NetWitness® Platform XDR

Version 12.3.0.0

Alerting with ESA Correlation Rules User Guide

NETWITNESS
Platform XDR

Contact Information

NetWitness Community at https://community.netwitness.com contains a knowledge base that answers common questions and provides solutions to known problems, product documentation, community discussions, and case management.

Trademarks

License Agreement

Third-Party Licenses

Note on Encryption Technologies

Distribution

Miscellaneous

September, 2023

Contents

Getting Started with ESA

This topic covers quick start topics for NetWitness Event Stream Analysis (ESA) to help you get started in using ESA. The following topics are designed to assist you in working with ESA Correlation Rules.

- Best Practices helps you to understand how to best set up, deploy, and create rules.
- Troubleshoot ESA helps you to troubleshoot different aspects of ESA, including rule writing and deployment.
- View Memory Metrics for Rules helps you to work with memory metrics to understand memory usage for ESA services.

In NetWitness version 11.5 and later, There are only two services that can run on an ESA host:

- **ESA Correlation (ESA Correlation rules):** Creates alerts from ESA rules.
- **Contexthub Server (Context Hub):** Runs only on an ESA primary host. Contexthub Server provides enrichment lookup capability in the Respond and Investigate views. For information, see the *Context Hub Configuration Guide*.

Note: The Event Stream Analytics Server (ESA Analytics) service is not supported in NetWitness Platform version 11.5 and later.

The first service is the ESA Correlation service that creates alerts from ESA rules, also known as ESA Correlation Rules, which you create manually or download from Live.

In NetWitness 11.3 and later, the ESA Correlation service replaces the Event Stream Analysis service and is also known as ESA Correlation Server. The ESA Correlation service provides the same services as the Event Stream Analysis service with the added benefit of enabling you to specify different data sources for your ESA correlation rules. Like the Event Stream Analysis service, the ESA Correlation service installs on the ESA Primary and ESA Secondary host types.

The second service is the Contexthub Server service, which provides enrichment lookup capabilities in the Respond and Investigate views. It runs only on an ESA Primary host. For information, see the *Context Hub Configuration Guide*. Go to the NetWitness All Versions Documents page and find NetWitness Platform guides to troubleshoot issues.

IMPORTANT: The NetWitness server (Admin server), ESA primary host, and ESA secondary host must all be on the same NetWitness Platform version.

How ESA Generates Alerts

The ESA Correlation service runs rules that specify criteria for problem behavior or threatening events in your network. When ESA detects a threat that matches rule criteria, it generates an alert.

To generate alerts, ESA performs the following functions:

1. Gathers data
2. Runs ESA rules against the data
3. Captures events that meet rule criteria
4. Generates alerts for those captured events

Data Source Configuration Changes

In NetWitness version 11.3 and later, the ESA Correlation service enables you to specify different data sources for different sets of rules. Instead of adding data sources, such as Concentrators, to the entire ESA Correlation service, you can specify different data sources for each ESA rule deployment. An ESA rule deployment includes an ESA Correlation service with its associated data sources and a set of ESA rules. For example, you may want to use Concentrators with HTTP packet data in one deployment and Concentrators with HTTP log data in another deployment. For more detailed information, see Deploy Rules to Run on ESA.

An Endpoint Risk Scoring Rules Bundle is available in NetWitness Platform

An Endpoint Risk Scoring Rules Bundle, which contains approximately 400 rules, comes with NetWitness 11.3 and later. Endpoint risk scoring rules only apply to NetWitness Endpoint. You can add the Endpoint Risk Scoring Rules Bundle to an ESA rule deployment in the same way that you would add any ESA rule. However, you must specify endpoint data sources (Concentrators) in the ESA Rule Deployment.

The ESA Correlation service can process endpoint risk scoring rules, which generate alerts that are used in risk scoring calculations to identify suspicious files and hosts. To turn on risk scoring for NetWitness Endpoint, you must deploy endpoint risk scoring rules on ESA. For instructions, see "Deploy Endpoint Risk Scoring Rules on ESA" in the *ESA Configuration Guide*. For complete information on configuring NetWitness Endpoint, see the *NetWitness Endpoint Configuration Guide*.

Go to the NetWitness All Versions Documents page and find NetWitness Platform guides to troubleshoot issues.

Best Practices

Best practices provide guidelines to help you write and manage rules, deploy rules, and maintain system health for your ESA services.

Understand Event Stream Analysis Rule Types

The ESA Correlation service provides advanced stream analytics such as correlation and complex event processing at high throughputs and low latency. It is capable of processing large volumes of disparate event data from Concentrators. However, when working with Event Stream Analysis, you should be aware of the factors that affect resource usage in order to create effective rules.

Each event that is received by ESA is evaluated to determine if it may trigger a rule. There are three types of rules that can be deployed in order to determine what the ESA engine should do with the incoming event. Each of these rule types have different impacts on system resource utilization. All three rule types may be created via the Rule Builder, Advanced Event Processing Language (EPL) rules, or downloaded via RSA Live. The table below lists the rule type and the impact this rule may have on system resources.

Rule Type	Description
Simple Filter Rule	This rule has no correlation to other events. At ingestion time, this rule is evaluated against a set of conditions, and if those conditions are met an alert is generated. If no conditions match, the event is quickly released by the engine to free up memory usage. These rules do not take up memory since the events are not retained beyond the initial evaluation. The memory resource usage does not increase as more simple filter rules are deployed. However, if the filter condition is too generic, it is possible that this rule can generate too many alerts, which will strain the system resources for the storage and retrieval of these alerts. For example, you might write a rule to generate an alert when HTTP network activity arrives over a non-standard HTTP port.
Event Window Rule	This rule evaluates a set of events over a time period for specific conditions. At ingestion time, the rule is evaluated against a set of conditions. If those conditions are met, the event is retained in memory for a specific amount of time. After the specified time passes, the events are removed from the time window if the number of events collected does not meet the threshold to trigger an alert. The memory consumption of such rules is highly dependent on the incoming event rate (traffic), the amount of data per event, and the time length specified in the event window. Each matching event is retained in memory until the time window has passed, so the longer the time window, the greater the potential volume. For example, you might write a rule that generates an alert if a user has five failed login attempts within a ten minute time frame.

Rule Type	Description
Followed By Rule	This rule evaluates a chain of incoming events to determine if the sequence of events matches a particular condition. At ingestion time, the rule is evaluated against a set of conditions. If the conditions are met, one of two actions occurs: • If this is the first event of the sequence, a new event thread is started, and the event is retained as the head of the sequence. • If the event belongs to an existing event thread, it is added to that sequence. In both cases, the event is retained in memory. The amount of resource usage is particularly sensitive to the customer environment for this type of rule. If the filter condition generates many event threads, resources are consumed for each new thread (in addition to the event). Additionally, if the end of the event thread is never met (that is, an alert is never generated), then the entire event is saved in memory indefinitely. For example, you might write a rule to generate an alert when a user fails to log in to a server, then performs a successful login, and then creates a new account.

Note: ESA sends alerts to NetWitness Respond for processing and the alerts are eventually stored in a database. If your rule creates too many alerts, it can slow down another part of the system.

When writing and deploying rules, you should be aware that rule memory usage and alert generation consume system resources. The sections below are designed to help you keep your usage at a healthy level and monitor for problems if systems are becoming overloaded.

Best Practices for Writing Rules

These are general guidelines for writing rules.

- **Create alerts for actionable events**. The purpose of an alert should be to notify you of an event that requires immediate and specific action. For events that do not require action, or only require you to have awareness of the event, you can create a report.
- **Validate your ESA rules within the Rule Builder or Advanced EPL Rule Builder before you deploy them**. To prevent errors in your rules and confirm that they generate the expected alerts, you can test the rule logic with JSON data within the rule builders. This capability is available in NetWitness Platform version 11.5 and later. For more information, see Validate an ESA Rule and Validate an Advanced EPL Rule .
- **Configure new rules as trial rules so you can observe how they react in your environment**. If you deploy new rules as trial rules, they will be disabled if the configured memory threshold is exceeded.
- **Configure Alert notifications only after your rule testing and tuning is complete**. This can help ensure you do not get flooded with notifications if a rule behaves differently than you expect.

- **Rules need to be specific so that you limit resource usage**. Use the following guidelines to limit usage:
  - Make the filters on the rule exclude all but the necessary events for the rule to fire accurately.
  - Make the size of your windows (window time for correlation) as small as possible.
  - Limit the events that you include in the window: For example, if you only want to see IDS events, ensure that you only include those events in your time window.
- **Add Memory Thresholds to ESA rules that use memory**. For example, if a rule contains windows or pattern matching, configure a memory threshold for that rule. If the rule goes over the allotted memory threshold, it gets disabled individually and an error is displayed for that rule on the  **(Configure) > ESA Rules > Services** tab. This capability is available in NetWitness Platform version 11.5 and later.
- **Rules need to be tuned to an alert level that is manageable**. If you are flooded with alerts, then the purpose and of an alert is lost. For example, maybe you want to know about encrypted traffic to other countries. But, you could limit the list to countries that are known risks. This limits the volume of alerts to a level you can manage.

For more best practice information for writing ESA rules, see ESA Rule Writing Best Practices.

Best Practices for Working with RSA Live Rules

These are guidelines for RSA Live Rules.

- **Deploy RSA Live rules in small batches**. Not every rule is suited to every environment. The best way to ensure your RSA Live rules are successful is to deploy them in small batches so you can test them in your environment. If you deploy small batches, it's much easier to tell if a particular rule has an issue.
- **Read the rule descriptions provided with RSA Live rules**. ESA rules are not “one size fits all.” Not all rules will work in your environment. The rule descriptions tell you which parameters you will need to modify to successfully deploy a rule in your environment.
- **Set your parameters**. RSA Live rules have parameters that need to be modified. If you do not modify your parameters, the rule may not work or it may exhaust your memory.
- **Deploy new rules as trial rules so you can observe how they react in your environment.** If you deploy new rules as trial rules, they will be disabled if the configured memory threshold is exceeded. For more details, see Working with Trial Rules.

Best Practices for Deploying Rules

These are general guidelines for deploying rules.

- **Deploy rules in small batches so you can observe how they react in your environment.** Not all environments are the same, and a rule will need to be tuned for memory usage, alert volume, and

effective detection of events.

- **Test rules before you configure alert notifications**. Configure Alert notifications only after your rule testing and tuning is complete. This can help ensure you do not get flooded with alerts if a rule behaves differently than you expect.
- **Monitor system health as a part of your deployment process**. When you deploy rules, monitor your system's health as a part of your deployment process. You can view total memory usage for your ESA in the Health and Wellness tab. For more information, see "View Detailed Statistics in Health and Wellness" in Troubleshoot ESA.

Best Practices for System Health

These are general guidelines for system health.

