file in `.csv` format. If you also have an associated feed definition file in `.xml` format, which describes the structure of the feed data file, you can use the feed definition file to create a feed. The Custom Feed wizard can create the feed based on a feed data file, or based on a feed data file and corresponding feed definition file.

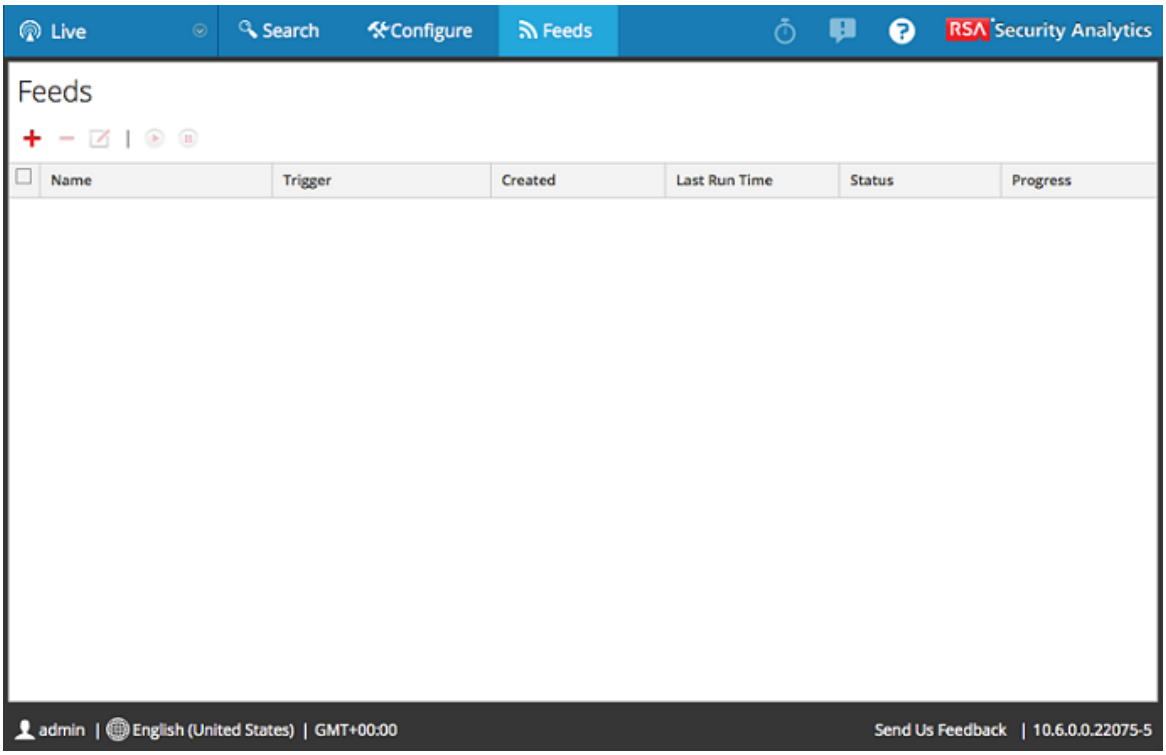
After completing this procedure, you will have created a custom feed.

The feed data file (`.csv`) and optionally the feed definition file (`.xml`) must be available on the local file system for an on-demand custom feed. For a recurring custom feed, the files must be available at a URL that is accessible to the Security Analytics server.

To create a custom feed:

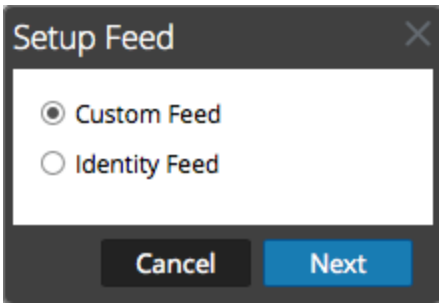
1. In the **Security Analytics** menu, select **Live > Feeds**.

   The Feeds view is displayed.



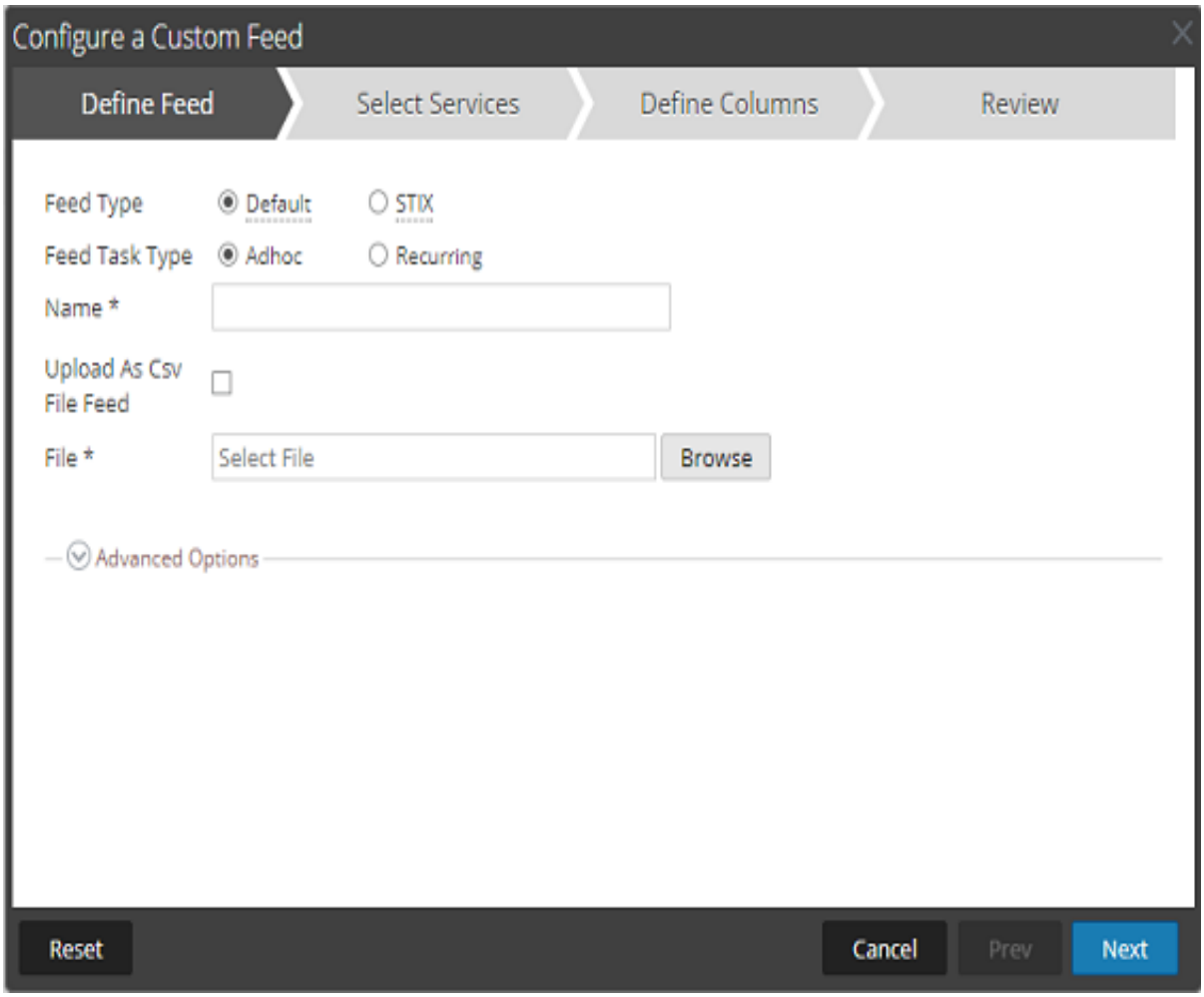
2. In the toolbar, click  .

   The Setup Feed dialog is displayed.



3. To select the feed type, click **Custom Feed** and **Next**.

   The Configure a Custom Feed wizard is displayed, with the Define Feed form open.



4. To define an on-demand feed task that executes once, select **Adhoc** in the **Feed Task Type** field and do one of the following:

   a. (Conditional) To define a feed based on a CsvFileFeed file, select the **Upload as Csv File Feed** checkbox, type the feed **Name**, select a .csv content **File** from the local file system, and click **Next**. If you do not select the checkbox, the .csv file will be a

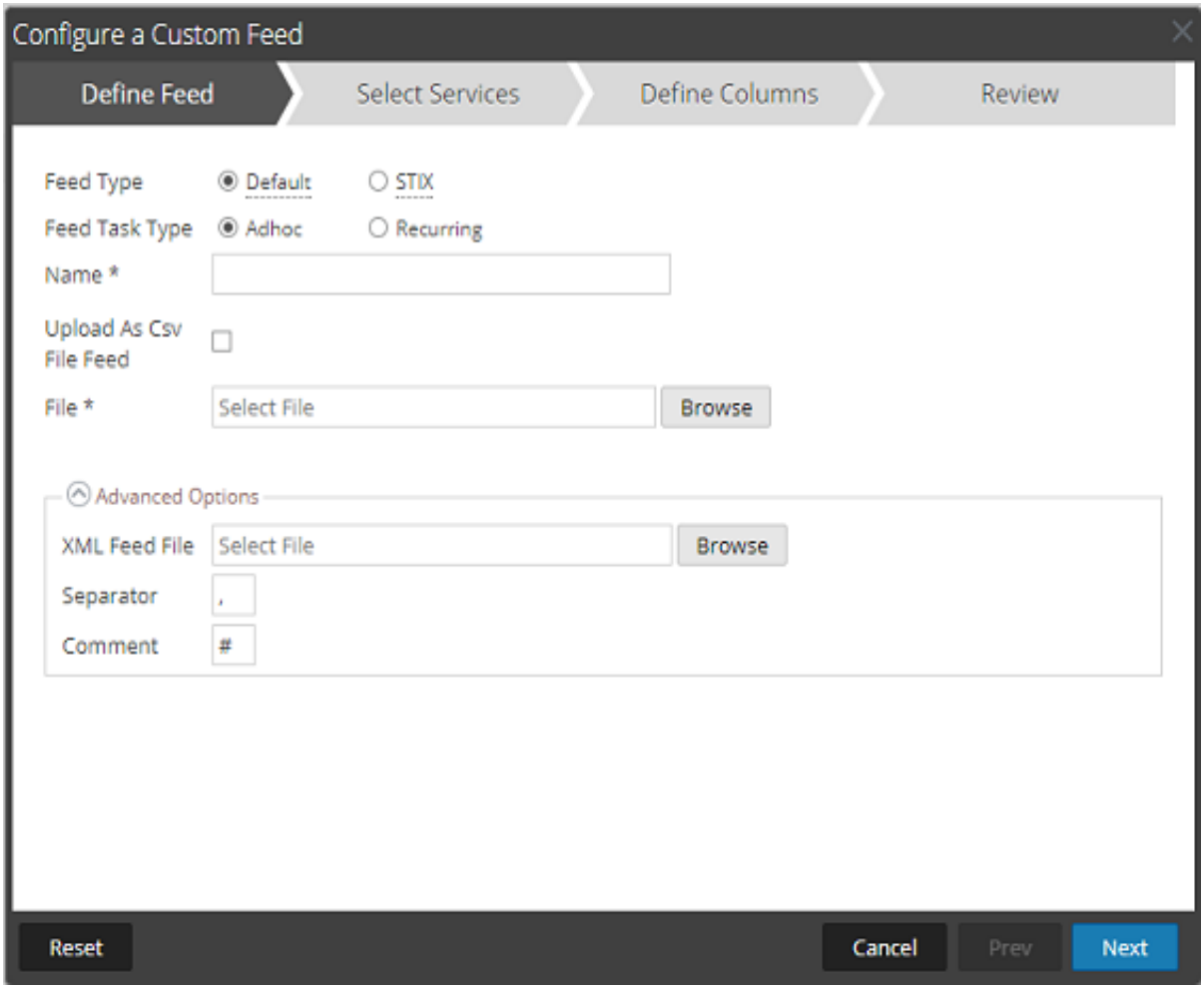
FlatFileFeed file. For more information, see Create and Deploy Custom Feeds Using a Wizard.

Note: When you select the **Upload as Csv File Feed** checkbox, the XML feed options under **Advanced** are unavailable.

b. (Conditional) To define a feed based on an XML feed file, select **Advanced Options**.

Note: Ensure that the **Upload as Csv File Feed** checkbox is deselected.

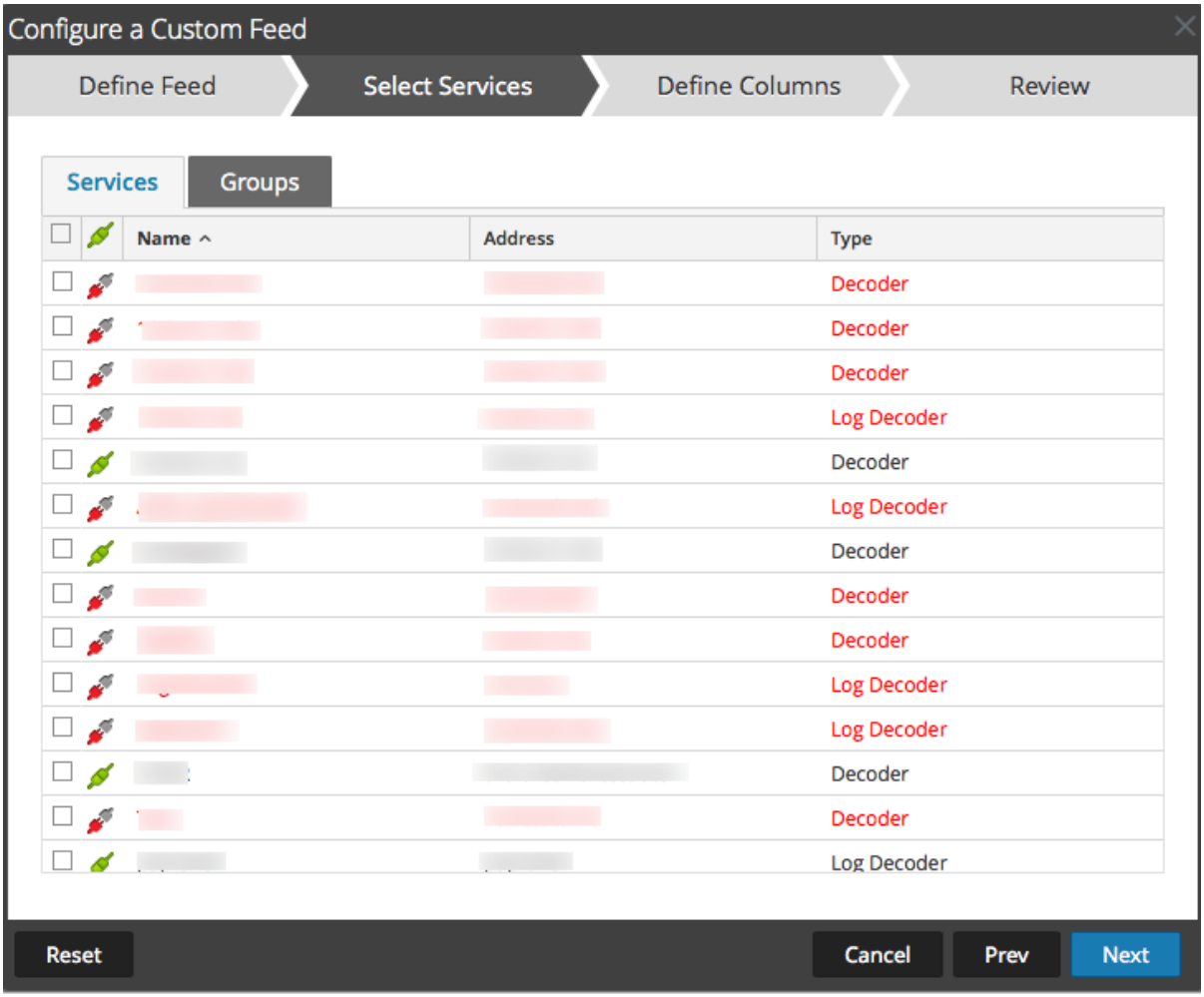
The Advanced Options are displayed:



c. Select an XML feed file from the local file system, choose the **Separator** (default is comma), and specify the **Comment** characters used in the feed data file (default is #), and click **Next**.

d. The Select Services form is displayed. This is an example of the form for a feed based on a feed data file with no feed definition file. If you are defining a feed based on a feed

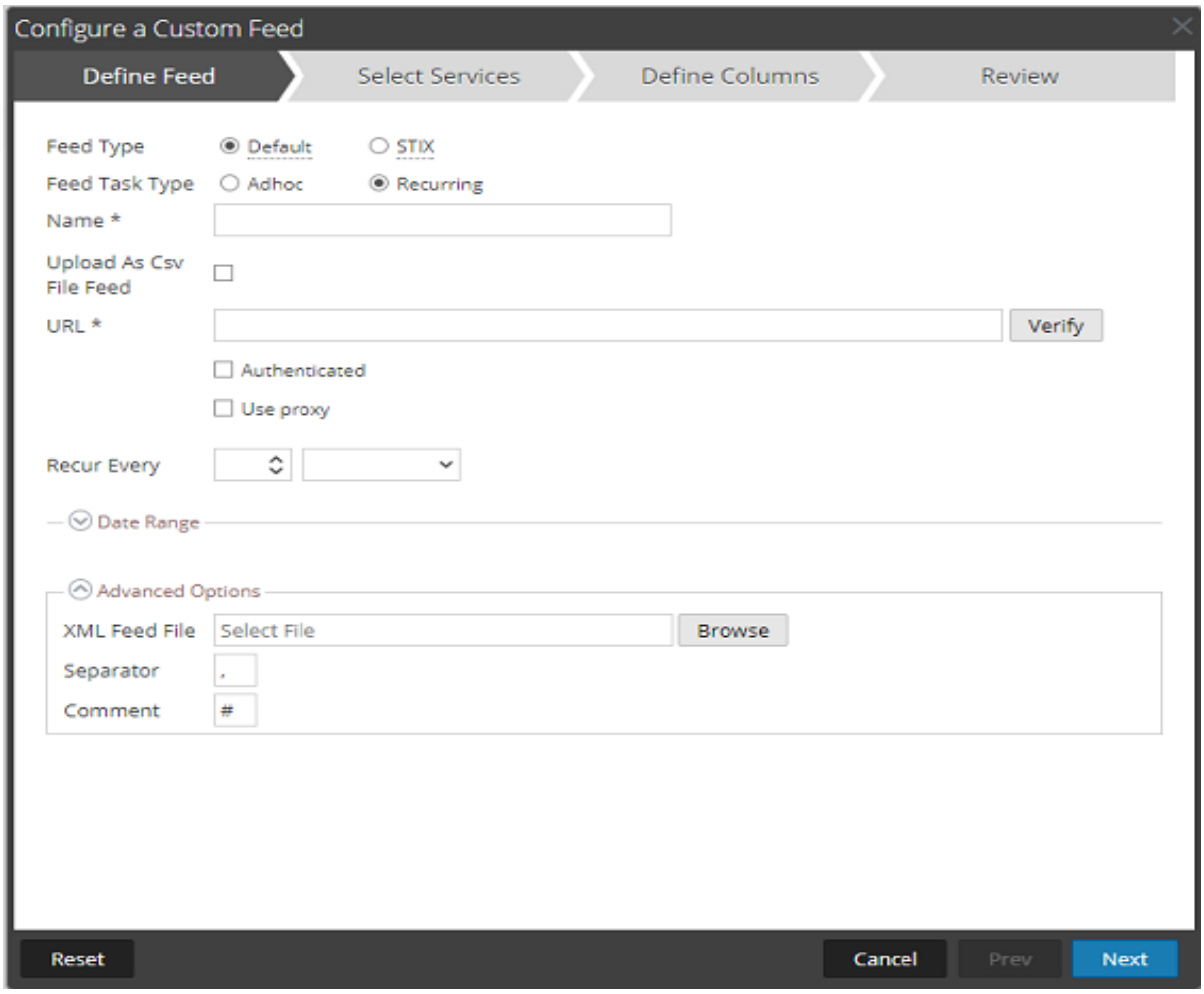
definition file, the Define Columns tab is not needed.



5. To define a recurring feed task that executes repeatedly at specified intervals, during a specified date range.

   a. Select **Recurring** in the **Feed Task Type** field.

      The Define Feed form includes the fields for a recurring feed.



b. In the **URL** field, enter the URL where the feed data file is located, for example, `http://<hostname>/<feeddatafile>.csv`, and click **Verify**.

   Security Analytics verifiies the location where the file is stored, so that Security Analytics can check for the latest file automatically before each recurrence.

c. (Optional) If the URL has restricted access and requires authentication using your username and password, select **Authenticated**.

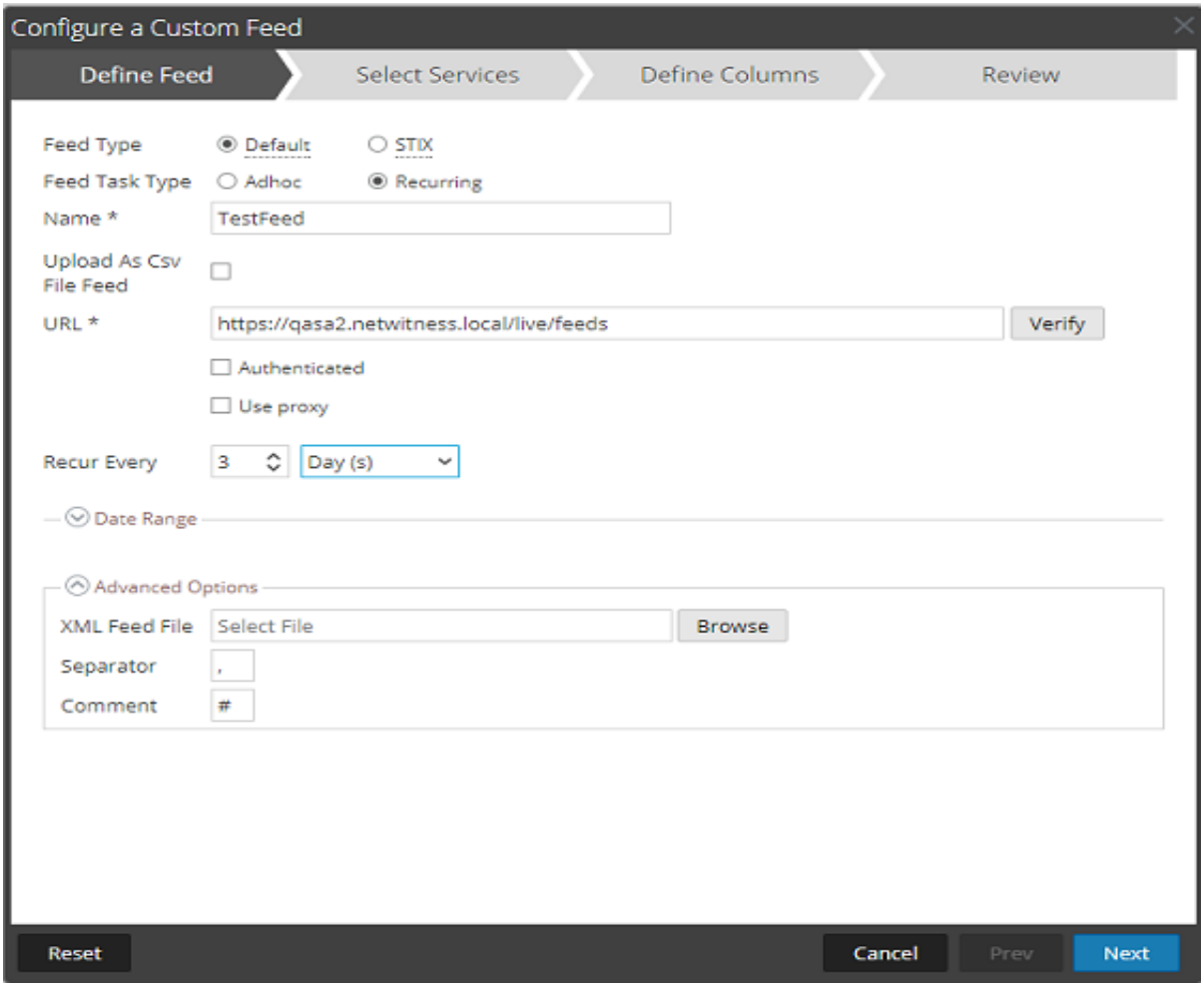
   Security Analytics provides your user name and password for authentication to the URL.

d. If you want the Security Analytics server to access the Feed URL through a proxy, select **Use Proxy**. For more information on configuring a proxy, see the **Configure Proxy for Security Analytics** topic in the *System Configuration Guide*. By default, the **Use Proxy** checkbox is not selected.

e. To define the interval for recurrence, do one of the following:

- Specify the number of minutes, hours, or days between recurrences of the feed.
- Specify recurrence every week, and select the days of the week.

f. To define the date range for the execution of the feed to recur, specify the **Start Date** and time and the **End Date** and time.



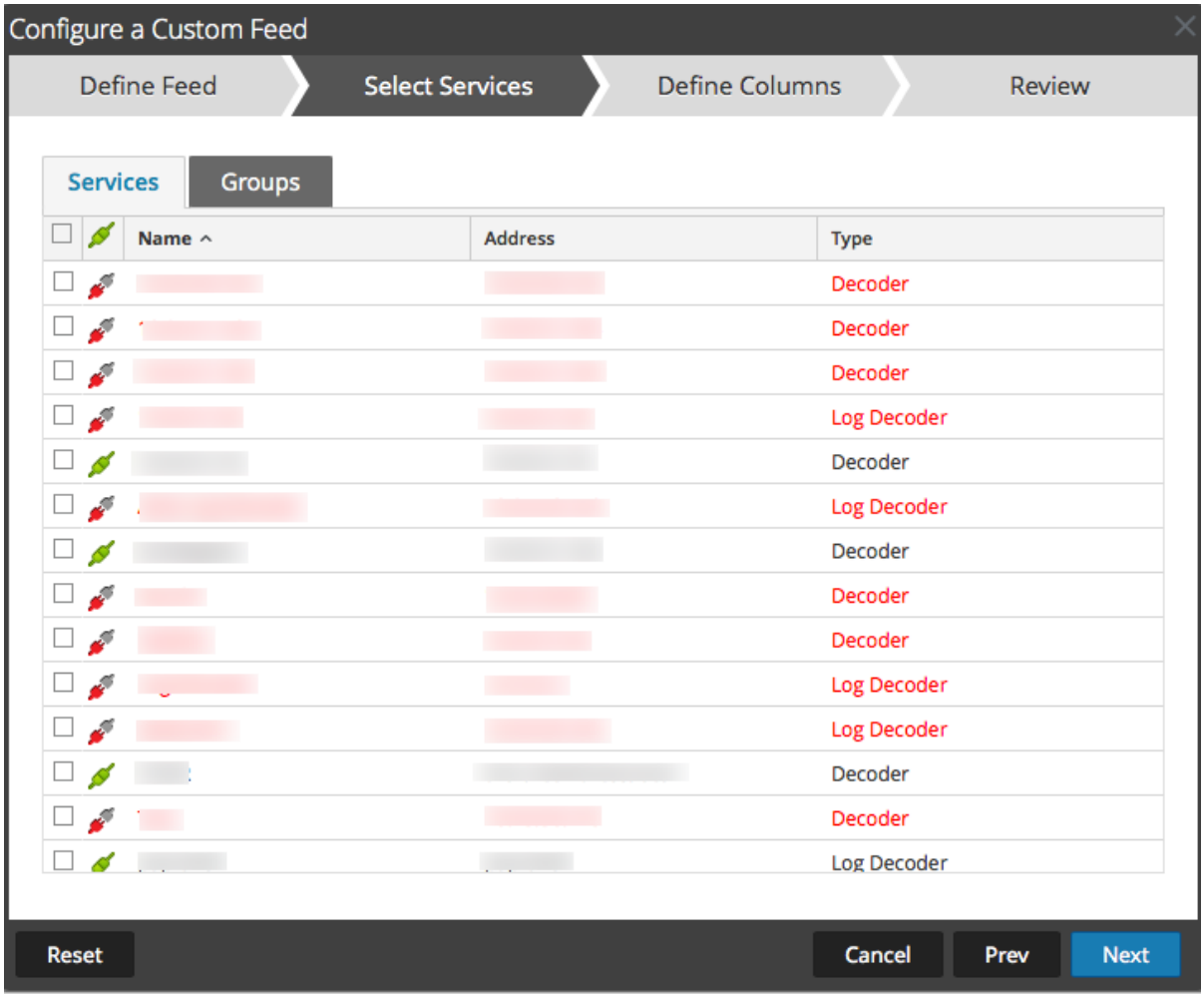
6. (Conditional) If you want to define a feed based on an XML feed file:

- Type the feed **Name**, select **Advanced Options**.

  The Advanced Options fields are displayed.

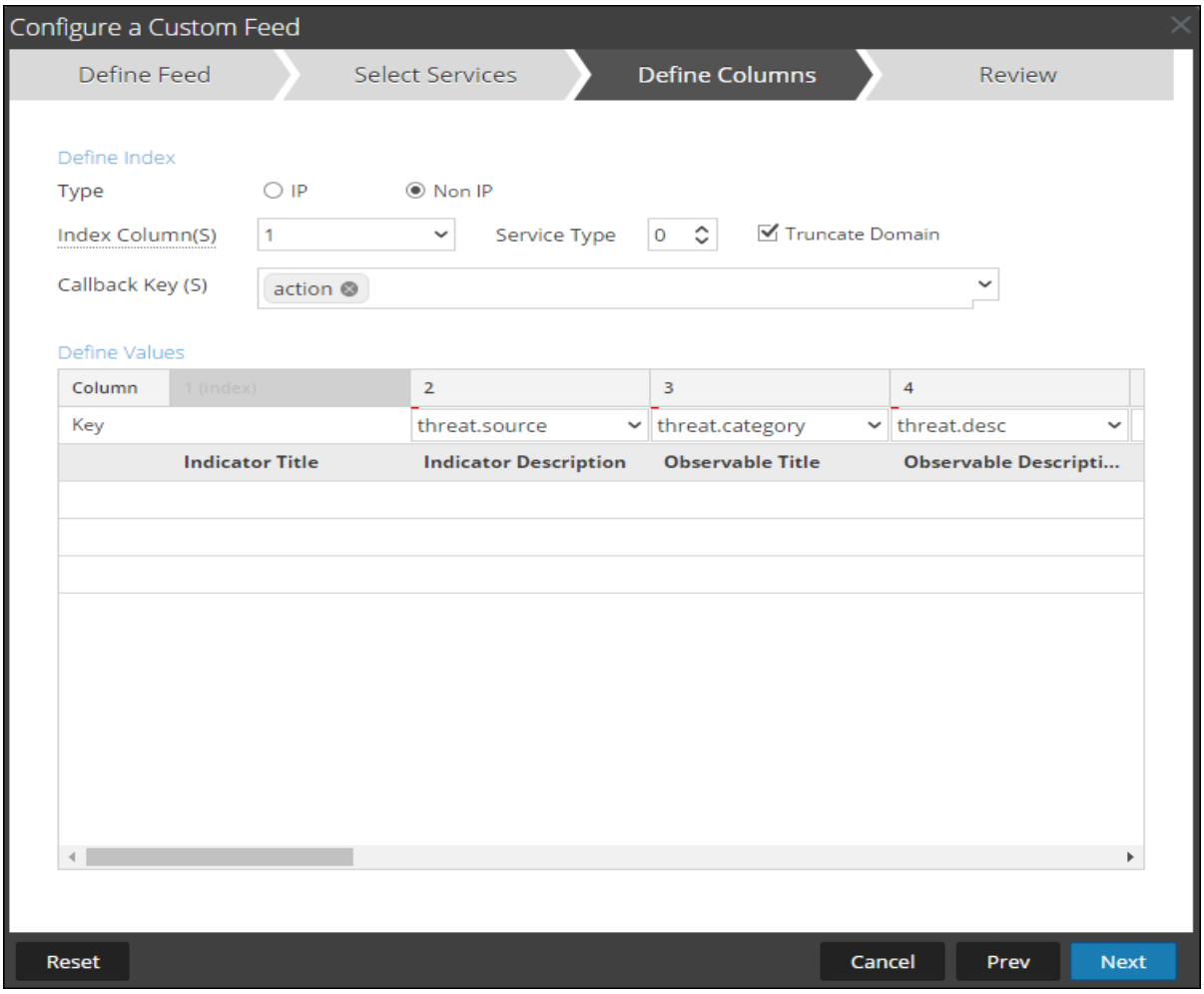
- Select an XML feed file from the local file system, choose the **Separator** (default is comma), specify the **Comment** characters used in the feed data file (default is #) and click **Next**.

The Select Services form is displayed.

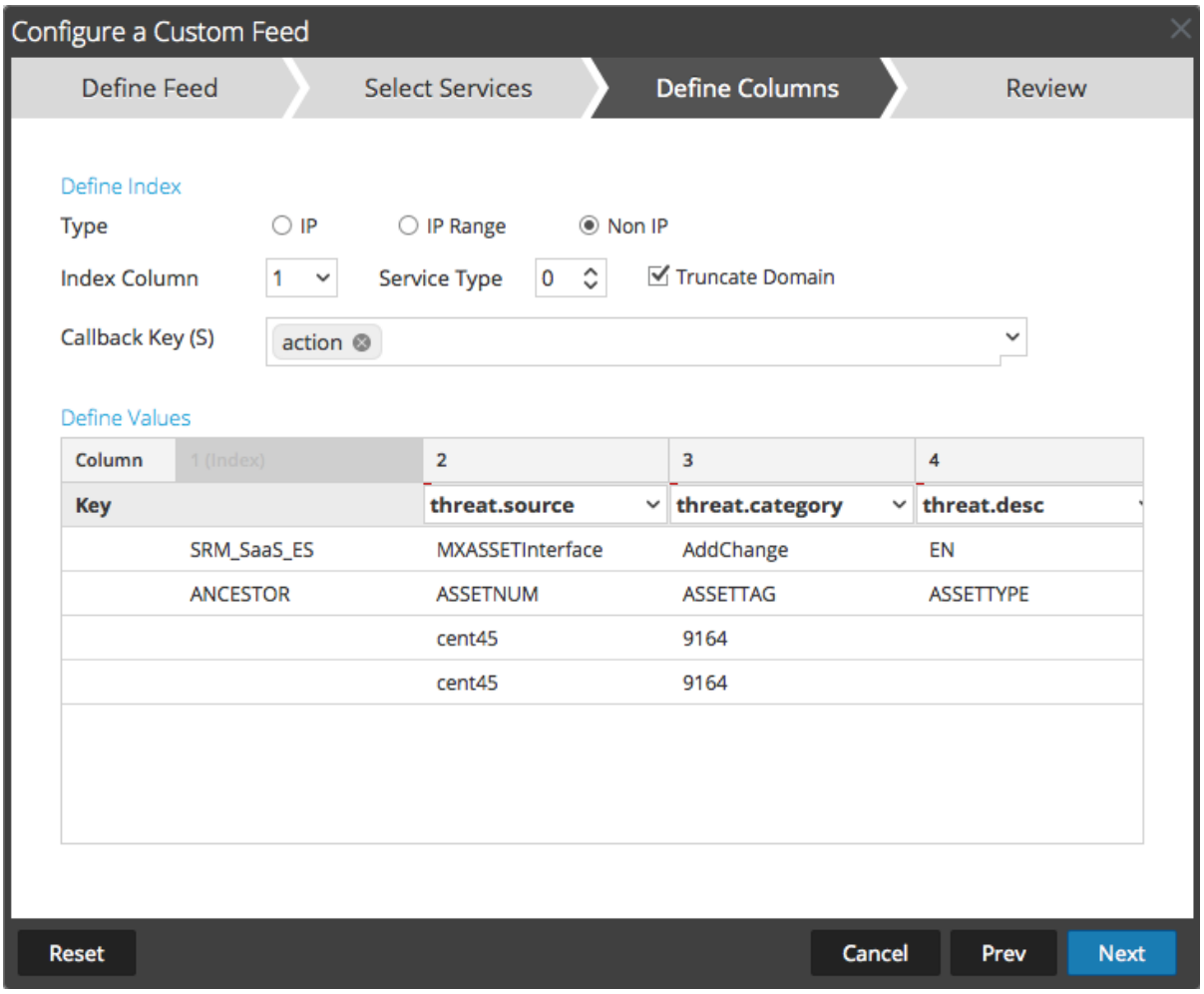


7. To identify services on which to deploy the feed, do one of the following:
   a. Select one or more Decoders and Log Decoders, and click **Next**.
   b. Click the **Groups** tab and select a group. Click **Next**.

      The Define Columns form is displayed.
8. To map columns in the Define Columns form:
   a. Define the Index type: **IP**, **IP Range**, or **Non IP**, and select the index column.
   b. (Conditional) If the index type is **IP** or **IP Range** and the IP address is in CIDR notation, select **CIDR.**
   c. (Conditional) If the index type is **Non IP**, additional settings are displayed. Select the service type and **Callback Keys,** and optionally select the **Truncate Domain** option.

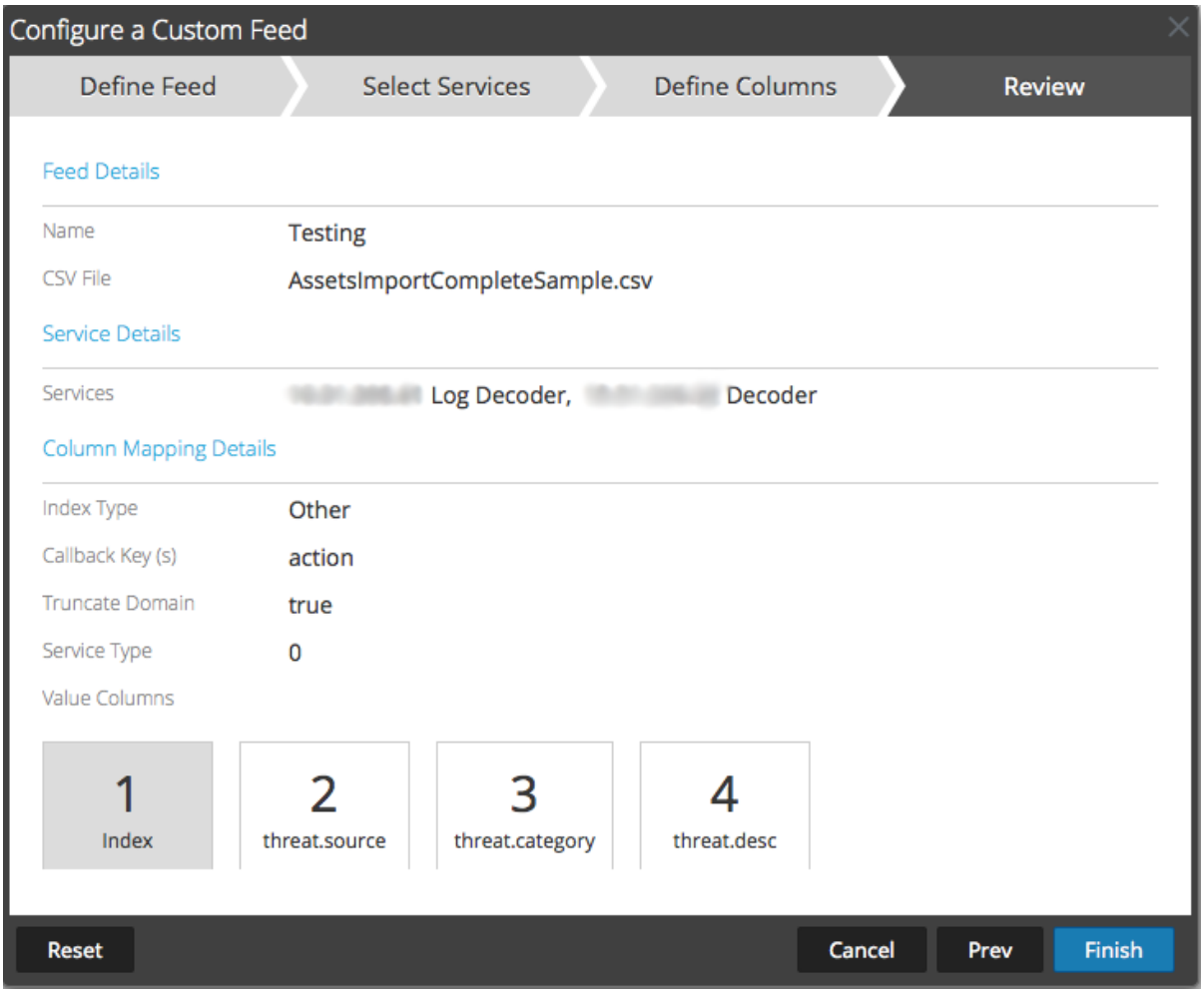


d. Select the language key to apply to the data in each column from the drop-down list. The meta displayed in the drop-down list is based on the meta available for the service define values. You can also add other meta based on advanced expertise.