- **Set up new rules as trial rules**. A common issue is that new rules may cause memory issues. To prevent this, you can set up new rules as trial rules. If the configured memory threshold is met, all trial rules are disabled to prevent the system from running out of memory. For more information about trial rules, see Working with Trial Rules.
- **Set up thresholds in Health & Wellness to alert you if memory usage is too high**. There are metrics in NetWitness Health & Wellness that track memory usage. You can set up alerts and notifications to send you an email if those thresholds are crossed. For more information about the memory statistics you can view, see View Memory Metrics for Rules.
- **Monitor memory metrics for each rule in Health & Wellness**. For each rule, you can view the estimated memory usage in Health & Wellness. You can use this information to ensure that rules do not use too much memory. For more information about the memory statistics you can view, see View Memory Metrics for Rules.

Troubleshoot ESA

This section describes common issues that may occur while using ESA, and it suggests common solutions to these problems.

Troubleshoot ESA Correlation Services

Problem	Possible Causes	Solutions
On the NetWitness Platform Dashboard, the ESA service appears in red to indicate it is offline. In the **(Configure) > ESA Rules** view, the following message appears: "The Service is either offline or not reachable."	Several	When an ESA Correlation service is offline, there are many possible causes. However, a common issue is that you have created a rule that uses excessive memory and causes the ESA service to fail. To troubleshoot this problem, see Steps to Troubleshoot Memory Issues with an ESA Service Offline. Other common causes might be that your firewall is blocking the connection between the ESA and NetWitness Platform, or the ESA Correlation service machine may be down. To bring up ESA Services: Go to **(Admin) > Services**, select your ESA service, and then select **> Start**. If your ESA service is stopping and restarting in a loop, you may need to call Customer Support to get the services to start.

Problem	Possible Causes	Solutions
On the NetWitness Platform Dashboard, the ESA service appears in red to indicate it is offline. In the **(Configure) > ESA Rules** view, the following message appears: "The Service is either offline or not reachable.	Configuration issues	If your system has been recently upgraded, you may have made a configuration error. Go to **(Admin) > Services**, select your ESA service and then select **> Edit**. In the Edit Service dialog, click **Test Connection**. If the connection fails, you likely have a configuration error. Attempt to fix your configuration error and try again.
Suddenly, an ESA Correlation service is completely unresponsive and appears to have crashed. The ESA Correlation Log file shows a Too Many Open Files error.	Connectivity issues	The most likely cause for this error is a connectivity issue between the ESA Correlation service and a data source used in an ESA rule deployment, such as a Concentrator or Decoder. 1. Restart the ESA Correlation service. 2. Go to **(Admin) > Services**, select your ESA service and then select **> Restart**. 3. Check the connectivity from ESA to the data source. Look for connection errors in the ESA Correlation logs. You can use SSH to get in the system and go to: `/var/log/netwitness/correlation-server/correlation-server.log`. For example: `Error: com.rsa.netwitness.streams.RecordStreamException: admin@<ip address>:56005:: com.rsa.netwitness.streams.RecordStreamException: connect::com.rsa.asoc.transport.nw.session.NextgenException: Failed to connect to <ip address>:56005` If there are network connectivity issues, fix the issues and then restart the ESA Correlation service again to see if it fixes the problem. If you have a data source with intermittent connectivity, you should remove it from the ESA rule deployment.

Troubleshoot RSA Live Rules for ESA

Problem	Possible Causes	Solutions
I imported a group of rules from RSA Live, and now my ESA service is crashing. Why?	You may not have configured the parameters for the RSA Live rule to tune it for your environment.	Each rule in RSA Live has a description that includes the parameters you must configure and prerequisites for your environment. Review this description to see if the rule is appropriate for your environment. To ensure that you deploy rules safely in your environment, configure new rules as trial rules to test them in your environment. Trial rules add a safeguard for testing new rules. For details on this, see Deploy Rules as Trial Rules.

I imported a group of rules from RSA Live, and while the rules deployed without errors, they were later disabled.	Not all RSA Live rules are meant for every environment. You may not have the correct meta in your ESA for the rule to run.	You can verify that a rule was disabled by going to  **(Configure) > ESA Rules > Services > Deployed Rule Stats**. If the rule is disabled, the green icon does not display next to the rule.
		If a rule deployed correctly but was disabled, check the logs for exceptions related to the rule. Specifically, check to see if the rules were disabled due to missing meta. To do this, go to the ESA Correlation logs. You can use SSH to get in the system and go to: `/var/log/netwitness/correlation-server/correlation-server.log`.
		Then, search for a message similar to the following:
		`"Property named '<meta_name>' is not valid in any stream"`
		For example, you might see:
		`Failed to validate filter expression '(medium=1 and streams=2 or medium=3...(238 chars)': Property named 'tcp_flags_seen' is not valid in any stream`
		If a similar message displays, you may need to add a custom meta key to the Log Decoder or Concentrator. To do this, follow these instructions: "Create Custom Meta Keys Using Custom Feed" in the *Decoder and Log Decoder Configuration Guide*.

Troubleshoot ESA Rules

Problem	Possible Causes	Solutions
I have an ESA rule that is not getting deployed and is not creating alerts.	A meta key that the rule uses is a string array type, but it shows as a string type on ESA.	Check to see if any string array meta keys that the rule uses are configured as string array types on ESA. Go to  **(Configure) > ESA Rules > Settings** tab (Meta Key References). • If it shows `string[]`, it is configured as a string array type on ESA. This is fine. • If it shows `string` without the brackets, it is configured as a string type and you need to fix it on ESA. In the ESA Correlation service Explore view, go to `correlation/stream`. Add string array meta keys to the **multi-valued** list to allow them to be used as an array in ESA rules. Go back to the Meta Key References and click the refresh icon (). Verify that the meta keys with a string array type show a value of `.string[]`. For additional details, see "Configure Meta Keys as Arrays in ESA Correlation Rules" in the *ESA Configuration Guide*.
I set up notifications for a rule, but we are not receiving them. The `correlation-server.log` file does not show any errors. Why?	Correlation-server successfully sent the notification messages to integration-server, but when integration-server tried to send the notifications to their destination, it failed.	When troubleshooting notifications, check both the ESA Correlation service log files (`/var/log/netwitness/correlation-server/correlation-server.log`) AND the Integration-Server log files on the NetWitness Server (`/var/log/netwitness/integration-server/integration-server.log`). For an example, see Integration-Server SMTP Notification Error Example. **Note:** For any notification-related troubleshooting, check the `integration-server log` file in addition to the log file of the service creating the notification.

Problem	Possible Causes	Solutions
I created a rule with an enrichment, added an SMTP notification, and deployed my rule. We are not receiving SMTP notifications. Why?	You do not have a template that met the criteria to parse the events.	Check the ESA Correlation service log files to see if the SMTP notification failed: `/var/log/netwitness/correlation-server/correlation-server.log`. For more details on the notification error, check the Integration-Server log file on the NetWitness Server (also known as Node 0, Admin server, or NWServer): `/var/log/netwitness/integration-server/integration-server.log`. If you use an ESA rule that has an enrichment, such as a Context Hub list, you must create a custom template. You can duplicate a default template and adjust it for your enrichment. See SMTP Notification Error Example below for a notification error example. For information on creating a custom ESA template, see "Define a Template for ESA Alert Notifications" in the *System Configuration Guide*. Go to the NetWitness All Versions Documents page and find NetWitness Platform guides to troubleshoot issues.
I created a custom rule (via the Rule Builder or Advanced EPL), and my rule is not firing. Why?	You may have connectivity issues.	Check the Offered Rate statistic on the  **(Configure) > ESA Rules > Services** tab. Select the ESA service and then look at the statistics on the tab for the Deployment. If the Offered Rate is zero, then the ESA service is not receiving data from Concentrators. Check the ESA Correlation log files for connectivity issues: `/var/log/netwitness/correlation-server/correlation-server.log`. If the offered rate is not zero, the meta key name and type used in the rule likely doesn't match the meta key present in events. Check to see if the meta key name and type used in the rule is valid by searching for the meta key name in  **(Configure) > ESA Rules > Settings** tab (Meta Key References).

Problem	Possible Causes	Solutions
I created a custom rule (via the Rule Builder or Advanced EPL), and my rule is not firing. Why?	There may be a problem with the rule.	**If a specific rule is not firing**, go to **(Configure) > ESA Rules > Services** to see if the rule was disabled. In the **Deployed Rule Stats** section, a rule that is disabled displays a clear enabled button (instead of the green enabled button). You can also check Events Matched field. Go to **(Configure) > ESA Rules > Services**. From there, you can see the number of events that were matched in the **Events Matched** column. If no events matched, check the logic of your rule for errors. For example, check the syntax for uppercase and lowercase errors, and check the time window. If the rule still doesn't fire, consider simplifying the logic of the rule to see if it fires when there is less complexity.
After a recent upgrade, I am not seeing alerts and I am seeing disabled rules.	There may be a problem with the ESA rule deployment.	Deploy the ESA rule deployments again. Create a Deployment steps in the Live Services Management Guide provides more information on deploying rules using the ESA Correlation service. If this does not resolve the issue, check the ESA Correlation log files for more information: `/var/log/netwitness/correlation-server/correlation-server.log`.
After an update or upgrade to 11.3.0.2 or later, if I try to make an adjustment to some rules, I get an error when trying to save them.	The Ignore Case option may be selected for a meta key that does not contain alphabetic values, such as IP address.	In NetWitness Platform 11.3.0.2 and later, the Ignore Case option has been removed from the ESA Rule Builder - Build a Statement dialog for meta keys that do not contain text values. Adding Ignore Case on meta keys which do not contain alphabetic values causes additional processing to occur for no added benefit. In the ESA Rule Builder - Build a Statement dialog, check to see if you have any meta keys that do not contain alphabetic characters, for example, ip_src and ip_dst. If you do, clear the **Ignore Case** checkbox for those meta keys and try to save the rule again.

Problem	Possible Causes	Solutions
After an upgrade to 11.3.0.2 or later, I see a warning message in the ESA Correlation service log file showing a difference between the multi-valued and default-multi-valued parameter meta key values. Why?	You do not have the required meta keys on ESA Correlation that the Endpoint, UEBA, and Live content rules need to work.	If you want to use the latest Endpoint, UEBA, and Live content rules, add the necessary meta keys to the `multi-valued` and `single-valued` parameter fields. For detailed information and instructions, see "Update Your ESA Rules for the Required Multi-Value and Single-Value Meta Keys" in the *ESA Configuration Guide*. For example warning messages, see Example ESA Correlation Server Warning Message for Missing Meta Keys.
Meta keys marked as sensitive for data privacy are still included in notifications and alerts for some rules.	In ESA rules that do not select every piece of meta from the session (that is, using **`select *`**), you may see that data privacy (if enabled) and the **Pivot to Investigate > Navigate** link accessed from a context tooltip in the Respond Incident Details view does not work.	The following steps apply to all released versions of NetWitness 11.3 and later. In 11.4 and later, you do not need to follow these steps for data privacy. However, you need to follow these steps if you want to enable the **Pivot to Investigate > Navigate** link accessed from a context tooltip in the Respond Incident Details view. 1. Add the ESA generated `event_source_id` meta key to the `index-concentrator-custom.xml` file. 2. Add the `event_source_id` meta key to the SELECT statement within any ESA rule that does not select every piece of metadata from the session. 3. After the Concentrator changes take effect, redeploy the ESA rule deployment that contains the ESA rule. To do this, see Update any ESA Rule that Selects Only Certain Meta Keys from the Session to Include event_source_id. For NetWitness 11.4 and later, to resolve the data privacy issue, see How to Remove Sensitive Meta Keys Globally from All Alerts.