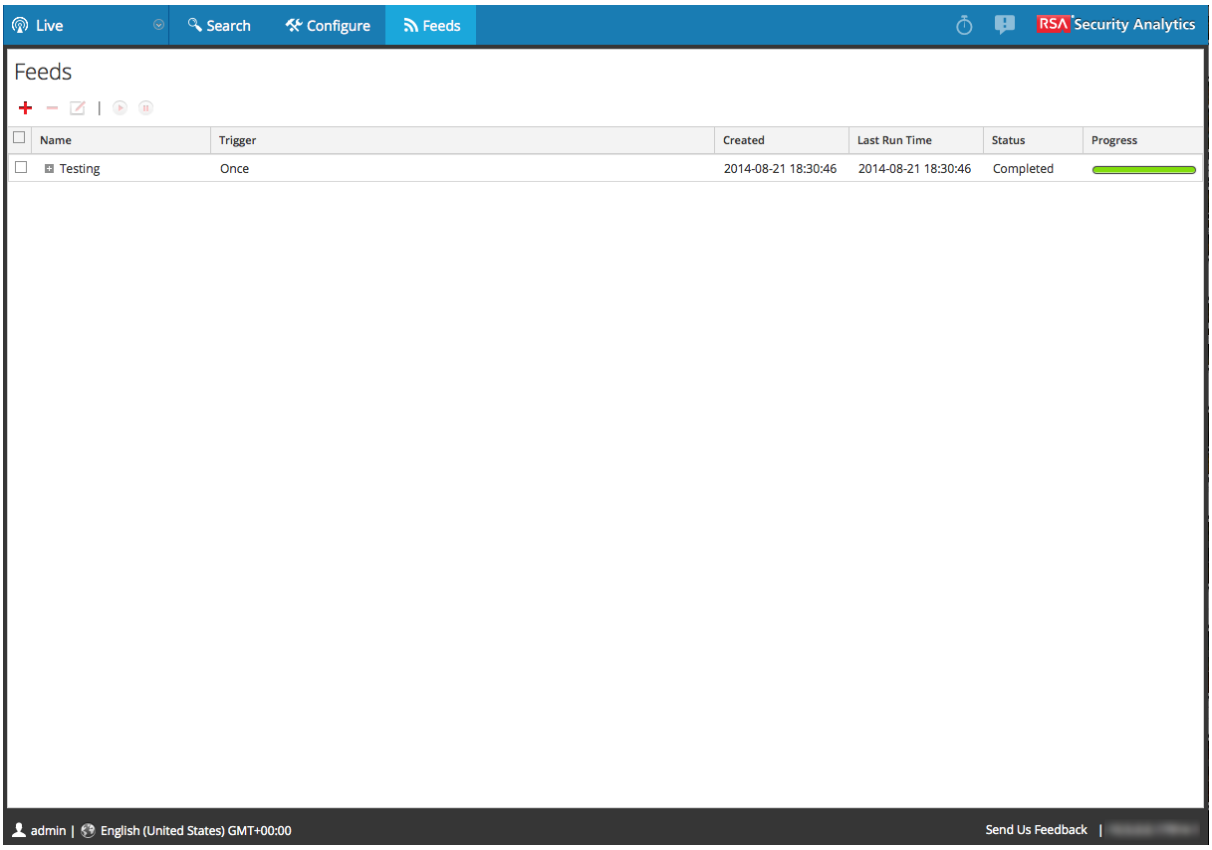
e. Click **Next**.

The Review form is displayed.



9. Anytime before you click **Finish**, you can:
   - Click **Cancel** to close the wizard without saving your feed definition.
   - Click **Reset** to clear the data in the wizard.
   - Click **Next** to display the next form (if not viewing the last form).
   - Click **Prev** to display the previous form (if not viewing the first form).

10. Review the feed information, and if correct, click **Finish**.
11. Upon successful creation of the feed definition file, the Create Feed wizard closes, and the feed and corresponding token file are listed in the Feed grid and progress bar tracks completion. You can expand or collapse the entry to see how many services are included, and which services were successful.


Live
Search
Configure
Feeds
RSA Security Analytics
Feeds
Name
Trigger
Created
Last Run Time
Status
Progress
Testing
Once
2014-08-21 18:30:46
2014-08-21 18:30:46
Completed
admin | English (United States) GMT+00:00
Send Us Feedback |

Create an Identity Feed

You can easily create an Identity feed and populate it to selected Decoders and Log Decoders. After completing this procedure, you will have created an Identity feed.

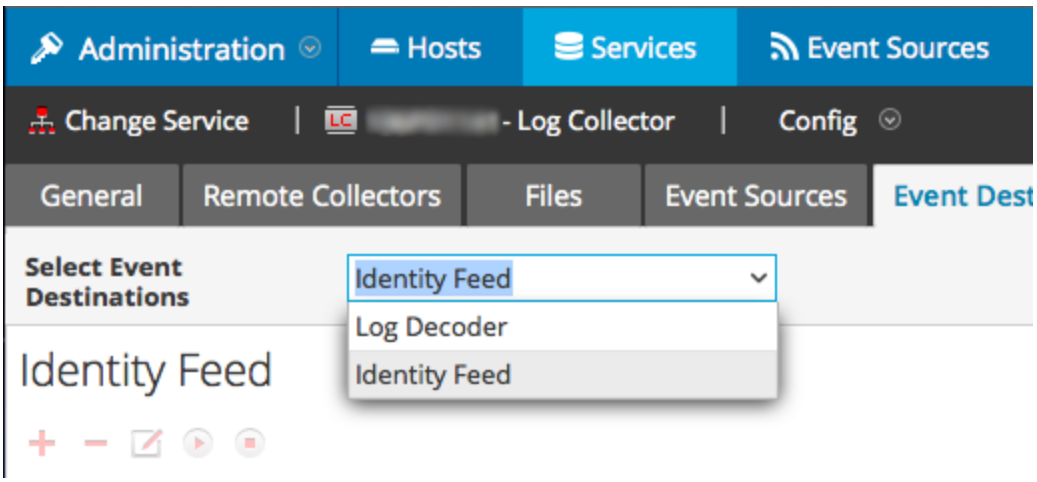
Prerequisites

In order to create an identity feed, you need to have:

- A Log Collector service with an Identity Feed Event Processor
- A Log Collector service with Windows Collection configured and enabled

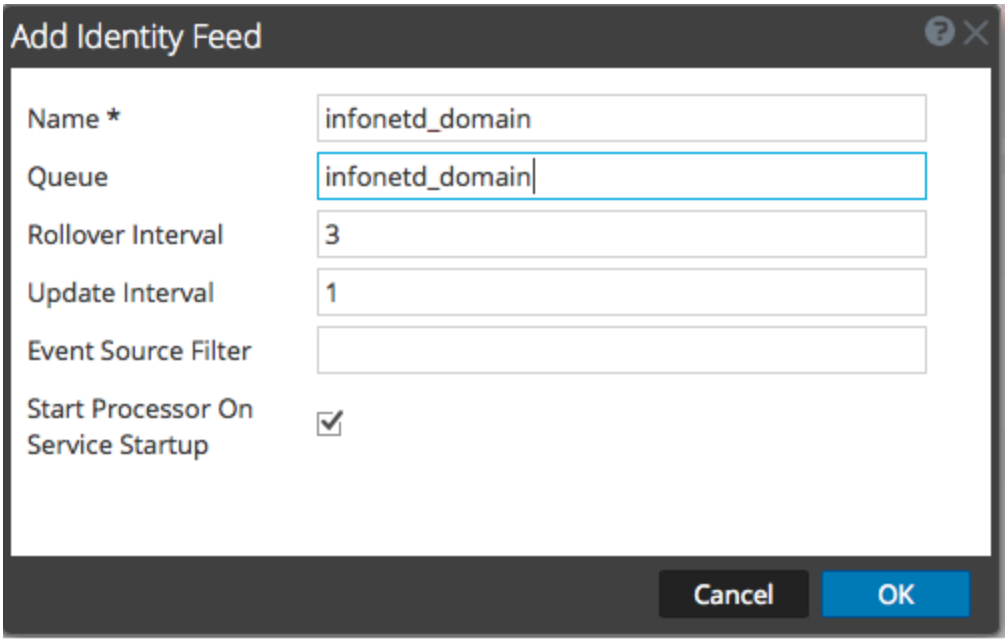
Create an Identity Feed

1. Add a destination for the feed.
   a. In the **Security Analytics** menu, select **Administration > Services**.
   b. In the **Services** grid, select a **Log Collector** service.
   c. Click ⊙ under **Actions** and select **View > Config**.
   d. Select the **Event Destinations** tab.
   e. In the Select **Event Destinations** field, select **Identity Feed**.



   f. Click + and enter a unique name for the feed.

The Queue name identifies the feed within the log collector. Use the name of the feed for the Queue.

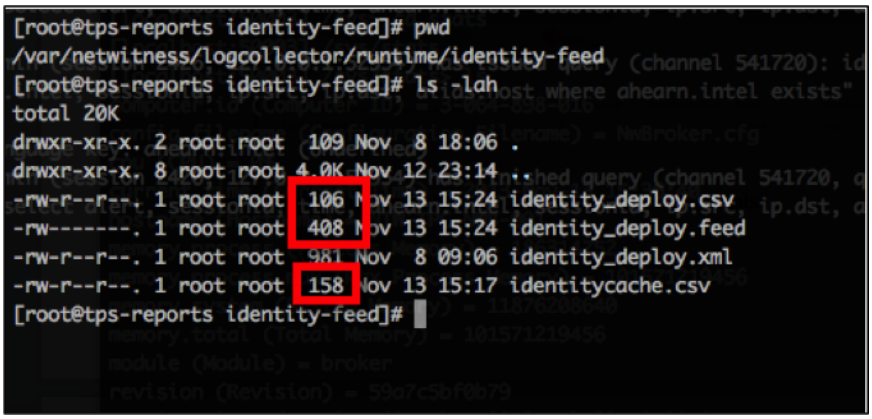


g. Click **OK**.

2. Test generation of messages.

a. Have users log into Windows boxes on the domain to generate the appropriate log messages on the domain controllers for testing.

b. Verify that data is written to the feed files. SSH to the Log Decoder/Collector or Virtual Log Collector being configured. Navigate to `/var/netwitness/logcollector/runtime/identity-feed` and verify that the **Identity_deploy** files are getting populated with data.

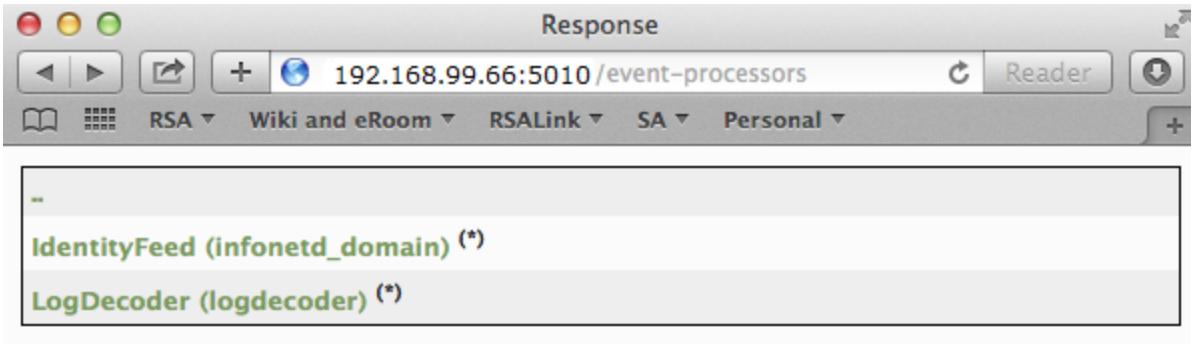


c. Open up a web browser (Non-Internet Explore browsers preferred) and log in to the REST interface of the Log Collector. Use administrative credentials when logging in. For

example, if the IP address of your log collector is 192.168.99.66, the URL would be:

- SSL not enabled: **http://192.168.99.66:50101/event-processors**
- SSL enabled: **https://192.168.99.66:50101/event-processors**

The browser screen should look like this:

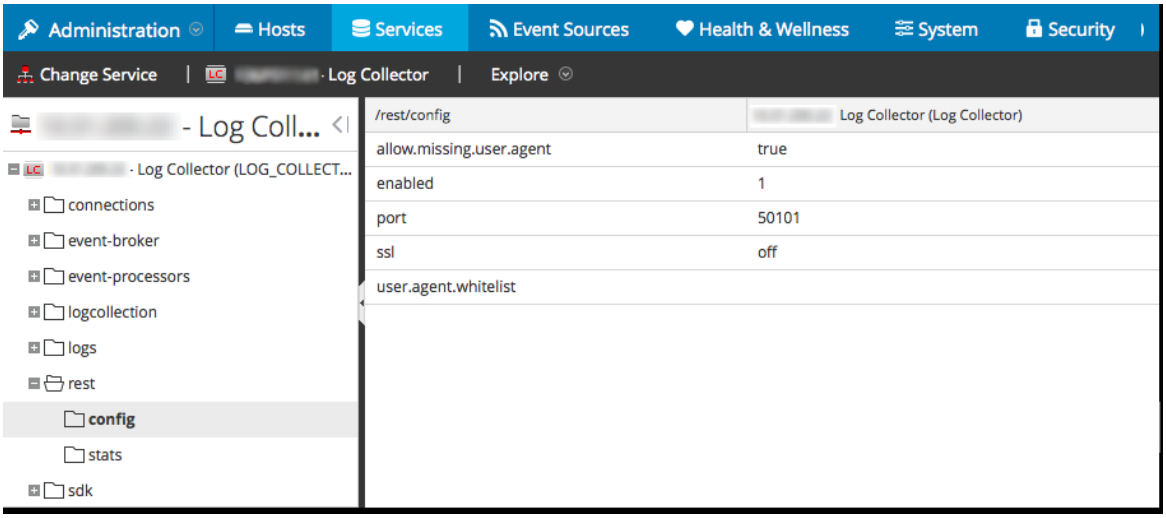


Notice the screen contains the name of the identity feed you created earlier (**infonetd_domain**, in this example).

For the identity feed to function correctly, port 50101 must be active on the Log Collector, and you must determine whether SSL encryption is active.

d. From the Security Analytics menu, select **Administration > Services > *<Log Collector being setup>* > Actions > View > Explore**.

e. In the left pane, expand **rest > config**.



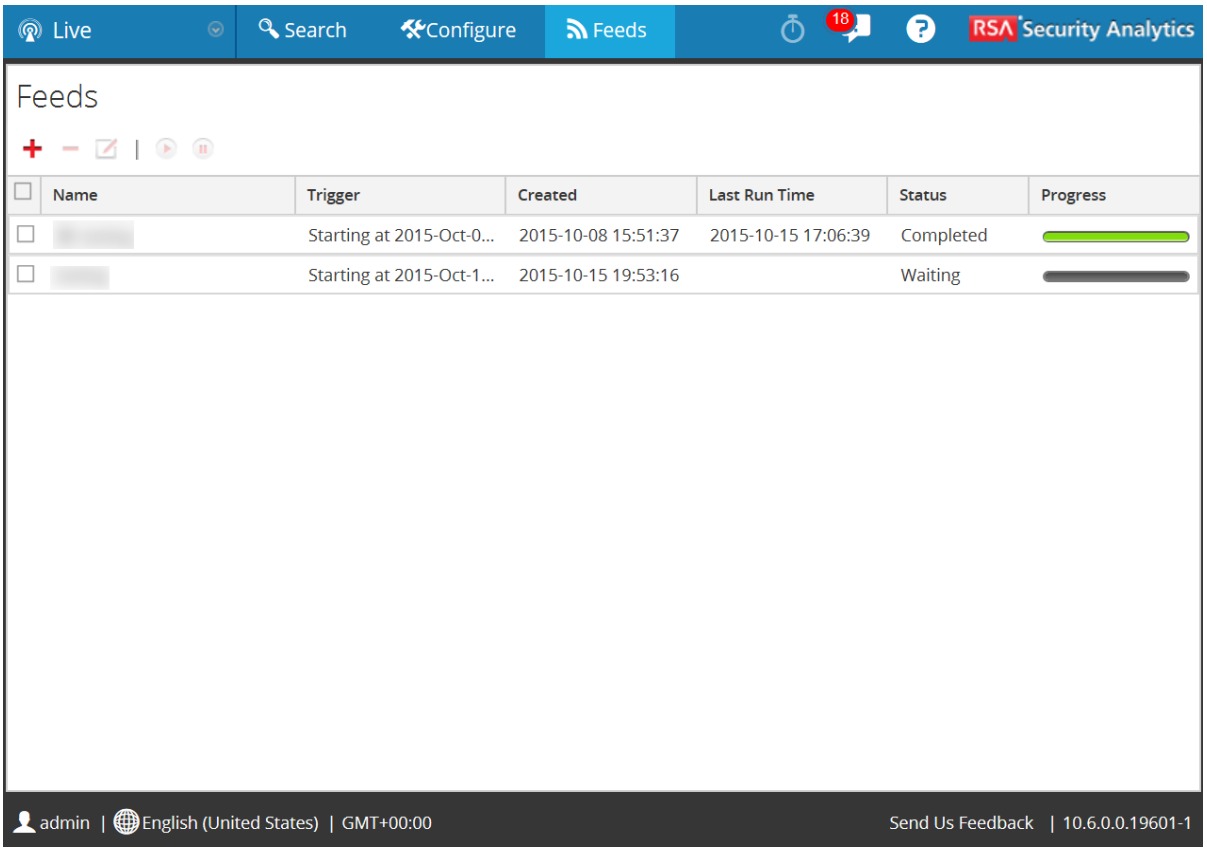
For REST to be active, **enabled** must be set to **1**.

f. Note the value for **ssl**. If SSL should be enabled for your environment, this must be set to **on**.

Note: If you changed the setting for either the **enabled** or **ssl** option you must restart the Log Collector service before moving forward.

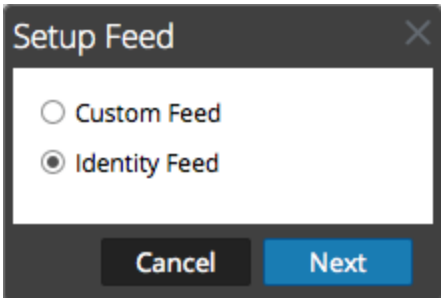
3. In the **Security Analytics** menu, select **Live > Feeds**.

   The Feeds grid is displayed.



4. In the toolbar, click  .

   The Setup Feed dialog is displayed, with Identity Feed selected by default.



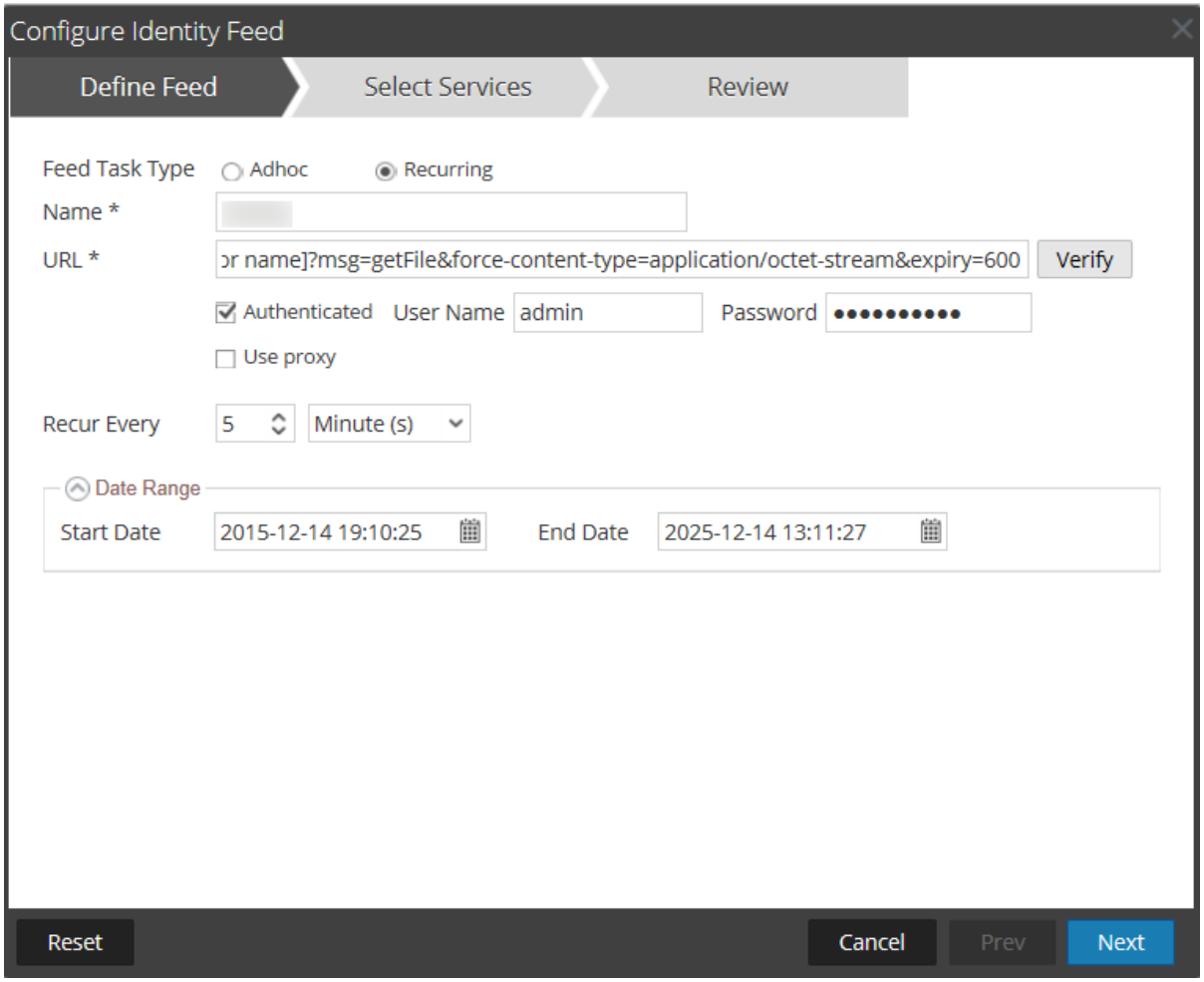
5. Select **Identity Feed** and click **Next**.

   The Configure Identity Feed panel opens with the **Define Feed** tab displayed.

6. (Conditional) You can create an on-demand or recurring feed.

- To define an on-demand Identity feed task that executes once, select **Adhoc** in the **Feed Task Type** field, type the feed **Name**, and browse for and open the feed.
- To define a recurring Identity Feed task that executes on a recurring basis, select **Recurring** in the **Feed Task Type** field.

  The **Define Feed** form includes the fields for a recurring feed.



Note: Security Analytics verifies the location where the file is stored, so that Security Analytics can check for the latest file automatically before each recurrence.

7. Fill in and verify the URL field.
   a. In the **URL** field, enter the URL where the feed data file is located. This is the REST API interface that was setup earlier. You need to know the following information to construct the URL:
      - The IP address of the log collector being used to construct the Identity Feed file.
      - The identity queue name, as set in step 2c.

- Whether or not SSL is enabled on the log collector REST port, as set in step 2f.

You construct this value as follows:

- SSL enabled: `https://<LogCollector>:50101/event-processors/<ID Event processor name>?msg=getFile&force-content-type=application/octet-stream&expiry=600`
- SSL not enabled: `http://<LogCollector>:50101/event-processors/<ID Event processor name>?msg=getFile&force-content-type=application/octet-stream&expiry=600`

So, using our example from earlier, the complete value that you would enter into this field is as follows:

```
http://192.168.99.66:50101/event-processors/infonetd_domain?msg=getFile&force-content-type=application/octet-stream&expiry=600?msg=getFile&force-content-type=application/octet-stream&expiry=600
```

b. For the URL verification to work correctly, it is important that the Security Analytics UI server can access the log collector's REST API port (50101). This can be tested by going to the Security Analytics UI server via SSH. Once there, run the following command:

- SSL enabled: `curl -vk https://<ip of log collector>:50101`
- SSL not enabled: `curl -v http://<ip of log collector>:50101`

If the curl command does not connect then there may be a network firewall or routing issue between the Security Analytics UI server and the Log Collector.

Example of Bad connection:

```
* About to connect() to 192.168.99.66 port 50105 (#0)
* Trying 192.168.99.66... No route to host
* couldn't connect to host
* Closing connection #0
curl: (7) couldn't connect to host
```

Example of Good connection:

```
* About to connect() to 192.168.99.66 port 50105 (#0)
* Trying 192.168.99.66... connected
* Connected to 192.168.99.66 (192.168.99.66) port 50105 (#0)
> GET / HTTP/1.1
```

```
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu)
libcurl/7.19.7 NSS/3.19.1 Basic ECC zlib/1.2.3 libidn/1.18
libssh2/1.4.2
> Host: 192.168.99.66:50105
> Accept: */*
>
< HTTP/1.1 401 Unauthorized
< Content-Length: 71
< Connection: Keep-Alive
< Pragma: no-cache
< Expires: -1
< Cache-Control: no-cache, no-store, must-revalidate
< WWW-Authenticate: Basic realm="NetWitness"
< Content-Type: text/xml; charset=utf-8
<
<?xml version="1.0" encoding="utf-8"?>
<error>401 Unauthorized</error>
* Connection #0 to host 192.168.99.66 left intact
* Closing connection #0
```

8. The REST API requires a username and password when attempting to pull the **identity_deploy.csv** file from the log collector. This can be any username and password that is available on the service itself. For details, see the "Services Security View" topic in the *Hosts and Services Guide*.

   To see which accounts are available, navigate to **Administration > Services > *<log collector being setup>* > Actions > View > Security**.

   Under the Users table, you see all the users that can be used in this step. It is suggested that a separate user account is created specifically for this setup, and is used nowhere else in the environment, for added security. For details, see the "Add a User and Assign a Role" topic in the *System Security and User Management Guide*.

9. To define the interval for recurrence, do one of the following:

   - Specify the number of minutes, hours, or days between recurrences of the feed.
   - To define the date range for the execution of the feed to recur, specify the **Start Date** and time and the **End Date** and time.

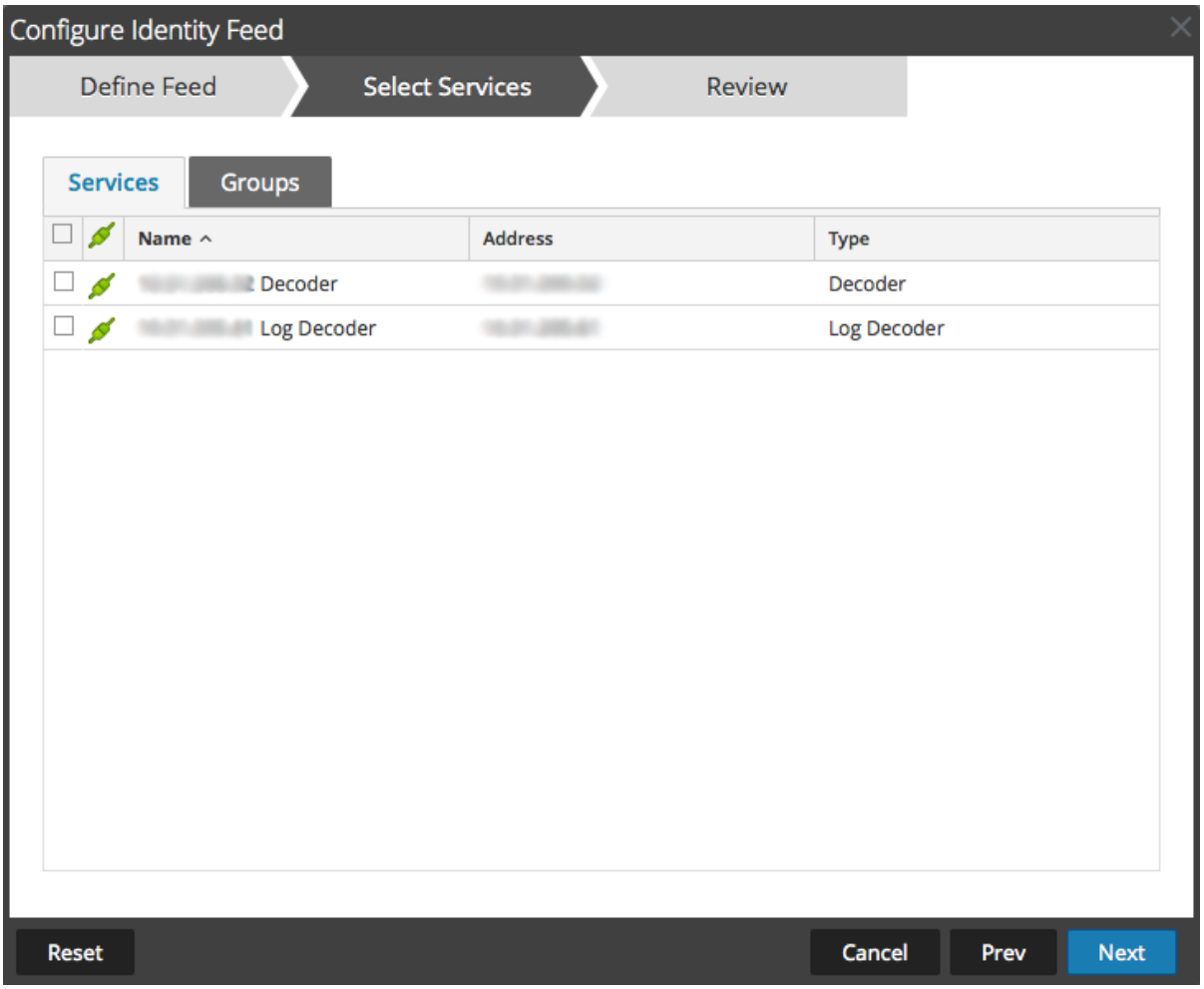
10. If using SSL encryption, you need to install the REST API SSL certificate for the log Collector into the Security Analytics UI server. For details, see Import the SSL Certificate.

    If, after importing the SSL certificate, the verification of the URL still fails, see Cannot Verify Identity Feed URL.

11. Click **Verify** to verify your identity feed configuration before you proceed to the Select Services form.

12. Click **Next**.

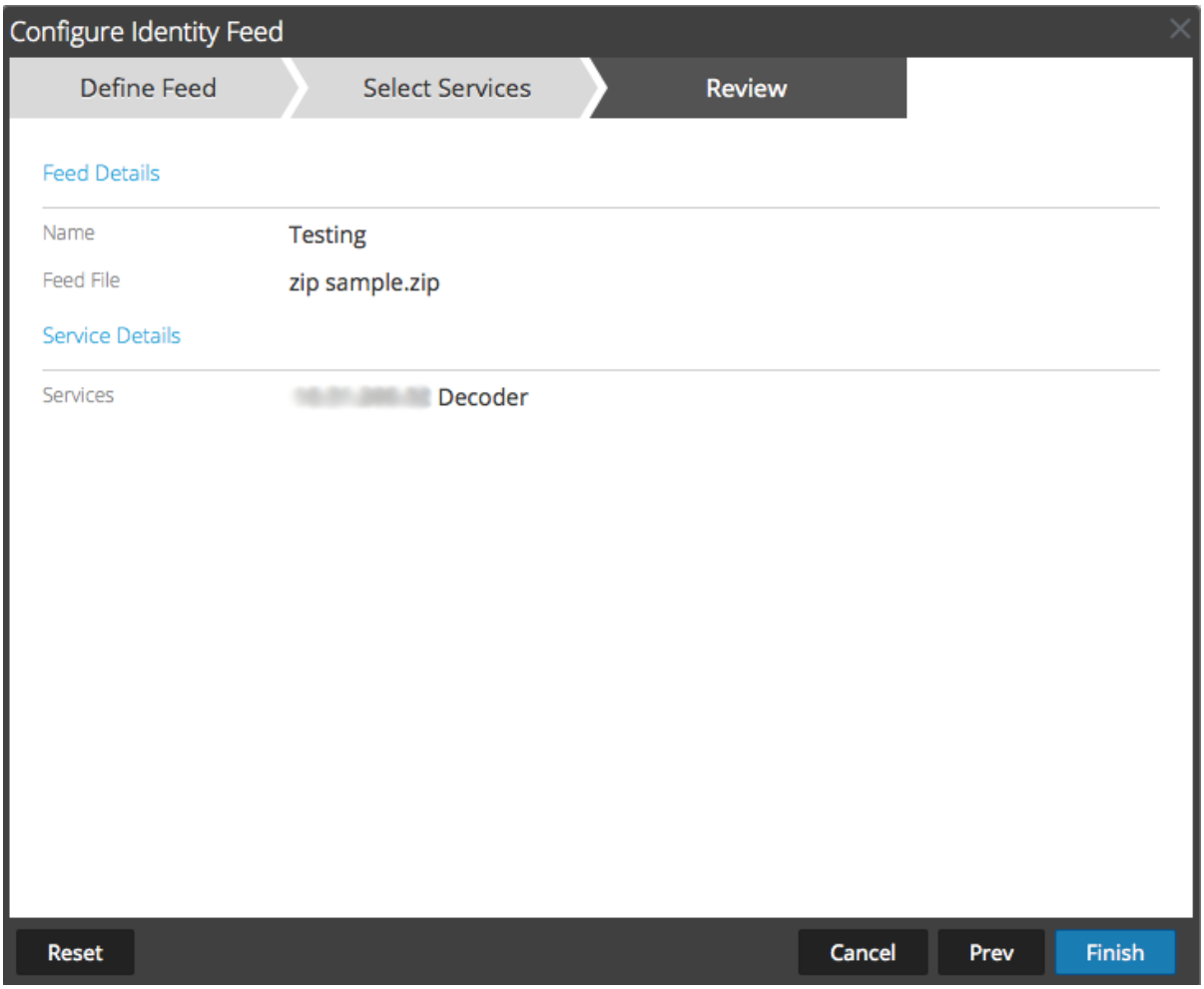
    The Select Services form is displayed.



13. To identify services on which to deploy the feed, select one or more Decoders and Log Decoders and click **Next**.

14. Click the **Groups** tab, select a group, and click **Next**.

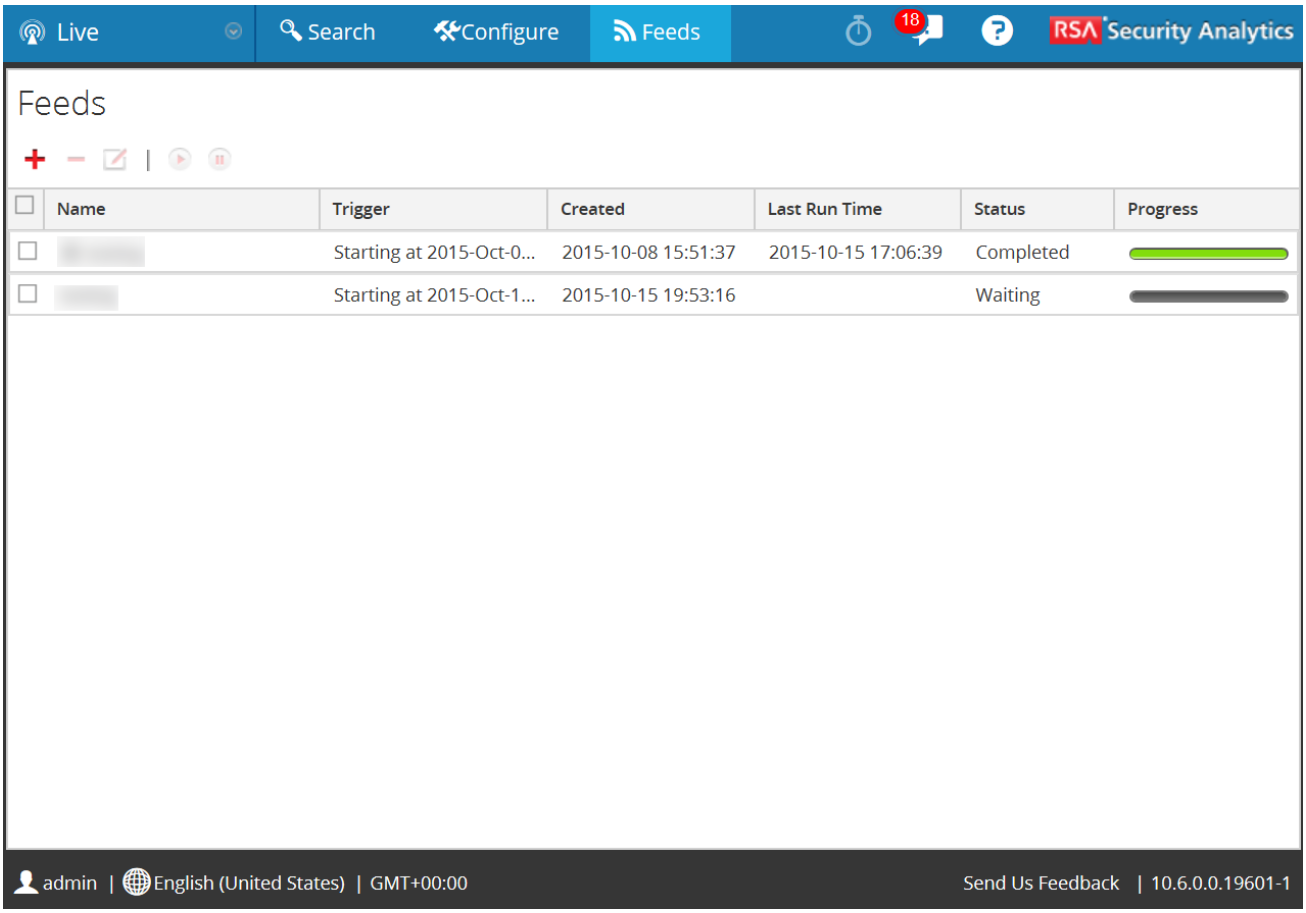
    The Review form is displayed.



Note: If a group of devices with Decoders and Log Decoders is used to create recurring or custom feeds and this group is deleted, you can edit the feed and add a new group to the feed.

15. Anytime before you click **Finish**, you can:
    - Click **Cancel** to close the wizard without saving your feed definition.
    - Click **Reset** to clear the data in the wizard.
    - Click **Next** to display the next form (if not viewing the last form).
    - Click **Prev** to display the previous form (if not viewing the first form).
16. Review the feed information, and if correct, click **Finish**.

Upon successful creation of the feed definition file, the Create Feed wizard closes, and the feed and corresponding token file are listed in the Feed grid and progress bar tracks completion. You can expand or collapse the entry to see how many services are included, and which services were successful.



Import the SSL Certificate

If SSL is configured on the Identity feed's Log Collector, follow these steps to import the Log Collector's SSL certificate into the Security Analytics UI server key store. If this certificate is not imported, the Security Analytics UI server will be unable to pull the Identify feed file from the Log Collector.

1. To pull the SSL certificate off the log collector, SSH into the Security Analytics UI server and run the following command:

   ```
   echo -n | openssl s_client -connect <HOST>:<PORT> | sed -ne
   '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >
   /tmp/<SERVERNAME>.cert
   ```

   This command saves the SSL certificate to `/tmp/<SERVERNAME>.cert`.

   For example:

   ```
   echo -n | openssl s_client -connect 192.168.99.66:50101 | sed
   -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >
   /tmp/logcollector.cert
   ```

2. To import the SSL certificate into the Security Analytics UI server, SSH into the UI server and run the following command:

   ```
   keytool -importcert -alias <name an alias for the cert> -file
   <the cert file pathname> -keystore /etc/pki/java/cacerts
   ```

   For example:

   ```
   keytool -importcert -alias logcollector01 -file
   /tmp/logcollector.cert -keystore /etc/pki/java/cacerts
   ```

3. The system requests a password. Enter the password for the keystore on the Security Analytics UI server, not for the jetty keystore. The default password is **changeit**.

4. Restart **jettysrv** to allow jetty to read the new certificate in the store.

Cannot Verify Identity Feed URL

If the Identity feed URL cannot be verified, and you are using SSL, make sure you followed the steps in Import the SSL Certificate.

If there are still issues, it is possible that the internal name of the certificate does not match the hostname of the log collector. The following procedure checks this.

1. SSH to the Security Analytics UI server.

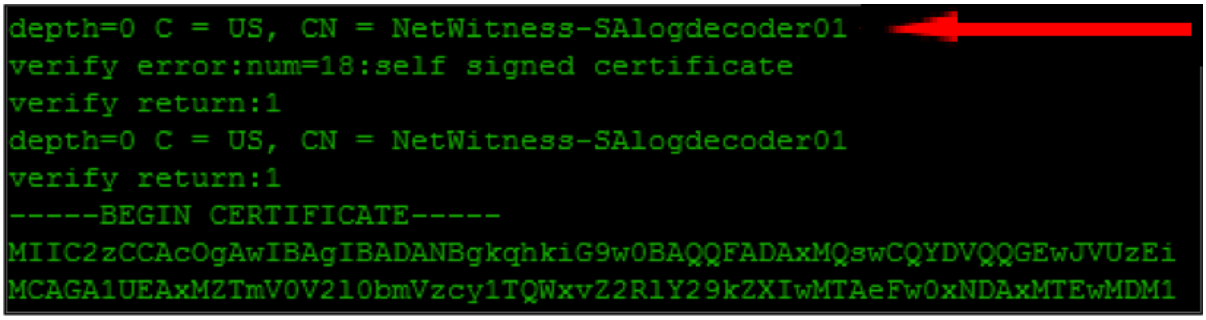
2. Run the following command to output the CN name of the SSL cert:

   ```
   echo -n | openssl s_client -connect <log decoder>:50101 | sed
   -ne '/BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p'
   ```

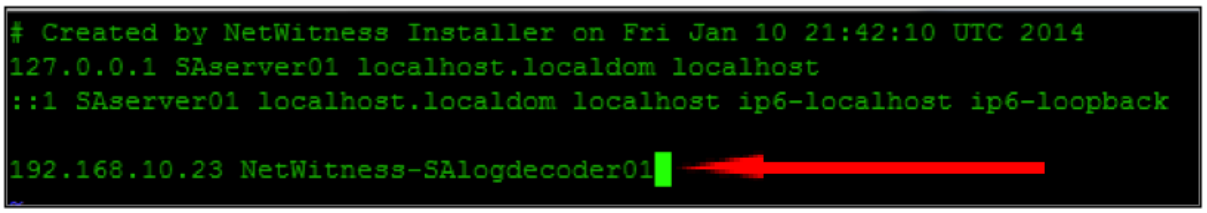
   Example:

   ```
   echo -n | openssl s_client -connect salogdecoder01:50101 |
   sed -ne '/BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p'
   ```

3. Retrieve the CN name of the SSL certificate.

4. Edit the **/etc/hosts** file and add the IP address and CN name to the file.



5. Restart the network service on the appliance.
6. Confirm that the name placed in the **/etc/hosts** file is used instead of the FQDN or IP address in the Identity feed URL.
7. Re-verify the Identity feed URL.

Edit a Custom Feed

This topic provides instructions for editing a custom feed using the Custom Feed Wizard.

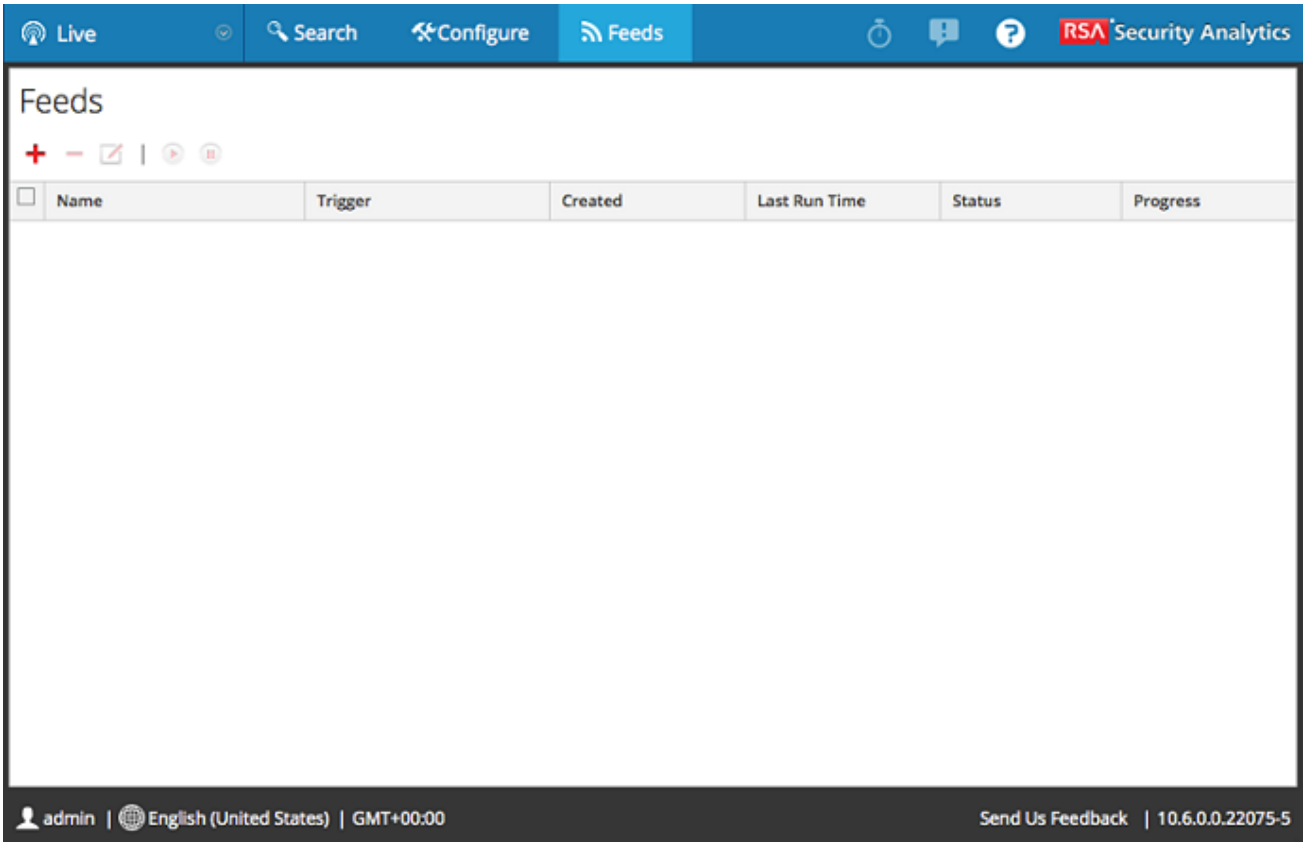
Completing this procure will result in:

- An existing custom feed opened.
- The feed (**.zip** format) or the file used to create the feed (**.csv** or **.xml**) downloaded and edited.
- The feed recreated with the updated file and new feed specifications.

To edit an existing feed:

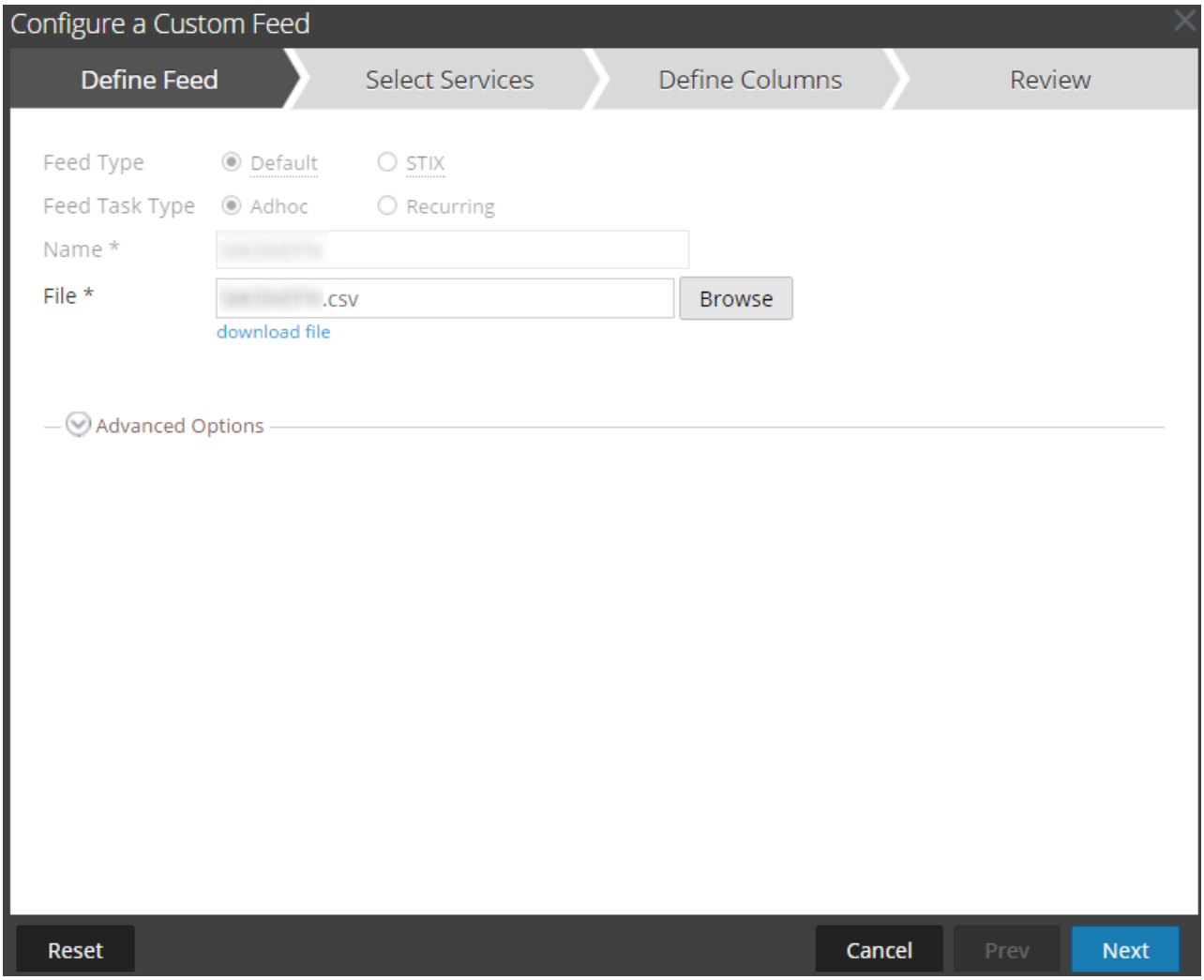
1. In the **Security Analytics menu**, select **Live > Feeds**.

   The Feeds view is displayed.



2. In the toolbar, select a feed and click  .

   The Configure Custom Feed or Configure Identity Feed panel opens in the Custom Feed wizard.



3. If you want to edit the feed file:
   a. Click **download file**.

      For an Identity feed, the .zip file is downloaded. For a custom feed, the .csv or .xml file is downloaded to your local file system.

      Note: For a STIX feed, the .xml file is downloaded to your local file system.

   b. Edit and save the file.

   c. In the **Define Feed** tab, browse for and open the edited file.
4. Edit any other parameters in the **Define Feed** tab, **Select Services** tab, and **Define Columns** tab that apply to the type of feed.
5. Anytime before you click **Finish**, you can:

- Click **Cancel** to close the wizard without saving your changes.
- Click **Reset** to clear the data in the wizard.
- Click **Next** to display the next form (if not viewing the last form).
- Click **Prev** to display the previous form (if not viewing the first form).

6. In the **Review** tab, review the feed information, and if correct, click **Finish**.

The feed is added to the feeds list and progress bar tracks completion. Upon successful creation of the feed definition file, the Create Feed wizard closes, and the feed and corresponding token file is listed in the Feed grid. You can expand or collapse the entry to see how many services are included, and which services are successful.

Use Custom Parsers

This topic provides instructions for using custom parsers in RSA Security Analytics.

RSA Security Analytics has the ability to upload parsers from your local system and delete these parsers.

Procedures

Upload Parsers to a Decoder or Log Decoder

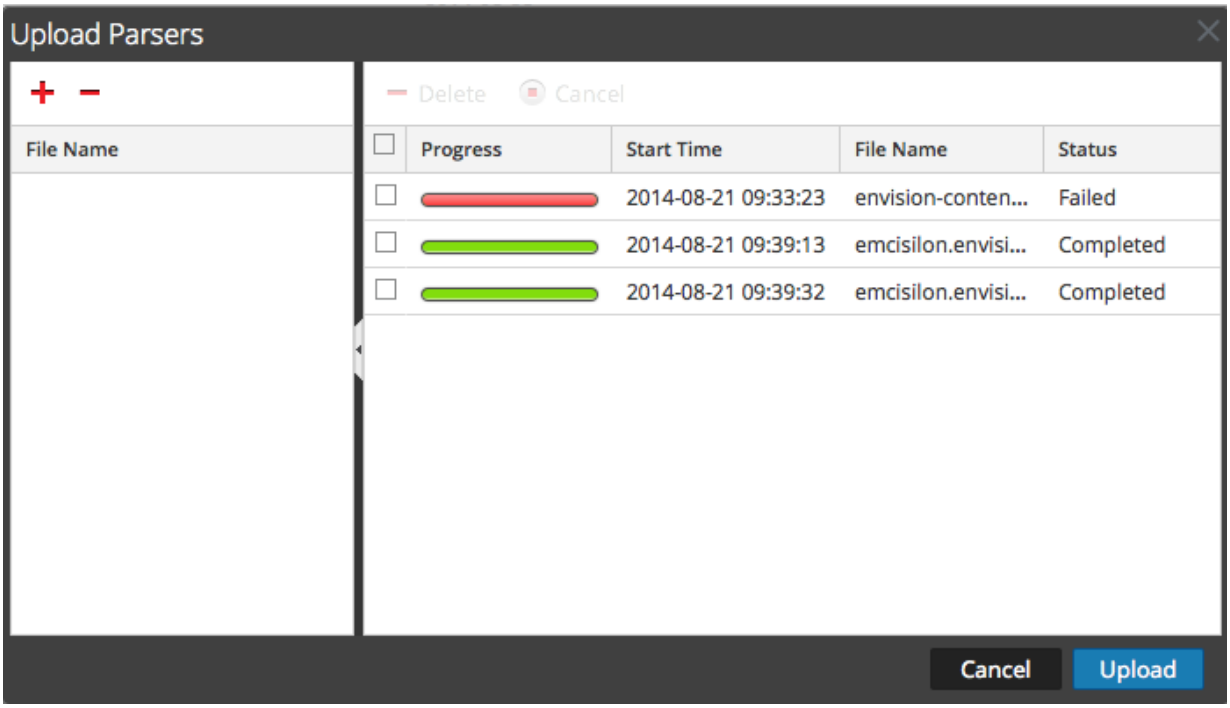
The Upload option in the Service Config view > Parsers tab displays the Upload Parsers dialog, in which you can manage the uploading of parsers to a Decoder or Log Decoder. In the File grid, you prepare a list of parsers for uploading. You can add files from a directory structure, and delete files from the grid if you decide that you don't want to upload a particular file. When the list is ready, clicking Upload starts the upload process.

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **> View > Config.**

   The Config view for the selected service is displayed.
3. Click the **Parsers** tab.

4. Click Upload.

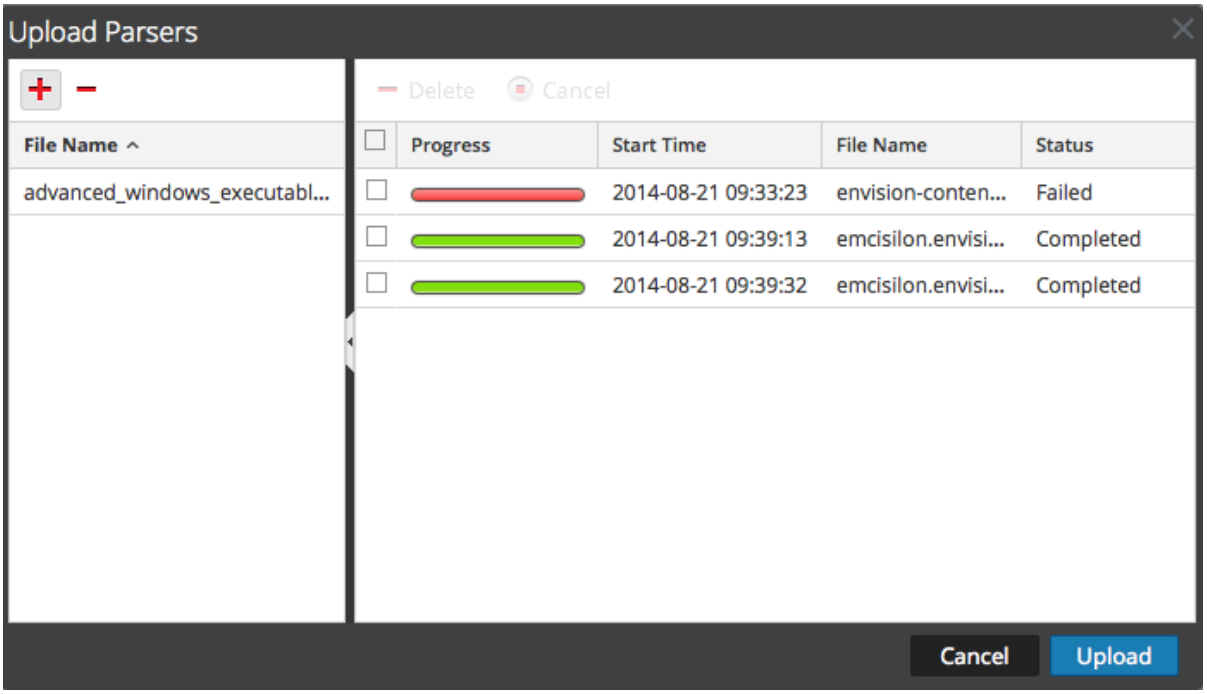
   The Upload Parsers dialog is displayed.



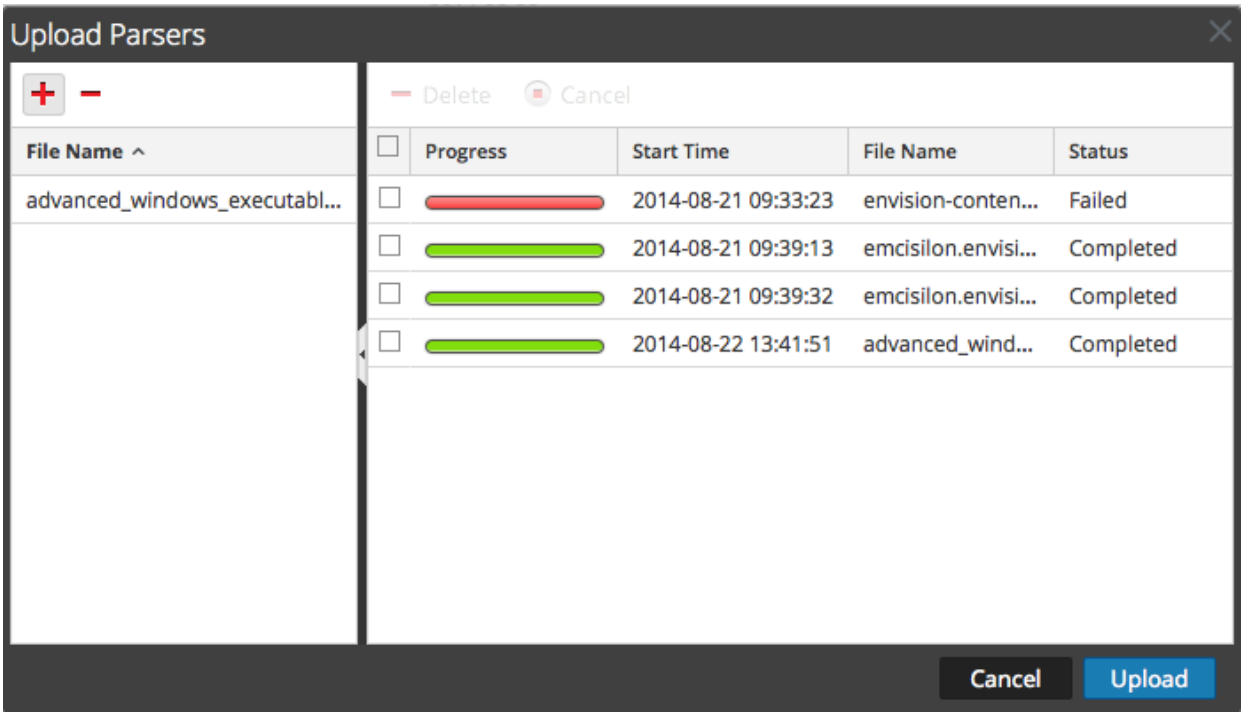
5. Click .

   A file selection dialog is displayed.

6. Select the **.flex**, **.parser**, and **.lua** files to be updated, and click **Open**.
   The dialog closes, and the selected files are displayed in the File grid.



7. Click **Upload**.
   The Upload Job grid shows the progress of the upload jobs with each job representing a file being uploaded.



8. Use any of the Upload grid tools to manage the upload of selected jobs: pause and resume, cancel, and delete.
   Once a job is complete, it is deployed on the Decoder and listed with the deployed parsers in Parsers tab.

Manage Upload Jobs

You can use any of the Upload grid tools to manage the upload of selected jobs: pause, resume, cancel, and delete.

- To cancel uploading a set of parsers while the upload is in queue or progress, click  Cancel.
- To pause uploading a set of parsers, if the upload is not yet complete, click  Pause.
- To resume uploading a set of parsers after a pause, click  Resume.
- To delete an upload job, click  Delete.

Delete Deployed Parsers

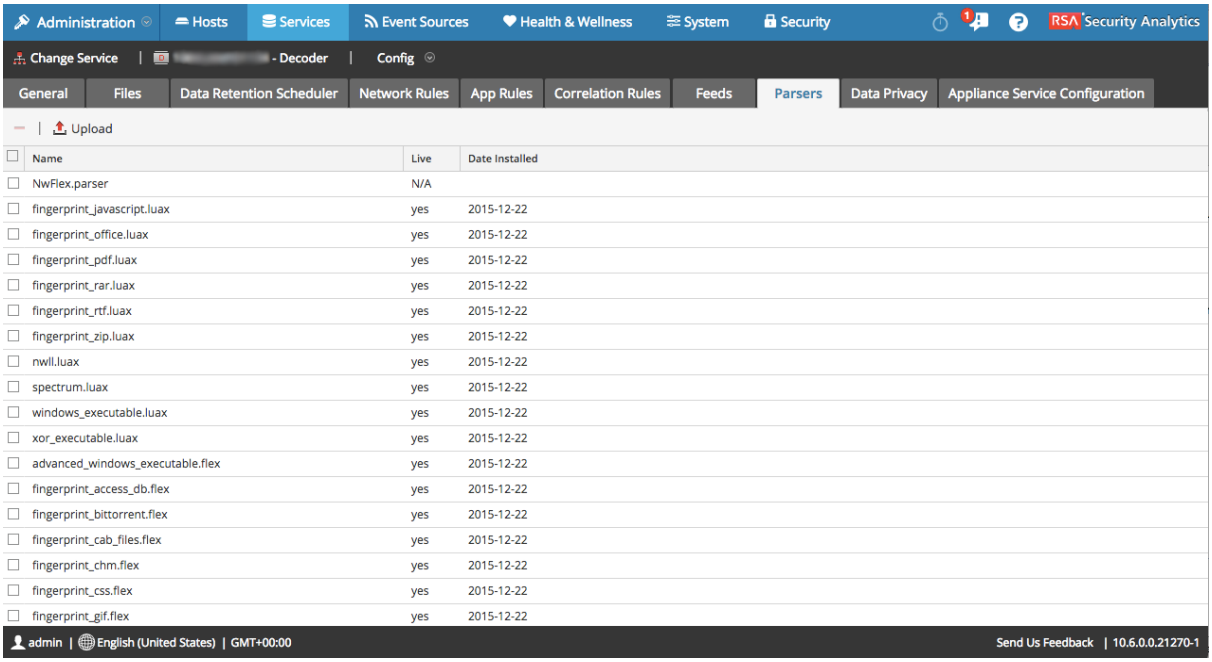
The **Delete** option in the Service Config view > Parsers tab provides a way to delete deployed parsers from a Decoder or Log Decoder. Parsers can be added and removed while a Decoder is running without affecting capture.

Note: Unless otherwise stated, any reference to Decoders applies to Log Decoders as well.

To delete a parser from a Decoder:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **> View > Config.**
   The Services Config view for the selected service is displayed.

3. Click the **Parsers** tab.



4. In the **Parsers** tab, select one or more parsers to delete.
5. Click .

   A dialog requests confirmation that you want to delete the parsers.
6. If you want to delete the parsers, click **Yes**.

   The parsers are removed from the Decoder immediately.

Configure 10G Capability

This topic guides administrators in how to tune a Packet Decoder specifically for high speed packet capture.

This guide applies when capturing packets on a 10G interface card. Packet capture at high speeds requires careful configuration and pushes the Decoder hardware to its limits, so please read this entire topic when implementing a 10G capture solution.

RSA Security Analytics Version 10.6.2 provides support for high-speed collection on the Decoder. You can capture network packet data from higher speed networks and optimize your Packet Decoder to capture network traffic up to 8Gb/sec sustained and 10Gb/sec burst, depending on which parsers and feeds you have enabled.

Note: You can skip to Configure 10G Decoder if you are starting with new Series 5 hardware.

Enhancements introduced to facilitate capture in these environments include the following:

• Utilization of **pf_ring** capture driver capability to leverage commodity 10G Intel NIC card for high speed capture.

• Introduction of assembler.parse.valve configuration. Configuration automatically disables application parsers when certain thresholds are exceeded to limit risk of packet loss. Once disabled, network layer parsers are still active. Once stats fall below exceeded thresholds, application parsers will automatically re-enable.

• Introduction of parallel.values configuration on the Concentrator for query optimizations.

Hardware Prerequisites

- Series 4S Decoder
- Intel 82599-based ethernet card, such as the Intel x520. All RSA-provided 10G cards meet this requirement.
- 96 GB of DD3-1600 memory in **dual-rank** DIMMs. Single rank DIMMs may decrease performance by as much as 10%. To determine the speed and rank of the installed DIMMs, run the command `dmidecode -t 17`.
- Sufficiently large and fast storage to meet the capture requirement. Storage considerations are covered later in this topic.

Software Prerequisites

- Linux kernel package obtained from RSA. Only Linux kernel packages provided by RSA are supported.
- The pfring package that matches the currently installed kernel. The kernel version must match the pfring version exactly.

10G Decoder Installation

Perform the following steps to install the Security Analytics 10.6.2 10G Decoder:

Prerequisites

• SA-S4H-P-DEC or SMC-S4H-P-DEC platforms built on the Dell R620 Platform

• SMC-10GE-* 10G Intel 520 NIC installed (available from RSA)

• Packet Decoders updated to 10.6.2

• Each Packet Decoder configured with a minimum of 2 DACs or SAN connectivity.

Note: Refer to Storage Considerations in this document prior to update, as physical re-cabling may be required.

• Dell R620 BIOS v1.2.6 or later. It is recommended that customers update to the latest v2.2.3 BIOS, but is not required for 10G if they are running v1.2.6 or later.

Note: BIOS revisions earlier than v1.2.6 have issues properly identifying the location of the 10G capture card within the system. It is important to update the BIOS before installing packages, as the packages use information provided by the BIOS to initialize the system.

BIOS Installation Instructions

1. Download BIOS v2.2.3 from the following location:

http://www.dell.com/support/home/us/en/19/Drivers/DriversDetails?driverId=V7P04

2. Download the Update Package for Red Hat Linux file.

3. Copy the file to the Security Analytics server.

4. Login as **root**.

5. Change the permissions on the file to execute.

6. Run the following file:

```
./BIOS_V7P04_LN_2.2.3.BIN
```

7. The system will request a reboot when complete.

Note: The BIOS installation procedure takes approximately 10 minutes.

Update 10G Decoder

1. Update the Decoder appliance to the version 10.6.2 release, including any and all OS patches. The minimum version of the security patch applied is RSA Security Analytics Version 10.6.2. This release requires Linux kernel package:

`kernel- 2.6.32-642.6.2`, which is the kernel release for RSA Security Analytics Version 10.6.2.

2. Ensure that the kernel, pfring, and numactl versions are as follows:

- `kernel- 2.6.32-573.12.1.el6.x86_64`
- `pf_ring 6.0.3-8598.2.6.32.642.6.2`
- `numactl-2.0.9-2.el6 .x86_64.rpm`

Install 10G Decoder

Download the latest version of the pfring rpm package from smcupdate

`pfring-6.0.3-8598.2.6.32.573.12.1.el6.x86_64.rpm`

For more information, refer to RSA SecurCare: https://knowledge.rsasecurity.com.

2. Via ssh, install the packages using the following command once the files are scp'ed to the Decoder:

```
rpm -ivh pfring*
```

Note: Be sure to perform the following checks:

a. Check for the el6 rpm using the following command:

```
rpm -qa |grep numactl*
```

b. Check to ensure the version is `numactl-2.0.9-2.el6 .x86_64.rpm`

Note: If the update step above is performed prior to upgrading the BIOS, the following steps need to be performed:

- Uninstall the packages via **rpm –e** command.
- Update BIOS to v2.2.3
- Run the **rpm** commands to install the necessary packages again.

3. Ensure that the kernel, pfring, and numactl versions are as follows:

- `kernel- 2.6.32-573.12.1.el6.x86_64`

- `pfring-6.0.3-8598.2.6.32.573.12.1.el6.x86_64.rpm`
- `numactl-2.0.9-2.el6 .x86_64.rpm`

4. Reboot the Decoder appliance (full system restart is required to ensure the **pf_ring** drivers load correctly).

5. Once the Decoder reboots, you can verify that the installation was successful if you see additional **PFRINGZC** interfaces available under the options for "Capture Interface Selected" (shown below).

Configure 10G Decoder

Once updated, perform the following steps to configure the 10G Decoder:

1. From the **Decoder Explorer** view, right click **Decoder** and select **Properties**.

2. In the **Properties** drop-down menu, select **reconfig** and enter the following parameters:

```
update=1 op=10g
```

3. From the **Decoder Explorer** view, right click database and select Properties.

4. In the **Properties** drop-down menu, select **reconfig** and enter the following parameters shown in the following screen capture:

```
update=1 op=10g
```

5. Select the capture port adapter. Options for this include:

a. Single port capture - **PFRINGZC,p1p1 or PFRINGZC,p1p2**

b. Capture off both ports –

i. Select **PFRINGZC,P1P1**

ii. In Explorer view, `set capture.device.params = device=zc:p1p2,zc:p1p1`

c. Ensure that the selected capture hardware is on the correct NUMA node.

From an ssh session to the appliance, execute the following statement:

```
cat /sys/class/net/<interface_name>/device/numa_node
```

where `<interface_name>` is the selected capture interface (for example, `p1p1`).

If the result is 0 (zero), no additional configuration is necessary.

If not, add the result as the parameter core to the capture parameters, as shown below:

```
/decoder/config/capture.device.params: core=1
```

This change requires a service restart to take effect.

Note: Based on hardware configuration, the capture ports may be identified with a different name other than **p1p1/p1p2** but will always have the prefix **PFRINGZC**. For example, on some appliances these ports may be identified as **`eth4 / eth5`**. To capture from eth4, select **`PFRINGZC,eth4`**. To capture from **eth5**, select **PFRINGZC,eth5**.

6. If the write thread is having trouble sustaining the capture speed, you can try the following:

Change **/datebase/config/packet.integrity.flush** to normal.

Note: You can try adjusting the **packet.file.size** to something higher, but keep the file size under 10 GB, as the whole file is buffered in memory at these speeds.

7. (Optional) Application parsing is extremely CPU intensive and can cause the Decoder to drop packets. To mitigate application parsing induced drops, the setting **/decoder/config/assembler.parse.valve** can be set to **true**. This will result in the following:

• When session parsing becomes a bottleneck, application parsers (HTTP, SMTP, FTP, etc.) will be temporarily disabled.

• Sessions are not dropped when the application parsers are disabled, just the fidelity of the parsing performed on those sessions.

• Sessions parsed when the application parsers are disabled will still have associated network meta (NETWORK parser).

• The statistic **/decoder/parsers/stats/blowoff.count** displays the count of all sessions that bypassed application parsers (network parsing is still performed).

• When session parsing is no longer a potential bottleneck, the application parsers are automatically re-enabled.

8. The assembler session pool should be large enough that it is not forcing sessions.

• To determine if sessions are being forced by the statistic **/decoder/stats/assembler.sessions.forced** (it will be increasing) and **/decoder/stats/assembler.sessions** will be within several hundred of **/decoder/config/assembler.session.pool**.

• RSA Security's test site used the following configuration at just under 10G:

/decoder/config/assembler.session.pool was set to **1000000**

and **/decoder/stats/assembler.sessions** would average **630K**.

An alternative method for Steps 1 through 4 listed above can be used to configure the 10G Decoder by performing Steps 1, 2, 3, and 4 explained below. Steps 5 through 8 listed above are required if you are using this method.

1. Update session and packet pool settings to the following values (under **/decoder/config**):

a. **pool.packet.pages = 1000000**

b. **pool.session.pages = 300000**

2. Packet write block size under (**/database/config/packet.write.block size**) must be set to exactly 4 GB, or for Version 10.6+, use **filesize**.

Note: This configures the Decoder to buffer the file with huge pages and write using direct I/O for maximum performance.

3. Update parse thread settings to the following values (under **/decoder/config**).

a. **parse.threads =12**

4. Select the capture port adapter. Options for this include:

a. Single port capture - **PFRINGZC,p1p1 or PFRINGZC,p1p2**

b. Capture off both ports –

i. Select **PFRINGZC,P1P1**

ii. In Explorer view, **set capture.device.params = device=zc:p1p2,zc:p1p1**

Note: Based on hardware configuration, the capture ports may be identified with a different name other than **p1p1/p1p2** but will always have the prefix **PFRINGZC**. For example, on some appliances these ports may be identified as **eth4 / eth5**. To capture from **eth4**, select **PFRINGZC,eth4**. To capture from **eth5**, select **PFRINGZC,eth5**.

Storage Considerations

When capturing at higher speed rates, the storage system holding the packet and meta databases must be capable of the necessary throughput for reads and writes to disk. Supported options for DAC and SAN configurations are outlined below.

Using the Series 4S Hardware (With Two or More DAC Units)

The Decoder head unit is equipped with a hardware-RAID SAS controller card providing connectivity to the DAC. In most deployments these are configured such that the DACs are daisy-chained off a single port on the SAS card. To support higher speed environments, a minimum of two DACs are required per Decoder and must each be connected directly to the SAS card. To accommodate two DACs, connect the first DAC to one port on the SAS card, and then connect another DAC to the other port on the SAS card. For environments with more than two DACs, chain them off each port in a balanced manner. This may require a re-cabling of DACs in an existing deployment, but should not affect data that has already been captured on the Decoder.

If adding new capacity, use the currently available NwMakeArray script to provision the DAC units. The script automatically adds one DAC per execution (i.e., if adding three DACs, then the script must be run three times), adding them to NwDecoder10G's configuration as separate mount points. The independent mount points are important, as it allows the NwDecoder10G to segregate write I/O from capture from the read I/O needed to satisfy packet content requests.

Using SAN Storage

The Decoder will allow any storage configuration that can meet the sustained throughput requirement. Note that the standard 8Gbit FC link to a SAN is not sufficient to read and write packet data at 10G, thus environments utilizing SAN are required to configure connectivity to the SAN utilizing multiple FCs.

Parsing and Content Consideration for Packet Capture

Capturing and performing enrichment against raw packets can present unique challenges at any capture rate. With higher session and packet rates in 10G, parsing efficiency is paramount. A single parser can have a detrimental effect on the system, ultimately resulting in packet drops. Testing performed for 10G capture included baseline parsers as well as combinations of feeds, rules, and other content accessible via RSA Live. Whether a customer is updating a currently deployed system or deploying a new system, it is recommended they utilize the following best practices to minimize risk for packet loss. One caveat is if you are updating a current 10G deployment but not adding any additional traffic. For example, a current Decoder capturing off a 10G card at 2G sustained should see no difference in performance, unless part of the update also entails adding additional traffic for capture.

10G Best Practices

1. Incorporate baseline parsers (except SMB/Webmail, both of which generally have high CPU utilization) and monitor to ensure little to no packet loss.

2. When adding additional parsers, add only one or two parsers at a time.

3. Measure performance impact of newly added content, especially during peak traffic periods.

• If drops start occurring when they did not happen before, disable all newly-added parsers and enable just one at a time and measure the impact. This helps pinpoint individual parsers causing detrimental effects on performance. It may be possible to refactor it to perform better or reduce its feature set to just what is necessary for the customer use case.

• Although lesser performance impacts, feeds should also be reviewed and added in a phased approach to help measure performance impacts.

• Application Rules also tend to have little observable impact, though again, it is best not to add a large number of rules at once without measuring the performance impact.

Finally, making the recommended configuration changes outlined in the Configuration section will help minimize potential issues.

Aggregation on a 10G Decoder to Other Security Analytics Components

With the initial release, aggregation from the Packet Decoder to a Concentrator is supported. Deployments utilizing Malware Analytics, Event Stream Analysis, Warehouse Connector, and Reporting Engine are expected to impact performance and can lead to packet loss. Due to the high volume of session rates, the following configuration changes are recommended:

• Nice aggregation on the Concentrator limited the performance impact on the 10G Decoder

/concentrator/config/aggregate.nice = true

• Due to the high volume of sessions on the concentrator, you may consider activating "parallel values" mode on the concentrator by setting **/sdk/config/parallel.values** to true. This will improve investigation performance when the number of sessions per second is above 30,000.

• Further review for content and parsing will be required for deployments where utilization of other SA Components are desired (i.e., Warehouse, Malware Analysis, ESA, and Reporting Engine).

Decoder

Storage Considerations

When capturing at 10G line rates, the storage system holding the packet and meta databases must be capable of sustained write throughput of 1400 MBytes/s.

There are several ways to achieve such high sustained throughput. Here we describe one such possible solution, though other storage architectures are possible.

Using the Series 4S hardware, with two DAC units

The Series 4S is equipped with a hardware RAID SAS controller capable of an aggregate 48Gbit/s of I/O throughput. It is equipped with 8 external 6 Gbit ports, organized into two 4-lane SAS cables. The recommended configuration for 10G is to balance at least 2 DAC units across these two external connectors. For example, connect 1 DAC to one port on SAS card, and then connect another DAC to the other port on the SAS card. As you add more DACs, chain them off of each port in a balanced manner.

As you add capacity, use the NwMakeArray script to provision the DAC units. This will automatically add them to NwDecoder10G's configuration as separate mount points. The independent mount points are important as it allows the NwDecoder10G to segregate write I/O from capture from the read I/O needed to satisfy packet content requests.

Other Storage Configurations (SAN, etc.)

The Decoder will allow any storage configuration that can meet the sustained throughput requirement. Note that the standard 8Gbit FC link to a SAN is not sufficient to store packet data at 10G, thus in order to use a SAN it may be required to perform aggregation across multiple targets using a software-RAID Scheme.

Parsing at High Speeds

Obviously, parsing raw packets at high speeds presents unique challenges. Given the high session and packet rates, parsing efficiency is paramount. A single parser that is inefficient (spends too long examining packets) can slow the whole system down to the point where packets are dropped at the card. For initial 10G testing, start with only native parsers (except SMB/WebMail). Use the native parsers to establish baseline performance and with little to no packet drops. Do not download any Live content until this has been done and the system is proven to capture without issue at high speeds.

After the system has been operational and running smoothly, Live content should be added very slowly - especially parsers. Parsers can have a dramatic effect on performance. Here are some rules of thumb:

Tested Live Content

The following parsers can all (not each) be run at 10G on our test data set:

- MA content (7 Lua parsers,1 feed, 1 application rule)
- 4 feeds (alert ids info, nwmalwaredomains, warning and suspicious)
- 41 application rules
- DNS_verbose_lua (disable DNS)
- fingerprint_javascript_lua
- fingerprint_pdf_lua
- fingerprint_rar_lua
- fingerprint_rtf_lua
- MAIL_lua (disable MAIL)
- SNMP_lua (disable SNMP)
- spectrum_lua
- SSH_lua (disable SSH)
- TLS_lua
- windows_command_shell
- windows_executable

NOT TESTED:

- SMB_lua, native SMB disabled by default
- html_threat

OTHER:

HTTP_lua reduces capture rate from >9G to <7G. At just under 5G, this parser can be used in place of the native without dropping (in addition to the list above). xor_executable will push parse CPU to 100% and the system can drop significantly at any time due to parse backup.

Aggregation on a 10G Decoder

A 10G Decoder can serve aggregation to a single concentrator while running at 10G speeds.

1. Concentrator aggregates between 45-70k sessions/sec
2. The 10G Decoder is capturing between 40-50k sessions/sec.
   With content identified above, this is about 1.5 to 2 million meta/sec.
3. Turn on nice aggregation on the Concentrator to limit the performance impact on the Decoder.
   /concentrator/config/aggregate.nice=true
4. Due to the high volume of sessions on the concentrator, you may consider activating **parallelvalues**mode on the concentrator by setting **/sdk/config/parallel.values** to **true**. This will improve investigation performance when the number of sessions per second is above 30k.

If multiple aggregation streams are necessary, it would be less impactful on the Decoder to aggregate from the Concentrator instead.

Configure Syslog Forwarding to Destination

This topic provides instructions for forwarding collected Syslog messages from a Log Decoder to another Syslog receiver.

In addition to collecting Syslog messages, you can configure the Log Decoder to forward Syslog messages to another Syslog receiver. Security Analytics forwards Syslog messages after it has parsed the messages and before it writes the messages to the Log Decoder.

Note: You must configure Syslog Forwarding using the steps defined in this topic under **Procedure** using the **Explore** view.

Prerequisites

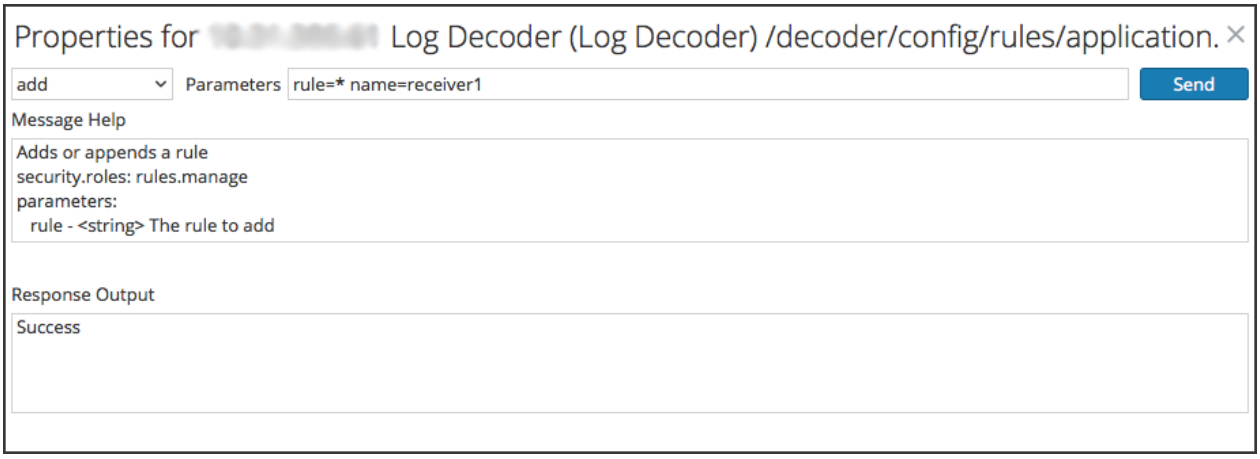
The Log Decoder must be in the **Started** state.

Procedure

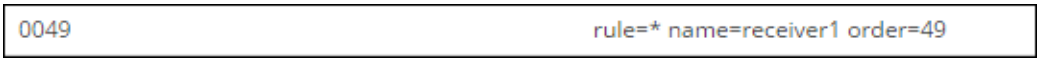
To configure Syslog Forwarding:

1. Configure Log Decoder application layer rules (Application rules) to tag Syslog messages with meta that instructs Security Analytics to forward the messages:
   a. In the **Services** view, select a Log Decoder, and in the Actions column, select  **> View > Explore**.
   b. Go to the **/decoder/config/rules/application** node, right-click **application,** and click **Properties**.
   c. In the **Properties** view, specify the **add** command with the following parameters: **rule=<query> name=<name>** (Example 1, **rule=*name=receiver1,** Example 2, **rule="device.type='winevent_nic'" name=receiver1**)

d. Click **Send**.



Security Analytics creates the **name=receiver1 rule=* order=<n>** rule. Security Analytics inserts the order number (for example, **order=49**) based on when you set up the rule.



e. Go to the **/decoder/config/rules/application** node and click the **name=receiver1 rule=* order=49** rule.

f. Add **alert forward** parameters to the rule's parameters.

All other rule parameters have the same meaning as they do in other application rules.

The following Application rule example selects all logs with the * rule. It creates an alert meta with the value "**receiver1**" and tags the entire log for forwarding it to the syslog forwarding destination. You can define as many different forwarding rules as you need with the same name or unique names.

2. Define Syslog forwarding destinations and enable forwarding.

a. In the **Services** view, select a Log Decoder, and in the **Actions** column, select  **> View > Explore**.

b. In the **/decoder/config/logs.forwarding.destination** parameter, specify the destination. For example:
TLS Connections: **receiver1=tls:receiver1.netwitness.local:6514**
UDP Connections: **receiver1=udp:receiver1.netwitness.local:514**
TCP Connections: **receiver1=tcp:receiver1.netwitness.local:514**

logs.forwarding.destination	receiver1=tcp:10.31.244.44:514 receiver2=tcp:10.31.244.46:514 receiver3=tcp:10.31.244.48:514

Note:
You can configure:
- Multiple rules to forward logs to the same destination.
- Multiple rules to forward logs to multiple destination.