Problem	Possible Causes	Solutions
The **Pivot to Investigate > Navigate** link does not work in a context tooltip accessed from Respond.	In ESA rules that do not select every piece of meta from the session (that is, using `select *`), you may see that data privacy (if enabled) and the **Pivot to Investigate > Navigate** link accessed from a context tooltip in the Respond Incident Details view does not work.	For NetWitness 11.3 and later (including 11.4), see Update any ESA Rule that Selects Only Certain Meta Keys from the Session to Include event_source_id.
I added a data source filter to the data sources in my ESA rule deployment. It was working fine and then all of a sudden, I stopped receiving alerts.	If there are any adjustments to an application rule on the Decoders that are mapped to the data sources used in your data source filter, the filter does not work for that rule since the application rule used in the filter no longer exists.	The optional data source filter is available in NetWitness 11.5 and later. If an application rule linked to a data source filter is modified on a Decoder, the filter must be removed, added again, and redeployed. The changes take effect on ESA after the deployment is redeployed.
I added a data source filter to the data sources in my ESA rule deployment and I see an "Invalid header size" error while communicating with Core services in the ESA Correlation log file.	You are filtering out a large portion of the traffic and less sessions match the aggregation criteria.	In the Explore view node list for an ESA Correlation service, select **correlation > stream**. Decrease the `max-sessions` parameter from 10,000 to a lower session count and restart the ESA Correlation service. For step-by-step instructions, see .
I created a rule using a Context Hub list and it was working properly, but suddenly, the rule stopped firing and ESA is not processing any rules.	If a Context Hub list that is used by ESA Correlation is renamed or deleted, ESA will not be able to access the list and may halt processing for all rules.	Do not rename or delete a Context Hub list that is used in a deployed ESA rule. ESA Correlation will also not be able to access a Context Hub list if you delete it and then add it again with the same name while the rule is deployed. If you rename a Context Hub list or recreate the Context Hub list with the same name, update the ESA rules that use that Context Hub list, and then redeploy the ESA rule deployments that contain those rules.

SMTP Notification Error Example

The following SMTP notification error example is an excerpt from a `correlation-server.log` file, which shows an error message for sending notifications with unsupported templates. In this example, there is a rule that is configured with the GeoIP enrichment, which has a hash table as one of its fields (the GeoIPLookup meta). Because the default SMTP template is only designed to deal with metas that are either singular values or arrays that contain only singular values, such as `"ip.src":"1.1.1.1"` and `"action":["fw:inbound-network-traffic"]`, sending the email notification fails due to the array containing a hash table.

```
FTL stack trace ("~" means nesting-related):

- Failed at: ${value!""} [in template "smtp.ftl" in macro "value_of" at line
1, column 152]

- Reached through: @value_of metadata[key] [in template "smtp.ftl" at line 85,
column 141]

----

...

For "${...}" content: Expected a string or something automatically convertible
to string (number, date or boolean), or "template output" , but this has
evaluated to an extended_hash (LinkedHashMap wrapped into
f.t.DefaultMapAdapter):

==> value!"" [in template "smtp.ftl" at line 1, column 154]
```

Integration-Server SMTP Notification Error Example

The following SMTP notification error example is an excerpt from an `integration-server.log` file, which shows a failure when the Integration-server attempts to send an email notification to the email notification server. In this case, you should check the email notification server configuration in the Global Notifications settings (**(Admin) > System > Global Notifications > Servers** tab).

```
2019-10-09 18:53:42,015 [-SMTP-5c45c867e4b03b89a49b78ba] WARN
Notification|SMTP dispatch failed (Reason: Sending the email to the following
server failed : email.server.com:25)

2019-10-09 18:53:42,100 [-SMTP-5c45c867e4b03b89a49b78ba] WARN
SystemOperation|Failed to forward ResolvedNotification
{server=5c45c867e4b03b89a49b78ba, destination=5c45c854e4b03b89a49b78b9,
content-length=30681}

java.lang.IllegalArgumentException: org.apache.commons.mail.EmailException:
Sending the email to the following server failed : email.server.com:25
```

Example ESA Correlation Server Warning Message for Missing Meta Keys

If you see a warning message in the ESA Correlation server error logs for missing multi-valued meta keys, there is a difference between the `default-multi-valued` parameter and `multi-valued parameter` meta key values, and the new Endpoint, UEBA, and Live content rules will not work. The same is true for missing single-valued meta keys. Completing the "Update the Multi-Valued and Single-Valued Parameter Meta Keys for the latest Endpoint, UEBA, and RSA Live Content Rules" procedure in the *ESA Configuration Guide* should fix the issue.

Multi-Valued Warning Message Example

```
2019-08-23 08:55:07,602 [ deployment-0] WARN Stream|[alert, alert_id,
browserprint, cert_thumbprint, checksum, checksum_all, checksum_dst,
checksum_src, client_all, content, context, context_all, context_dst,
context_src, dir_path, dir_path_dst, dir_path_src, directory,
directory_all, directory_dst, directory_src, email_dst, email_src,
feed_category, feed_desc, feed_name, file_cat, file_cat_dst, file_cat_
src, filename_dst, filename_src, filter, function, host_all, host_dst,
host_orig, host_src, host_state, ip_orig, ipv6_orig, OS, param, param_
dst, param_src, registry_key, registry_value, risk, risk_info, risk_
suspicious, risk_warning, threat_category, threat_desc, threat_source,
user_agent] are still MISSING from multi-valued
```

Single Value Warning Message Example

```
2019-08-23 08:55:07,602 [ deployment-0] WARN Stream|[accesses, context_
target, file_attributes, logon_type_desc, packets] are still MISSING
from single-valued
```

Update any ESA Rule that Selects Only Certain Meta Keys from the Session to Include event_source_id

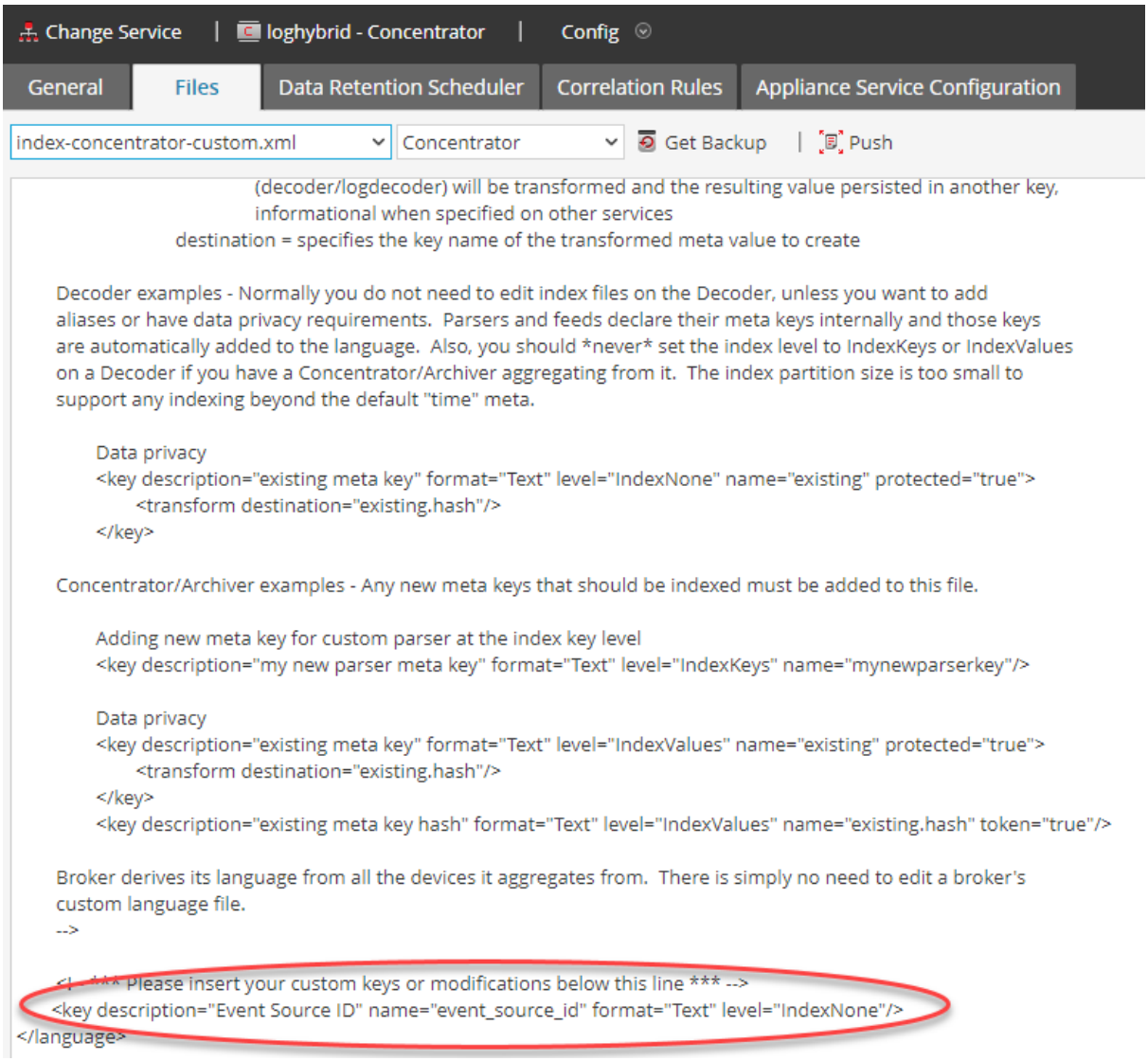
In ESA rules that do not select every piece of meta from the session (that is, using **`select *`**), you may see that data privacy (if enabled) and the **Pivot to Investigate > Navigate** link accessed from a context tooltip in the Respond Incident Details view does not work.

The following steps apply to all released versions of NetWitness 11.3 and later. In 11.4 and later, you do not need to follow these steps for data privacy, instead, see How to Remove Sensitive Meta Keys Globally from All Alerts. However, you need to follow these steps if you want to enable the **Pivot to Investigate > Navigate** link accessed from a context tooltip in the Respond Incident Details view.

Note: Do not use any Esper keyword as custom meta keys since this causes an error while creating an ESA Rule. For Esper keywords, see Reserved keywords.

1. Update the **`index-concentrator-custom.xml`** file to include the ESA generated **`event_source_id`** meta key. If you do not add the meta key, ESA cannot recognize it and the rule will fail to deploy.
   The following figure shows the file configured for the custom meta key "**`Event Source ID`**" with index settings of "**`IndexNone`**" with a format of "**`Text`**".