For TLS connections, the certificate of the forwarding destination must be validated. The certificate authority that signed the destination's certificate must be present in the Log Decoder's CA trust store and the certificate must reside on the destination or Syslog receiver. Refer to the **Configure Certificates** topic in the *Log Collection Configuration Guide* for information about manipulating the Log Decoder's CA trust store.

c. In the **/decoder/config/logs.forwarding.enabled** parameter, specify **true**.

logs.forwarding.enabled	true

Related Topic

- Configure Application Rules

Create Custom Meta Keys Using Custom Feed

This topic provides information on how to add custom meta keys, using custom feed in the Log Decoder.

You can create custom meta keys to retrieve data, to investigate and analyze the logs and packets. Custom meta keys enable you to add an enrichment context for the log and packet data. This document highlights the configuration changes to reflect the custom meta keys in the Concentrator, ESA, Archiver, Warehouse Connector, and Reporting Engine schema.

Here is a example of creating the custom meta key in the Log Decoder. In this scenario, an organization wants to track the location of an asset such as a printer. So, a custom meta key **source location** is introduced which indicates the location of the asset, for example the Printer1, which is located in the 'Fifth Floor A wing'.

Note: Custom meta keys can be created in Decoder as well. Make sure to select the index.decoder.xml file when you create a custom meta in the Decoder.

Procedure

Add custom meta key in Log Decoder

To add custom meta keys using custom feed:

1. In the **Security Analytics** menu, select **Administration** > **Services** > **Log Decoder.**
2. Select a service and click  > **View** > **Config** > **Files** tab > **index-logdecoder-custom.xml.**

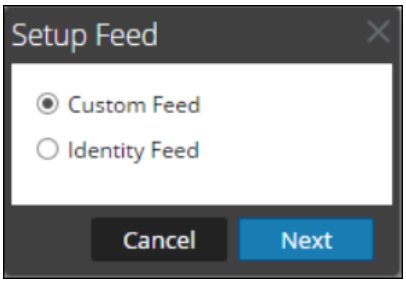
```
<Language>
  <?xml version="1.0" encoding="utf-8"?>
  <Language level="IndexNone" defaultAction="Auto">
  <!-- Reserved Meta key for Feed -->
  <Key description="Source Location" level="IndexNone"
name="location.src" format="Text"/>
 </Language>
```

3. Restart the Log Decoder service. In the Services view, click  > **Restart**.

Deploy feed in Live

To deploy the feed in the live environment:

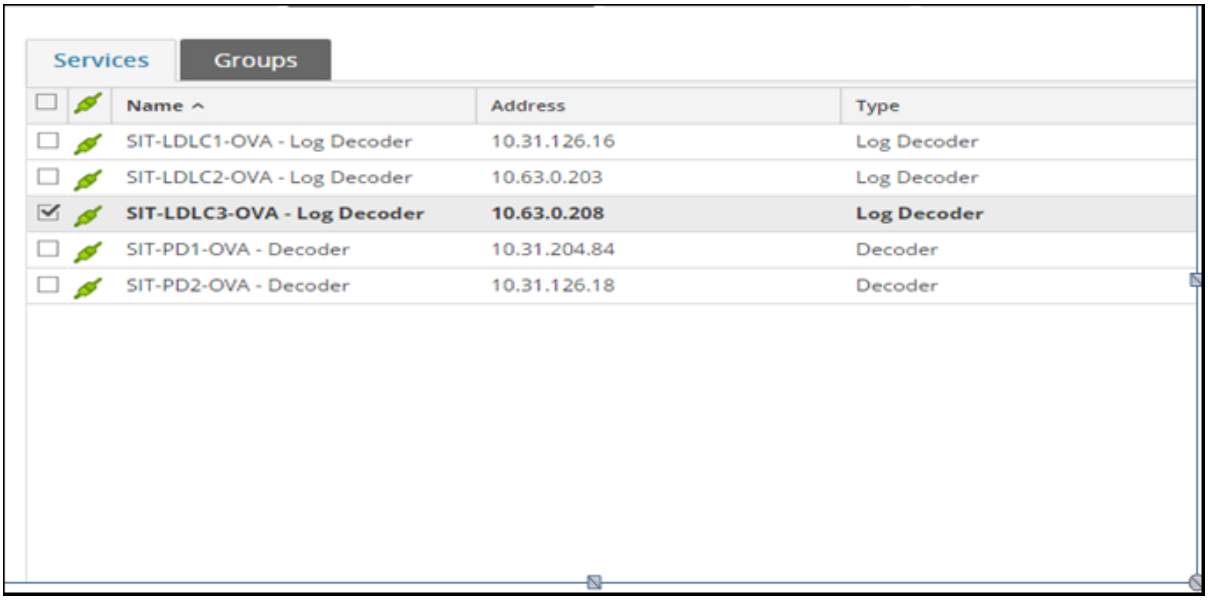
1. In the **Security Analytics** menu, select **Live** > **Feed.**
2. In the toolbar, click  .
   The Setup Feed dialog is displayed.



   To select the feed type, click **Custom Feed** and **Next**.
   The Configure a Custom Feed wizard is displayed, with the Define Feed form open.
   Enter the name and upload the Feed CSV file.

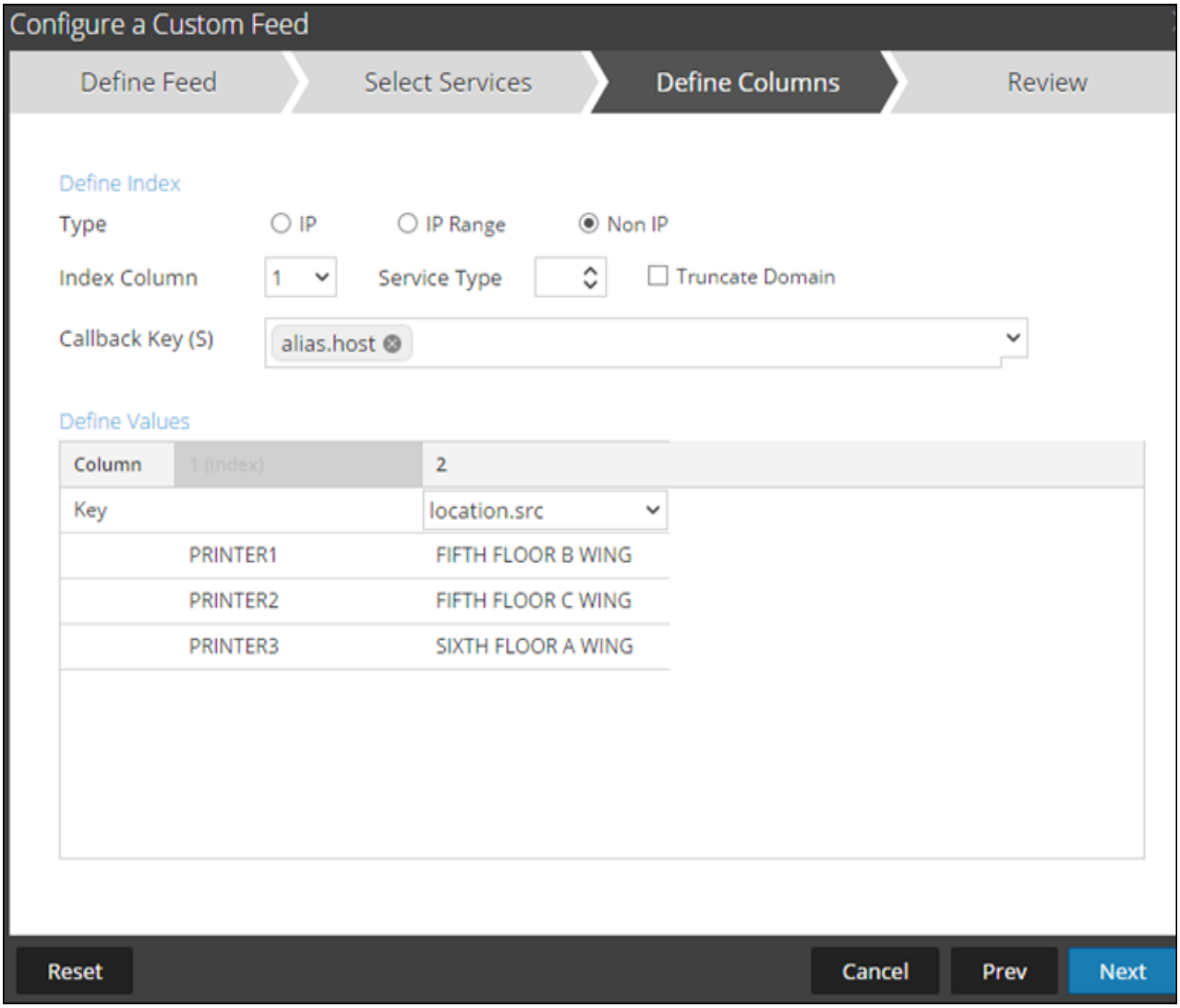
> **Note:** For a STIX feed you must upload the `.xml` file.

3. Click **Next**.
4. Select the **Log Decoder** service, where the feed needs to be uploaded.



5. In the Define Index section, select the index type, index column, and callback key. In the Define Values section, enter the custom meta key.
   The contents of the .csv file are displayed in the feed wizard. In this case, the first column displays the asset hostname and the second column indicates the asset location.

Note: The Source IP should be indexed by selecting the type as 'IP' as the ip.src. and ip.dst are in IPv4 format.



In this scenario, a custom meta key location.src (location source) is added by indexing the hostname (alias.host). In this example, the printer hostname are populated in meta key 'alias.host'. So, select 'alias.host' as callback key, and index type as 'Non IP' in the Feed Wizard as shown below. In the Define Values section, select the custom meta key from the drop down menu.

7. Click **Next**.
8. Click **Done**.

For more information on the feed wizard, see Create and Deploy Custom Feeds Using a Wizard.

Add the custom meta entry in Concentrator index file

To add the custom meta entry in the concentrator index file:

1. In the **Security Analytics** menu, select **Administration** > **Services** > **Concentrator**.
2. Click > **View** > **Config** > **Files** tab > **index-concentrator-custom.xml.**
3. Add the custom meta entry in the Concentrator index file.

```
<Language>
 <?xml version="1.0" encoding="utf-8"?>
 <Language level="IndexNone" defaultAction="Auto">
  <!-- Reserved Meta key for Feed -->
  <Key description="Source Location"  level="IndexValues"
name="location.src" format="Text"  valueMax="10000"
defaultAction="Open"/>
 </Language>
```

4. Restart the Concentrator services. In the Services view, click > **Restart**.

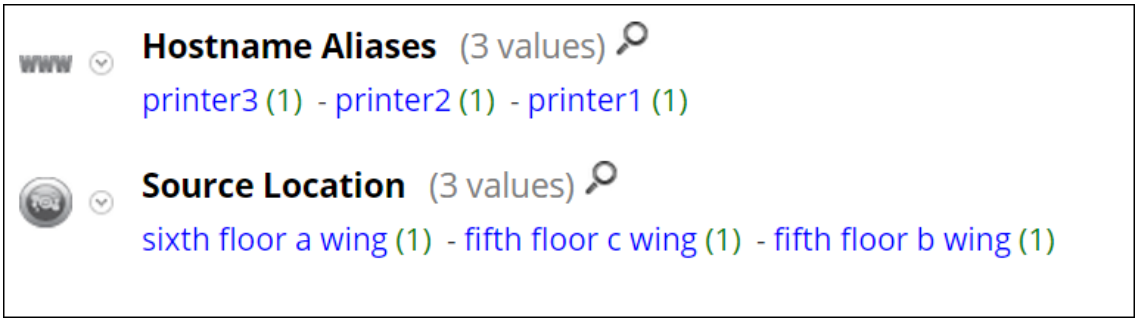
Note: In case of the Broker, the Broker derives its index from the Concentrator from which it aggregates. So you do not need to create custom meta in the broker. If you have not indexed the meta key in the concentrator, the broker will not display in the investigation.

Investigate

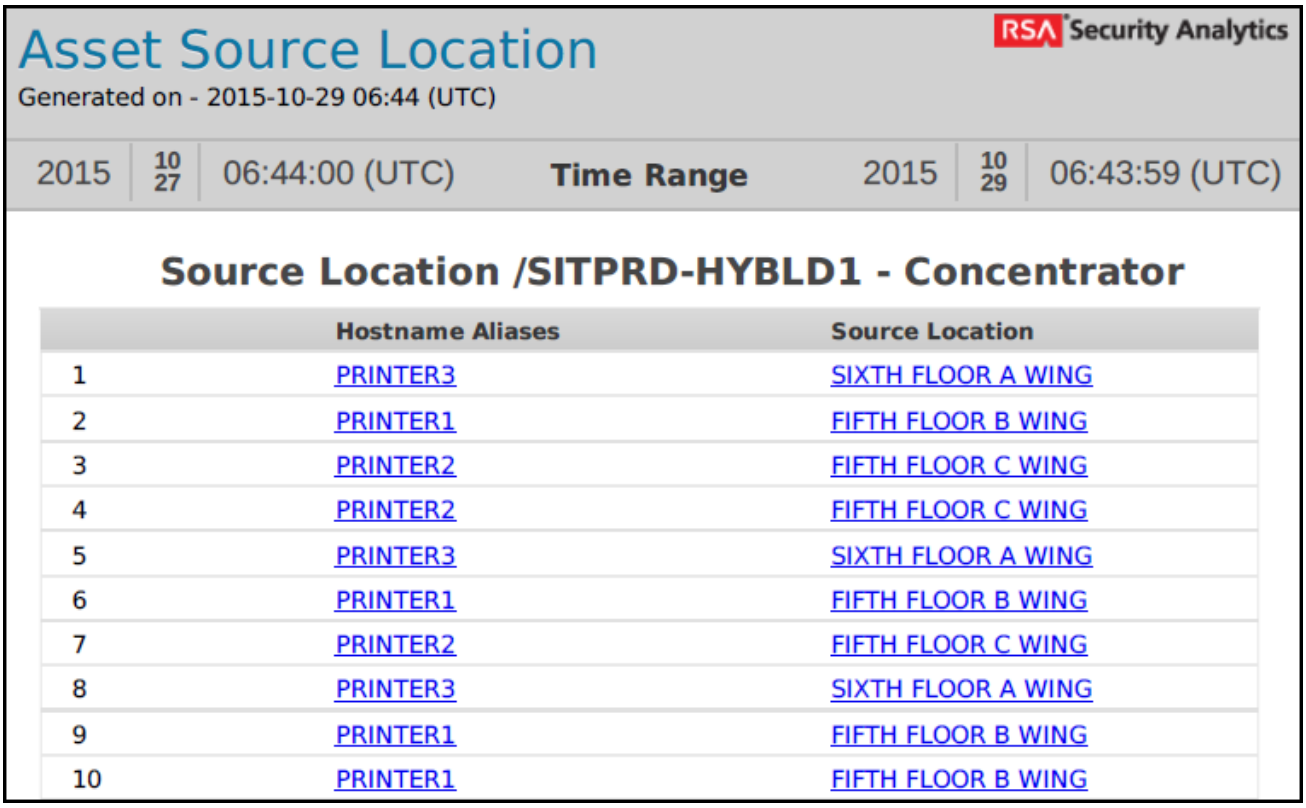
Note: Make sure that you logout and login from the Security Analytics User Interface, before you can view the custom meta key in Investigation.

To investigate on the custom meta key:

1. In the **Security Analytics** menu, select **Investigation** > **Navigate**.
2. Select a Concentrator service.
3. Click **Navigate**.



Here is an example of a report executed on the concentrator.



Asset Source Location

RSA Security Analytics

Generated on - 2015-10-29 06:44 (UTC)

2015 | 10 27 | 06:44:00 (UTC) | Time Range | 2015 | 10 29 | 06:43:59 (UTC)

Source Location /SITPRD-HYBLD1 - Concentrator

	Hostname Aliases	Source Location
1	PRINTER3	SIXTH FLOOR A WING
2	PRINTER1	FIFTH FLOOR B WING
3	PRINTER2	FIFTH FLOOR C WING
4	PRINTER2	FIFTH FLOOR C WING
5	PRINTER3	SIXTH FLOOR A WING
6	PRINTER1	FIFTH FLOOR B WING
7	PRINTER2	FIFTH FLOOR C WING
8	PRINTER3	SIXTH FLOOR A WING
9	PRINTER1	FIFTH FLOOR B WING
10	PRINTER1	FIFTH FLOOR B WING

Additional Procedures

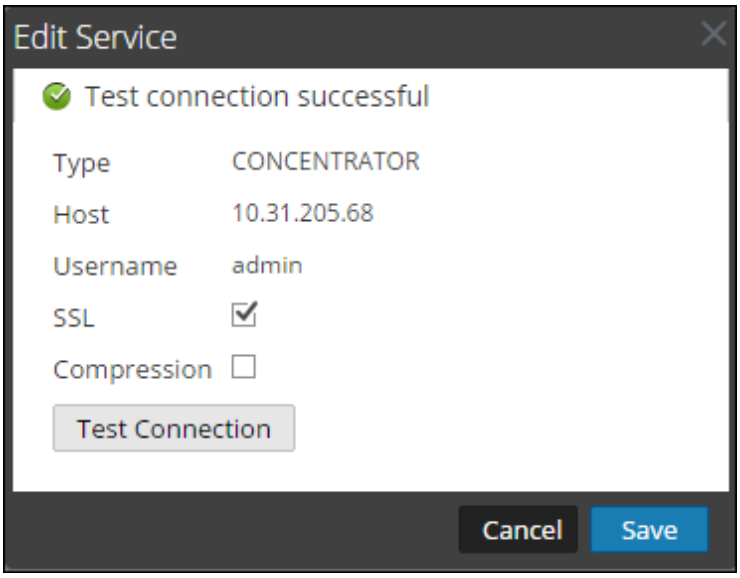
The following procedures must be executed if you have Warehouse Connector, Archiver, Reporting Engine and ESA configured.

Update the Schema in ESA

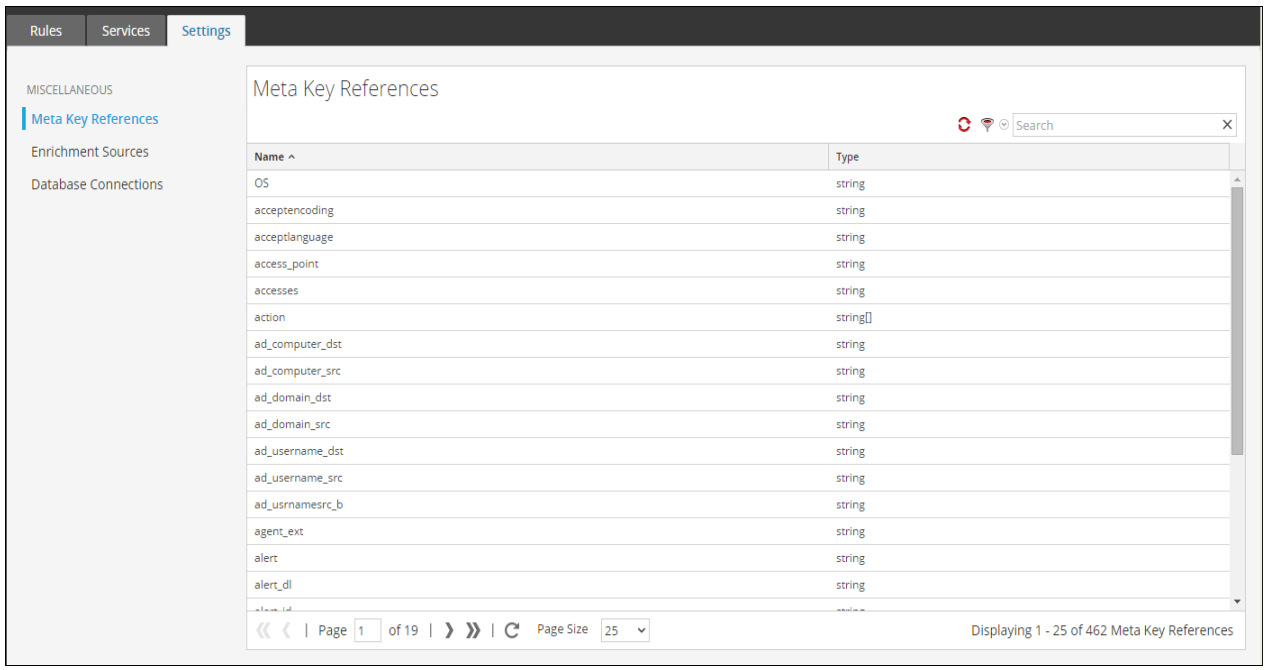
Before you update the schema in ESA, the custom meta key should be indexed in the concentrator.

To update the schema ESA rules and to be able to use the new custom meta keys:

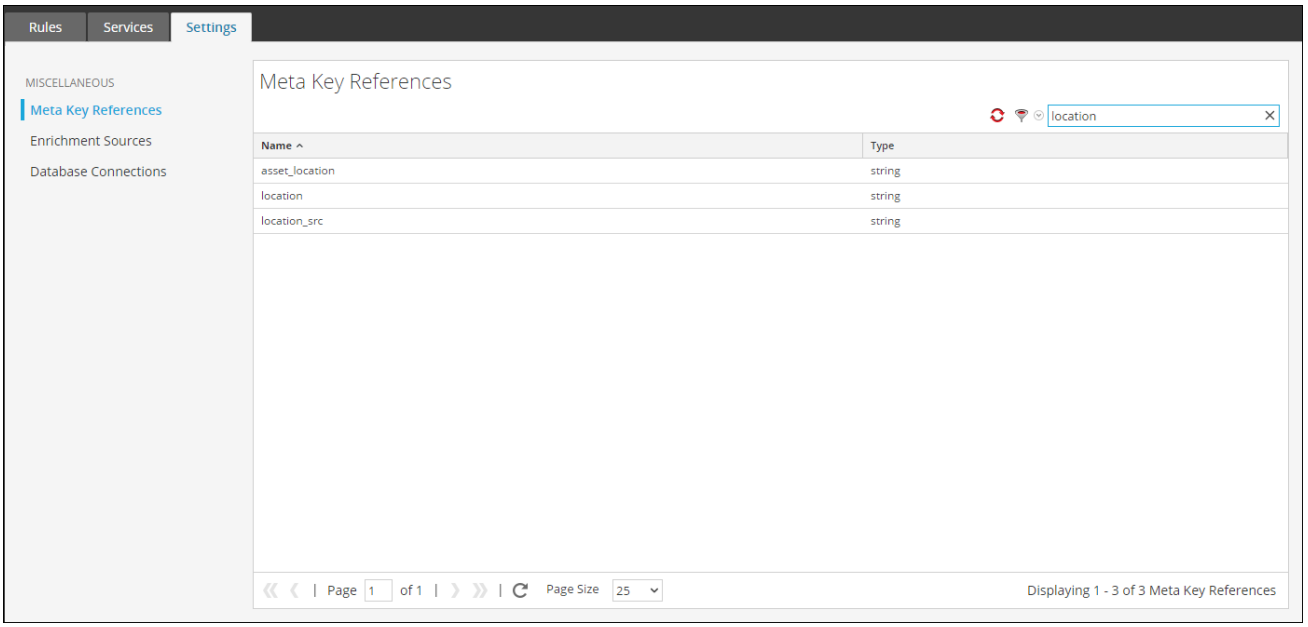
1. In the **Security Analytics** menu, select **Administration** > **Services** > **ESA- Event Stream Analysis** > **View** > **Config**.
2. Edit the Concentrator Datasource.
3. Click **Test Connection.**



4. Click **Save** after the connection is successful.
5. Click **Apply**.
6. Navigate to **Alerts** > **Configure** > **Settings**.



7. Click the **Search** tab and search for the name of the custom meta key.
   The custom meta key name and type is displayed.



Update the Schema in Archiver

If you want to configure the Security Analytics Archiver, using the new custom meta keys, you need to update the Archiver schema in the Reporting Engine.

To update the Archiver schema in Reporting Engine:

1. In the **Security Analytics** menu, select **Administration** > **Services** > **Archiver**.
2. Click on > **View** > **Config** > **Files** > **index-archiver-custom.xml**.
3. Add the custom meta entry in the Archiver index file.

```
<Language>
  <?xml version="1.0" encoding="utf-8"?>
  <Language level="IndexNone" defaultAction="Auto">
  <!-- Reserved Meta key for Feed -->
  <Key description="Source Location" level="IndexValues"
name="location.src" format="Text"
  valueMax="10000" defaultAction="Open"/>
 </Language>
```

4. Restart the Archiver service. Click on > **Restart.**

   The Archiver schema gets updated with the custom meta key.

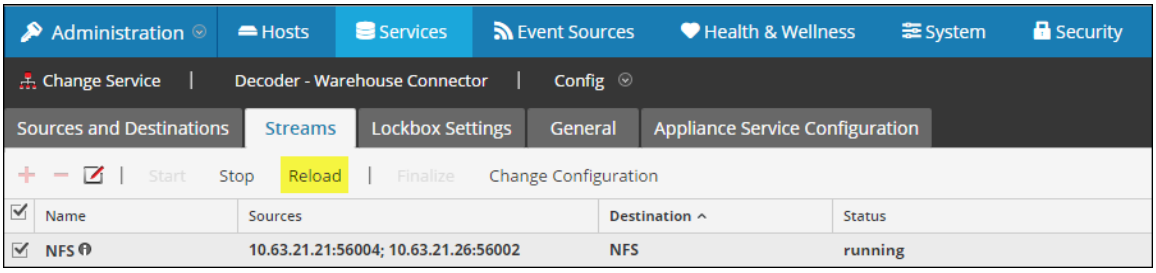
Update the Schema in Warehouse Connector

If you want to configure the Security Analytics Warehouse with custom meta and use it in warehouse report then you need to update the Warehouse schema in the Reporting Engine.

If the Log Decoder or Decoder, where the custom meta key is added, is one of the sources in the Warehouse Connector stream, you need to update the schema in the Warehouse Connector.

To update the Warehouse schema in the Reporting Engine:

1. In the **Security Analytics** menu, select **Administration** > **Services** > **Warehouse Connector**.
2. Click on > **View** > **Config** > **Files** tab > **index-logdecoder-custom.xml.**
3. Select the stream and click **Reload**.
   The warehouse connector pulls the schema from the downstream devices (log decoder/decoder).



For more information on streams, see the **Configure Streams** topic in the *Warehouse Connector Configuration Guide*.

Update the Schema in Reporting Engine

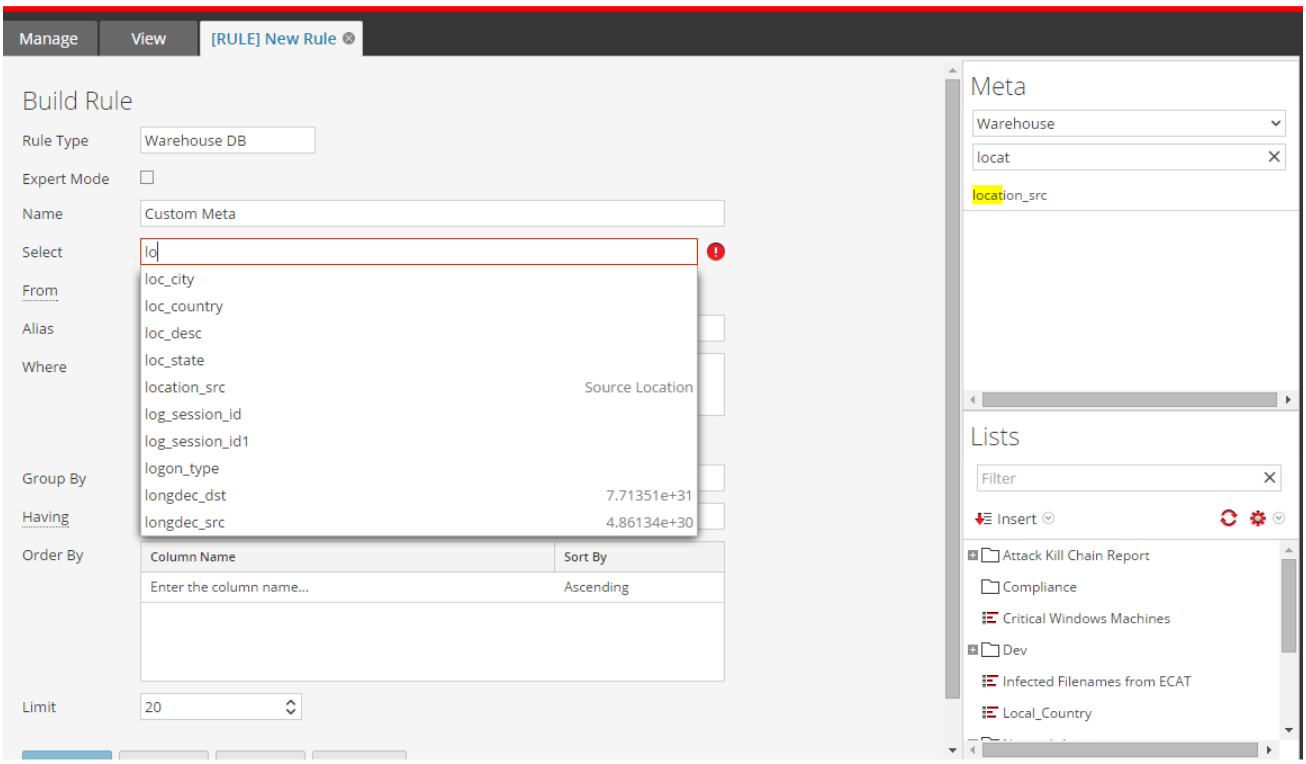
To update the schema in Reporting Engine:

1. In the **Security Analytics** menu, select **Administration**> **Services** > **Reporting Engine.**
2. Click on > **Restart.**

Note: Restart the Reporting Engine or wait for thirty minutes for the schema to be updated.

To view the custom meta key:

1. Navigate to **Reports** > **Rules**.
2. In the toolbar, click .
3. Select Warehouse DB.
4. In the Build Rule page, search for the custom meta from the right panel of the page.
   The custom meta key is displayed.


Manage
View
[RULE] New Rule
Build Rule
Rule Type
Warehouse DB
Expert Mode
Name
Custom Meta
Select
lo
From
Alias
Where
Group By
Having
Order By
Limit
20
loc_city
loc_country
loc_desc
loc_state
location_src
Source Location
log_session_id
log_session_id1
logon_type
longdec_dst
7.71351e+31
longdec_src
4.86134e+30
Column Name
Sort By
Enter the column name...
Ascending
Meta
Warehouse
locat
location_src
Lists
Filter
Insert
Attack Kill Chain Report
Compliance
Critical Windows Machines
Dev
Infected Filenames from ECAT
Local_Country

Configure Parser Mappings

This topic tells administrators how to configure event source mapping on a Log Decoder.

The Log Collector discovers the event source type on a per-message basis. If the correct parser is not identified for the event source, the messages common to the same event source types are misclassified. The misclassified messages do not populate event source rules and alerts, and the reports do not have the correct data. If there are multiple event source types associated with an IP address, it makes it difficult for the parsers to identify the exact event source from which the logs are generated.

If you map an IP address to its event source type, the Log Decoder can identify the event source from which the log is generated. When messages are delivered to the Log Decoder from a mapped event source, only the assigned parsers are queried to find event matches.

You can assign event source types to IPV4, IPV6, or the hostname value of the event source. You can also assign multiple event source types to a single IP address. You can also use the Log Collector ID when different event source types with the same IP address are sent to different Log Collectors.

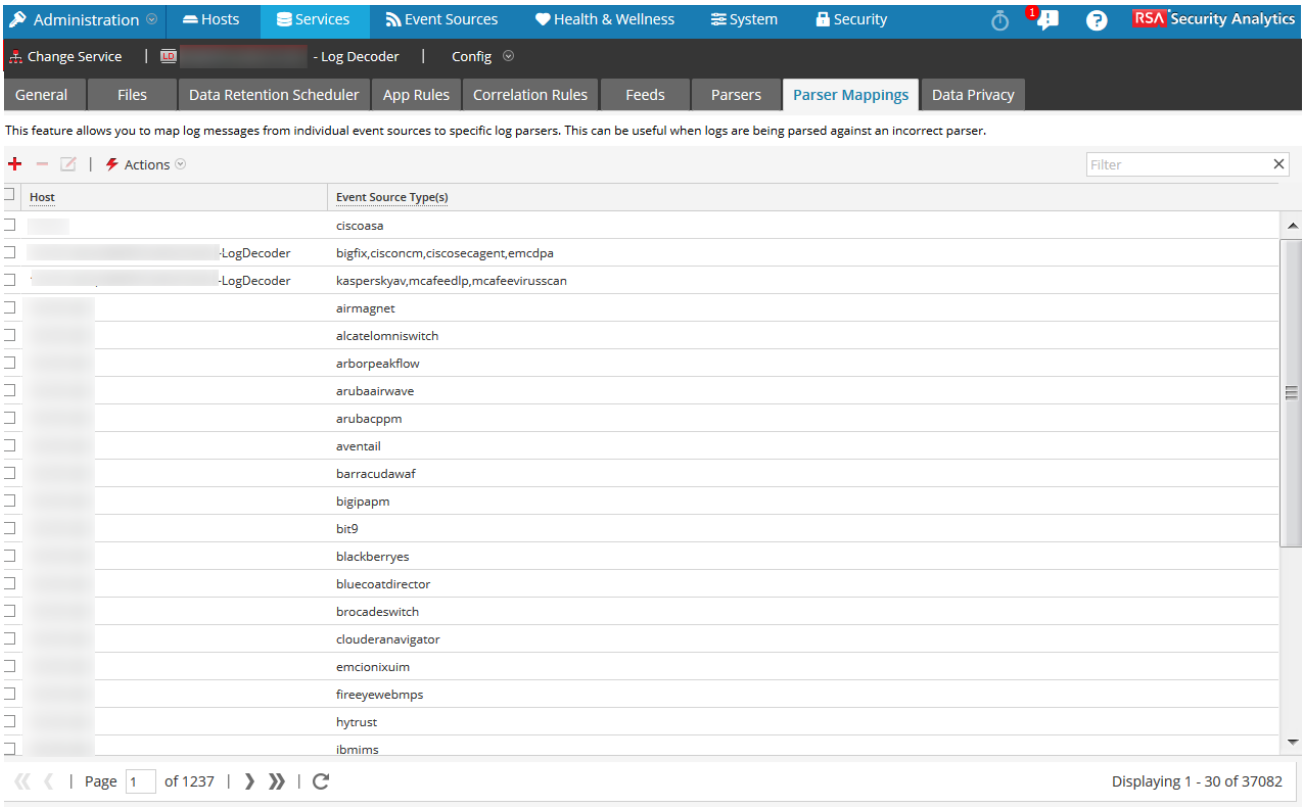
Update IP to Event Source Mapping

To update an IP to event source mapping:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a **Log Decoder**, and in the **Actions** column, select  **> View > Config**.

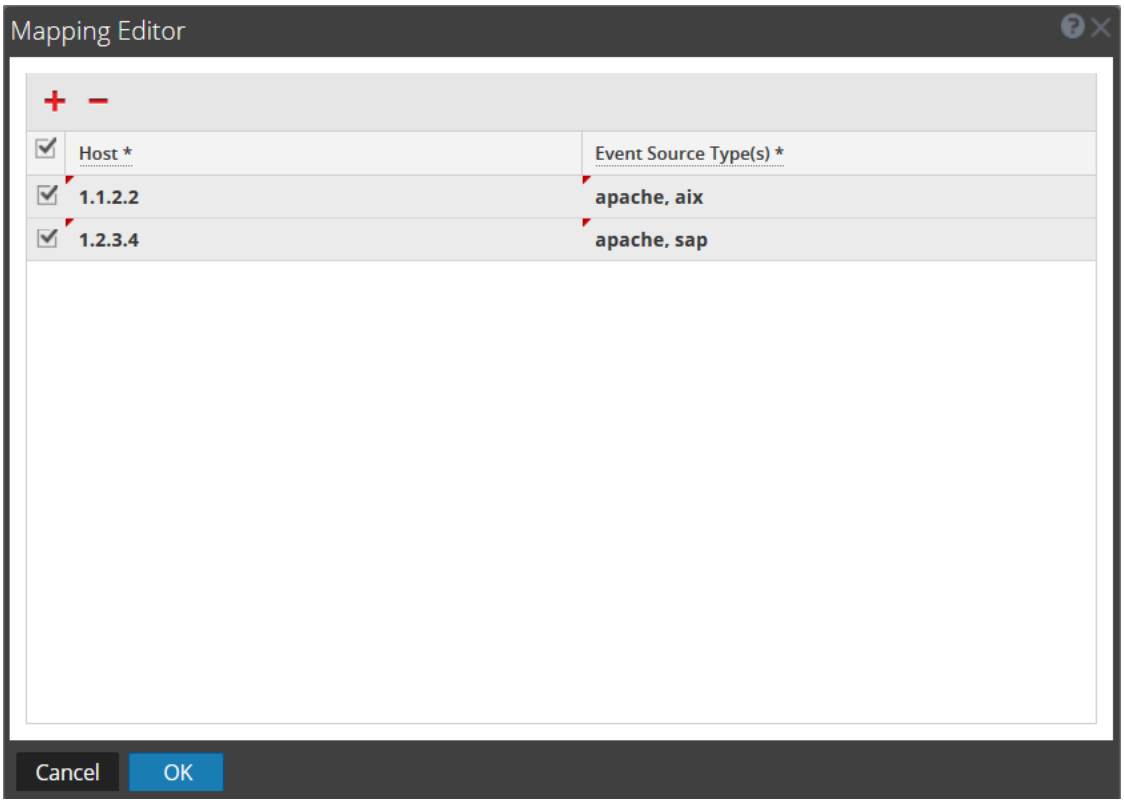
   The Services Config view is displayed.
3. Select the **Parsers Mapping** tab.

   The Parser Mappings tab is displayed.



4. Click + .

   The Mapping Editor is displayed.



5. Any of the following mappings can be defined:

- **One Host and One Event Source Type**
  - In the **Host** field, enter the hostname.
    For example: `10.0.0.1`
  - In the **Event Sources**(s) field, enter the event source type.
    For example: `apache`
- **One Host and One or More Event Source Types**
  - In the **Host** field, enter the hostname.
    For example: `10.0.0.1`
  - In the **Event Source(s)** field, enter the event source type.
    For example: `apache,sap,aix`
- **One Host, One Log Collector, and One Event Source Type**
  - In the **Host** field, enter the hostname and Log Collector ID.
    For example: `10.0.0.1,LC-1`.
  - In the **Event Source(s)** field, enter the event source type.
    For example: `apache`
- **One Host, One Log Collector ID, and One or More Event Source Types**
  - In the **Host** field, enter the hostname and Log Collector ID.
    For example: `10.0.0.1,LC-1`

- In the **Event Source(s)** field, enter the event source type.
  For example: `apache,sap,aix`

Note: The event source types are processed in the order you enter the parsers and if one or more parsers matches a log, the first parser in the list is queried. The Host/IP can be IPv4, IPv6, or Hostname.

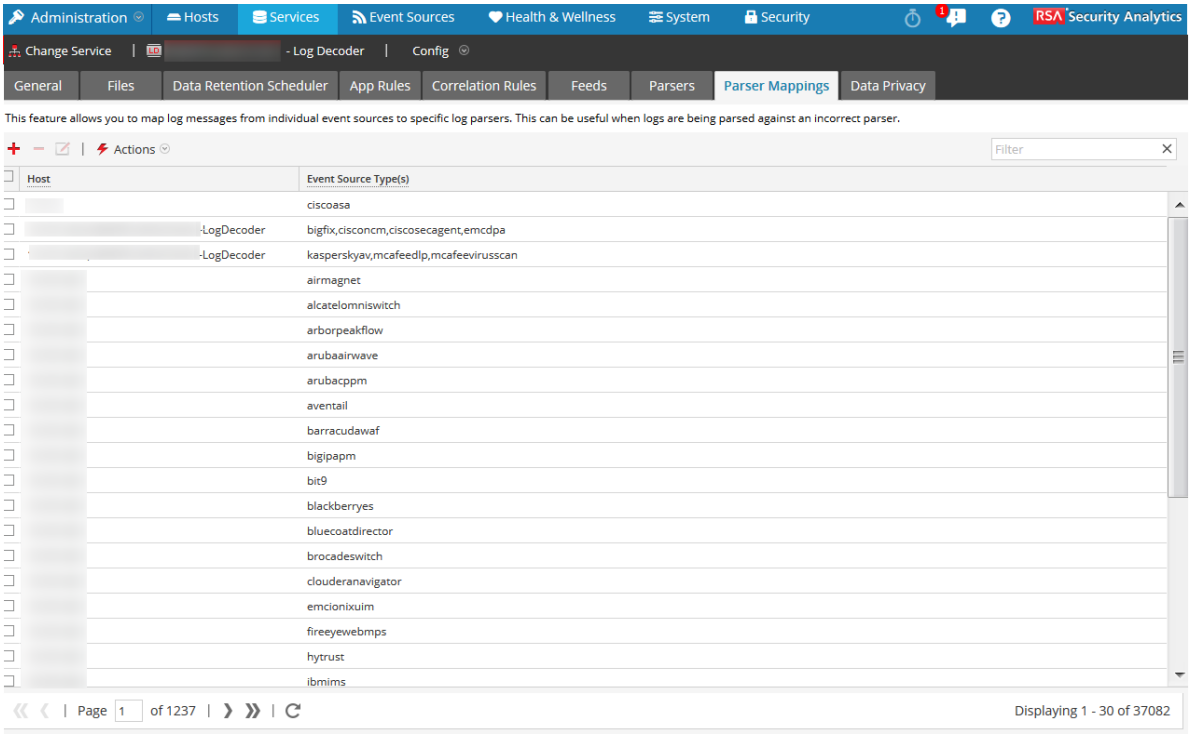
6. Click **OK**.
   The Parser Mapping is added.
7. To cancel the parser mappings selection, click **Cancel**.

Read IP to Event Source Type Mappings

To read an IP to event source type mappings:

1. In the Security Analytics menu, select **Administration > Services.**
2. Select a Log Decoder service.
3. In the Actions column, select  > **View** > **Config**.
   The Services Config view is displayed.
4. Select the **Parsers Mapping** tab.

The mappings are displayed.