   <key description="Event Source ID" name="event_source_id" format="Text" level="IndexNone"/>



To save and deploy the new setting on the NetWitness host, select the **Apply** button. To force the change, restart the Concentrator service or you can wait until the next polling interval for the change to be recognized.

The XML file can also be deployed to other NetWitness hosts by clicking the **Push** button and selecting the destination NetWitness host. Only deploy the XML file to a NetWitness host that runs that service (that is, other Concentrators).

2. Update any ESA rule that selects only certain meta from the session to include the ESA generated **`event_source_id`** meta key. Add the **`event_source_id`** meta key to the **`SELECT`** statement. See the following example rule.

Example Rule with event_source_id

```
@RSAAlert
SELECT user_dst, reference_id, hostname, event_source_id FROM
Event(
```

```
device_class='Windows Hosts',
reference_id IN ('4624' , '4625'),
user_dst IS NOT NULL,
user_dst NOT LIKE '%$',
user_dst NOT IN ('ANONYMOUS LOGON','SYSTEM')
)
.win:time_batch(10 Minutes)
GROUP BY user_dst
HAVING COUNT(distinct hostname) >= 15;
```

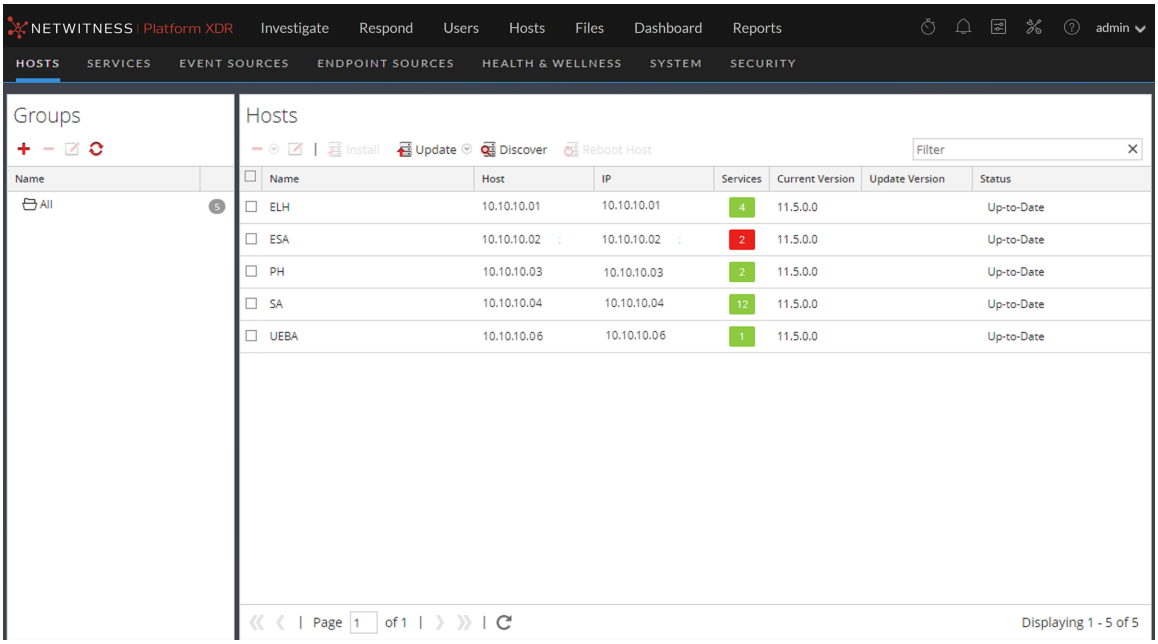
3. After the Concentrator changes takes effect, redeploy the ESA rule deployment that contains the ESA rule.

For additional information, see How ESA Handles Sensitive Data. For information on the stragegy and benefits of obfuscating data, see the *Data Privacy Management Guide*.

Steps to Troubleshoot Memory Issues with an ESA Service Offline

Step 1: Verify that your Host Is Running

The first step to troubleshooting is to ensure that your host is running. To do this, go to  **(Admin) > Hosts**. If the host is down, the system parameters will not display (updating host information can sometimes be delayed), the **Services** display in red, and you may see an error message.

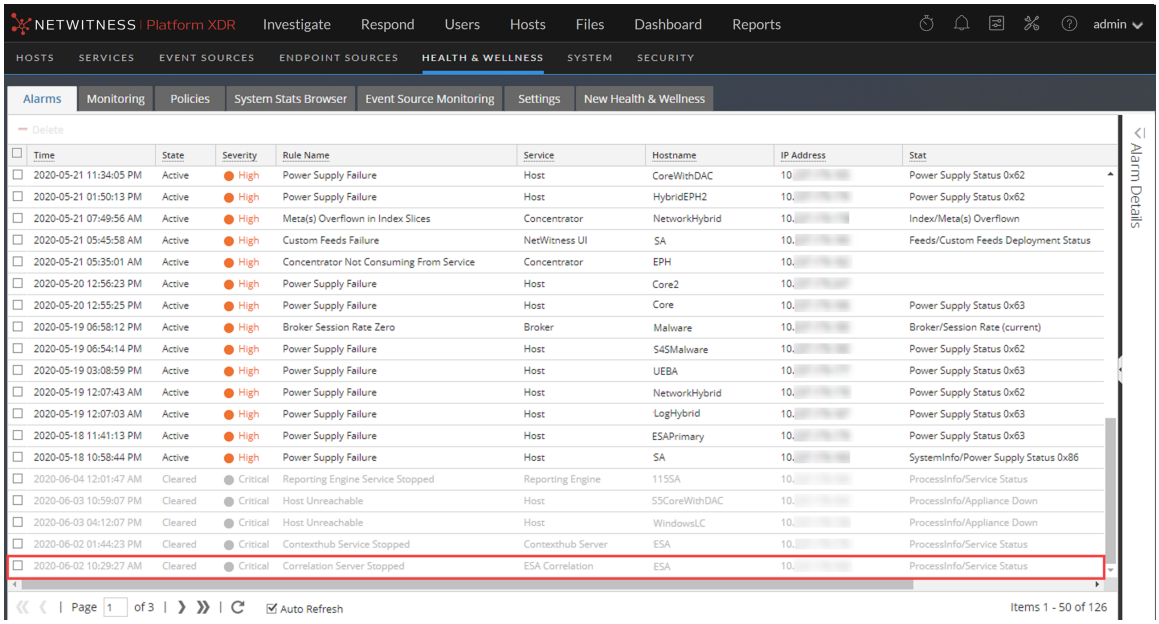


If your host is down, contact your NetWitness Administrator to restart it. Otherwise, go to Step 2.

Step 2: View Detailed Statistics in Health & Wellness

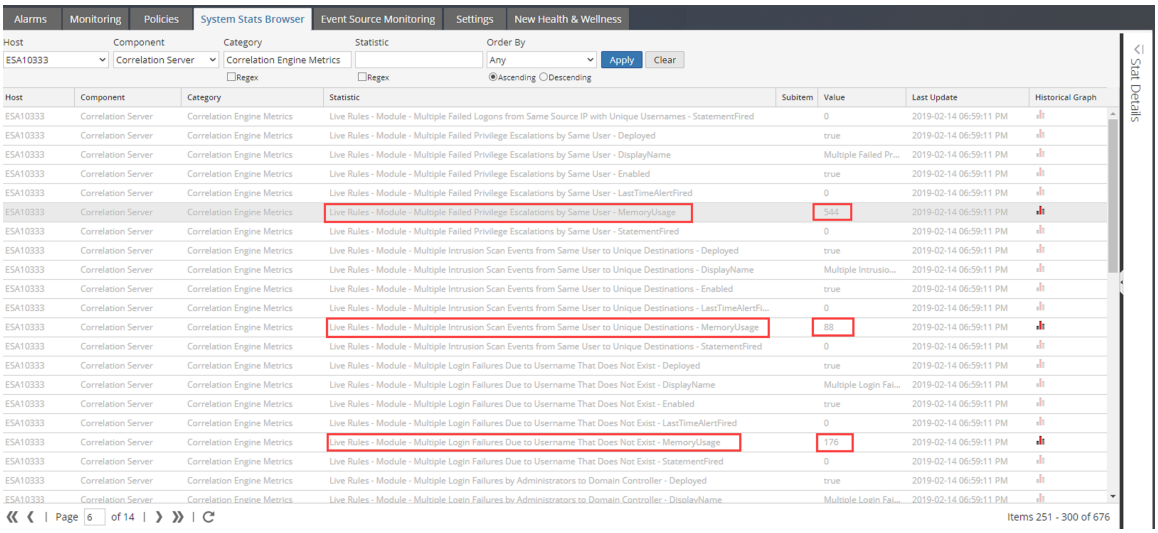
If your ESA service is down, you can go to Health & Wellness and view the **last known metrics** to see where potential issues are occurring. The most common problem is that your ESA service is exceeding memory thresholds, which causes it to stop or fail. In NetWitness 11.5 and later, see also View Health Statistics and Trends for ESA Correlation in New Health & Wellness.

1. Go to **(Admin) > Health & Wellness > Alarms** to see if the ESA triggered any alarms. Look for the following alarms for ESA Correlation:
   - Correlation Server in Critical State
   - Correlation Server in Unhealthy State
   - Correlation Server Stopped



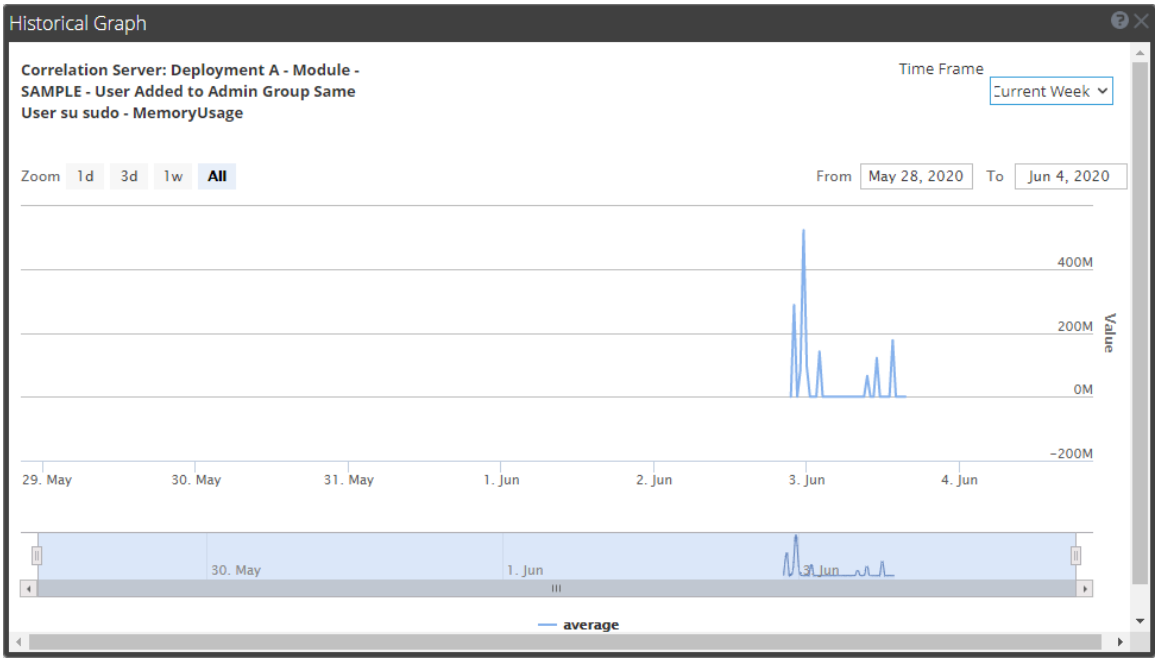
2. Go to **(Admin) > Health & Wellness > System Stats Browser** to see the memory metrics for each rule's performance. To view the metrics, enter the following and click **Apply**:

Host	Component	Category
<your host>	Correlation Server	Correlation Engine Metrics

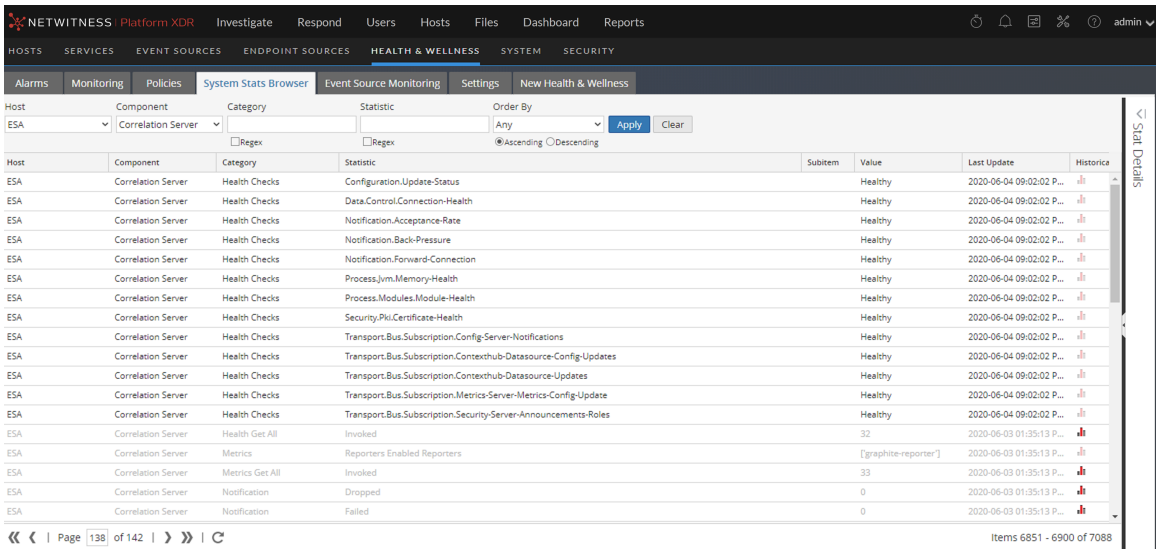


The name of the rule is in the **Statistic** column and the memory usage in bytes is in the **Value** column.

3. Click  to view a historical view of memory usage for the rule in the **Historical Graph** column.



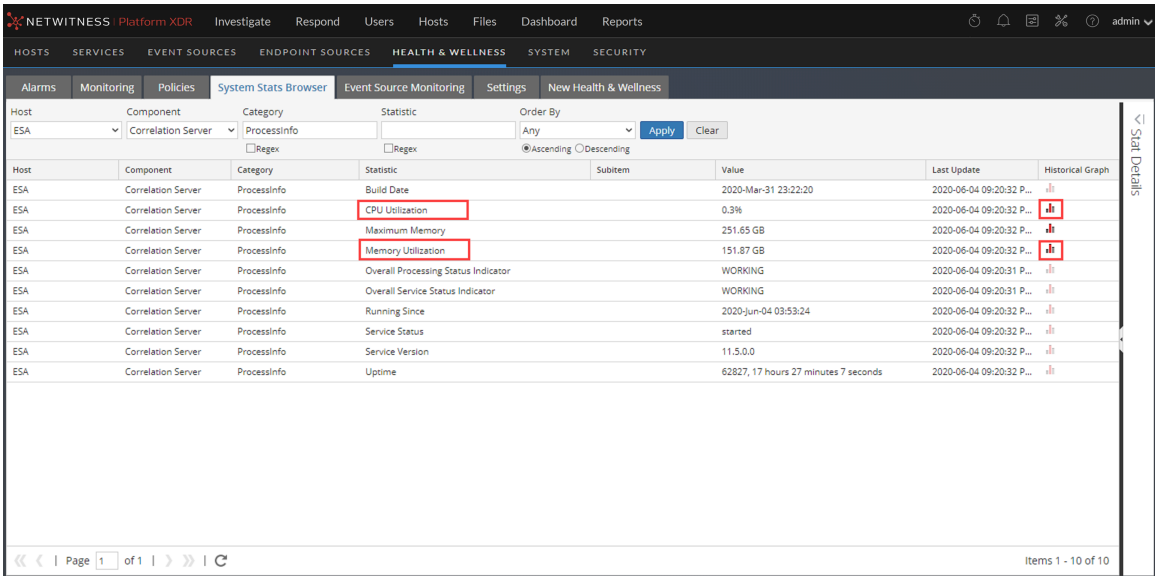
4. In the **System Stats Browser**, you can also see details of your ESA Correlation service performance.



Select your host, and use the following filters to view the following statistics:

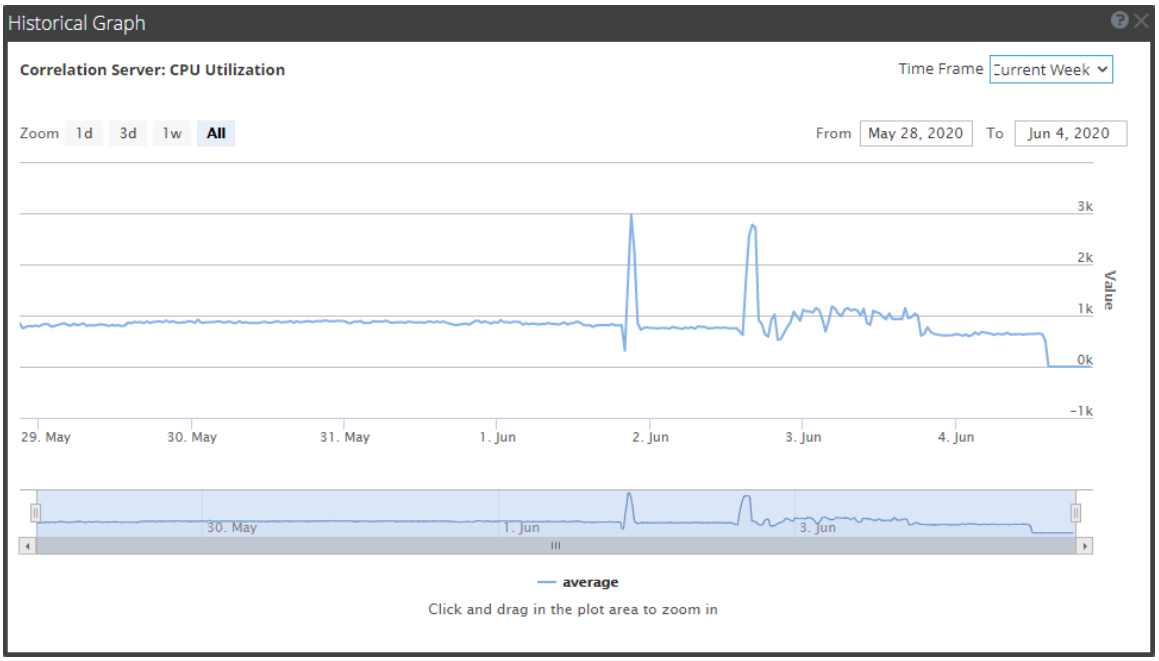
Host	Component	Category	Statistic	Example
<your host>	Host	SystemInfo	CPU Utilization	49.74%
<your host>	Host	SystemInfo	Memory Utilization	72.87%
<your host>	Host	SystemInfo	Used Memory	183.14 GB
<your host>	Host	SystemInfo	Total Memory	251.65 GB
<your host>	Host	SystemInfo	Uptime	289546, 3 days 8 hours 25 minutes 46 seconds
<your host>	Correlation Server	Process jvm	Memory Total Max	163 GB
<your host>	Correlation Server	Process jvm	Memory Total Used	13.50 GB
<your host>	Correlation Server	ProcessInfo	CPU Utilization	0.3%
<your host>	Correlation Server	ProcessInfo	Maximum Memory	251.65 GB
<your host>	Correlation Server	ProcessInfo	Memory Utilization	151.87 GB

The following figure shows the location of the ESA Correlation service CPU and Memory Utilization statistics.

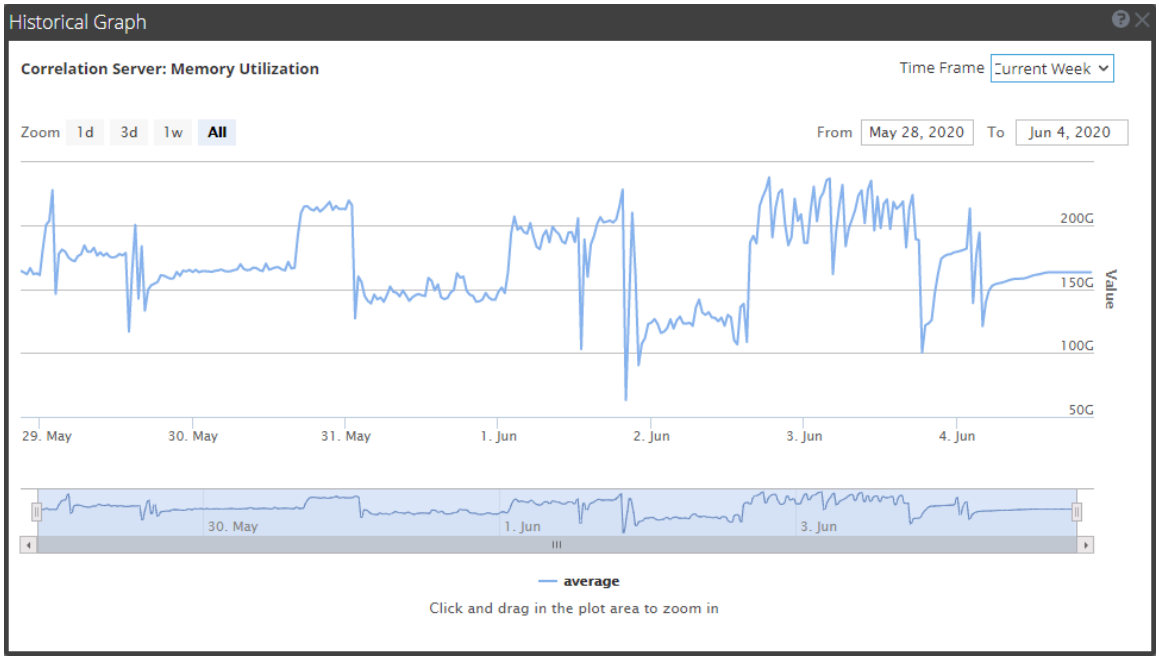


5. Click  to view a historical view of CPU and memory utilization.

   The following figure shows the historical graph of **CPU utilization**.