Edit an IP to Event Source Type Mapping

To edit an IP to event source type mapping:

1. In the Security Analytics menu, select **Administration** > **Services**.
2. Select a Log Decoder service.
3. In the Actions column, select > **View** > **Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. Select the mapping you want to edit.

   Note: You can only edit one mapping at a time.
6. Click
7. In the **Event Source(s)** field, modify the event source(s).

   Note: The host is not editable and the field is disabled.

8. Click **OK** to accept the edited Event Source.
9. To cancel the changes, click **Cancel**.

Delete an IP to Event Source Type Mapping

To delete an IP to event source type mapping:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a Log Decoder service.
3. In the Actions column, select  **> View > Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. Select the mapping you want to delete.
6. Click .

   The mapping is deleted and the grid is refreshed.
7. To cancel the changes, click **Cancel**.

Sort the Hostname or Event Source Type

To sort the hostname or event source type:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a Log Decoder service.
3. In the Actions column, select  **> View > Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. To sort a column, click in the column header.

Event Source Type(s) are applied for your selected IP address. Logs are parsed against the parsers in the order they are listed.

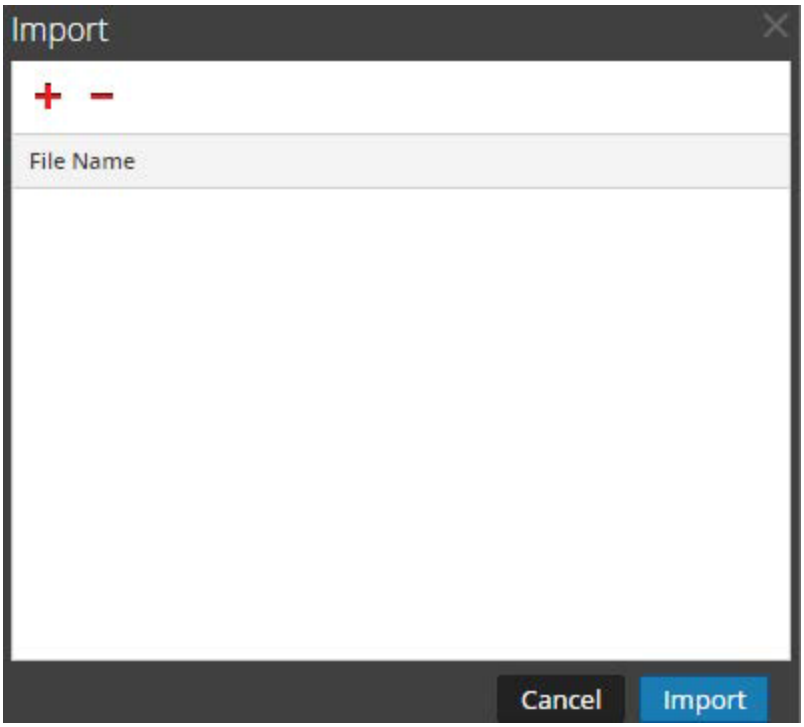
Import IP to Event Source Mapping Entries

To import IP to event source mapping entries:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a Log Decoder service.
3. In the Actions column, select > **View > Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. Select **Actions > Import**.

   The Import dialog is displayed.



6. Click .
7. Select the file you want to import and click **OK**.
8. To load the parser, click **Import**.

Note: You can only import one .csv file at a time.

Export IP to Event Source Mapping Entries

To export IP to event source mapping entries:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a Log Decoder service.
3. In the Actions column, select  **> View > Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. Select the mappings you want to export.
6. Select **Actions > Export > Selection**.

   The Export Selection dialog is displayed.

7. Enter the file name and click **Export**.

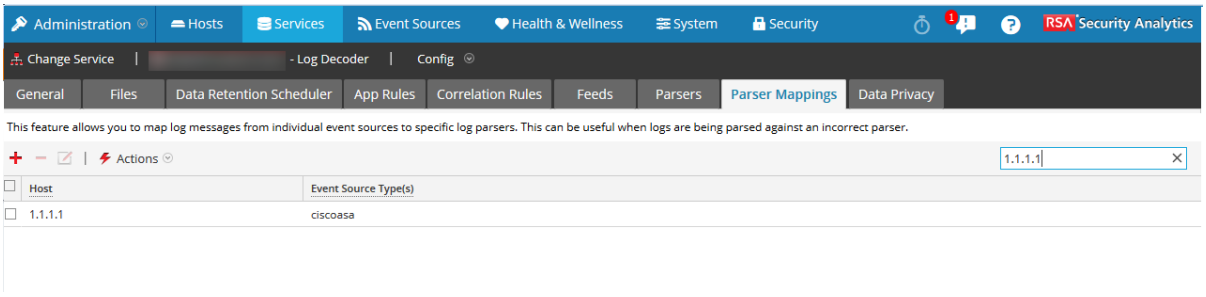
Search IP to Event Source Mapping Entries

To search IP to event source mapping entries:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a Log Decoder service.
3. In the Actions column, select  **> View > Config**.

   The Service Config view is displayed.
4. Select the **Parser Mappings** tab.
5. In the Parsers Mappings toolbar, enter the Host or Event Source in the **Filter** field.
6. Click **Enter**.

The Hosts or Event Sources that match the names entered in the **Filter** field are displayed.



Fix Rules with Deprecated Syntax

After an update to Security Analytics 10.6.2, the user interface highlights any rules with deprecated syntax. It is important to correct the syntax for the highlighted rules because they may contain ambiguous syntax, which can cause unexpected results. The Rule Editor provides additional tooltips. After you fix the rules, the highlights disappear.

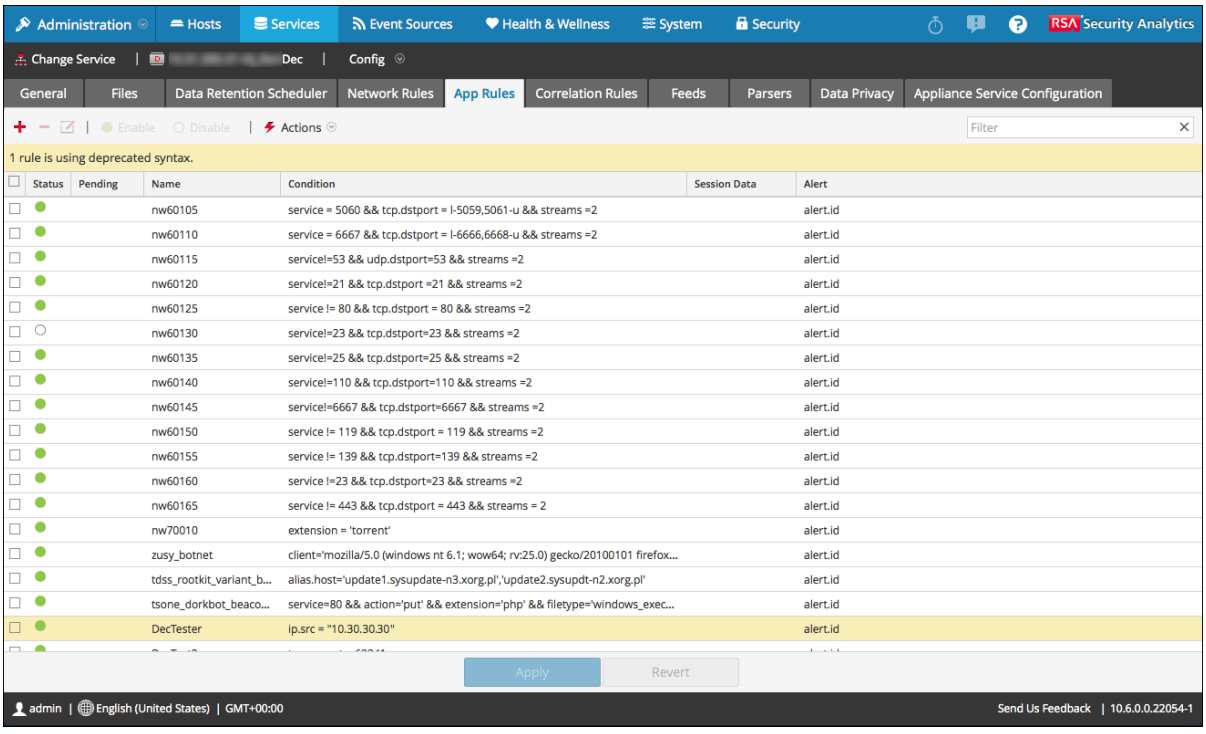
Rule and Query Guidelines provides guidelines that all queries and rule conditions in Security Analytics must follow. It also provides information about strict mode configuration as well as valid and deprecated syntax.

Procedure

To correct rules with deprecated syntax:

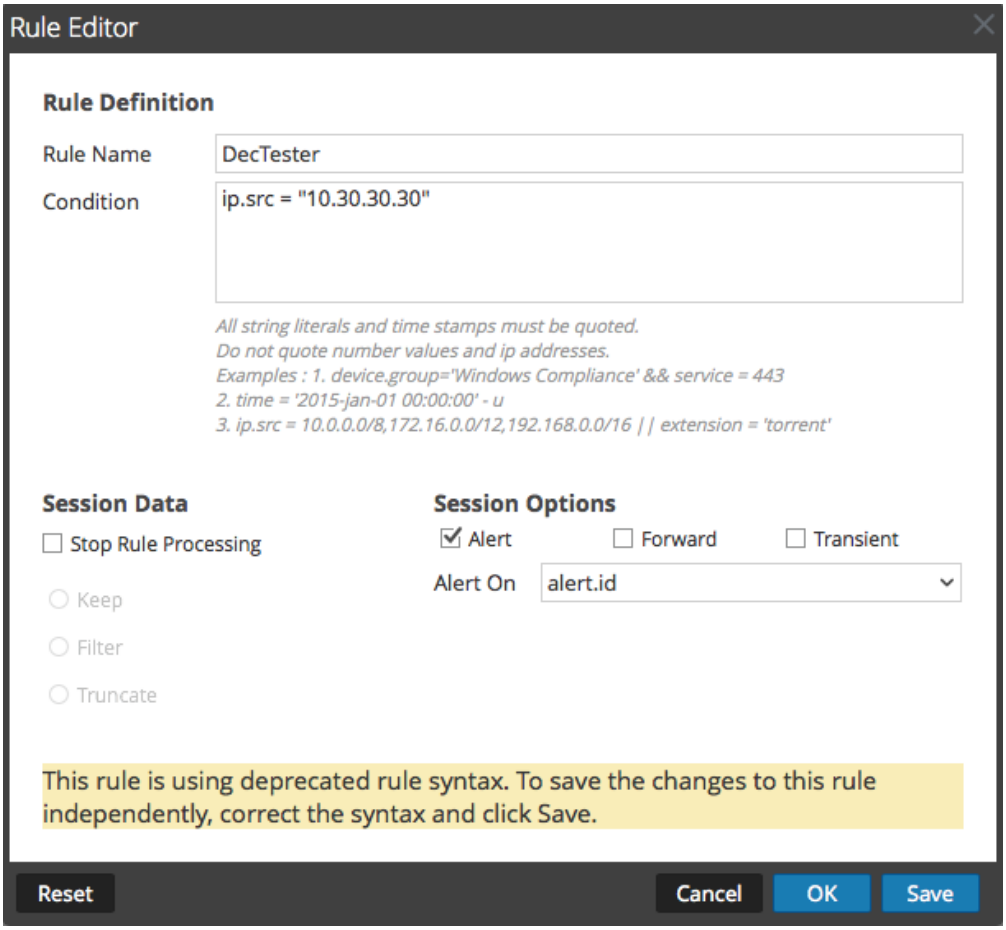
1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Decoder service and  **> View > Config**.
3. In the **Services Config** view, select one of the Rules tabs: Network Rules, App Rules, or Correlation Rules.

   The Rules tab for the selected rule type shows the number of rules using the deprecated syntax and the deprecated rules are highlighted.



4. Select a deprecated rule and click  .

   The Rules Editor shows additional information for the deprecated rule and it includes an additional Save option.



5. In the **Condition** field, correct the rule syntax.

   All string literals and time stamps must be quoted. Do not quote number values and IP addresses. Rule and Query Guidelines provides additional details.

   For example, if the deprecated rule condition is `ip.src="10.30.30.30"`, correct the syntax by removing the quotes: `ip.src=10.30.30.30`

6. Do one of the following:

   - To correct the rule individually, click **Save**.

     The corrected rule is applied independently to the Decoder service. The corrected rule appears on the Rules tab without highlights.

   - To correct the rule and apply the rule to the Decoder service later with other rules, click **OK**.

The corrected rule appears on the Rules tab without highlights. The rule is not applied to the Decoder service.

Enable or Disable Lua and Flex Parsing Systems

This topic tells administrators how to enable or disable Lua and Flex parsing systems on a Decoder or Log Decoder.

The settings to enable or disable Lua and Flex parsing systems are configured correctly by default and you do not typically have to change them. However, you may need to adjust these settings at the request of RSA Customer Care or for troubleshooting purposes.

In addition to configuring individual parsers, you can enable and disable all Lua parsing as well as all Flex parsing in the Services Explore view. You enable and disable the Lua parsing and Flex parsing systems settings separately, but they work in the same way.

- If you **disable** the Lua/Flex parsing system, the Lua/Flex parsing system is disabled and no Lua/Flex parsers are loaded.
- If you **enable** the Lua/Flex parsing system, the Lua/Flex parsing system is enabled and individual Lua/Flex parsers are enabled and disabled following the current individual configurations.

Procedure

To enable or disable Lua and Flex parsing systems on a Decoder or Log Decoder:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a Decoder or Log Decoder and  **> View > Explore.**
   The Services Explore view for the selected service is displayed.
3. In the Node list, navigate to and select **/decoder/parsers/config**.
4. In the Monitor panel:
   - To enable the Lua parsing system, in the value field for `lua.enabled`, type **yes**.
   - To disable the Lua parsing system, in the value field for `lua.enabled`, type **no**.
   - To enable the Flex parsing system, in the value field for `flex.enabled`, type **yes**.
   - To disable the Flex parsing system, in the value field for `flex.enabled`, type **no**.

Map IP Address to Service Type

This topic describes the procedure to map an IP address to a service type for log parsing.

The Log Collector discovers event source type on a per-message basis. If the correct parser is not used for the specific event source, the messages that are common between event source types are misclassified. The misidentified messages will not populate service rules and alerts, and the reports will not have proper information. Also, if there are multiple services associated with an IP address, it can be difficult for the parsers to identify the exact service from which the log is generated.

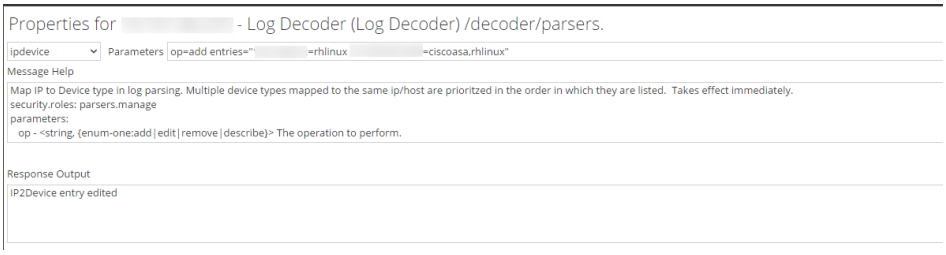
If you map an IP address to its services, the log decoder can identify the service from which the log is generated. When messages come into the log decoder from a mapped service, the assigned parsers are loaded to find event matches.

You can assign service types to IPV4, IPV6 or hostname value of the event source. You can also assign multiple service types to a single IP address. You can also use the CollectorID when different service types with the same IP address are sent to different collectors.

Procedure

To map an IP address to a service type, do the following:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Log Decoder, and in the **Actions** column, select  **> View > Explore**.
3. Go to **/decoder/parsers** node, right-click **parsers**, and select **Properties**.
4. In the **Properties** view, specify the **ipdevice** command with the following parameters: `op=add/remove entries="ipaddress=service"` (for example, `op=add entries="10.100.201.300=ciscoasa"`)
5. Click **Send**.



IPdevice Command

In the **ipdevice** command, three operations are available:

- *add*: This operation adds or updates entries in the ipdevice map. Multiple space delimited address/type pairs may be specified.
  `op=add entries="<address>=<service type>"`
- *remove*: This operation removes entries from the ipdevice map. Multiple space delimited address/type pairs may be specified.
  `op=remove entries="<address>"`
- *describe*: This operation returns the values currently in the ipdevice map.

Time Zone Support

The Log Decoder currently has the ability to configure the system so that a given log device source can be associated with a time zone so that the event can be correctly converted to UTC across all devices.

Three time zone formats are currently accepted and are shown in the following examples:

1. **Olson format**:
   `America/Anguilla`
2. **POSIX formats**:
   `EST5EDT`
   `AST2:45ADT0:45,M4.1.6/1:45,M10.5.6/2:45`
3. **Offset by Hours formats**:
   `EST`
   `-500`

Note: Offset by Hours time zone formats do **not** change for Daylight Savings Time.

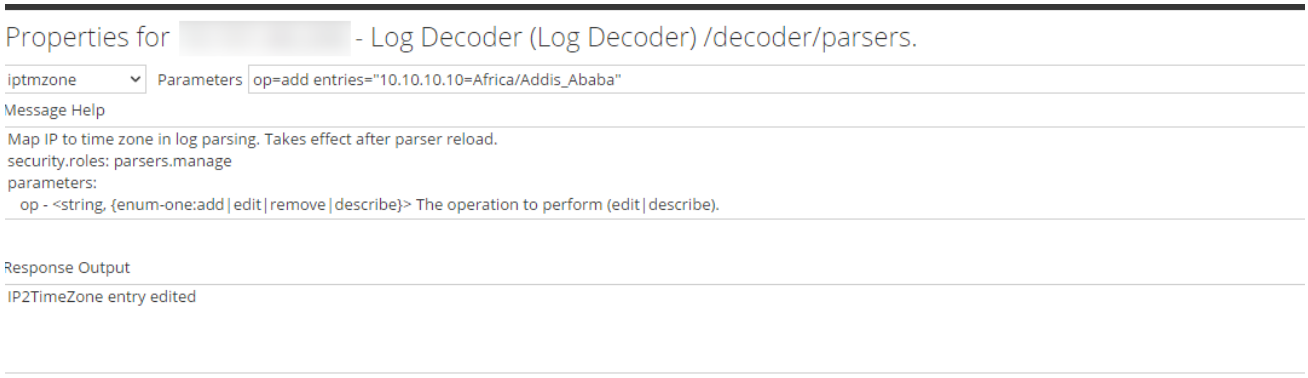
Result

Security Analytics maps the IP address to a time zone in the log decoder. Event time meta is updated according to their respective mappings.

Procedure

To map an IP address to a time zone, do the following:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Log Decoder, and in the **Actions** column, select  **> View > Explore**.
3. Go to **/decoder/parsers** node, right click **Parsers**, and select **Properties**.
4. In the **Properties** view, specify the iptmzone command with the following parameters:
   `op=add entries="ipaddress=timezone" (for example, op=add entries="10.10.10.10=Africa/Addis_Ababa")`
5. Click **Send**.



iptmzone Command

In the **iptmzone** command, three operations are available:

- *add*: This operation adds or updates entries in the iptmzone map. Multiple space delimited address/type pairs may be specified.
  `op=add entries="<address>=<time zone>"`
- *remove*: This operation removes entries in the iptmzone map. Multiple space delimited address/type pairs may be specified.
  `op=remove entries="<address>"`
- *describe*: This operation returns the values currently in the iptmzone map.

Examples

The following examples provide instances for mapping IP addresses to time zones:

- If you want to map two different entries with different IPV4 values and time zone, enter the following parameter in the **iptmzone** command and click **Send**
  `"op=add entries=”10.10.10.10=America/Anguilla`

```
10.10.10.11=Pacific/Rarotonga”
```

- If you want to remove an entry for a single IPV4 value and time zone, enter the following parameter in the **iptmzone** command and click **Send**.

```
"op=remove entries=10.5.245.9"
```

- If you want to create a single entry for an IPV6 value and time zone, enter the following parameter in the **iptmzone** command and click **Send**.

```
op=add entries=”2001:DB8:85A3::8A2E:370:7334=America/Anguilla”
```

- If you want to map a single device to a time zone or offset, you can create an entry by using:

  `op=add entries="<address>=<time>"`
  Where `<address>` is an IPV4, IPV6, or hostname and where `<time>` is an integer offset or a time zone Olson, or POSIX format. Enter the following parameter in the **iptmzone** command and click **Send**.

  For example:
  `op=add entries="10.168.0.2=EST5EDT"`

  Alternately, you can enter the following parameter in the **iptmzone** command and click **Send**.

  For example:

```
op=add entries="10.168.0.2=America/Anguilla
2001:DB8:85A3::8A2E:370:7334=0500
nwappliance21=EST5EDT,M3.2.0/2,M11.1.0"
```

Event Time Support

A parameter `mdformat` has been added to the **iptmzone** message in */decoder/parsers* on a Log Decoder.
The Log Decoder currently has the ability to change the dates in the logs to have the following format:
mdy or dmy (month/day/year or day/month/year)

By default, the value for the date is set to **none**.

Note:
. Event time meta in a parser does not account for dmy and mdy.
. The day/month/year change can only be done from the Rest API. There is no UI capability to do this.
. Currently, there is no functionality to detect this scenario and adjust parsing automatically.

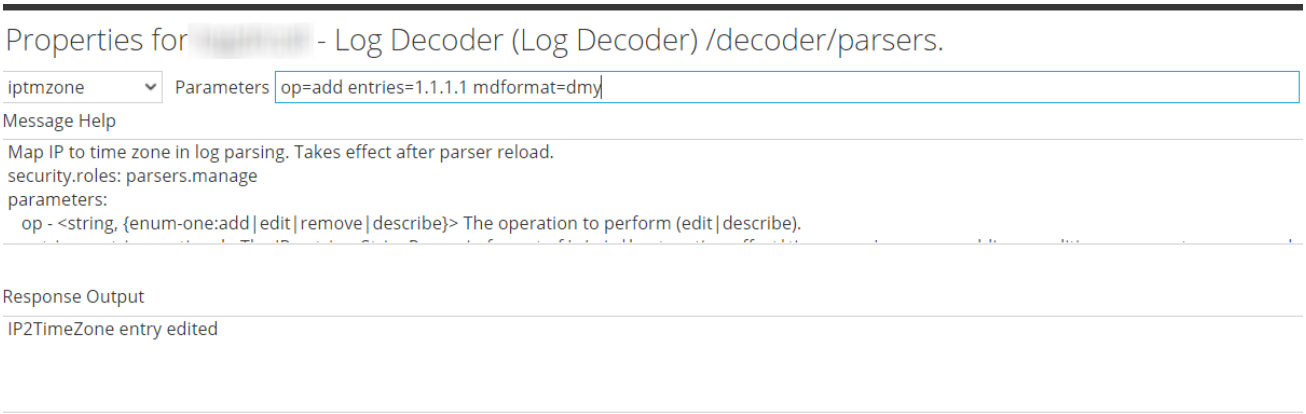
Result

Security Analytics maps the IP address along with the date format in the Log Decoder. Event time meta is updated according to their respective mappings.

Procedure

To change the date format, do the following:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. In the **Services** view, select a Log Decoder, and in the **Actions** column, select  **> View > Explore**.
3. Go to **/decoder/parsers** node, right click **Parsers**, and select **Properties**.
4. In the **Properties** view, select **iptmzone** from the drop-down list, and specify the command with the following parameters:
   `op=add entries="ipaddress" mdformat="dmy or mdy or none"(for example, op=add entries="1.1.1.1" mdformat=dmy)`
   Where,
   entries is the Device IP address, and
   mdformat can be dmy, mdy, or none.
5. Click **Send**.



Missing Year Support

If there is no year associated with a format string, the parser attempts to populate the year heuristically. In most cases, the value is populated based on the current year. The current year is assigned as the message year (UTC), as per the clock on the Log Decoder.

The following are the various other scenarios:

1. Latent log during transition to new year.
2. Logs from forward time zones (or skewed clocks) just before new year transition.
3. Latent logs received where a leap day cannot be successfully assigned to an appropriate leap year.

Latent log during transition to new year

If the day is more than 31 days into the future, the year is decremented.

For example,

The message date is Dec-31. The current date is 2018-Jan-1. The temporary assigned date, 2018-Dec-31, will be more than 31 days into the future.

This will cause the year component to be decremented to 2017, resulting in a reasonable message time.

Logs from forward time zones (or skewed clocks) just before new year transition

If the day is more than 334 days into the past, the year is incremented.

For example,

The message date is Jan-1. The current date is 2017-Dec-31. The temporary assigned date, 2017-Jan-1, will be more than 334 days into the past.

This will cause the year component to be incremented to 2018, resulting in a reasonable message time.

Latent logs received where a leap day cannot be successfully assigned to an appropriate leap year

If the day (Feb 29) is invalid (say, the assigned year is NOT a leap year), the relative position to the current time cannot be calculated. To do this, the year is decremented in an attempt get a valid time stamp.

Because the position of the leap day relative to the transition to the new year, along with the expectation that no logs would be received this far into the future, we do not ever make an attempt to increment the year to produce a valid time stamp.

After the year is decremented, the same logic is then followed (refer to statement 1 and 2). If the result is still an invalid day. The parser throws and a valid message time cannot be parsed.

Limitations

Note: Currently, there is no mechanism to assign a year to the import of logs.

As this pertains to leap days, if you find the correct leap year, an inconsistent data would result. For example, if a set of messages is two years old and during a leap year, only a single day would remain which is accurate. Because of the heuristic described above, the remaining set of messages would have time stamps one year too.

For data consistency, usually you cannot find valid leap years beyond 1 month and less than 11 months.

Examples

The following examples provide instances for logs with year and logs without year:

- If the log is with year, event time is generated as below:

```
%emcavamar: 1704^^2017-04-07^^07:50:02^^1^^<event-source NodeID="avamar"
ProgramName="com.avamar"/>^^1270626^^SYSTEM^^ERROR^^OK^^MCS:DPN_
Proxy^^Internal server error^
<MESSAGE
        id1="1:01"
        id2="1"
        eventcategory="1605020000"
        functions="&lt;@msg:*PARMVAL($MSG)&gt;&lt;@event_time:*EVNTTIME
        ($MSG,'%W-%G-%F %H:%U:%O',fld2,fld3)&gt;"
        content="&lt;fld1&gt;^^&lt;fld2&gt;^^&lt;fld3&gt;^^&lt;fld4&gt;^^&lt;&lt;event-source
        NodeID=&quot;&lt;hostname&gt;&quot;
        ProgramName=&quot;&lt;fld8&gt;&quot;/&gt;^^&lt;fld9&gt;^^&lt;category&gt;^^&lt;se
        verity&gt;^^&lt;event_type&gt;^^&lt;agent&gt;^^&lt;event_description&gt;" />
```

- If the log is without year, event time is still generated, The current year is assigned as the message year (UTC), as per the clock on the Log Decoder.

%emcavamar: 1704^^04-07^^07:50:02^^1^^<event-source NodeID="avamar" ProgramName="com.avamar"/>^^1270626^^SYSTEM^^ERROR^^OK^^MCS:DPN_ Proxy^^Internal server error^

<MESSAGE

id1="1:01"

id2="1"

eventcategory="1605020000"

functions="<@msg:*PARMVAL($MSG)><@event_time:*EVNTTIME ($MSG,'%G-%F %H:%U:%O',fld2,fld3)>"

content="<fld1>^^<fld2>^^<fld3>^^<fld4>^^<<event-source NodeID="<hostname>" ProgramName="<fld8>"/>^^<fld9>^^<category>^^<severity>^^<event_type>^^<agent>^^<event_description>" />

Upload Log File to a Log Decoder

This topic describes the method for importing a log file to a Log Decoder.

There are occasions when you want to analyze a log file that is not available on the service you are using. You can upload a log file captured on another service to Security Analytics. Log filenames are of the type **.log**.

When a log file is uploaded to a Log Decoder, the Log Decoder analyzes and generates meta for each log it contains. These logs are added to the already decoded logs on the Log Decoder and are available for analysis. Security Analytics includes a filename tracking option that makes searching for a particular set of logs easier. When the log file is uploaded with file tracking, the Log Decoder adds meta to each log based on the uploaded filename. You can then filter sessions for analysis using that meta.

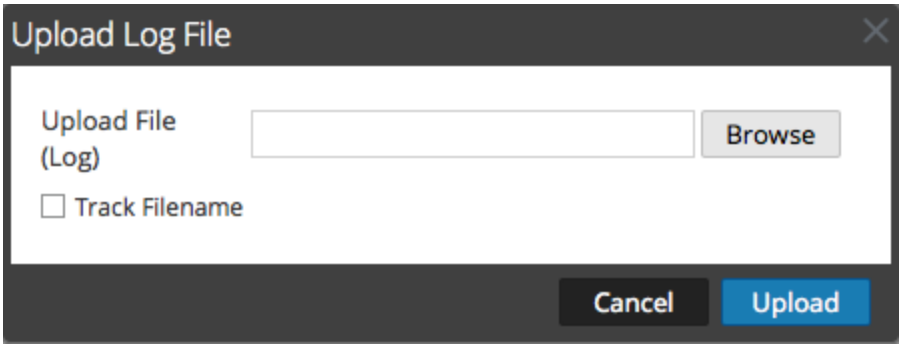
The option to upload a log file is dimmed when other Log Decoder operations prevent an upload from occurring. For example, when the Log Decoder is capturing logs.

Procedure

To import a log file to an Log Decoder:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. Select a Log Decoder in the **Service** grid, and select  **> View > System**.

   The Services System view for the Log Decoder is displayed.
3. In the toolbar, click **Upload Log File**.



4. To choose a log file, click **Browse**.

   A directory view is displayed.
5. Select the log file that you want to upload.

   The filename is displayed in the **Upload File** field.

6. If you want the Log Decoder to add meta to the logs based on the filename, click the checkbox next to **Track Filename**.
7. To upload the file, click **Upload**.
   The selected file is uploaded and a status message indicates that the file is uploaded. The log file is available for analysis.

Upload Packet Capture File

This topic explains how to import a packet capture file to a Decoder.

There are occasions when you want to analyze a packet capture file that is not available on the service you are using. You can upload a file captured on another service to Security Analytics. Supported packet capture file types are **pcap** and **pcap.gz**.

When a packet capture file is uploaded to a Decoder, the Decoder creates sessions from the packet capture file packets. These sessions are added to the already decoded sessions on the Decoder and are available for analysis. Security Analytics includes a filename tracking option that makes searching for a particular set of sessions easier. When the packet capture file is uploaded with file tracking, the Decoder adds meta to the sessions based on the uploaded filename. You can then filter sessions for analysis using that meta.

The option to upload a packet capture file is dimmed when other Decoder operations prevent an upload from occurring; for example, when the Decoder is capturing packets.

Procedure

To select and upload a packet capture file:

1. In the **Security Analytics** menu, select **Administration >Services**.

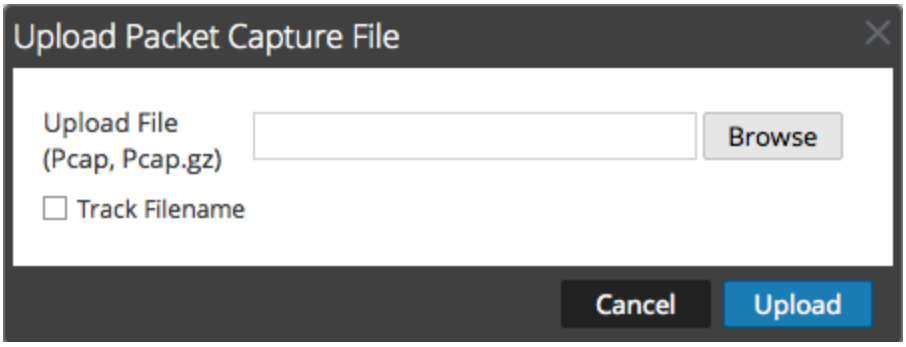
   The Administration Services view is displayed.

2. Select the Decoder name, and  **> View > System**.

   The Services System view for the Decoder is displayed.

3. In the toolbar, click **Upload Packet Capture File**.

   The **Upload Packet Capture File dialog** is displayed.



4. To choose a capture file, click **Select**.

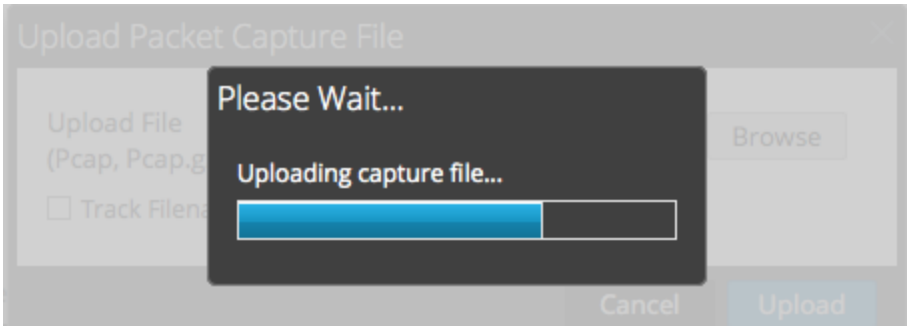
   A directory view is displayed.

5. Browse the directory and select the packet capture file that you want to upload.

The filename is displayed in the **Upload File(pcap,pcap.gz)** field.

6. If you want the Decoder to add meta to the sessions based on the filename, click the checkbox next to **Track Filename**.
7. To upload the file, click **Upload**.

   A progress bar shows upload progress.



Upload time varies depending on the size of the file. When the file upload is complete, a status message is displayed. The file is now available for investigation.

Verify Decoder System Information

This topic introduces features in the System view that pertain specifically to Decoders and Log Decoders.

When a service is first added to Security Analytics, default values for the system configuration parameters are in effect. You can edit these values to tune performance.

System Configuration

Name	Config Value
Compression	0
Port	50005
SSL FIPS Mode	☐
SSL Port	56005
Stat Update Interval	1000
Threads	20

In most cases, the default values for compression, statistics update interval, and number of threads in the thread pool are set at a good point for optimal system performance. One parameter that you may want to change for your environment is the SSL setting, which by default is not enabled. When enabled, the security of data transmission is managed by encrypting information and providing authentication with SSL certificates.

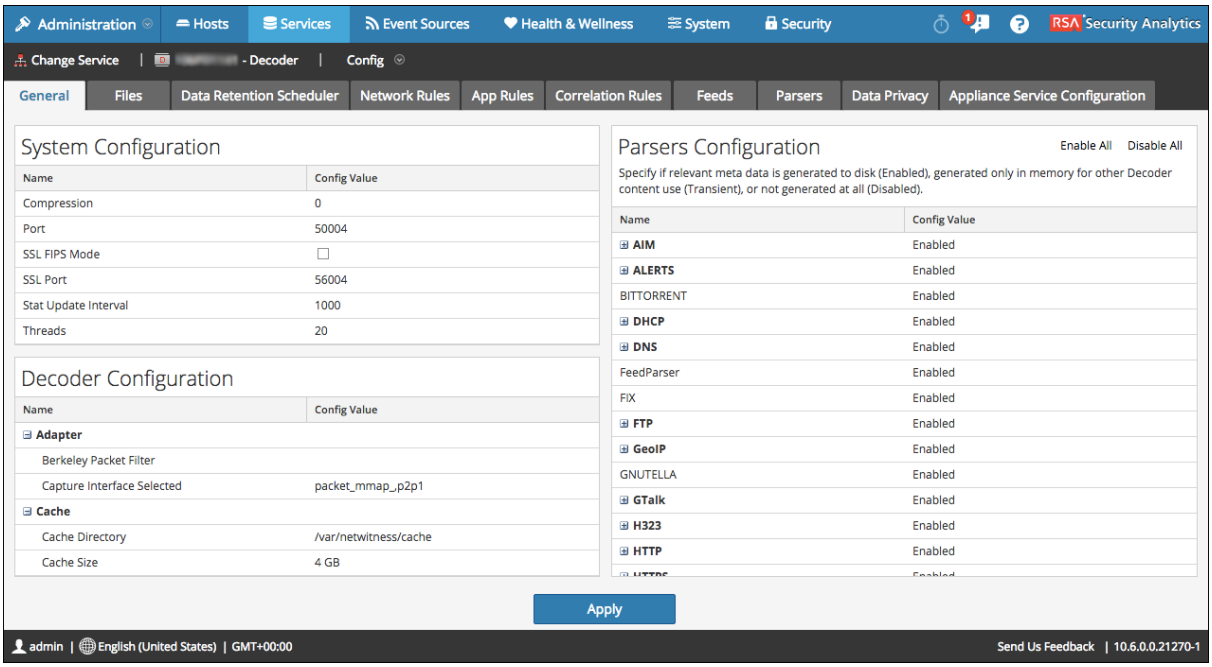
Procedure

To edit system configuration parameters:

1. In the **Security Analytics** menu, select **Administration > Services**.

2. In the **Services** view, select a Decoder or Log Decoder and  **>View > Config**.

   The Services Config view for the selected service is displayed.



3. Under **System Configuration**, click a field that you want to edit, and type a new value.
4. When finished editing, click **Apply**.

Configure a Log Decoder to Accept Protobuf

This topic describes the method for configuring a Log Decoder to accept logs in protobuf (Protocol Buffer) format.

There are occasions when you want to analyze log files that are in protobuf (Protocol Buffer) format.

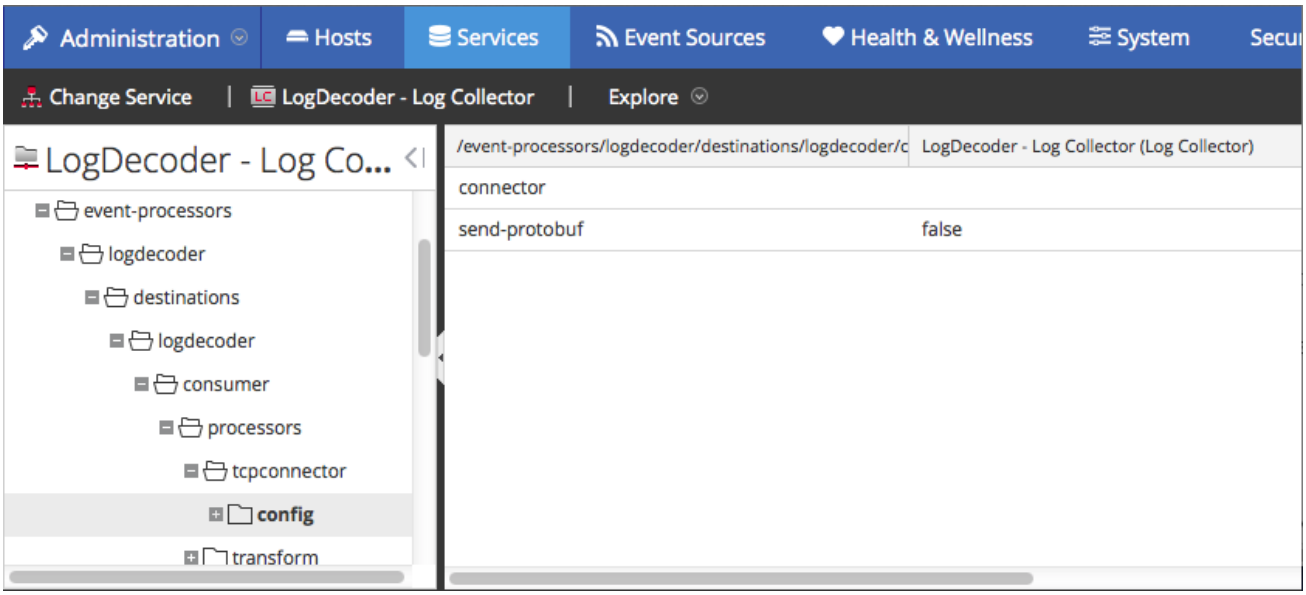
Procedure

To import a log file to a Log Decoder:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. Select a Log Decoder in the **Service** grid, and select  **> View > Explore**.

   The Explorer view for the Log Decoder is displayed.
3. Navigate to `event-processors/logdecoder/destinations/logdecoder/consumer/processors/tcpconnector/config`

   Your screen should look similar to the following.



4. For the **send-protobuf** field, select **false**, and change the value to **true.**
5. Navigate to `event-processors/logdecoder/destinations/logdecoder/consumer/processors/tcpconnector/config/connector/channel/tcp` and change the **port** value to **50202**.

6. Navigate to `event-processors/logdecoder/destinations/logdecoder/consumer/processors/tcp connector/config/connector/event` and change the following parameters:
   - Clear the **delimiter** field
   - Change **format** to **%text%**

Decoder and Log Decoder References

This topic is a collection of references, which describe the user interface for Decoders and Log Decoders in Security Analytics. These topics are presented in alphabetical order.

Use this section when you are looking for descriptions of the entitlements user interface and definitions of the features of the user interface.

The Security Analytics Services Config view provides a user interface for configuring Decoders and Log Decoders to capture data and controlling the type of traffic captured using rules, feeds, and parsers.