The following figure shows the historical graph of **Memory Utilization**.



If you are having a problem with memory or CPU utilization, continue to step 3.

Step 3: Bring up your ESA Services

1. Go to **(Admin) > Services**, select your ESA service, and then select **> Start**.
2. Return to the ESA Service to troubleshoot which rules have created memory issues.

If your ESA service is stopping and restarting in a loop, you may need to call Customer Support to get the services to start.

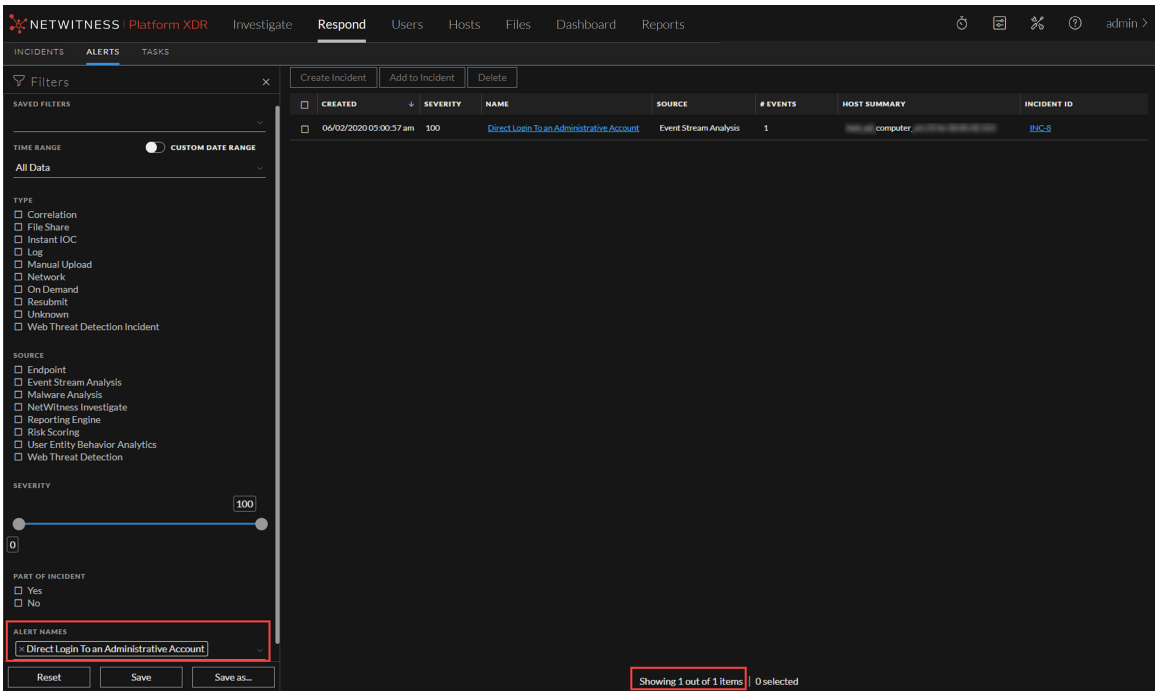
If you are able to start your ESA service without a shutdown, continue to step 4.

Step 4: Check the Alerts and Events Volume

After you are able to restart your ESA service without an immediate shutdown, you can review the stats for your rules to see which rules are consuming too many resources. Sometimes, ESA services fail because a rule is generating too many alerts or a rule is matching too many events. Check for both of these issues if you have determined that memory usage is causing your ESA service to shut down.

View Alert Summaries

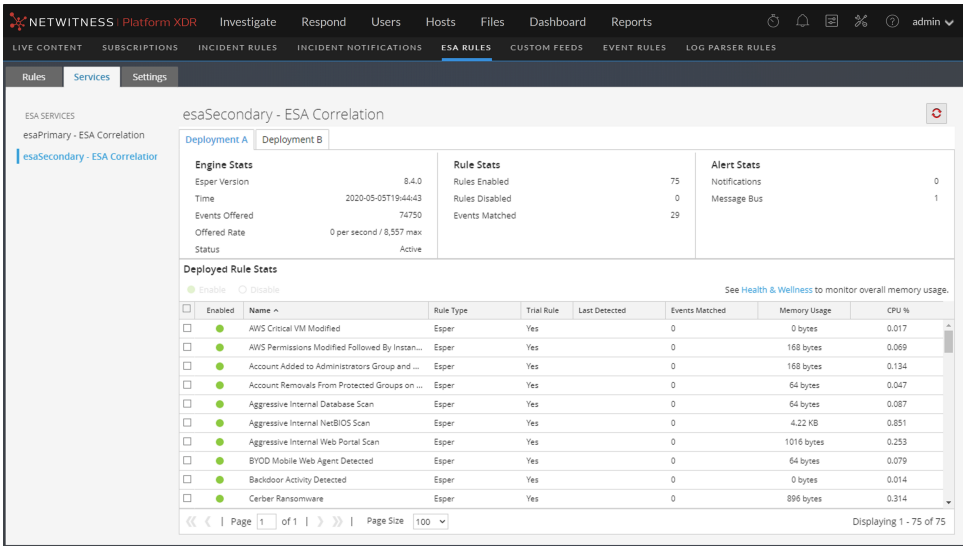
Rules that generate a high volume of alerts can overwhelm the system and cause it to fail or restart. To view the alert summaries, go to **Respond > Alerts**. In the **Filters** panel on the left, in the **Alert Names** section, select the alert name for the rule. The number of alerts with that name appears at the bottom of the Alerts list results. If the number is significantly high for a particular rule, you need to disable the rule and rewrite it to be more efficient.



To clear your filter, click **Reset**.

View Events Matched

Sometimes a rule matches too many events, which can use up excessive memory. This typically occurs if you create a large event window where a great number of events accumulate without triggering an alert. This is a problem because each event is stored in memory while the rule waits for the alert to trigger. To check for this issue, go to **(Configure) > ESA Rules > Services**. From there, you can see the number of events that were matched in the **Events Matched** column for the deployment. If a high number of events were matched for a given rule, you can investigate the rule further to see if you can make it more efficient.



Step 5: Disable and Repair the Rule that Caused Issues

Once you have determined the rules that need to be rewritten, disable them and rewrite rules so that they don't generate such a high volume of alerts or events. For pointers on how to write more efficient rules, see Best Practices.

Disable Rules

1. To disable rules, go to **(Configure) > ESA Rules > Services**, and select the rules you want to disable in the **Deployed Rules Stats** field.
2. Select **Disable** to disable the rules.

Edit Rules

1. To repair the rules, go to **(Configure) > ESA Rules > Rules tab > Rule Library**.
2. For each rule that you repair, do the following:
   a. Select the rule to edit and then select **> Edit**.
   b. Edit the rule to be more efficient. For instructions on creating rules, see Add Rules to the Rule Library
   c. When you are satisfied with your rule, you can save the rule as a trial rule to ensure that any memory issues do not affect ESA services performance. To do this, follow the steps listed in Working with Trial Rules.

Deploy Rules

To deploy rules, see topic Create a Deployment in the *Live Services Management Guide*.

Verify that the Rules are Enabled

After you deploy the ESA rules, they should automatically show as enabled. If not, you can enable the rules.

1. Go to **(Configure) > ESA Rules > Services** tab, and select the ESA service in the options panel.
2. On the deployment tab for the deployment that contains the rules, in the Deployed Rule Stats section, look at the status of the rules in the Enable column. Enabled rules show a green circle. If the rules show a white circle, you can enable the rules.
3. To enable rules, select the rules you want to enable and select **Enable** above the table.

(Optional) Check the ESA Correlation Log Files for More Information

Once you verify that your services are down and some potential causes for the system going down, check to see if the service is stopping and restarting in a loop. To do this, go to the ESA Correlation logs. You can use SSH to get in the system and go to: `/var/log/netwitness/correlation-server/correlation-server.log`.

ESA Rule Troubleshooting with Nw-Shell

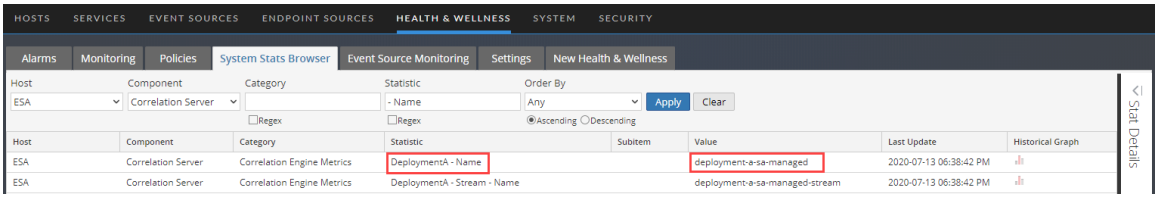
Note: This procedure applies to NetWitness Platform 11.3 and later versions.

The ESA Correlation service replaces the Event Stream Analysis service in NetWitness version 11.3 and later. As a result of this change, some settings are no longer available in the user interface. In addition to the standard troubleshooting methods available, you can use the **nw-shell** utility to perform advanced troubleshooting of the ESA Correlation service and rules. For detailed information on the nw-shell utility, see the *NetWitness Shell User Guide*.

Find Your Engine Name for Nw-Shell

Follow these steps to find your engine name using your ESA rule deployment name. Your engine name is required for ESA Nw-Shell troubleshooting. Locate the names of each deployment that you plan to troubleshoot.

1. In the NetWitness UI, go to **(Admin) > Health & Wellness > System Stats Browser**.
2. In the System Stats Browser, use the following filters and then click **Apply**.
   a. **Host**: Select your ESA host.
   b. **Component**: Select **Correlation Server**.
   c. **Statistic**: Type **- Name** (put a space between the dash and name).
3. In the **Statistic** column, locate your deployment followed by **- Name**. The name in the **Value** field is `YOUR ENGINE NAME`.



In the above example, the deployment name is DeploymentA and the engine name is `deployment-a-sa-managed`.

Connect to an ESA Correlation Server

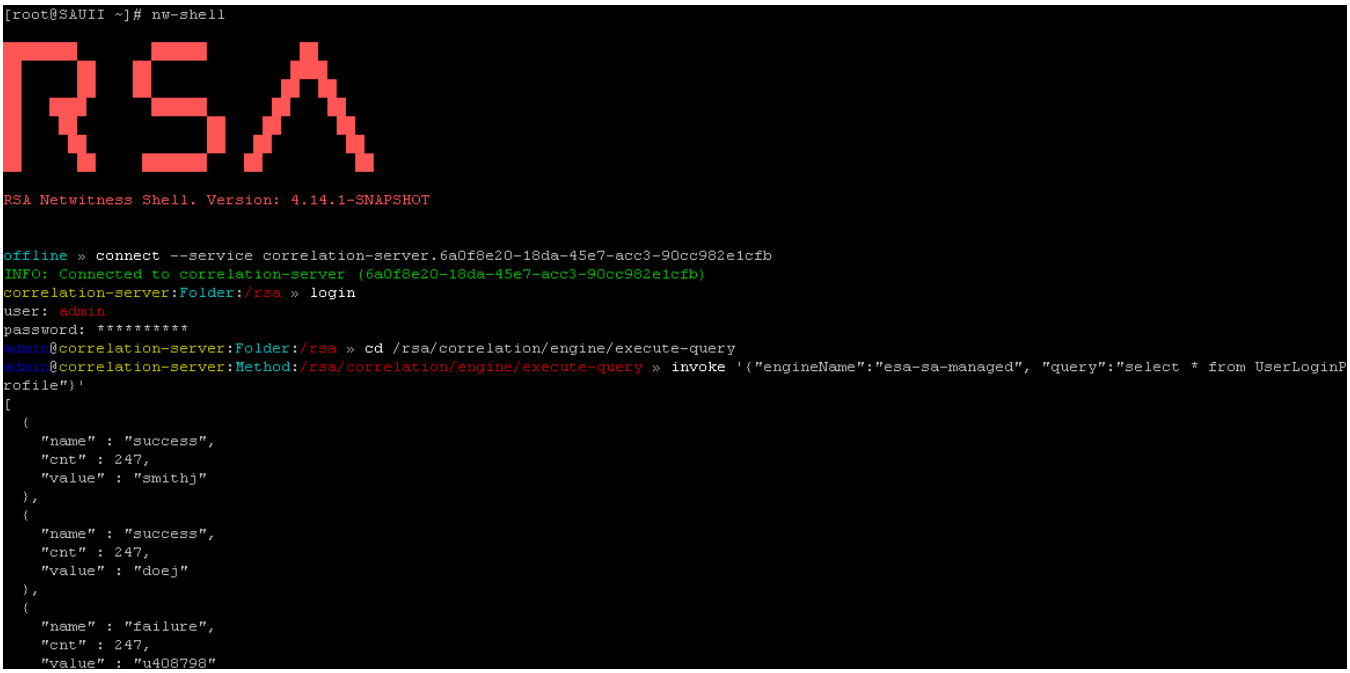
1. Log in to nw-shell:
   a. Connect via SSH to the NW server (head unit).
   b. After logging in and getting the command prompt, type `nw-shell`.
2. Connect to ESA:
   a. Go to your ESA physical host and type the command:
   `cat /etc/netwitness/correlation-server/service-id`
   b. Go back to the NW server and nw-shell and connect to the correlation server:
   `connect --service correlation-server.ID`
3. Log in to ESA with the admin user credentials.
   a. Type `login`.
   b. Enter the username and password of the admin user.

View the Contents of a Named Window

Use the execute-query method to view the contents of a named window.

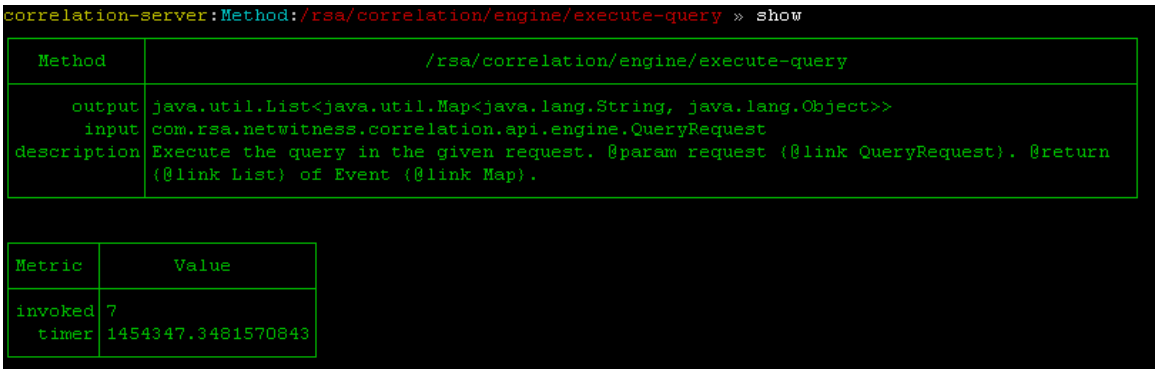
1. After you are connected to the ESA correlation service and authenticated, type:
   `cd /rsa/correlation/engine/execute-query`
2. Type `invoke '{"engineName":"<YOUR ENGINE NAME>", "query":"<YOUR QUERY>"}`
   - Where `<YOUR ENGINE NAME>` = the engine name that you located in Find Your Engine Name for Nw-Shell.
   - Where `<YOUR QUERY>` = the select statement into the named window.

Example: `invoke '{"engineName":"esa-sa-managed", "query":"select * from UserLoginProfile"}'`



See the Method Input and Output

Type `show` at the command line to see the input and output expected. All method invocation must be prefaced by `invoke`.



Obtain Correlation Server Metrics for ESA Rule Deployment Troubleshooting Using Nw-Shell

Note: This procedure is available in NetWitness Platform version 11.4.1 and later.

You can use Nw-Shell to view ESA Correlation Server metrics for each of your ESA rule deployments. These metrics show the number of sessions behind for the deployment data sources as well as the memory usage for the rules in the deployment.

1. Find the engine name to use for Nw-Shell. See Find Your Engine Name for Nw-Shell.
2. Connect to an ESA Correlation Server. See Connect to an ESA Correlation Server.
3. After you are connected to the ESA correlation service and authenticated, type:
   `cd /rsa/correlation/service/stats/get-condensed-metrics`
4. Type `invoke '<YOUR ENGINE NAME>'`
   Where `<YOUR ENGINE NAME>` is the engine name that you located in Find Your Engine Name for Nw-Shell.

Here is an example of the metrics output that you can obtain for your ESA rule deployment:

```
{
  "engineName" : "esa",
  "eventsOffered" : 1650,
  "maxEventsRate" : 2.337019271237945,
  "eventsRate" : {
    "count" : 1650,
    "oneSecRate" : 0.02129597415982235,
    "meanRate" : 0.05795554387594279,
    "oneMinuteRate" : 0.15485195471608212,
    "fiveMinuteRate" : 0.12419048320215775,
    "fifteenMinuteRate" : 0.11923922260543295
  },
  "streamMetrics" : {
    "pollingRate" : {
      "count" : 30100,
      "meanRate" : 1.0572440914676082,
      "oneMinuteRate" : 1.1497867142612552,
      "fiveMinuteRate" : 1.1213207743063618,
      "fifteenMinuteRate" : 1.1178158863433476
    },
    "positionTracking" : 475,
    "polling" : 30100,
    "bufferedRecords" : 0,
    "incomingRecords" : {
      "count" : 1650,
      "meanRate" : 0.05796815004652209,
      "oneMinuteRate" : 0.16275810590703457,
      "fiveMinuteRate" : 0.1253818211054517,
      "fifteenMinuteRate" : 0.11956773913684943
    },
    "outgoingRecords" : {
      "count" : 1650,
      "meanRate" : 0.05796815072883537,
      "oneMinuteRate" : 0.16275810590703457,
      "fiveMinuteRate" : 0.1253818211054517,
      "fifteenMinuteRate" : 0.11956773913684943
    },
    "sourceMetrics" : {
      "nw://admin@10.10.10.01:50005?compression=0&compressionLevel=6" : {
        "bufferedRecords" : 0,
        "incomingRecords" : {
          "count" : 1650,
          "meanRate" : 0.05796819356968545,
          "oneMinuteRate" : 0.16275810590703457,
          "fiveMinuteRate" : 0.12538181849661215,
          "fifteenMinuteRate" : 0.11956716772097528
```

```
        },
        "outgoingRecords" : {
          "count" : 1650,
          "meanRate" : 0.05796819300115741,
          "oneMinuteRate" : 0.16275810590703457,
          "fiveMinuteRate" : 0.12538181849661215,
          "fifteenMinuteRate" : 0.11956716772097528
        },
        "sessionsBehind" : 2,
        "sessionLastId" : "1645",
        "sessionRate" : "0",
        "lastReceivedSessionId" : 1645
      }
    }
  },
  "ruleMetrics" : [
    {
      "ruleName" : "create_persist",
      "memoryUsage" : 104,
      "cpuLockedTimePercentage" : 50.286,
      "cpuLockedTimeNanos" : 138253,
      "statementFired" : 0,
      "alertsFired" : 0
    },
    {
      "ruleName" : "test_persist",
      "memoryUsage" : 72,
      "cpuLockedTimePercentage" : 39.44,
      "cpuLockedTimeNanos" : 108433,
      "statementFired" : 0,
      "alertsFired" : 0
    },
    {
      "ruleName" : "test-per",
      "memoryUsage" : 0,
      "cpuLockedTimePercentage" : 10.275,
      "cpuLockedTimeNanos" : 28249,
      "statementFired" : 0,
      "alertsFired" : 0
    }
  ]
}
```

View Memory Metrics for Rules

This topic tells ESA rule writers how to view memory metrics for an ESA Correlation service and its associated ESA rules. You can see estimated memory usage for each rule running on a server, and you can use this information to modify your rule statements and conditions if they use too much memory.

Rules can sometimes consume more memory than you expect, causing ESA to slow down or stop. To see approximately how much memory a rule is using, you can view estimated memory usage for each rule in the Health & Wellness System Stats browser (you need permissions to access this module). You can use this information to modify your rules to be more efficient.

At a high level, you need to complete the following steps to use memory metrics to troubleshoot memory usage for rules:

1. Ensure you have the correct permissions to view the Health & Wellness module. For information on roles and permissions, see ESA Permissions.
2. View the memory statistics in Health & Wellness.
3. (Recommended) Configure Health & Wellness ESA policies to send an email if memory thresholds are exceeded. See "Manage Policies" in the *System Maintenance Guide* for instructions on sending email notifications.
4. Use the memory metrics data to modify rules to be more efficient, if necessary.

Note: You can also view memory metrics for ESA rules in the  **(Configure) > ESA Rules > Services** tab. See View Stats for an ESA Service.

Prerequisites

The following are requirements for using memory metrics:

- You must have the appropriate permissions to view Health & Wellness statistics.
- (Recommended) Configure the ESA Health & Wellness policy to send an email when memory thresholds are exceeded.

Note: Memory Metrics is always on for the ESA Correlation service; you do not have to enable it.