Topics

- Services Config View - Data Privacy Tab
- Services Config View - Feeds Tab
- Services Config View - Files Tab
- Services Config View - General Tab
- Services Config View - Parser Mappings Tab
- Services Config View - Parsers Tab
- Services Config View - Rules Tabs
- Services System View - Decoders

Services Config View - Data Privacy Tab

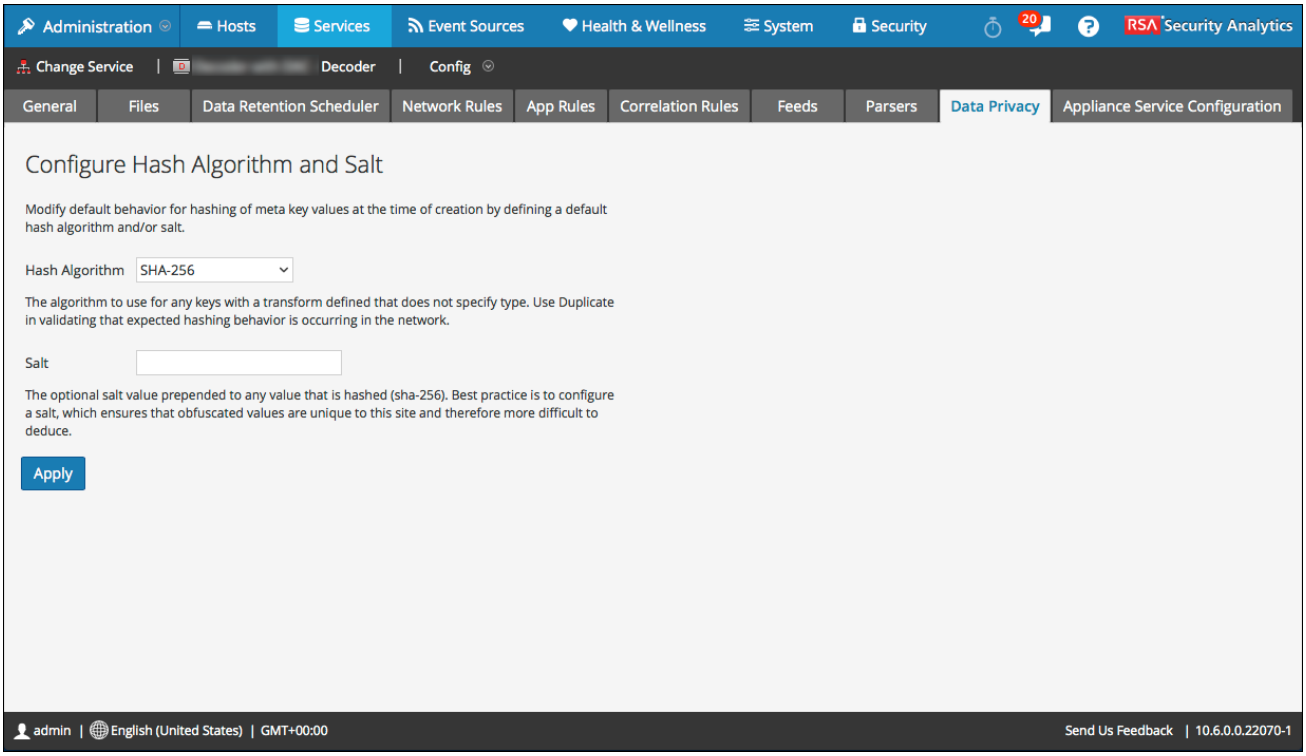
This topic provides a description of the configurable options for a Decoder or Log Decoder in the Data Privacy tab.

In the Data Privacy tab, Administrators can configure data privacy parameters for certain Core services. For the Decoder and Log Decoder, you can set the default hash algorithm and salt.

To access this tab:

1. In the **Security Analytics** menu, select **Administration > Services**.
2. Select a Decoder or Log Decoder service and click > **Config**.

   The General tab is displayed.
3. Click the **Data Privacy** tab.



Features

The Data Privacy tab has the Configure Hash Algorithm and Salt configuration settings. The following table describes the parameters in this tab.

Parameter	Description
Hash Algorithm	Displays a drop-down list of hash algorithms to use for any keys with a transform that does not specify algorithm type. Possible values are SHA-256 and Duplicate. Duplicate is a special algorithm available for administrators to use when validating that expected hashing behavior is occurring in the network. In versions of Security Analytics prior to 10.5, SHA-1 was available as a hash algorithm, but RSA does not recommend use of SHA-1.
Salt	Indicates the optional salt value prepended to any value that is hashed. Best practices for security purposes dictate a salt value that is no less than 100 bits or 16 characters in length. Configuring a value ensures that obfuscated values are unique to this site and therefore more difficult to deduce. For more information on this field, see the **Configure Data Obfuscation** topic in the *Data Privacy Management* guide.
Apply	Applies any changes.

Services Config View - Feeds Tab

This topic describes the features in the Decoder Services Config view > Feeds tab.

Feeds and parsers are FLEXPARSE programs loaded and compiled when either processing capture files in Investigation or capturing data with Decoders. Most commonly, they are used for static meta extraction and service identification.

Note: Unless otherwise stated, any reference to Decoders applies to Log Decoders as well.

Security Analytics uses feeds to create metadata based on externally defined meta values. A feed is a list of data that is compared to sessions as they are captured or processed. For each hit, additional metadata is created. This data can identify and classify malicious IPs or incorporate additional information such as department and location based on internal network assignments. Some examples of feeds include threat feeds to identify BOTNets, DHCP mappings, or even active directory information such as physical location or logical department.

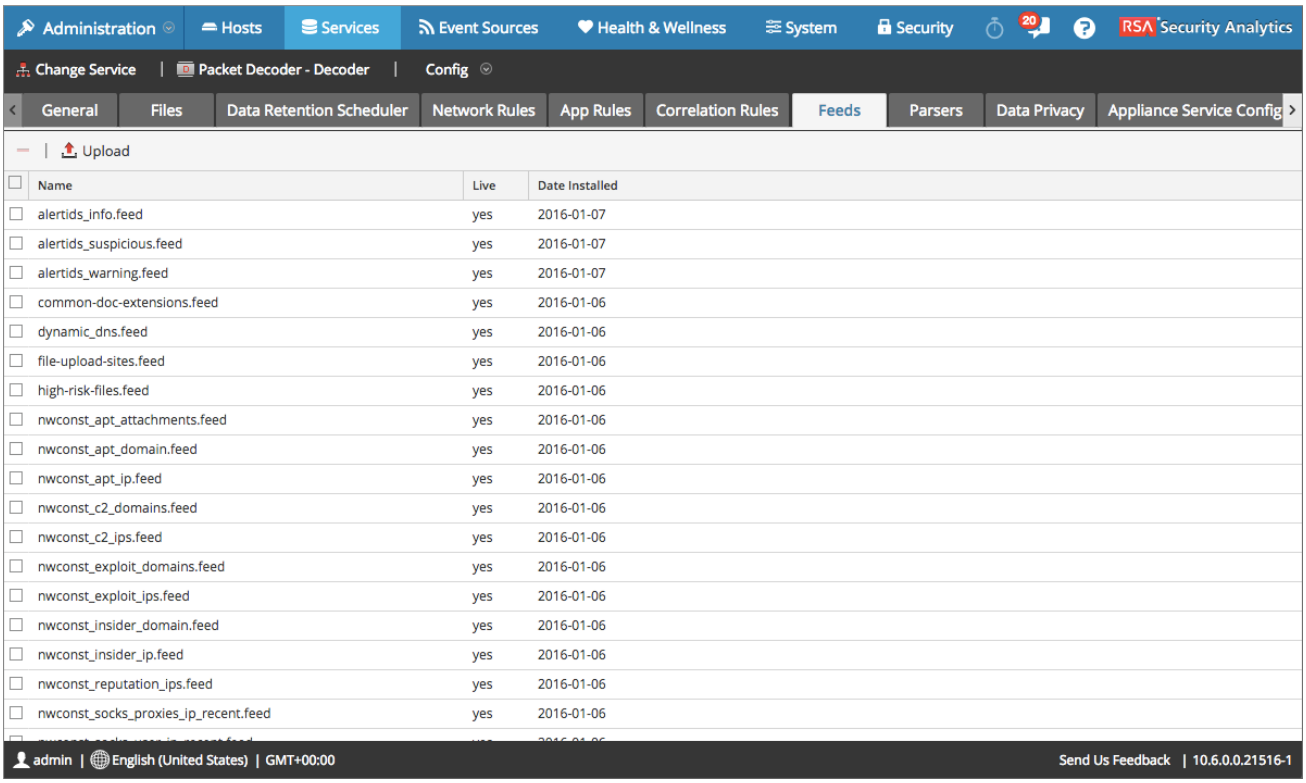
Feeds can be added, removed, and updated while a Decoder is running without affecting capture. The Services Config View > Feeds Tab provides a user interface for managing feeds on Decoders.

To display this view, do the following:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **Feeds** Tab.

This is an example of the Feeds tab.



Features

The Feed Grid lists all feeds that are currently deployed on the Decoder. The Feeds Tab Toolbar has options to work with feeds in the grid.

Feeds Tab Toolbar

Feature	Description
Upload	Displays the Upload Feeds dialog.
	Deletes the selected feeds.

Feed Grid

The Feed grid provides a listing of all currently deployed feeds for the Decoder.

Column	Description
Name	The name of the feed or the feed file.

Column	Description
Live	Indicates if the feed originated from Live. Possible values are **Yes**, **No**, or **N/A**. • **Yes** = Installed through Live • **No** = Installed through Security Analytics • **N/A** = The feed has no attributes file created by Security Analytics to track the installation date. The feed may have been installed manually, not through Security Analytics or Live. Manually installed feeds still function properly.
Date Installed	The date the feed was pushed to the service.

Topic

- Upload Feeds Dialog

Upload Feeds Dialog

This topic describes the features of the Upload Feeds dialog in the Services Config view > Feeds tab.

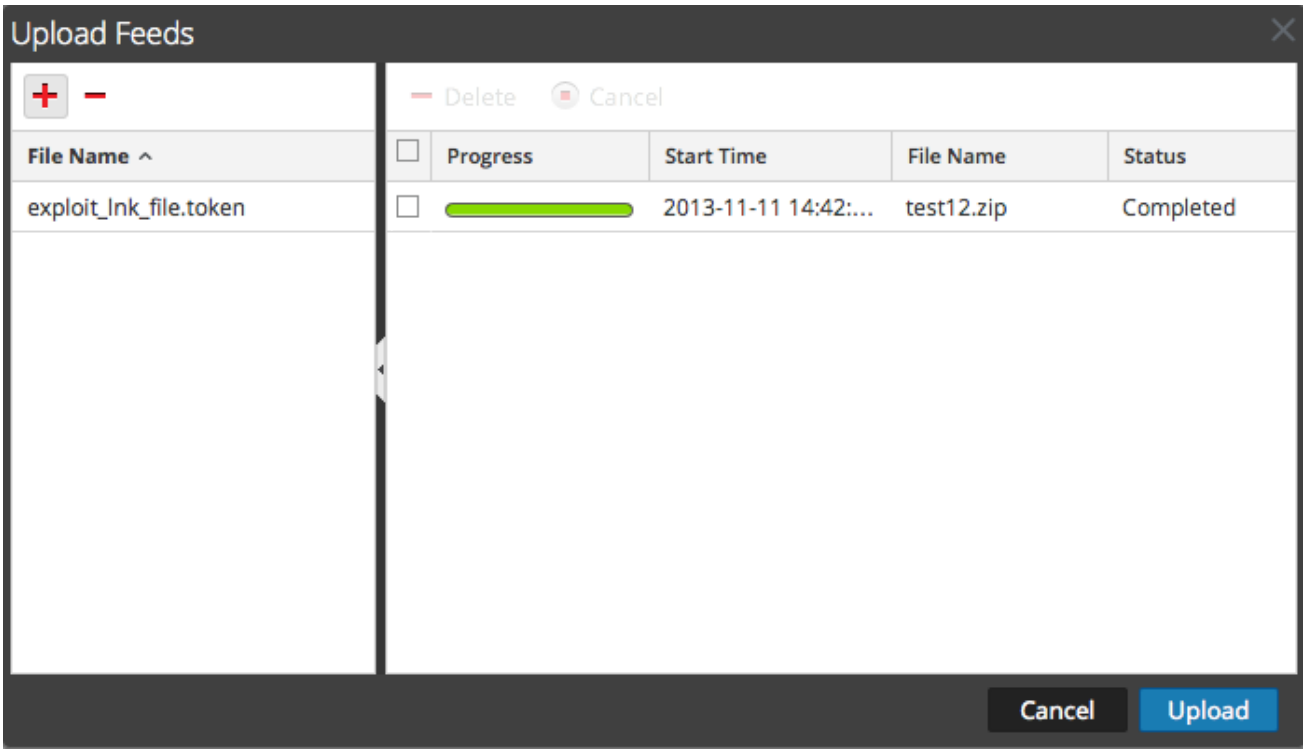
The **Upload** option in the Services Config view > Feeds tab displays the Upload Feeds Dialog, in which you can manage the uploading of feeds to a Decoder or Log Decoder.

You can access this view by doing the following:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **Feeds** Tab.
4. Click Upload.

This is an example of the Upload Feeds dialog.



File Grid

The File grid is the place to prepare a list of feeds for uploading. You can add files from a directory structure, and delete files from the grid if you decide that you don't want to upload a particular file. When the list is ready, clicking **Upload** starts the upload process.

Feature	Description
	Opens a view of the directory structure where you can select files to add to the File grid.
	Deletes the selected files from the File grid.
File Name	Lists the feed files you have added from a file system in preparation for uploading to a Decoder. When you click **Upload**, the files listed here are uploaded.

Upload Job Grid

The Upload Job grid provides a view of upload jobs started by clicking **Upload**.

Feature/Column	Description
	Deletes an upload job.
Progress	Displays progress of an upload job.
Start Time	Displays the start time of an upload job.
File Name	Lists filename of the feed being uploaded.
Status	Displays the status of upload job.

Upload Feeds Dialog Buttons

Feature	Description
Cancel	Closes the Upload Feed dialog.

Feature	Description
Upload	Starts uploading the feed files listed in the File grid. Each feed is listed in a separate row in the Upload Process grid.

Services Config View - Files Tab

This topic introduces the Decoder and Log Decoder configuration files that are visible in the Services Config view > Files tab.

The Decoder and Log Decoder configuration files are visible and editable in the Services Config view > Files tab. The **Edit Core Services Configuration Files** topic in the *Hosts and Services Getting Started Guide* provides general instructions for editing files.

Like other core Security Analytics services, both the Decoder and Log Decoder have an index file, and may also have a crashreporter, netwitness, and scheduler. The Decoder and Log Decoder index files are named **index-decoder.xml** and **index-logdecoder.xml.**

Note: This file type is available only for Log Decoder with Envision content installed. Table-map.xml and table-map-custom.xml will now show up but only if table-map.xml was found on the file system (e.g., it's a log decoder with envision content installed).

Filename	Description
GeoPrivate.ipl	This fixed parser takes the IP addresses and converts them to geographical locations. The locations are displayed through the Google Earth display.
NwFlex.parser	This is a generic parser definition language for extending the existing application protocol support of the Decoder.
feed-definitions.xml	Used to create custom feeds, this is the XML schema used by the Decoder to define a **feed** message when it creates a **.feed** file.
search.ini	This is the Search Parser configuration file, The Search Parser is a custom parser, used to generate metadata by scanning for pre -defined keywords and regular expressions.
wlan-config.xml	This is the wireless LAN configuration file (9/9/2009). This file controls the 802.11 parsers. Its chief purpose is to control decryption of raw 802.11 frames captured by the Decoder.

Related Topics

- Feed Definitions File
- Flex Parser
- Geo IP Parser
- Lua Parsers

- Search Parser
- Wireless LAN Configuration

Feed Definitions File

This topic introduces the feed definitions file, which is available for editing in the Services Config view > Files tab.

One of the files available for editing in the Services Config view > Files tab is **feed-definitions.xml**, the feed definitions file.

feed-definitions.xml

You can define feeds in the **feed-definitions.xml** file. The Decoder uses an XML schema to define feed messages when it creates a binary .feed file from the feeds defined here.

For details on the feed definition language, refer to the NextGen System Administrator Guide.

Flex Parser

This topic introduces the flex parsers.

One of the files available for editing in the Services Config view > Files tab is **NwFlex.xml**, the flex parser.

NwFlex.xml

There are two kinds of Flex parsers:

- **Service identification based solely on port.** These are parsers that use only the source or destination ports to identify the session application type (service). These are the most basic and easiest to define.
- **Service identification based on a found token(s).** These parsers use tokens to identify the service type. This is also an easy way to expand which service types are identified. These are important when identifying non-internet standard applications. These parsers require that the protocol has a definable token that can uniquely identify the service type.

Five common parser operations are:

- Match Port and Identify Immediately
- Match Port and Delay Identification
- Match Token and Identify Immediately
- Match Multiple Tokens
- Match Token and Create Metadata

Detailed language information and samples are provided in this topic. This topic describes the XML schema used to define a FlexParse file. The SML node, attribute, and values referenced in descriptive text are **bold**. The root node of every file must be the **parsers** node. Under that node there can be any number of parser nodes. Each parser node defines a single parser.
A parser node can have an optional **declaration** node and any number of **match** nodes.

Topics

- Arithmetic Functions
- Common Parser Operations

- General Functions
- Logging Functions
- Nodes
- Payload Functions
- Regex
- String Functions

Arithmetic Functions

This topic defines language for the flex parser arithmetic functions.

This topic defines language for the flex parser arithmetic functions. All numbers are 64-bit unsigned values and subject to both underflow and overflow, depending on the operation.

Language Definition

The following table provides language definitions.

Node Name	Attribute Name	Description
and		Performs bitwise AND between two numbers.
	name	Variable to AND result into.
	value	Number to AND into result.
or		Performs bitwise OR between two numbers.
	name	Variable to OR result into.
	value	Number to OR into result.
increment		Performs ADDITION of two numbers.
	name	Variable containing the initial value AND to receive ADDITION results.
	value	Number to ADD to initial value.
decrement		Performs SUBTRACTION of two numbers.
	name	Variable containing initial value AND to receive SUBTRACTION results.
	value	Number to SUBTRACT from initial value.
divide		Performs DIVISION of two numbers.

Node Name	Attribute Name	Description
	name	Variable containing the initial value AND to receive DIVISION results.
	value	Number by which to divide the initial value. Division by zero generates an error and stops any further processing of the current session by this parser.
modulo		Performs MODULO of two numbers.
	name	Variable containing the initial value AND to receive MODULO results.
	value	Number by which to divide the initial value. Division by zero generates an error and stops any further processing of the current session by this parser.
multiply		Performs MULTIPLICATION of two numbers.
	name	Variable containing the initial value AND to receive MULTIPLICATION results.
	value	Number by which to MULTIPLY the initial value.
shiftleft		Performs a binary shift left.
	name	Variable containing the initial value AND to receive shift results.
	value	Number of bits to shift by.
shiftright		Performs a binary shift right.
	name	Variable containing the inital value AND to receive shift results.
	value	Number of bits to shift by.

Common Parser Operations

This topic provides some examples of common parser operations.

This topic includes five common parser operations.

Match Port and Identify Immediately

```
<?xml version="1.0" encoding="utf-8"?>
<parsers
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation="parsers.xsd">
          <parser name="CustApp" desc="Acme Custom App" service="45324">
               <declaration>
                    <port name="port" value="45324" />
               <declaration>
               </match name="port">
                    <identify />
               </match>
          </parser>
</parsers>
```

Match Port and Delay Identification

```
<?xml version="1.0" encoding="utf-8"?>
<parsers
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="parsers.xsd">
     <parser name="MSRPC" desc="Microsoft RPC protocol" service=135">
          <declaration>
               <port name="port" value="135" />
               <number name="state" scope="session" />
               <session name="end" value="end" />
          </declaration>
          <match name="port">
```

```
            <assign name="state" value="1" />
        </match>
        <match name="end">
            <if name="state" equal="1" />
                <identify />
            </if>
        </match>
    </parser>
</parsers>
```

Match Token and Identify Immediately

```
<?xml version="1.0" encoding="utf-8?>
<parsers
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="parsers.xsd">
    <parser name="RDP" desc="Remote Desktop Protocol" service="3389">
        <declaration>
            <token name="signature" value="Cookie: mstshash=" />
        </declaration>
        <match name="signature">
            <identify />
        </match>
    </parser>
</parsers>
```

Match Multiple Tokens

```
<?xml version="1.0" encoding="utf-8"?>
<parsers
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="parsers.xsd">
    <parser name="MyServiceMultiToken" desc="Multiple Tokens"
    service="333">
        <declaration>
```

```
            <number name="state" scope="stream" />
            <token name="user" value="USER " />
            <token name="pass" value="PASS " />
            <session name="session" value="end" />
        </declaration>
        <match name="user">
            <or name="state" value="1" />
        </match>
        <match name="pass">
            <or name="state" value="2" />
        </match>
        <match name="session">
            <if name="state" equal="3">
                <identify />
            </if>
        </match>
    </parser>
</parsers>
```

Match Token and Create Metadata

```
<?xml version="1.0" encoding="utf-8"?>
<parsers xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="parsers.xsd">
    <parser name="SHELL" desc="Command Shell Identification">
        <declaration>
            <token name="cmd.exe" value=" (C) Copyright 1985-2001
            Microsoft Corp" options="linestart" />
            <meta name="client" key="client" format="Text" />
        </declaration>
        <match name="cmd.exe"
            <register  name="client" value="MS Command Shell" />
        </match>
    </parser>
</parsers>
```

General Functions

This topic defines language for the flex parser general functions.

General Functions Language Definition

Node Name	Attribute Name	Description
apptype		Gets the currently defined service type for the current session.
	name	A number variable to receive the current service type.
identify		Marks the session with the parser's service type if the service type has not already been identified.
assign		Assigns a value to a variable.
	name	The unique identifier assigned to the item in the declaration section.
	value	Optional. If specified, the action defined in the match is only applied when the declaration matches the given value.
getmeta		Retrieves the value of meta that generated a callback. This function will return empty results (0, zero length string) if called when there was no meta callback.
	name	The variable to receive the value of the meta that generated the callback.OK.
gettoken		Returns the current matched token.
	name	A string variable to receive the current matched token. If there is no current token, the variable is assigned an empty string.
end		This terminates the execution of the current **match** section.

Node Name	Attribute Name	Description
if		Compares two values. If the comparison is true, executes any sub-actions. Comparisons can be **number** or **string** types, as long as both values are the same type.
	name	The unique variable identifier assigned to the item in the **declaration** section.
	equal notequal less lessequal greater great-erequal and or	The operation value to compare. If true, any sub-actions are executed.
register		Adds metadata to the session.
	name	The unique identifier of a meta variable to be created, as defined in the **declaration** section.
	value	The value of the metadata to be created.
while		Compares two values and executes any sub-actions if the comparison is true. Comparisons can be **number** or **string** types, as long as both values are the same type.
	name	The unique variable identifier assigned to the item in the declaration section.

Node Name	Attribute Name	Description
	equal notequal less lessequal greater great-erequal and or	Specifies the operation value to compare. If true, any sub-action is executed. The **and** and **or** attributes signify bitwise operations and can only be applied to **number** variables.
call		Execute the specified **match** element. This can be any match element defined in the same flex parser regardless of how it was declared.
	value	The name of the match element, or a string variable containing the name of a match element. • If the match element name is specified, the parser will not load if the named matched element doesn't exist. • If a string variable is specified, the **call** element will execute any child elements that it may have if the string value resolves to a match element after executing the named match element. • If no **match** element can be found matching the string value, no action is taken.

Logging Functions

This topic defines language for the flex parser logging functions.

Logging functions provide a means for a flex parser to write to the system log. Logging functions can be extremely useful when creating a new flex parser, but should be kept to an absolute minimum when a flex parser is deployed to a production system.

Language Definition

Node Name	Attribute Name	Description
failure		Logs a message to the system log with the log level **Failure.**
	value	A string to include as the log message.
warning		Logs a message to the system log with the log level **Warning.**
	value	A string to include as the log message.
info		Logs a message to the system log with the log level **Info**.
	value	A string to include as the log message.
debug		Logs a message to the system log with the log level **Debug.**
	value	A string to include as the log message.

Nodes

This topic defines language for the flex parser nodes.

Nodes Language Definition

Node Name	Attribute Name	Description
parsers		The root node in each definition file.
	xmins:xsi	Defines the namespace to use for the schema inclusion. This attribute is not required; however, language definition is not possible without it. This node must have the following value: http://www.w3.org/2001/XMLSchema-instance
	xsi:noNamespaceSchemaLocation	Defines the XSD schema validation file used to validate the language definition. This attribute is not required; however, language definition is not possible without it. This node must have the following value: parsers.xsd
parser		The node that defines a single parser definition. This node must be directly under the **parsers** node. There can be more than one per file.

Node Name	Attribute Name	Description
	name	The name that uniquely identifies the parser. This name should be short and succinct. This is used by the system to allow enabling and disabling. It should contain only the letters [a-z] and [A-Z].
	desc	This node provides a friendly description of what the parser does.
	service	This is the unique number assigned to the session when identified.
declaration		The node that delineates the definition. Each of these definitions can have an associated **match** entry.
token		Specifies a definition for identifying a token somewhere in the session protocol. This defines a **match** callback when the specified tokens are encountered in a session payload. The **read** position is set to the byte immediately following the matched token.
	name	This is a unique identifier for the declaration.
	value	This is the exact token value to be identified.
	options	Options specify that the token should start on a new line or at an end of a line (**linestart** or **linestop**).

Node Name	Attribute Name	Description
meta-call-back		Registers a callback for the flex parser whenever meta of a specific format is created. This can be further qualified to generate callbacks only for sessions that have been identified as a specific apptype (e.g. 80 for http).
	name	Name of the match element to be executed when a callback occurs. (String)
	key	Name of the meta key that generates callbacks. (String)
	format	The data type of the meta key that will generate the meta.
	apptype	The meta callback is only generated if the session being parsed has been identified with the specified apptype. (Unsigned Integer, Optional)
number		Defines a numeric variable that can be referenced elsewhere within the parser definition. All numeric values are 64-bit unsigned values.
	name	This is a unique identifier for the declaration.

Node Name	Attribute Name	Description
	scope (optional)	Specifies when to reset the variable. This can either be for each side of a two-sided session or only after a new session is detected. The possible values are **global, constant, stream,** and **session** (default).
string		Defines a numeric variable that can be referenced elsewhere within the parser definition.
	name	This is a unique identifier for the declaration.
	scope (optional)	Specifies when to reset the variable. This can either be for each side of a two-sided session or only after a new session is detected. The possible values are **global, constant, stream,** and **session** (default).
port		Defines a **match** callback when a session is encountered using the specified port. The **read** position is set to the first byte of the first stream (client) in the session.
	name	This is a unique identifier for the declaration.
	value	This is the port number to identify.
session		Defines a **match** callback for session begin/end events. These events only occur if a token for the parser is encountered in the session.

Node Name	Attribute Name	Description
	name	This is a unique identifier for the declaration.
	value	Specifies that processing takes place at the beginning of a new session or at the end of a session (**begin** or **end**).
stream		Defines a **match** callback for stream begin/end events. These events only occur if a token for the parser is encountered in the stream.
	name	This is a unique identifier for the declaration
	value	Specifies that processing takes place at the beginning or at the end of a stream (**begin** or **end**).
function		Defines a **match** section that can be used as a generic function. No callbacks are associated with this declaration.
	name	This is a unique identifier for the declaration.
meta		Defines the type of data that the parser will create.
	key	Specifies the key name. The key needs to be 1-16 bytes in size.
	format	Specifies the variant type (e.g. **Text, IPv4, UInt32**). Refer to the SDK documentation for a full list.

Node Name	Attribute Name	Description
pattern		Defines a regular expression variable for use by the **regex** function
	name	This is a unique identifier for the declaration.
	scope (optional)	Specifies when to reset the variable. This can be for each side of a two-sided session or only after a new session is detected. Possible values are **global, constant, stream,** and **session** (default).
	value (optional)	Specifies a regular expression to assign to the pattern variable. This attribute is only valid when the **scope attribute** is set to **constant**.
match		The possible entries for taking an action once a match criterion has been found for a declaration. These nodes can be nested to provide deeper logic. There are several categories of execution elements (functions) that can appear as children of a match element: • General • Arithmetic • String • Payload

Payload Functions

This topic defines language for the flex parser payload functions.

These functions operate on a **read** position, set at the beginning of a **match** element.

Language Definition

Node Name	Attribute Name	Description
find		Searches the stream payload starting at the read position for a provided string value. If the value is found, the offset from the read position is returned. Any child elements will then execute. If not found, any child elements will not execute.
	name	A **number** variable to receive the offset from the **read** position where the match begins.
	value	A string to find.
	length (optional)	A limit to the length of the payload to be searched. If a limit is not provided, the remainder of the payload is searched. It is recommended to always use the smallest value possible here in order to reduce the effect on performance.
install-decoder		To enable tokens to match on payload data that may be fragmented or otherwise encoded. A scan decoder can be installed to pre-process a section of the payload before it is scanned for tokens. An example would be an HTTP response that uses the chunked transfer encoding with gzip content encoding. By parsing the HTTP header, the necessary type, offset, and length parameters can all be set, after which the HTTP response payload would appear to the token scanning as if neither encoding had been applied. However, this incurs significant overhead.

Node Name	Attribute Name	Description
	type	The type of decoder to install. Valid options are: gzip, deflate, chunked, chunked-gzip, chunked-deflate.
	offset	Offset from the current read position to begin decoding.
	length	The maximum payload length to decode.
isdecoding		Tests whether an installed decoder is currently active. If so, any children of this function will execute. This function has no parameters.
move		Moves the **read** position forward in the current stream by a specified number of bytes. If there is sufficient data in the stream, the **read** position is updated and any child elements will then execute. If not found, the **read** position remains unchanged and any child elements will not execute.
	value	The number of bytes to move the **read** position.
	direction (optional)	The direction to move the current read position. Can be **forward** (default) or **reverse.**
packetid		Returns the id of the packet for the current read position. It is possible for the result to be 0, which indicates that the packet id could not be determined.
	name	A number variable to receive the current packet id.
payload-position		Returns the current read position. This is a zero based index into the stream payload.
	name	A number variable to receive the current read position.

Node Name	Attribute Name	Description
read		Reads a specified number of bytes starting at the **read** position into a variable. If there is sufficient data in the stream, the **read** position is updated, the data read assigned, and any child elements will then execute. If not found, the **read** position remains unchanged and any child elements will not execute.
	name	The name of a **string** or **number** variable to receive stream data. If a **number** variable is provided, the bytes read are interpreted as a single unsigned numeric value.
	length	The number of bytes to read from a stream.
	endianess (optional)	The byte ordering to use when reading into a number variable. Can be **big** (default) or **little**. The attribute is invalid when reading into a **string** variable.

Regex

This topic defines language for the flex parser regex node.

Regex searches the stream payload starting at the **read** position for matches to a provided regular expression. If matches are found, the offset from the **read** position and, optionally the matched string, is returned. Any child elements execute. If no matches are found, child elements do not execute.

Language Definition

Attribute Name	Description
name	A **number** variable to receive the offset from the **read** position where the match begins.
value	A regular expression to find.
length (optional)	A limit to the length of the payload to be searched. If a limit is not provided, the remainder of the payload is searched. It is recommended to always use the smallest value possible here in order to reduce the effect on performance.
found (optional)	The name of a **string** variable to receive a matched string.

String Functions

This topic provides language definitions for the flex parser string functions.

String Functions Language Definition

Node Name	Attribute Name	Description
append		Attaches a number or string to the end of a **string** variable.
	name	The unique identifier of a string variable to which the specified value is to be attached.
	value	A number or string to attach.
find		Searches a string for a provided string value. If it is found, the position is returned and any child elements will execute. Otherwise, child elements will not execute.
	name	A **number** variable to receive the zero-based position, where the provided value string was found in the **in** string.
	value	A string to find.
	in	A string to search.
	length (optional)	A limit to the length of the **in** string to be searched. If a limit is not provided, all of **in** will be searched.
length		Assigns the length of a string to a **number** variable.
	name	A **number** variable to receive the length of the specified string.

Node Name	Attribute Name	Description
	value	A string value whose length is to be determined.
regex		Searches a string for matches to the provided regular expression. If a match is found, the position and, optionally, the matching string is returned. Any child elements will then execute. If not found, any child elements will not execute. Regular expression operations can adversely affect system performance.
	name	A number variable to receive the zero-based position, where the provided regular expression matched in the in string.
	value	A regular expression to be searched for.
	in	A string to search.
	length (optional)	A limit to the length of the **in** string to be searched. If a limit is not provided, all of **in** will be searched.
	found (optional)	The name of a string variable to receive the matched string.
substring		At least one of the optional attributes **from** and **length** must be specified.
	name	The unique identifier of a string variable to receive the extracted value.
	value	A string value from which to extract a substring.
	from (optional)	The zero-based position from which to begin the substring. If not specified, it defaults to zero.

Node Name	Attribute Name	Description
	length (optional)	The number of characters to extract. If not specified, it defaults to the remaining length of the string.
tolower		Converts a string to all lowercase letters.
	name	The name of a **string** variable to process.
toupper		Converts a string to all uppercase letters.
	name	The name of a **string** variable to process.
urldecode		Decode a string containing url-encoded characters.
	name	A string variable to receive the decoded string.
	value	A url-encoded string to decode.
base64decode		Decodes a base-64 encoded string.
	name	A string variable to receive the decoded string.
	value	A url-encoded string to decode.
uudecode		Decode a uuencoded string.
	name	A string variable to receive the decoded string.
	value	A uuencoded string. The header and trailing lines should not be included.
quotedprintabledecode		Decode a Quoted-printable encoded string.
	name	A string variable to receive the decoded string.
	value	A quoted-printable encoded string.
convert-ebcdic		Convert an EBCDIC string to its ASCII equivalent.
	name	A string variable to receive the decoded string.

Node Name	Attribute Name	Description
	value	A url-encoded string to decode.

Geo IP Parser

This topic introduces the Geo IP parser for Decoders.

One of the files available for editing in the Services Config view > Files tab is **GeoPrivate.ipl**, the Geo IP parser.

GeoPrivate.ipl

The Geo IP parser is a fixed parser that takes IP addresses and converts them to geographical locations. The locations are displayed through the Google Earth display.

The geolocation metadata in **GeoPrivate.ipl**, are added for both **ip.src** and **ip.dst**. The parser uses two external data files, **GeoCity.dat** and **GeoCountry.dat**, which are both stored in the application directory. There are up to eight metadata for each IP address as listed in the table below.

Metadata	Description
city.dst	Destination City
city.src	Source City
country.dst	Destination Country
country.src	Source Country
latdec.dst	Destination Decimal Latitude
latdec.src	Source Decimal Latitude
longdec.dst	Destination Decimal Longitude
longdec.src	Source Decimal Longitude

Lua Parsers

This topic introduces the Lua parsers.

One of the files available for editing in the Services Config view > Files tab is **NwLua.xml**, the Lua parser.

List of Lua Parsers

There are a number of Lua parsers available from Live. See SecurCare Online (SCOL) for:

- A complete list of these parsers
- Their interdependencies
- The Flex parsers that are subsumed by each Lua parser.

Five common parser operations are:

- Match Port and Identify Immediately
- Match Port and Delay Identification
- Match Token and Identify Immediately
- Match Multiple Tokens
- Match Token and Create Metadata

Search Parser

This topic explains how to configure a custom parser used on a Decoder to generate metadata by scanning for pre -defined keywords and regular expressions in the Services Config view > Files tab.

One of the files available for editing in the Services Config view > Files tab is **search.ini**, the search parser.

search.ini

The Search Parser is a custom parser used to generate metadata by scanning for pre -defined keywords and regular expressions. The parser searches the payload of a reconstructed session for string matches and can execute a regular expression search. You can configure the parser by editing the search.ini file.

Caution: The search parser can have a significant impact on system performance. It is important that both the search mechanism and the data to which it is applied to be well understood before creating new search definitions and enabling the search parser.

The search definition is used across all protocols. There are three basic search methods:

- Keyword: Search a stream for a specific set of words
- Pattern: Search a stream for a regular expression match
- Keyword + Pattern: Search a stream for a regular expression if it contains any of a given set of keywords.

For a detailed explanation, see Search Parser in the search.ini Search String Syntax.

search.ini Search String Syntax

This topic introduces search methods and syntax for use in Search parser.

The Search parser uses three basic search methods:

- Keyword: Search a stream for a specific set of words.
- Pattern: Search a stream for a regular expression match.
- Keyword+Pattern: Search a stream for a regular expression if it contains any of a given set of key words.

Syntax

```
Maxrecon=<max_size>Maxsearch=<max_ssearch_length>MatchLimit=<max_
matches_per_stream
Search Name
Services=<service_id_list>Keywords=<keyword_list>|Pat-
tern=<expression>Case=0|1
Proximity=<number_of_bytes>Recon=0|1
Raw=0|1
```

Parameters

Parameters used in this command:

Parameter	Description
autocheck	Automatically fixes all problems without prompting
header Only	Check/display the header of each file
chatty	Displays a hex dump of every object in the file (huge amount of data)
dump#-#	Indicates a zero-based object or range of objects in the file to output in hex to the console

Example

Following is an example of the command:

To check all NetWitness database files located in the Collection named Default. If any problems are found, the command will describe the problem and ask if you would like to fix it.

```
dbcheck C:\Documents and Settings\User\My Documents\NetWitness\ Investigations\Default\*.nw*
```

Wireless LAN Configuration

This topic introduces the wireless LAN configuration file for Decoders, which is in the Services Config view > Files tab.

wlan-config.xml

One of the files available for editing in the Services Config view > Files tab is **wlan-config.xml**, the wireless LAN configuration file.

It controls the 802.11 parsers. Its chief purpose is to control decryption of raw 802.11 frames captured by the Decoder. This file is optional. If decryption of 802.11 traffic is not desired, there is no need to create the file.

There are five link-level parsers related to wireless LAN packet capture:

- IEEE 802.11 parser (data frames and beacons only)
- Radiotap w/ 802.11 header
- Absolute Value Systems (AVS) w/ 802.11 header
- Prism II w/ 802.11 header
- CACE's "Per Packet Information" (PPI) w/ 802.11 header

The 802.11 wireless parsers introduced in 9.8 all share a single configuration file. This wlan - config.xml file is used to define any wireless access points the user may have in the network, and its primary purpose is to control decryption. The BSSID of the access point and the SSID that it's authoritative for is added to this file as well as all of the active default keys used by the access point.

A comprehensive discussion is provided in the chapter on Wireless Packet Capture in the NextGen System Administrator Guide.

Services Config View - General Tab

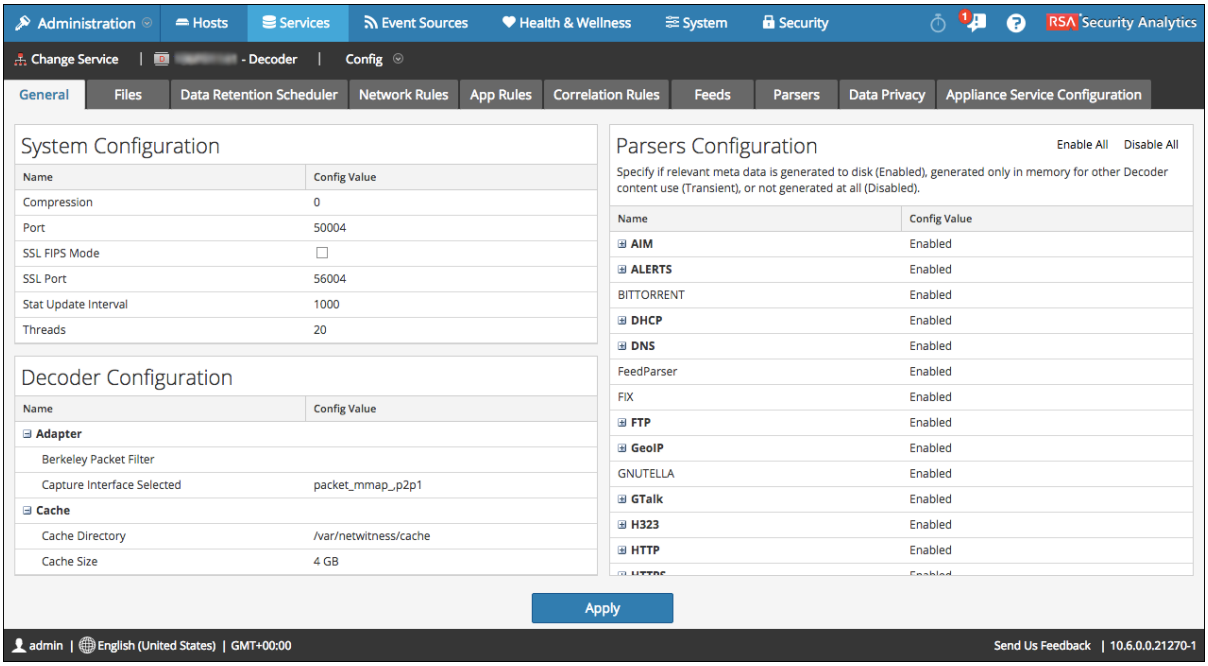
This topic introduces features of the Services Config view > General tab for Decoders and Log Decoders.

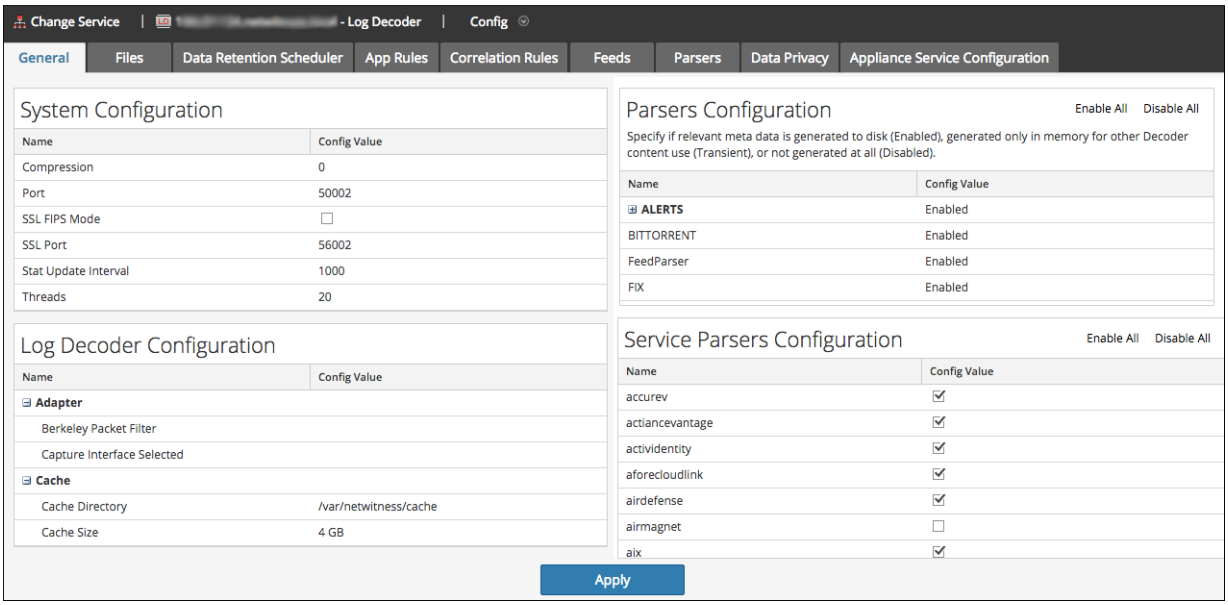
The General tab for a Decoder in the Services Config view provides a way to manage basic service configuration, configure data capture, and select the parsers that are applied to the captured data.

Settings that set up and tune data capture include:

- Adapter selection
- Cache specification
- Capture autostart and other capture parameters that affect cache, sessions, and timeouts
- Database file sizes
- Location of the hash directory

The first figure is an example of the General tab for a Decoder. The second is the General tab for a Log Decoder.





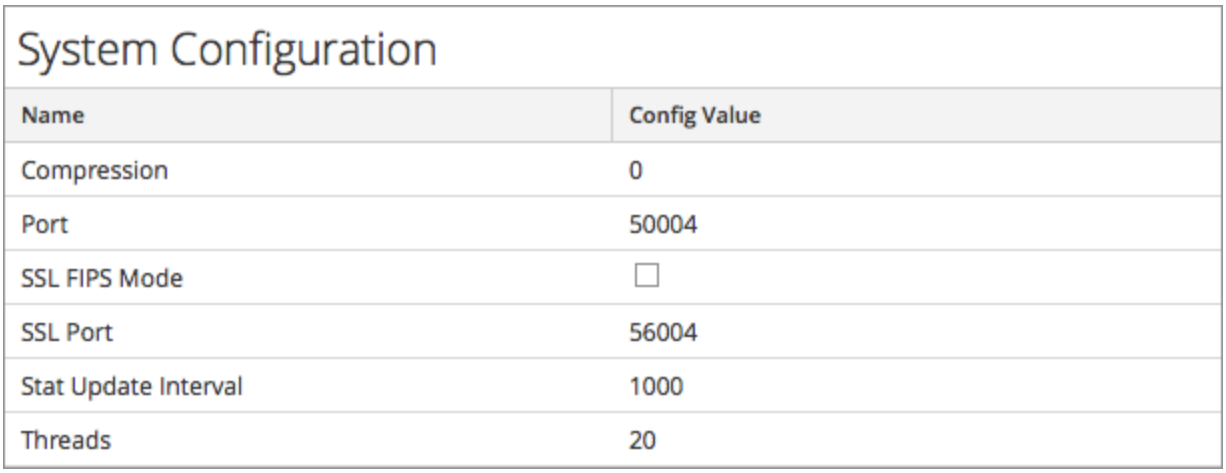
Features

These are the four major sections in the General tab for Decoders and Log Decoders:

- System Configuration
- Decoder Configuration
- Parsers Configuration
- Service Parsers Configuration (Log Decoders only)

System Configuration

The System Configuration section manages service configuration for a Decoder. When a service is first added, default values are in effect. You can edit these values to tune performance.



System Configuration

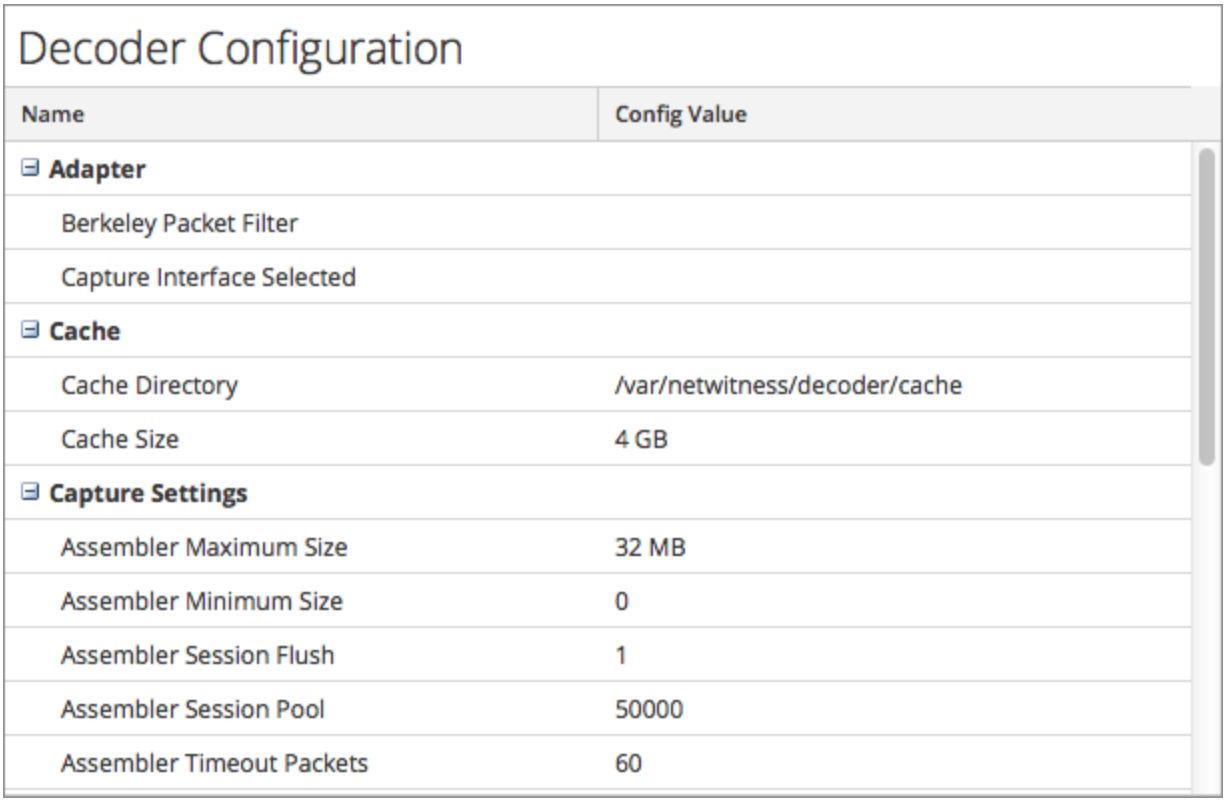
Name	Config Value
Compression	0
Port	50004
SSL FIPS Mode	☐
SSL Port	56004
Stat Update Interval	1000
Threads	20

The System Configuration section has these parameters.

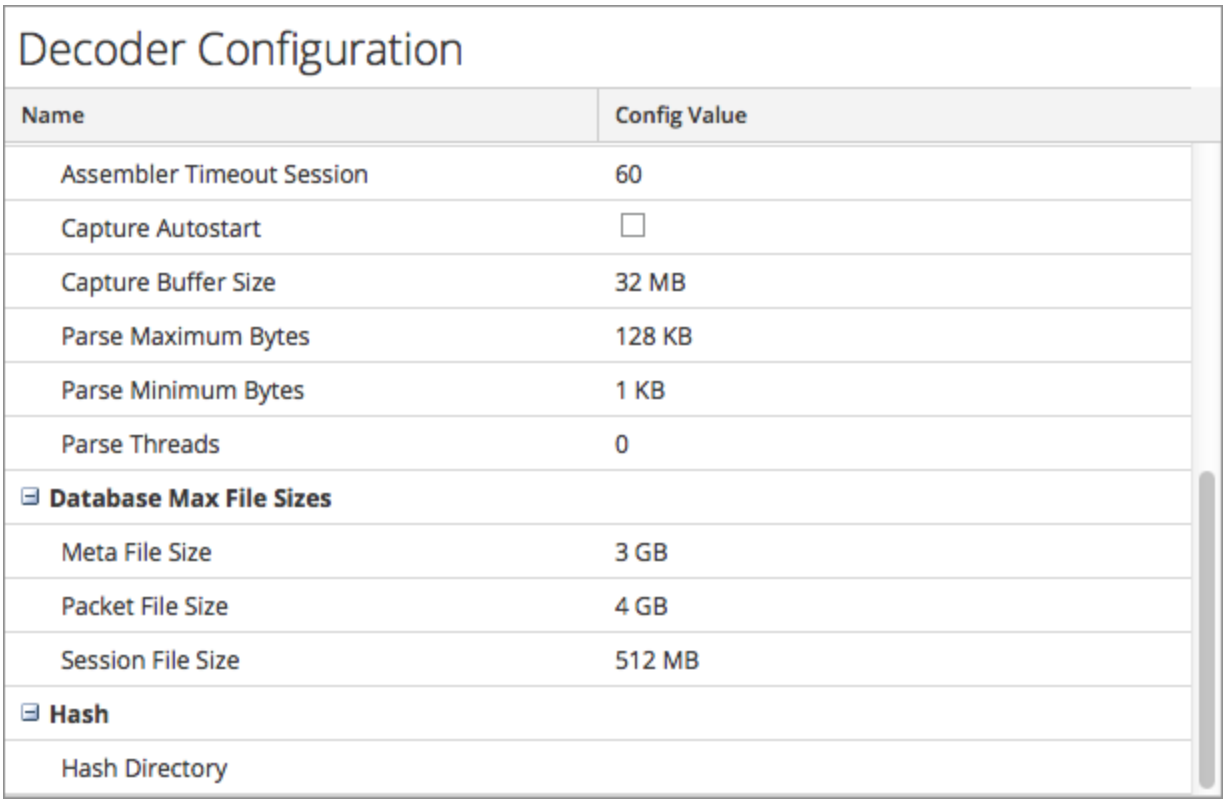
Parameter	Description
Compression	The minimum number of bytes that must be transmitted per response before compression. A setting of 0 disables compression. The default value is **0**. A change in value is effective immediately for all subsequent connections.
Port	Determines the port used by the service. **Note:** If you change the port number, ensure that you restart the service.
SSL FIPS mode	If enabled, all the data transferred in the network will be encrypted using SSL.
SSL Port	Indicates the port used for encrypting using SSL.
Stat Update Interval	The number of milliseconds between statistic updates on the system. Lower numbers cause more frequent updates and can slow down other processes. The default value is **1000**. A change in value is effective immediately.
Threads	The number of threads in the thread pool to handle incoming requests. A setting of **0** lets the system decide. A change takes effect on service restart.

Decoder Configuration

The Decoder Configuration section provides a way to view and edit service configuration parameters for a Decoder or Log Decoder. When a service is first added, default values are in effect. You can edit these values to manage traffic capture.



Scrolling to the bottom of the section reveals these additional Decoder Configuration parameters.



Adapter

Adapter parameters configure the network interface for capture. The table below describes the Decoder Adapter settings. The default network adapters available are set at installation. Consult your System Administrator for more information.

Adapter Parameter	Description
Berkley Packet Filter	Berkeley Packet Filters (BPF) are applied to the packet stream before the packets are copied to the Decoder adapter for analysis. This allows unwanted traffic to be efficiently discarded. However, any packets discarded are not accounted for in any Decoder statistics (capture rate, packets dropped, and packets filtered and total packets).

Adapter Parameter	Description
Capture Interface Selected	Select an adapter through which the Decoder captures packets. For the lower speed internal capture interface, use the **packet_mmap_,7,eth1** adapter, which corresponds to the monitor port located on the motherboard. There are six additional capture ports: • packet_mmap_,1,lo (bpf) • packet_mmap_,2,eth2 (bpf) • packet_mmap_,3,eth3 (bpf) • packet_mmap_,4,eth4 (bpf) • packet_mmap_,5,eth5 (bpf) • packet_mmap_,8,ALL (bpf) There are three wireless capture services available: • packet_netmon_ (Microsoft Netmon) • packet_mac80211_ (Linux mac80211) • packet_airport_ (Mac OS X AirPort)

The Decoder also supports system-level packet filtering defined using **tcpdump/libpcap** syntax. Specifying a Libpcap filter can efficiently reduce packet volume based on Layer 2 - Layer 4 attributes. A Libpcap filter is appropriate for use when a Decoder is receiving a traffic volume that is placing a load against the physical resources of the platform. In this scenario, the Decoder may consistently drop packets and have a large number of capture pages available (/decoder/stats/capture.pagefree is high).
The following is an example of a libpcap filter to keep only packets which do not have both source and destination addresses in the 10.21.0.0/16 subnet.
not (src net 10.21.0.0/16 and dst net 10.21.0.0/16)
For a full reference of the Libpcap filter syntax, see the main pages for:

- tcpdump (http://www.tcpdump.org/tcpdump_man.html).
- pcap-filter (http://www.unix.com/man-page/FreeBSD/7/pcap-filter/).

Cache

Cache parameters configure the cache directory and size for session cache files. The following table describes the cache settings.

Cache Parameter	Description
Cache Directory	The directory where session cache files are stored. The default value is **/var/netwitness/decoder/cache**. Change takes effect immediately.
Cache Size	The maximum size, in Megabytes (MB), that all files in the cache directory can attain before the oldest files are deleted. Once the threshold is reached, the cache size is reduced by 10%. The default value is **4 GB**. Change takes effect immediately.

Capture Settings

The Capture Settings section provides a way to configure operational capture settings.

Note: By default, no capture rules are defined when you first install Security Analytics. Unless there are rules specified, the packets are not filtered. You can define capture rules before beginning to capture data (see Configure Network Rules, Configure Application Rules, and Configure Correlation Rules).

This table describes the capture settings.

Capture Settings Parameter	Description
Assembler Maximum Size	Specifies the maximum size in bytes that a session's packet data size can attain. The default value is **32 MB**. Change takes effect immediately.
Assembler Minimum Size	Specifies the minimum size in bytes that a session must have in order to generate metadata. A value of **0** means every session has metadata generated. The default value is **0.** Change takes effect immediately.

Capture Settings Parameter	Description
Assembler Session Flush	Specifies whether a session is removed from the assembler when the session's last chain is removed from the assembler. The default value is **1**. • **2** = if the first packet of a session times out of assembler, the session is removed from assembler after parsing is complete. Any subsequent packets for this session create a new session in assembler. • **1** = If the last chain of a session times out of assembler, the session is removed from assembler. Any subsequent packets for this session create a new session in assembler. • **0** = If the last chain of a session times out of assembler, the session is left in assembler until it times out. Any subsequent packets for this session are filtered Change takes effect on service restart.
Assembles Session Pool	Specifies the number of entries in the session pool. The default value is **350000**. Change takes effect on service restart.
Assembler Timeout Packets	Specifies the number of seconds before a packet or chain is timed out. T default value is **60**. Change takes effect immediately.
Assembler Timeout Session	Specifies the number of seconds before a session is timed out. Default value is **60**. Change takes effect immediately.
Capture Autostart	Specifies whether capture begins automatically each time Decoder is started. When checked, the value = yes. When unchecked, the value = no. The default value is **no**. Change takes effect immediately.
Capture Buffer Size	The capture memory buffer allocation in Megabytes. Default value is **64 MB**. Change takes effect on service restart.

Capture Settings Parameter	Description
Parse Maximum Bytes	The maximum number of bytes to scan a stream for additional tokens. When the first token is found, the stream is scanned up to the set number of bytes, but no further. A setting of **0** removes the early termination and the full stream is scanned regardless of size. The default value is **128 KB**. Change takes effect immediately.
Parse Minimum Bytes	The minimum number of bytes to scan a stream for the first token. If no token is found within the set number of bytes, scanning is terminated. A setting of **0** removes the early termination and the full stream is scanned regardless of size. The default value is **1 KB**. Change takes effect immediately.
Parse Threads	The number of parse threads to use for session parsing. A value of **0** means let the server decide. The default value is **0**. Change takes effect on service restart.

Database Max File Sizes

The Database Max File Sizes section controls the maximum file size for various databases. The following table describes the parameters.

File Size Parameter	Description
Meta File Size	The maximum size in Gigabytes, of the meta database files. The default value is **3 GB**. Change takes effect on service restart.
Packet File Size	The maximum size in Gigabytes, of the packet database files. The default value is **4 GB**. Change takes effect on service restart.
Session File Size	The maximum size in Megabytes, of the session database files. The default value is **256 MB**. Change takes effect on service restart.

To calculate the drive sizes and free space for the meta, packet, and/or session, for your environment, perform the following:

1. In the Security Analytics menu, select **Administration > Services**.
2. Select a service and select  **> View > Explore**.
   The Service Explore View is opened.
3. In the Node List select **database** and right-click and select **Properties**.
   The Properties panel is displayed.
4. In the properties panel, from the drop-down list, select **reconfig** .
5. In the **Parameters** field, enter update = false.
6. Click **Send**.
   The Response Output displays the drive sizes and free space for the Meta, packet and session.

Hash

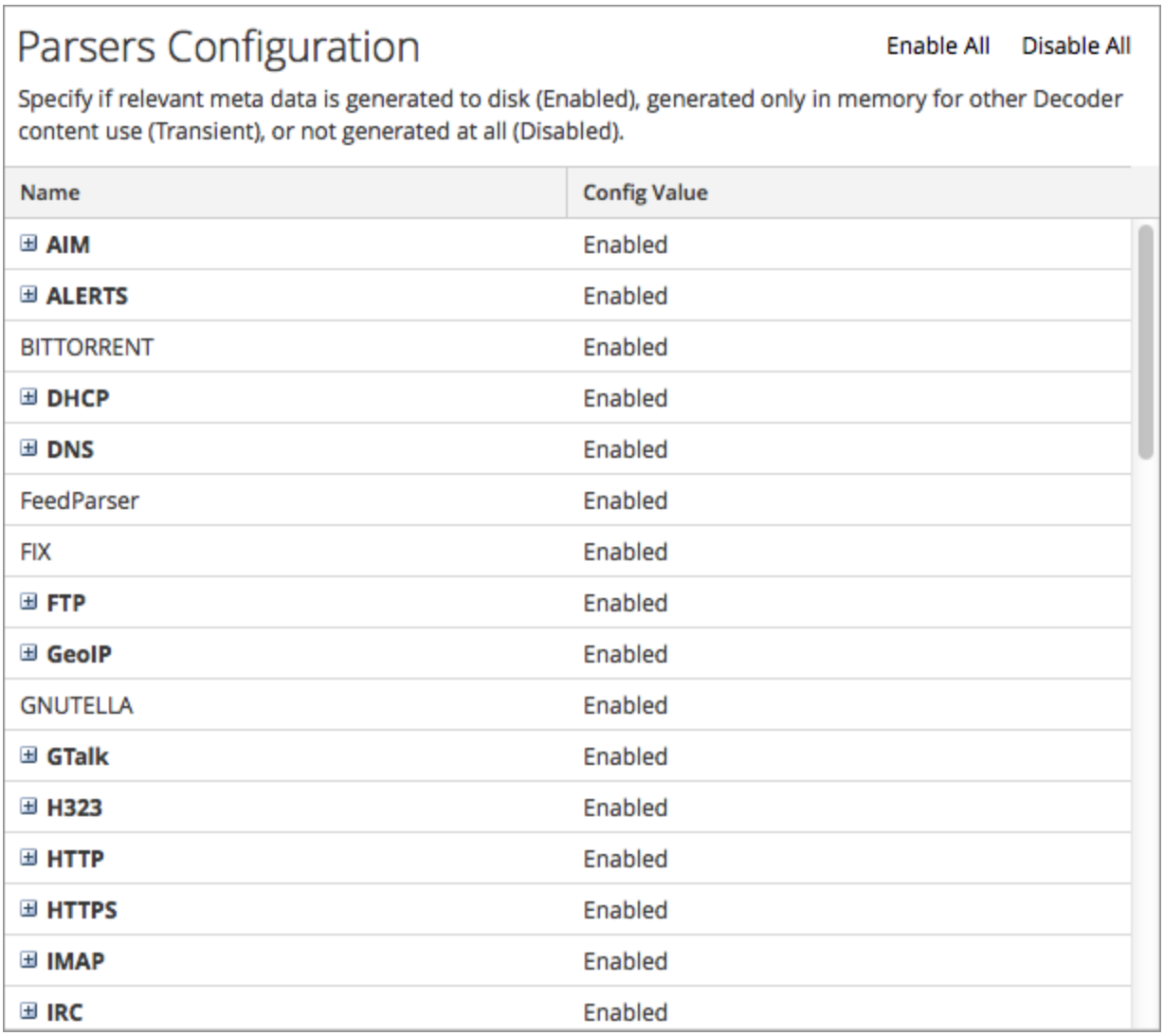
Controls data base file hashing options. There is a small performance penalty when hashing. The following table describes the hashing option.

Hash Parameter	Description
Hash Directory	The server directory where all hash files are written. If empty, each hash file is written to the same directory as the file being hashed. The default value is blank. Change takes effect on service restart.

Parsers Configuration

The Parsers Configuration panel provides a way to select parsers to use on the Decoder. Within some parsers, you can also configure the metadata that the parser creates.

Security Analytics has the ability to configure individual parsers that do not store generated metadata on disk (Transient option). This helps administrators to protect certain data and is usually done as part of a data privacy plan (see *Data Privacy Management*).



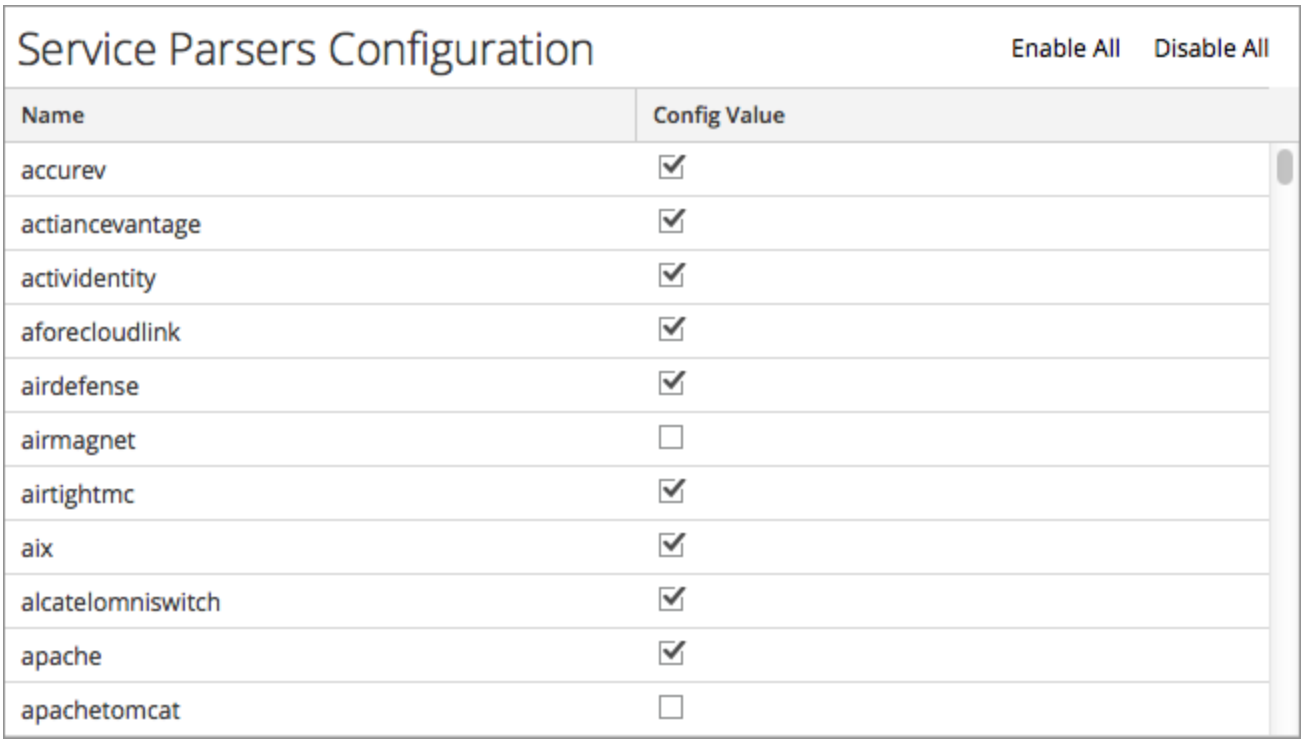
The following table describes the features of the Parsers Configuration section.

Feature	Description
Enable All **Disable All**	These options provide a way to quickly select either all parsers or no parsers.

Feature	Description
Name	The names of parsers available to the Decoder. A plus sign indicates that the metadata generated by the parser is configurable. Clicking the plus sign displays the metadata that the parser can create. In the example above, CMS_windows_ executable has three selectable metadata that the parser can create: alert.id, error, and filetype.
Config Value	A drop-down list changes the setting for the parser or metadata to **Enabled**, **Disabled**, or **Transient**. • When **Enabled**, the Decoder is using the parser to filter traffic. • When **Transient**, the Decoder is using the parser to filter traffic, and the generated metadata is not stored on disk. The transient metadata is available in memory to additional content (that is, parsers, feeds, and application rules) on that Decoder. • When **Disabled**, the Decoder is not using the parser. If the generated metadata for the parser is configurable, clicking the plus sign to expand the parser displays configurable meta keys and the same drop-down list selects the meta key the parser will create.

Additional Service Parsers Configuration for Log Decoder

The Service Parsers Configuration section provides a way to select Service parsers to use on the Log Decoder.


Service Parsers Configuration
Enable All
Disable All
Name
Config Value
accurev
actiancevantage
actividentity
aforecloudlink
airdefense
airmagnet
airtightmc
aix
alcatelomniswitch
apache
apachetomcat

Services Config View - Parser Mappings Tab

This topic provides a description of the configurable options for a Log Decoder in the Parsers Mappings tab.

In the Parsers Mappings tab, Administrators can configure log parser mappings for Log Decoder services. This feature is intended to track a subset of of Event Sources that is parsing against the wrong parser. The Parser Mappings tab must be enabled before you can see it in the Services Config view.

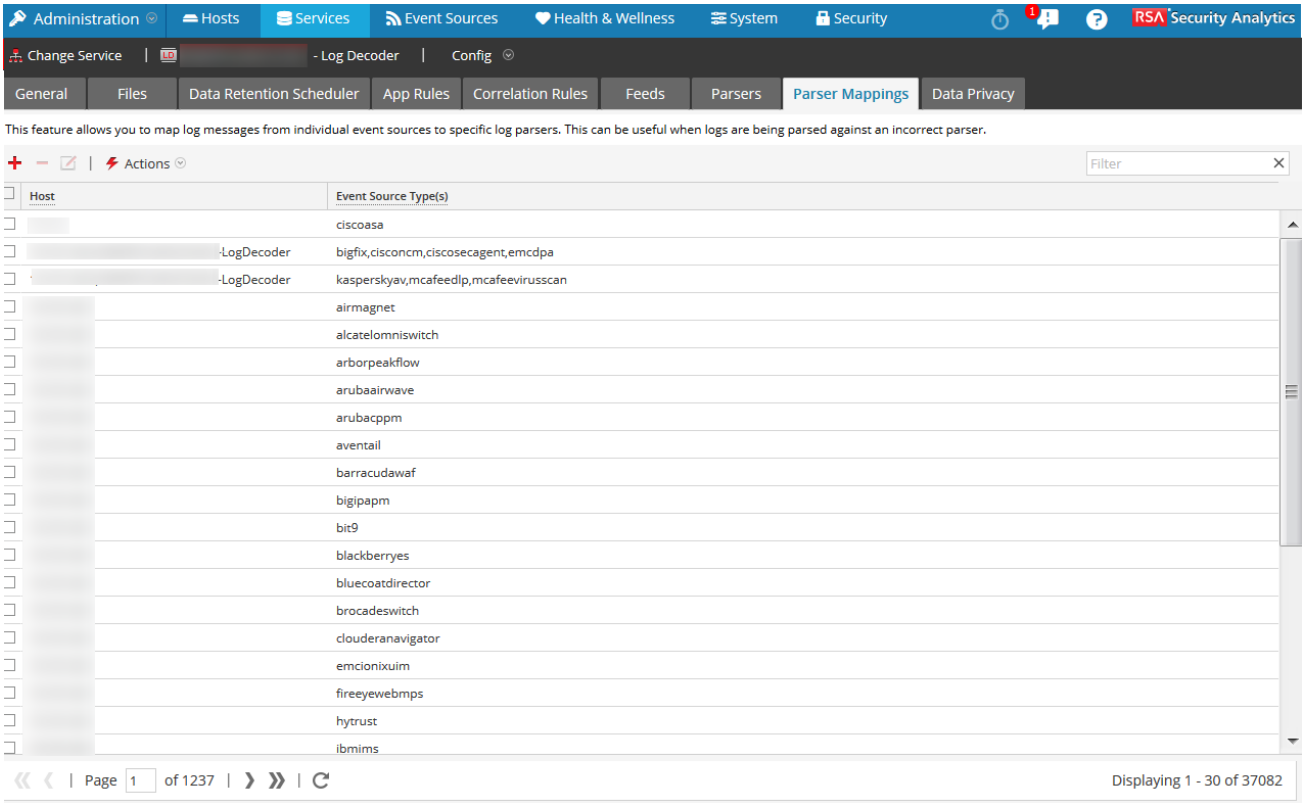
Procedures associated with the Parser Mappings tab are provided in Configure Parser Mappings.

To access this tab:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **Parsers Mapping** tab.

This is an example of the tab.



Features

The Parser Grid lists all parsers that are currently mapped on the Log Decoder. The Parser Tab Toolbar has options to work with parser mappings in the grid.

Parser Mappings Toolbar

The Parser Mappings Toolbar has options to work with parser mappings in the grid.

Feature	Description
	Add a parser mapping.
	Delete the selected parser mapping.
	Edit a parser mapping.
	Refresh the list of parser mappings.
	Display the Actions menu. • **Import** - Import a parser mapping to a file. • **Export** - Save a parser mapping to a file.

Parser Mappings Grid

The Parser Mappings grid lists all parsers that are currently mapped on the Log Decoder.

Parameter	Description
Host	Displays the IP address of the host.
Event Source	Displays the Event Sources that are parsing incorrectly.

Services Config View - Parsers Tab

This topic introduces the features of the Services Config View > Parsers tab.

In the Services Config view > Parsers tab, you can view deployed parsers on a Decoder, upload parsers, and delete deployed parsers. Parsers can be added and removed while a Decoder is running without affecting capture. Refer to Configure Feeds and Parsers for a general introduction to the use of parsers on Decoders.

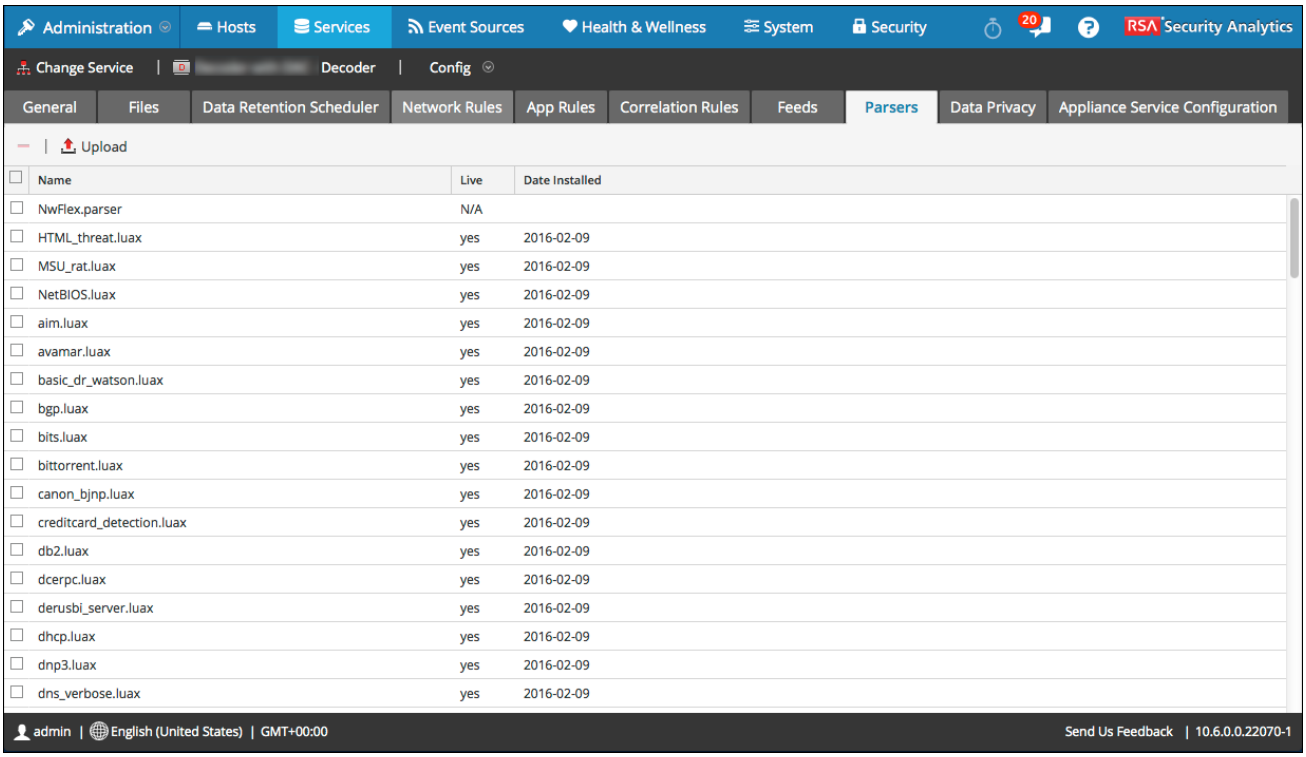
Note: Unless otherwise stated, any reference to Decoders applies to Log Decoders as well.

You can access this view by doing the following:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **Parsers** tab

This is an example of the Parsers tab.



Features

The Parser Grid lists all parsers that are currently deployed on the Decoder. The Parser Tab Toolbar has options to work with parsers in the grid.

Parsers Tab Toolbar

This is an example of the toolbar.

Feature	Description
Upload	Enables you to upload parsers to a Decoder or Log Decoder.
−	Requests confirmation that you want to delete the selected parsers. You can select **No** to cancel the deletion or select **Yes** to delete the selected parsers.

Parser Grid

The Parser Grid provides a listing of all currently deployed parsers for the Decoder.

Column	Description
Name	The name of the parser or the parser file.
Live	Indicates if the parser originated from Live. Possible values are **Yes**, **No**, or **N/A**. • **Yes** = Installed through Live • **No** = Installed through Security Analytics • **N/A** = The parser has no attributes file created by Security Analytics to track the installation date. The parser may have been installed manually, not through Security Analytics or Live. Manually installed feeds still function properly.
Date Installed	The date the parser was pushed to the service.

Services Config View - Rules Tabs

The Rules tabs in the Services Config view enable you to define and manage capture rules. Each type of rule has a grid with slightly different columns and different parameters in the Rule Editor dialog. Application and correlation rules apply to both Decoders and Log Decoders. Network rules apply only to packet Decoders.

Step 4. Configure Decoder Rules provides additional information.

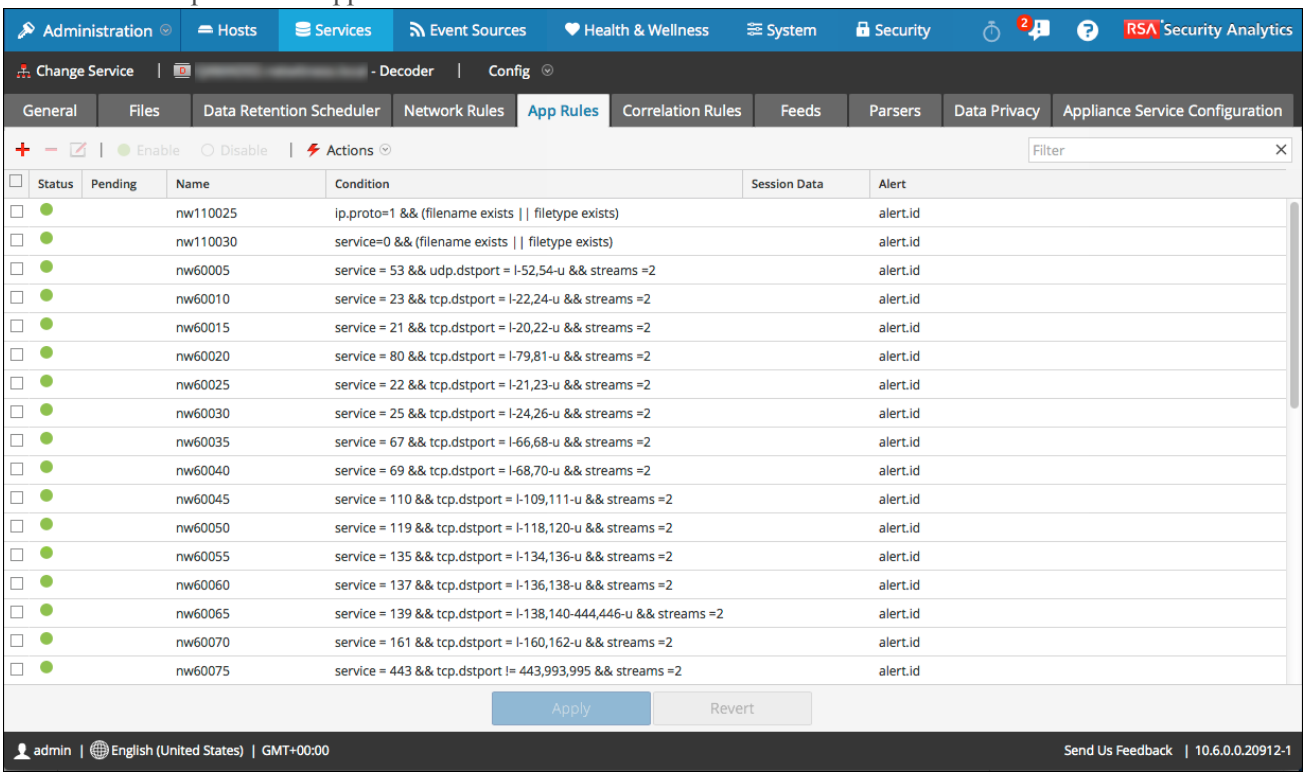
You can display this view by doing the following:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and **>View > Config**.

   The Config view for the selected service is displayed.
3. Click one of the rules tabs: **Network Rules**, **App Rules**, or **Correlation Rules**.

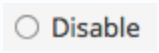
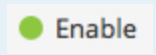
   The selected rules tab is displayed.

This is an example of the App Rules tab.



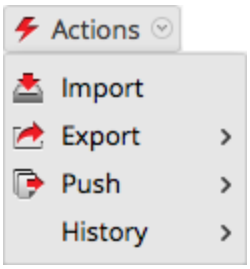
Rules Tab Toolbar

The toolbar is the same for all Config view > Rules tabs.

Feature	Description
Actions	Displays the **Actions** menu.
	Adds a new rule to a service.
	Deletes a rule from a service.
	Allows rule modification.
	Disables a rule (without deleting the rule).
	Enables (reactivates) a rule.
Filter	The input field for a search string. Security Analytics filters the rules dynamically as you type a search string. Clicking **x** clears the input field, restoring the unfiltered view.
Apply	Saves the changes made to rules and applies the configured rules to a service. Until you apply changes, it is possible to reload the rules as they were before current modifications.
Revert	Discards unsaved changes to the grid and reverts to the unedited rules.

Rules Actions Menu

The Actions menu has options that help to manage sets of rules.



Option	Description
Import	Imports a set of rules into the user interface so that it can be applied to a service. You can edit the rules before applying.
Export	Saves selected rules or all rules to an .nwr file on the client machine.

Push	Allows rules to be applied to other services (Decoders or Log Decoders) or Decoders belonging to a service group. When pushing, the rules can either be merged (update existing rules and append new ones) or replaced.
	• **Push > All.** Pushes all rules to other services. All rules on the target services are removed and replaced with all of the rules on the source service.
	• **Push > Selection.** Pushes selected rules to other services. You have two options:
	  • **Replace.** Deletes all rules on the target services and replaces them with the selected rules from the source service.
	  • **Merge.** Merges the selected rules with the existing rules on the target services.

History	Displays the last ten snapshots of rules applied through Security Analytics. You can select and apply (restore) a snapshot to the Decoder at anytime.

Rules Grid Context Actions

Within a rules grid, right-clicking a row displays the Rules Grid Context Menu.

Option	Description
Cut	Deletes the current rule.
Copy	Copies the current rule.
Paste Above	Pastes the copied rule above the current rule.

Option	Description
Paste Below	Pastes the copied rule below the current rule.
Edit	Edits the current rule.
Insert Below	Inserts imported rules below the current rule.
Insert Above	Inserts imported rules above the current rule.
Export Selection	Exports the selected rules.
Push Selected Rules	Pushes the selected rules to other services.

Topics

- App Rules Tab
- Correlation Rules Tab
- Network Rules Tab
- Rule and Query Guidelines

App Rules Tab

This topic describes the features for creating and managing application rules in the Services Config view > App Rules tab.

The App Rules tab enables you to manage application rules. Security Analytics applies application rules at the session level.

Step 4. Configure Decoder Rules provides additional information and Configure Application Rules provides instructions for creating application rules.

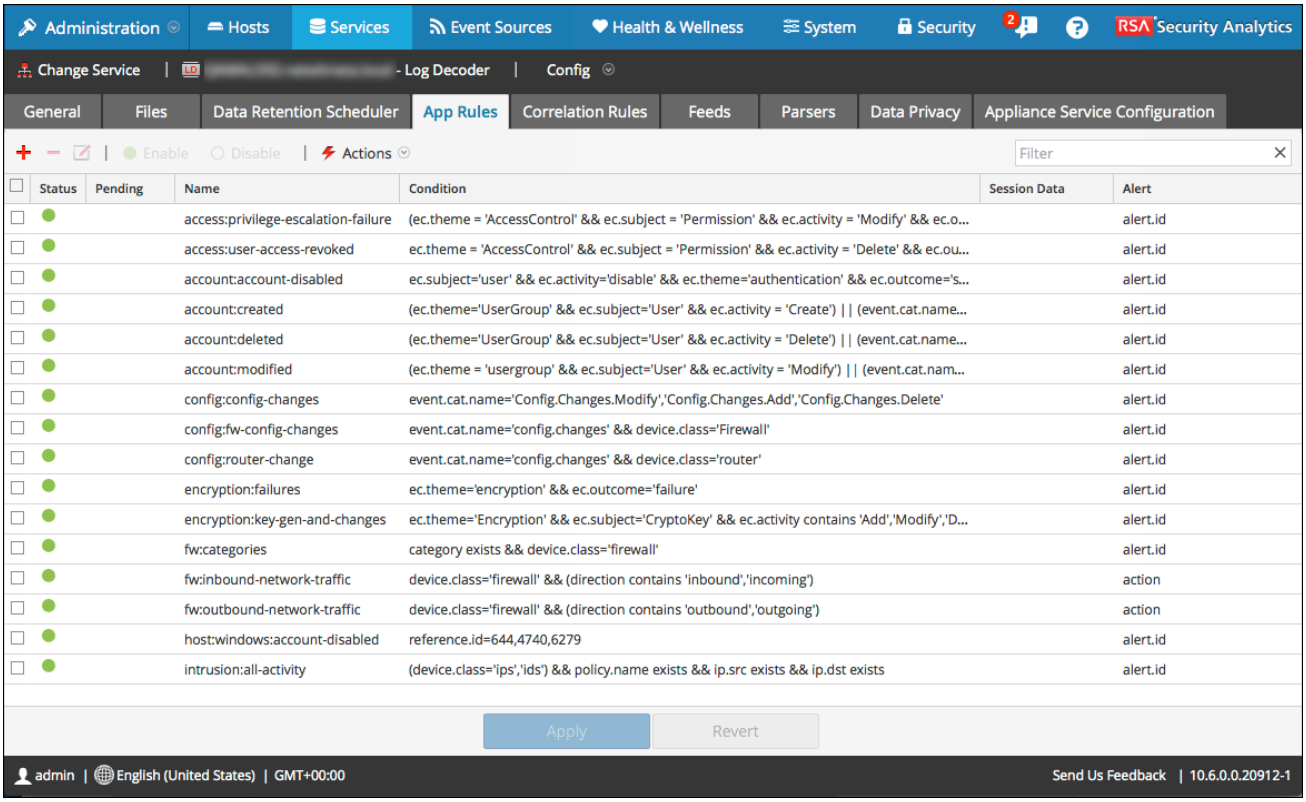
The toolbar on the App Rules tab is common to all types of rules. Services Config View - Rules Tabs provides information on the common rules toolbar and actions.

To access the App Rules tab:

1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a Decoder or a Log Decoder service and **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **App Rules** tab.

The following figure shows an App Rules tab.

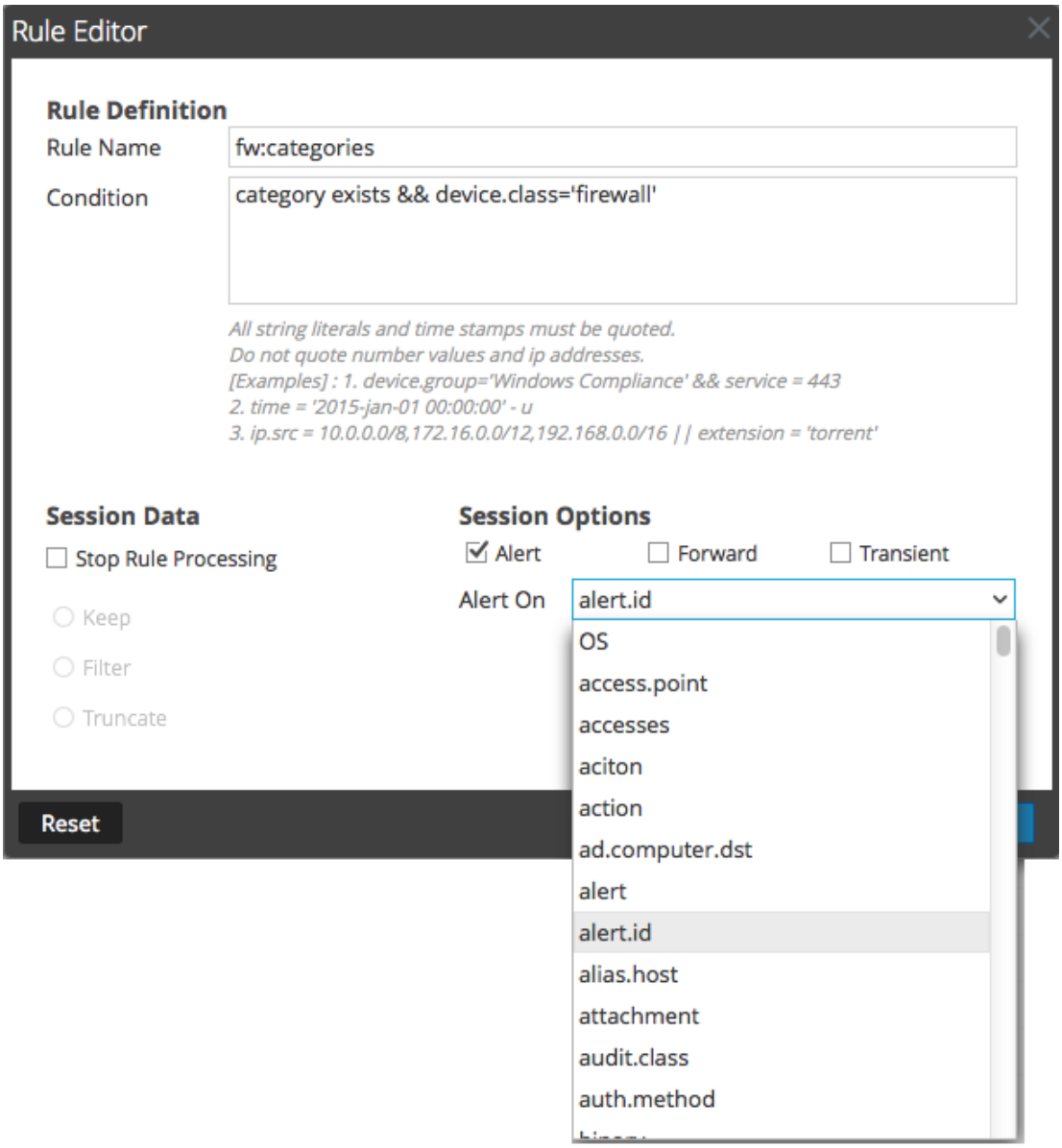


Application Rules Tab Columns

Column	Description
Pending	This column indicates whether a rule has pending changes. Rules that are currently active on the Decoder have no indicator. If the rule is new or has been modified, the column contains . Once the rules are applied, the pending indicator is removed.
Name	This is the rule name, a descriptive identifier for the rule.
Condition	This is the definition of the condition that triggers an action when matched.
Session Data	This column displays the Session Data action taken when a packet matches the rule. Possible values are **Filter**, **Keep**, or **Truncate**.
Alert	This column displays the name of the custom alert that the Decoder generates when metadata matches the rule.
Status	This column indicates whether the rule is enabled or disabled with a circle icon. If the circle is filled green, the rule is enabled. If the circle is empty, the rule is disabled.

Rule Editor Dialog

The following figure shows the Rule Editor dialog for an application rule.



The **Rule Editor** dialog provides the fields and options needed to define an application rule.

Field	Description
Rule Name	The descriptive name that identifies the rule.

Field	Description
Condition	The definition of the condition that triggers an action when matched. You can type directly in the field or build the condition in this field using meta from the Intelisense window actions. As you build the rule definition, Intellisense displays syntax errors and warnings. All string literals and time stamps must be quoted. Do not quote number values and IP addresses. Rule and Query Guidelines provides additional details.

The following table describes the Session Data actions and options.

Action	Description
Stop Rule Processing	If checked, further rule evaluation ends if the rule is matched, and the session is saved in accordance with the session action. If not checked, rule evaluation continues until all rules are evaluated.
Keep	The packet payload and associated metadata are saved when they match the rule.
Filter	The packet is not saved when it matches the rule.
Truncate	The packet payload is not saved when it matches the rule, but packet headers and associated metadata are retained.
Alert and **Alert On**	If **Alert** is checked, the packet generates a custom alert when metadata matches the rule. You can select the name of the alert in the **Alert On** field.
Forward	Enables the performance of syslog forwarding when the log matches the rule.
Transient	Prevents the alert metadata that is created from being written to the disk.

The following table describes Rule Editor dialog actions.

Action	Description
Reset	Resets the contents of the dialog to their values before editing; changes are discarded.

Action	Description
Cancel	Cancels any edits and closes the Rule Editor dialog.
OK	Saves the new rule or edited rule, and adds it to the rules grid. The Rule Editor dialog closes.
Save	(Rules with deprecated syntax only) Applies a corrected rule individually to the Decoder service. See Fix Rules with Deprecated Syntax.

Correlation Rules Tab

This topic describes the features for creating and managing correlation rules in the Services Config view > Correlation Rules tab.

The Correlation Rules tab enables you to manage correlation rules. Basic correlation rules are applied at the session level and alert the user to specific activities that may be occurring in their environment. Security Analytics applies correlation rules over a configurable sliding time window.

Step 4. Configure Decoder Rules provides additional information and Configure Correlation Rules provides instructions for creating correlation rules.

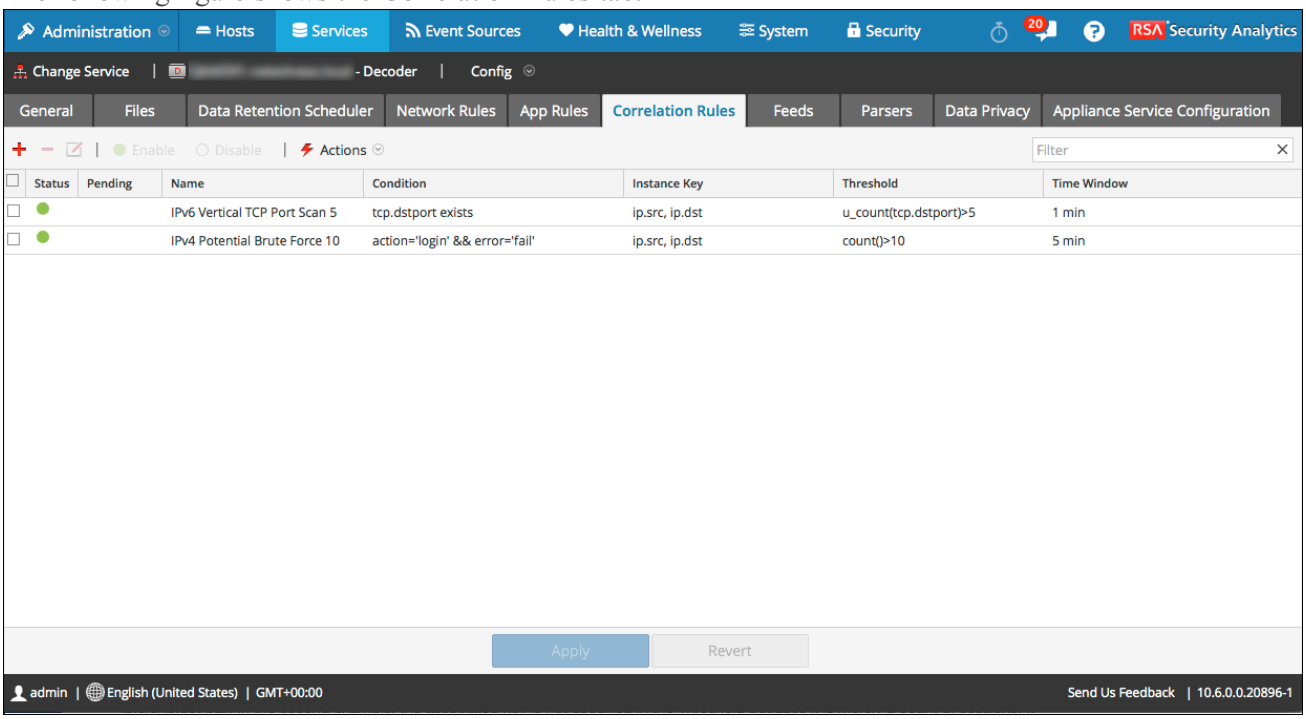
The toolbar on the Correlation Rules tab is common to all types of rules. Services Config View - Rules Tabs provides information on the common rules toolbar and actions.

To access the Correlation Rules tab:

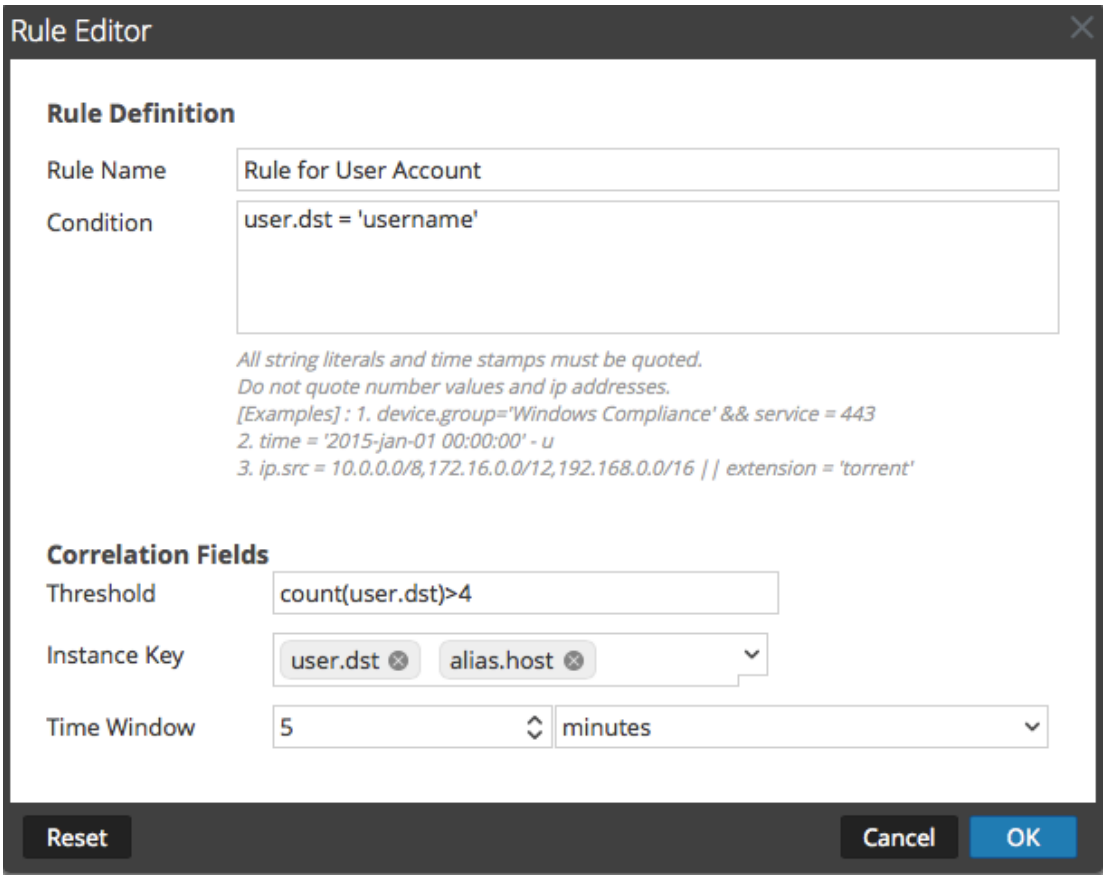
1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a service and  **>View > Config**.

   The Config view for the selected service is displayed.
3. Click the **Correlation Rules** tab.

The following figure shows the Correlation Rules tab.



The following figure shows the Rule Editor dialog for a correlation rule.



The following table describes the Correlation Rules tab columns.

Column	Description
Pending	This column indicates whether a rule has pending changes. Rules that are currently active on the Decoder have no indicator. If the rule is new or has been modified, the column contains . Once the rules are applied, the pending indicator is removed.
Name	This is the descriptive name for the rule.
Condition	This is the definition of the condition that triggers an action when matched. In conditions, all string literals and time stamps must be quoted. Do not quote number values and IP addresses. Rule and Query Guidelines provides additional details.

Column	Description
Instance Key	This is the target indicator to base the event upon. It can be a single primary key, such as ip.src or a compound primary key such as ip.src,ip.dst.
Threshold	This is the minimum number of occurrences required to trigger a correlation session and can include a associated key that identifies the meta type that were are counting to determine if the condition is satisfied. The correlation engine cannot use IPv4 or IPv6 as an associated meta type. Use one of these three arguments: • u_count(associated_key) = the count of unique values of the specified key. A key is required. • sum(associated_key) = the values of the specified key. a key is required. • count() = number of sessions, no associated key used. If included, it is ignored.
Time Window	This is the duration in hours, minutes, or seconds within which the threshold must be reached to trigger a correlation session.
Status	This column indicates whether the rule is enabled or disabled with a circle icon. If the circle is filled green, the rule is enabled. If the circle is empty, the rule is disabled.

The **Rule Editor** dialog provides the fields and options needed to define a network rule. The fields correspond exactly to the grid columns.

Action	Description
Reset	Resets the contents of the dialog to their values before editing; changes are discarded.
Cancel	Cancels any edits and closes the Rule Editor Dialog.
OK	Saves the new rule or edited rule, and adds it to the rules grid. The Rule Editor Dialog closes.
Save	(Rules with deprecated syntax only) Applies a corrected rule individually to the Decoder service. See Fix Rules with Deprecated Syntax.

Network Rules Tab

This topic describes the features for creating and managing network rules in the Services Config view > Network Rules tab.

The Network Rules tab enables you to manage network rules. Security Analytics applies network rules at the packet level. Network rules consist of rule sets from Layer 2, Layer 3, and Layer 4. Multiple rules can be applied to the Decoder. Rules can be applied to multiple layers (for example, when a network rule filters out specific ports for a specific IP address). Network rules apply only to packet Decoders.

Step 4. Configure Decoder Rules provides additional information and Configure Network Rules provides instructions for creating network rules.

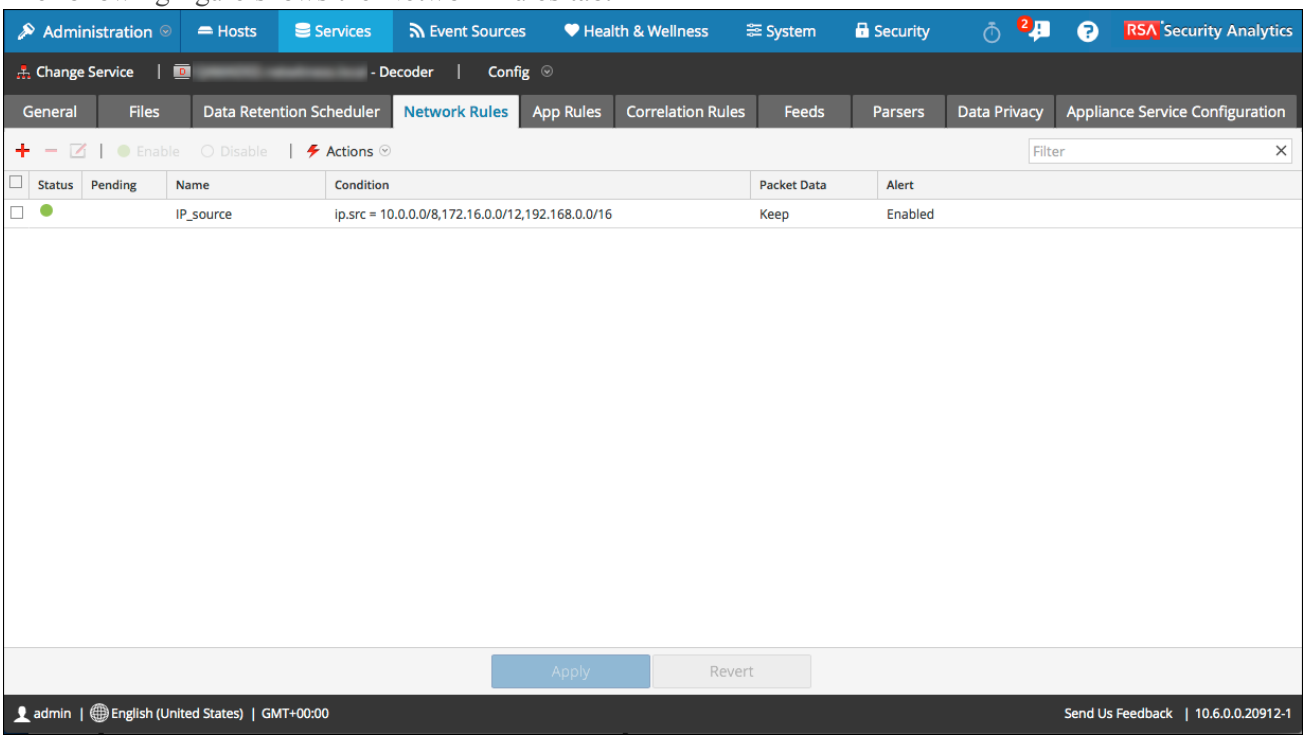
The toolbar on the Network Rules tab is common to all types of rules. Services Config View - Rules Tabs provides information on the common rules toolbar and actions.

To access the Network Rules tab:

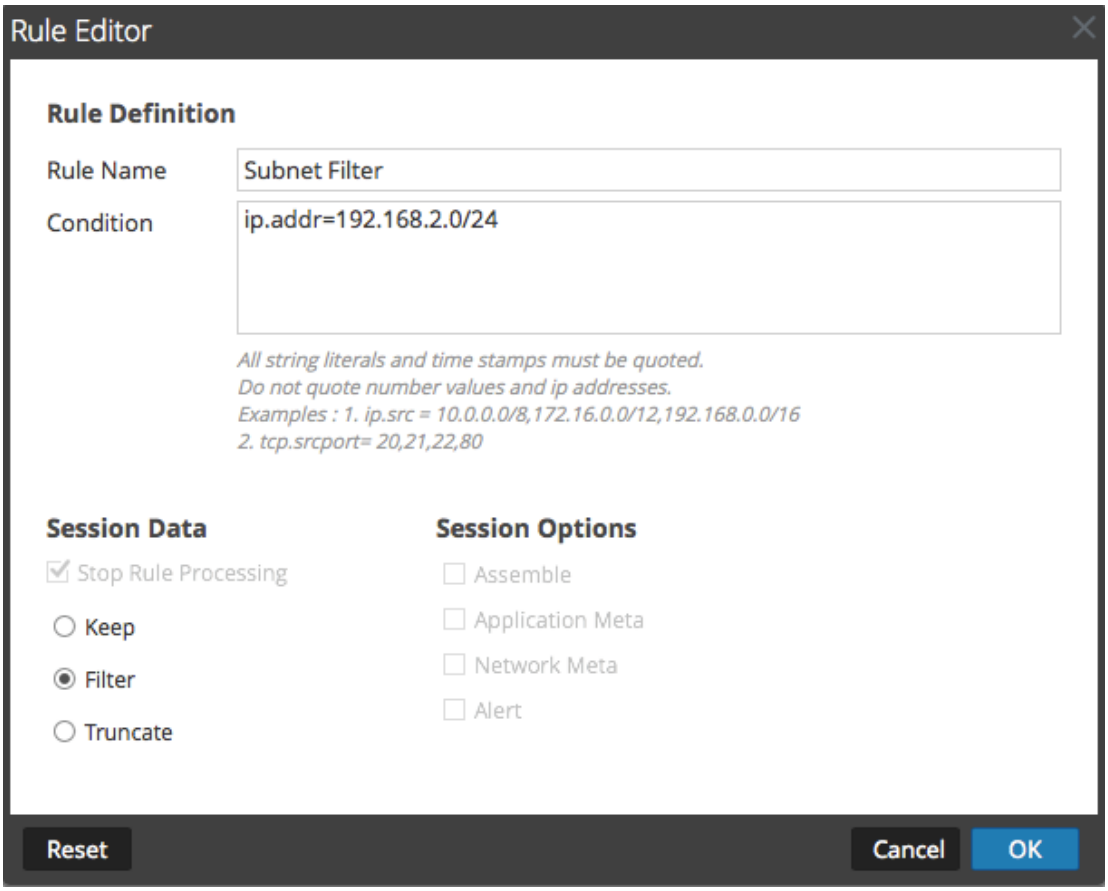
1. In the **Security Analytics** menu, select **Administration > Services.**
2. Select a Decoder service and select  **> View > Config**.

   The Config view for the selected service is displayed.
3. Select the **Network Rules** tab.

The following figure shows the Network Rules tab.



The following figure shows the Rule Editor dialog for a network rule.



Features

The following table describes the columns in the Network Rules grid.

Column	Description
Pending	This column indicates whether a rule has pending changes. Rules that are currently active on the Decoder have no indicator. If the rule is new or has been modified, the column contains . Once the rules are applied, the pending indicator is removed.
Name	This is the rule name, a descriptive identifier for the rule.
Condition	This is the definition of the condition that triggers an action when matched.
Packet Data	This column displays the Session Data action taken when a packet matches the rule. Possible values are **Filter**, **Keep**, or **Truncate**.

Column	Description
Alert	This column indicates whether the Decoder generates a custom alert when metadata matches the rule. Possible values are **Enabled** or **Disabled**.
Status	This column indicates whether the rule is enabled or disabled with a circle icon. If the circle is filled green, the rule is enabled. If the circle is empty, the rule is disabled.

The **Rule Editor** dialog provides the fields and options needed to define a network rule.

The following table describes the Rule Definition fields.

Field	Description
Rule Name	The descriptive name that identifies the rule.
Condition	The definition of the condition that triggers an action when matched. You can type directly in the field or build the condition in this field using meta from the Intellisense window actions. As you build the rule definition, Intellisense displays syntax errors and warnings. In conditions, all string literals and time stamps must be quoted. Do not quote number values and IP addresses. Rule and Query Guidelines provides additional details. Supported Meta Keys in Network Rules describes the meta keys that Security Analytics supports for use in network rule conditions.

The following table describes the Session Data actions.

Action	Description
Stop Rule Processing	If checked, further rule evaluation ends if the rule is matched, and the session is saved as indicated. If not checked, rule evaluation continues until all rules are evaluated.
Keep	The packet payload and associated meta are saved when they match the rule.
Filter	The packet is not saved when it matches the rule.

Action	Description
Truncate	The packet payload is not saved when it matches the rule, but packet headers and associated meta are retained.

The following table describes the session options.

Option	Description
Assemble	If checked, the assembler assembles the packet chain when it matches the rule.
Network Meta	The packet generates network metadata when it matches the rule.
Application Meta	The packet generates application metadata when it matches the rule.
Alert	The packet generates a custom alert when metadata matches the rule.

The following table describes Rule Editor dialog actions.

Action	Description
Reset	Resets the contents of the dialog to their values before editing; changes are discarded.
Cancel	Cancels any edits and closes the Rule Editor dialog.
OK	Saves the new rule or edited rule, and adds it to the rules grid. The Rule Editor dialog closes.
Save	(Rules with deprecated syntax only) Applies a corrected rule individually to the Decoder service. See Fix Rules with Deprecated Syntax.

Supported Meta Keys in Network Rules

Network rules consist of rule sets from Layer 2, Layer 3, and Layer 4. Multiple rules can be applied at the packet level to a Decoder. Rules can be applied to multiple layers (for example, when a network rule filters out specific ports for a specific IP address). You can create and manage network rules in the Services Config view > Network Rules tab.

Supported Meta Keys in Network Rule Conditions

The following table describes the meta keys that Security Analytics supports for use in network rule conditions.

Meta Key	Description
eth.addr	Ethernet source or destination address. Commonly known as the MAC address.
eth.dst	Destination Ethernet address. This is the same as the Ethernet address field except that it selects only packets where the destination address matches the selected value(s).
eth.src	Same as Ethernet destination except that it focuses on the source address.
eth.type	Ethernet frame type.
hdlc.type	Frame type of the HDLC frame.
ip.addr	IPv4 source or destination address in standard form. IP addresses can be entered in CIDR notation for subnets.
ip.dst	Destination IPv4 address in standard form. IP addresses can be entered in CIDR notation for subnets.
ip.proto	IPv4 protocol field.
ip.src	Source IPv4 address in standard form. IP addresses can be entered in CIDR notation for subnets.

Meta Key	Description
ipv6.addr	IPv6 source or destination address in hex format. Generally IPv6 addresses are written as eight groups of four hex digits, thus expressing the entire 128 bit address length. Supports notation to represent multiple blocks of 0000 in an address. Does not support CIDR notation.
ipv6.dst	Destination IPv6 address in hex format.
ipv6.proto	IPv6 protocol field. This maps to the Next Header field in the IPv6 header and uses the same values as the IPv4 protocol field.
ipv6.src	Source IPv6 address in hex format.
tcp.dstport	Destination TCP port.
tcp.port	TCP source or destination port.
tcp.srcport	Source TCP port.
udp.dstport	Destination UDP port.
udp.port	UDP source or destination port.
udp.srcport	Source UDP port.

Rule and Query Guidelines

All queries and rule conditions in RSA Security Analytics Core services must follow these guidelines:

All string literals and time stamps must be quoted. Do not quote numbers, MAC or IP addresses.

For example:

- `extension = 'torrent'`
- `time='2015-jan-01 00:00:00'`
- `service=80`
- `ip.src = 192.168.0.1`

Note: The space on the right and the left of an operator is optional.
For example, you can use `service=80` or `service = 80`.

Rule Examples

The following table shows examples of rule conditions. You can use rule conditions for log retention collections in an Archiver and for application, network, and correlation rules on a Decoder, Log Decoder, or Concentrator. Rule conditions are also used in all `Where` clauses in all Core database queries.

For detailed information on rule syntax in Security Analytics, see **Where Clauses** in the **Queries** topic in the *RSA Security Analytics Core Database Tuning Guide*.

Rule Name	Condition
ComplianceDevices	device.group='PCI Devices' || device.group='HIPPA Devices'
HighValueWindows	device.group='Windows Compliance'
MediumValueWindows	device.type='winevent_nic' && msg.id='security_4624_security'
LowValueWinLogs	device.type='winevent_nic' && msg.id='security_4648_security'
LowValueProxyLogs	device.class='proxy' && msg.id='antivirus_license_expired'
GeneralWindows	device.type='winevent_nic'

Strict Mode Configuration for Security Analytics 10.6

Since version 10.2, Security Analytics has been using a modern parser for rules and queries that strictly defines valid syntax. When a Core service encounters deprecated syntax, it writes a warning in the Security Analytics logs about the deprecated syntax. Security Analytics now enforces strict parsing for new application, network, and correlation rules. The previous generation legacy parser, now deprecated, allows syntax that is ambiguous, which can cause unexpected results. While Security Analytics 10.6 continues to support the deprecated syntax, future versions will no longer support it.

After you update to Security Analytics 10.6, rules with deprecated syntax are highlighted in the user interface. The Rule Editor provides additional tooltips. After you fix the rules, the highlights disappear. See **Fix Rules with Deprecated Syntax** in the *Decoder and Log Decoder Configuration Guide*.

The **/decoder/config/rules/rule.errors** and **/concentrator/config/rules/rule.errors** stats, introduced in 10.6, contain the count of rules with errors. If **rule.errors** is non-zero, Security Analytics generates a Health and Wellness alert to indicate that you need to fix the rules.

In addition, there is a migration path for queries from external systems. After an update from an earlier version, the system operates in deprecated mode (controlled by **/sdk/config/query.parse**). In deprecated mode, the service continues to use the legacy parser for all queries that fail strict parsing. The errors will be logged and a message will be streamed back to the client informing them of the strict parsing failure. But the query still executes and returns results just like previous versions. You should monitor the logs and external clients for any reports, dashboards, rules, and so on, that are written in deprecated syntax and fix those problems as they arise.

After you resolve the issues, you can switch all core services (Decoders, Log Decoders, Concentrators, Brokers, and Archivers) to strict mode and monitor them for issues. Strict mode does not use the legacy parser and any parsing violations return errors. You should perform this task before any major upgrade after 10.6, as the legacy parser will be removed in future releases and there will not be an option to operate in deprecated mode.

All new installs operate in strict mode by default. If you plan to add a new appliance to an existing infrastructure running in deprecated mode, in the Explore view (Administration > Services > Select a service and in the Actions menu, select View > Explore), you can switch **/sdk/config/query.parse** to deprecated mode until the whole stack has moved to strict mode.

In Security Analytics 10.6, all rule validation will always operate in strict mode, in order to eliminate creating syntax problems.

Valid Syntax with the Modern Parser

The following are valid syntax rules using the modern parser:

- All text types must quote literal values. Example: `username = 'user1'`
- Quotes can use single or double quotes, but they must match (you cannot start with a single quote and finish with a double quote)
- If the literal value has a quote, you can escape it or use a different starting quote character. Both of the following examples are valid (use backslash for the escape character):
  - `username = "User's"`
  - `username = 'User\'s'`
- To use a backslash in a literal string, escape it using an extra backslash: `\\`
- All time types should use quotes for dates in this form: `time = 'YYYY-MM-DD HH:MM:SS'`
- All time types that are the number of seconds since EPOCH (Jan 1, 1970), should not be quoted.
  Example: `time = 1448034064`
- EVERYTHING else is unquoted: IP Address, Ethernet Addresses, Numerics, and so on.
  Example: `service = 80 && ip.src = 192.168.1.1/16`

Ambiguous Syntax Examples with the Legacy Parser

Here is an example of ambiguous syntax with the deprecated legacy parser:

```
select * where alias.host = server-xeon
```

In the query above, it seems logical that the query author wants to return all meta where alias.host is equal to `server-xeon`. Unfortunately, that does not happen with the legacy parser. Instead, it parses that query as `select * where alias.host = 'server'-'xeon'`. So it converts it to a range query (values BETWEEN 'server' and 'xeon', using the unquoted dash - operator) and since that range will probably return results that have nothing to do with `server-xeon`, users may assume that there is a problem in the query engine. In 10.6 strict mode, you will get this error:

expecting <quoted_string> here: "server-xeon"

This immediately tells the user why the query failed to parse correctly.

Here is another example of ambiguous syntax with the deprecated legacy parser:

```
select * where username=lastname,firstname
```

In Security Analytics language, you can specify multiple values by separating them with commas (an implicit OR operator). But did the query author mean `'lastname,firstname'` or did the author want to search for two values, `'lastname'` and `'firstname'`? Again, it is ambiguous, but accepted in the legacy parser (which turns it into `'lastname' OR 'firstname'`). With the modern parser, you must be completely clear in your intent, either `'lastname,firstname'` or `'lastname','firstname'`.

The following example fails to parse correctly:

```
select * where username = lastname, firstname
```

At first glance, you may not notice that there is a space between the comma and firstname. Because there are no quotes, this fails to parse correctly with the legacy parser, but it does not return a parse error! Instead, it actually executes (by discarding firstname completely because of the whitespace issue). What is worse, you would not know that the executed query is not what you submitted unless you determine through some other means that the result set is incomplete. With strict parsing enabled, this returns a parse error, which is what should always happen. At least in prior versions (and deprecated mode), the logs show a message immediately after the audit log for the query: **Rule '<rule name>' in deprecated format. Please make sure all text values are surrounded with quotes.**

Services System View - Decoders

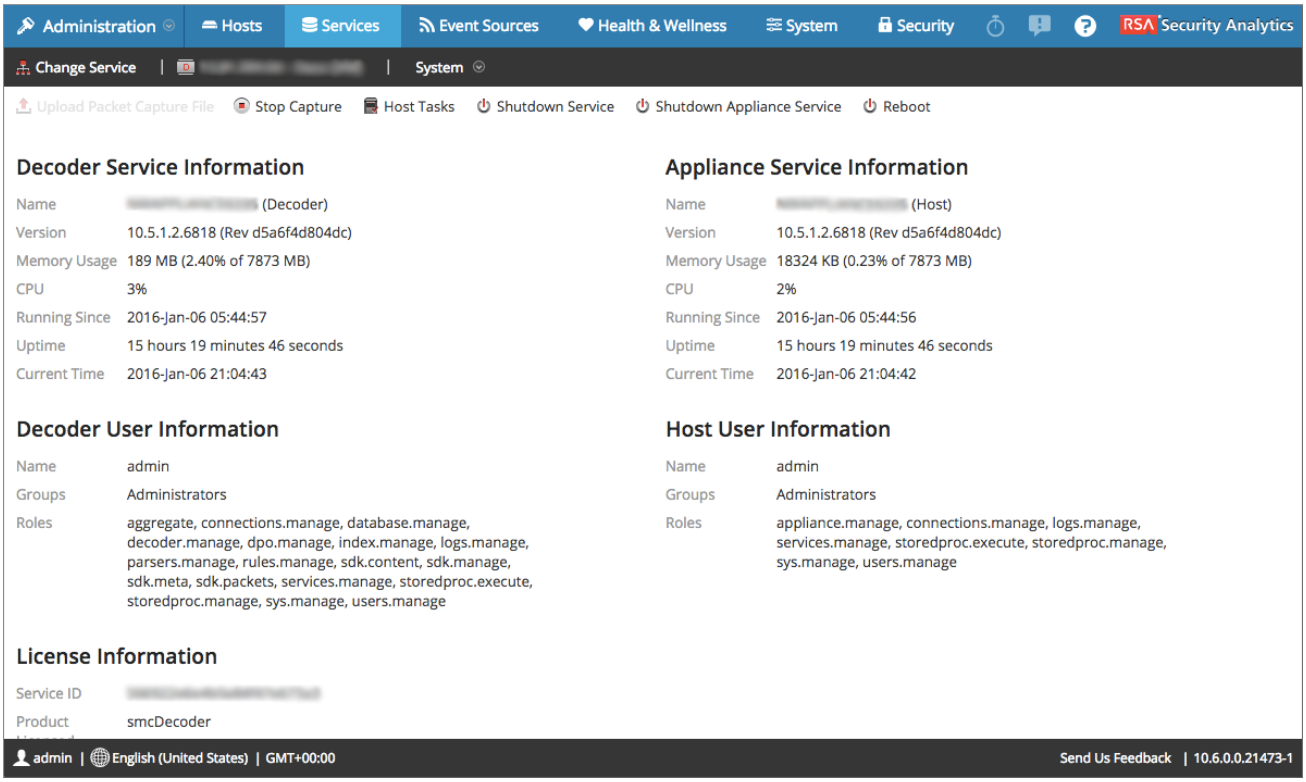
This topic introduces features in the System view that pertain specifically to Decoders and Log Decoders.

A Log Decoder is a special type of Decoder, and is configured and managed in a similar way to a Decoder. Therefore, most of the information in this section refers to both types of Decoders. Differences for Log Decoders are noted.

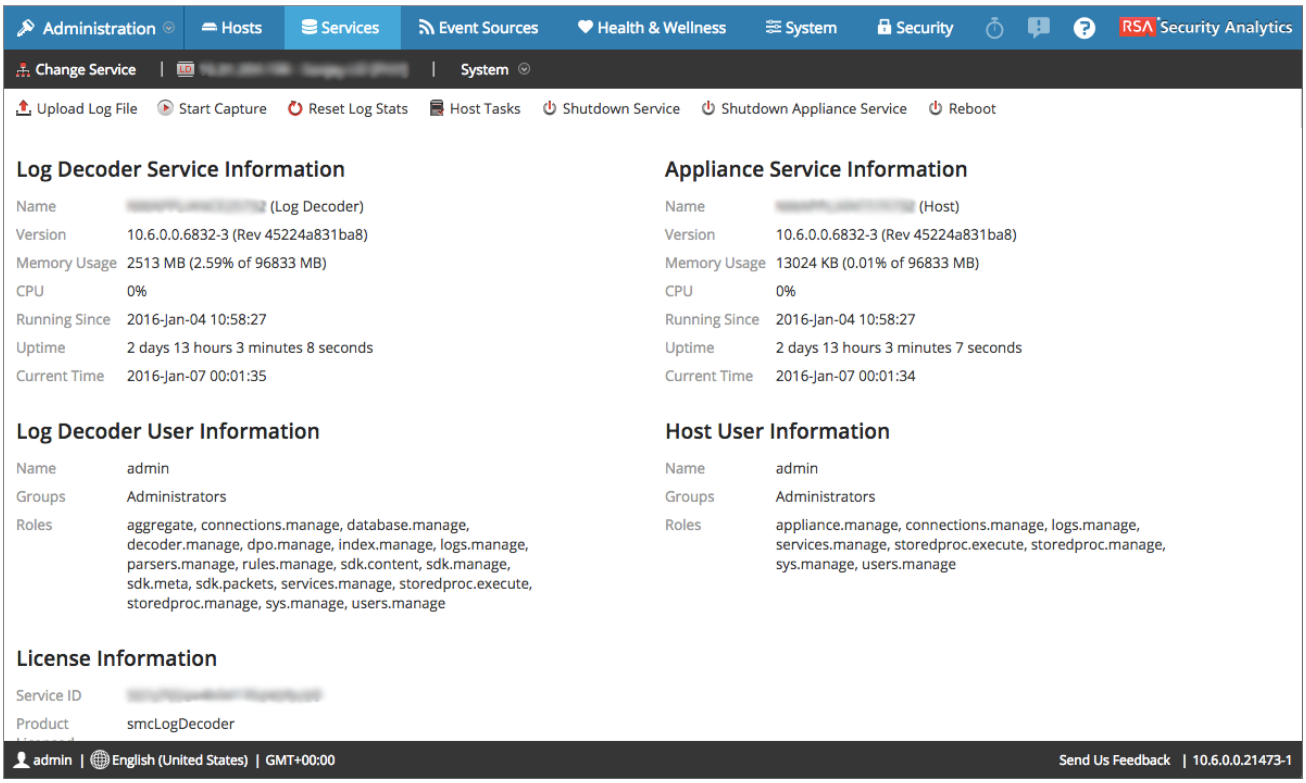
To access the Services System view for a Decoder:

1. In the **Security Analytics** menu, select **Administration > Services**.
   The Administration Services view is displayed.
2. Select a Decoder or Log Decoder, and select  **> View> System**.

This is an example of the Services System view for a Decoder.



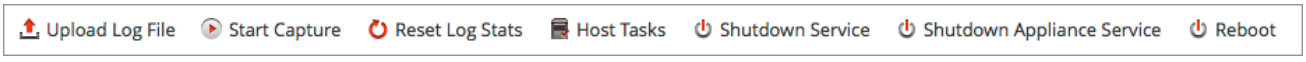
This is an example of the Services System view for a Log Decoder.



Features

Service Info Toolbar

These two toolbars illustrate the options specific to Decoders and Log Decoders.



In addition to the common options in the Services System view toolbar, you can start and stop capture of packets or logs. The upload file options are different for the standard Decoder (packet capture file) and the Log Decoder (log file).

Action	Description
Upload Packet Capture File	Displays a dialog that provides a way to select a packet capture (.pcap) file for upload to the selected Decoder. For more information, see Upload Packet Capture File. **Note:** This option does not apply to Log Decoders.
Upload Log File	Displays a dialog that provides a way to select a log (.log) file for upload to the selected Log Decoder. For more information, see Upload Log File to a Log Decoder.
Start/Stop Capture	Starts packet capture on the selected Decoder. When packet capture is in progress, the option in the toolbar changes to Stop Capture, and the option to upload a file is unavailable.

Related Topic

- **Services System View** in the *Hosts and Services Getting Started Guide*