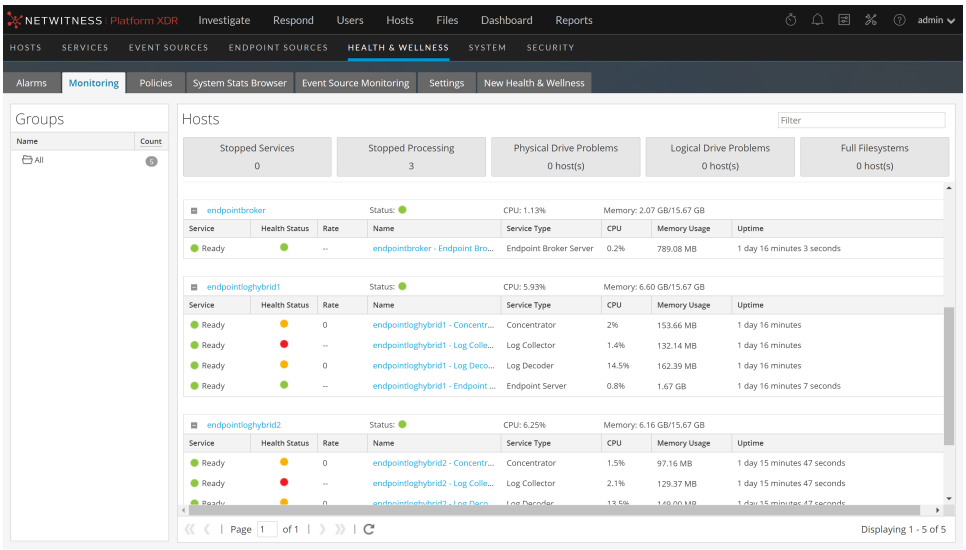
View Health Statistics and Trends for ESA Correlation in New Health & Wellness

In NetWitness version 11.5 and later, New Health & Wellness provides improved and intuitive dashboards, monitors, and visualizations. The ESA Correlation Overview dashboard provides health statistics and trends on ESA rule deployments.

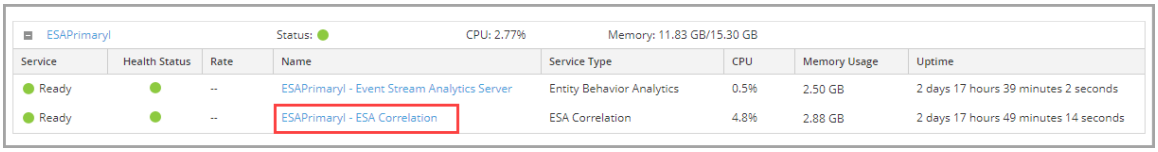
For more information, see "Monitor New Health and Wellness" and "Appendix A: New Health and Wellness Dashboards / ESA Correlation Overview Dashboard" in the *System Maintenance Guide*.

View Memory Metrics for an ESA Correlation Service in Health & Wellness

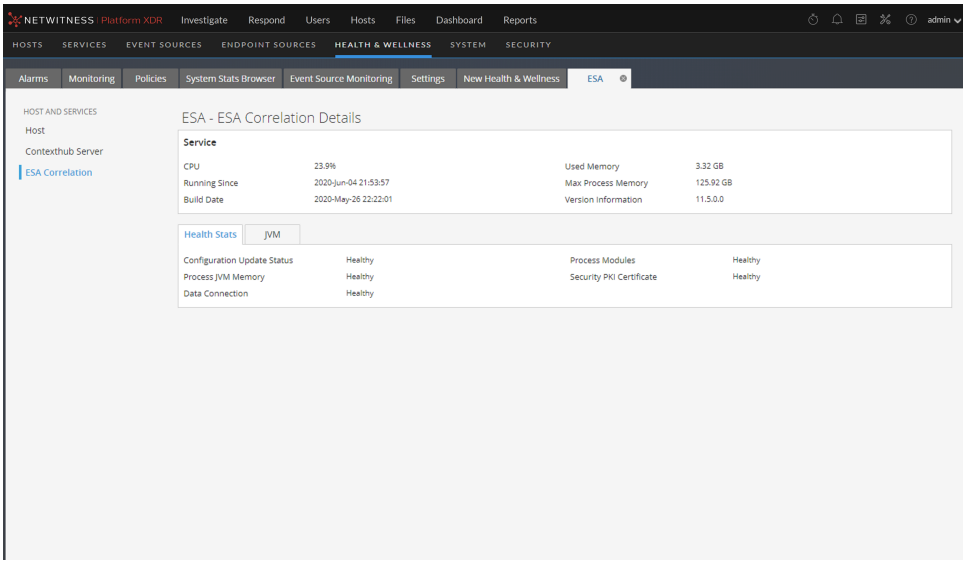
1. Go to **(Admin) > Health & Wellness > Monitoring** tab.



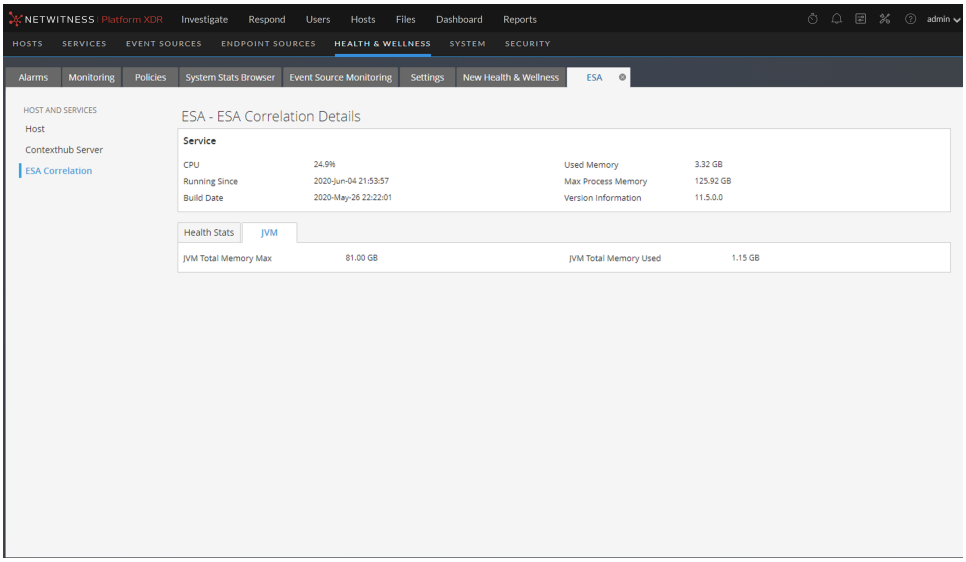
2. Locate your host and click the link in the **Name** field for your ESA Correlation service, for example, ESA- ESA Correlation.



3. On the tab for your ESA host, click the **Health Stats** tab.
   You can view the health status of the ESA Correlation service.



4. Click the **JVM** tab.
   You can view the JVM total memory used by the selected ESA Correlation service.

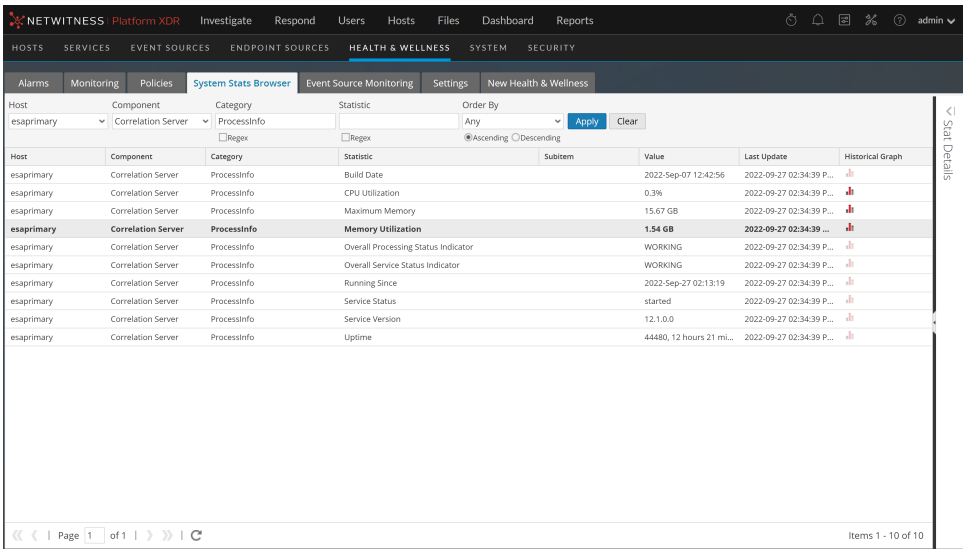


Note: You can also view memory metrics for the ESA Correlation service in the **(Configure) > ESA Rules > Services** tab. See View Stats for an ESA Service.

View Memory Metrics for an ESA Correlation Service and its ESA Rules

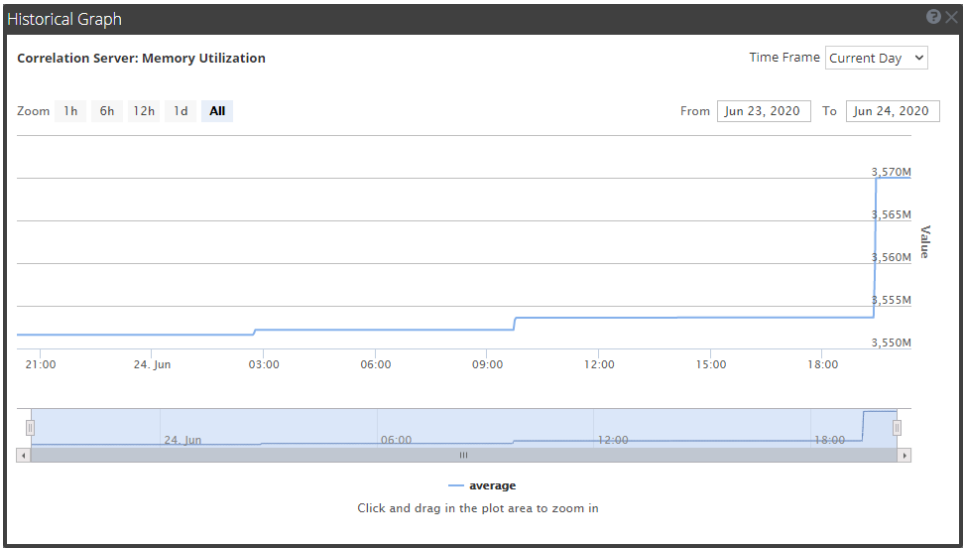
1. Go to  **(Admin) > Health & Wellness > System Stats Browser**.
2. To view memory metrics for an ESA Correlation service, in the **Host** field, select your ESA host. Select **Correlation Server** for **Component**, enter **ProcessInfo** for **Category**, and then click **Apply**.

Host	Component	Category
<your host>	Correlation Server	ProcessInfo



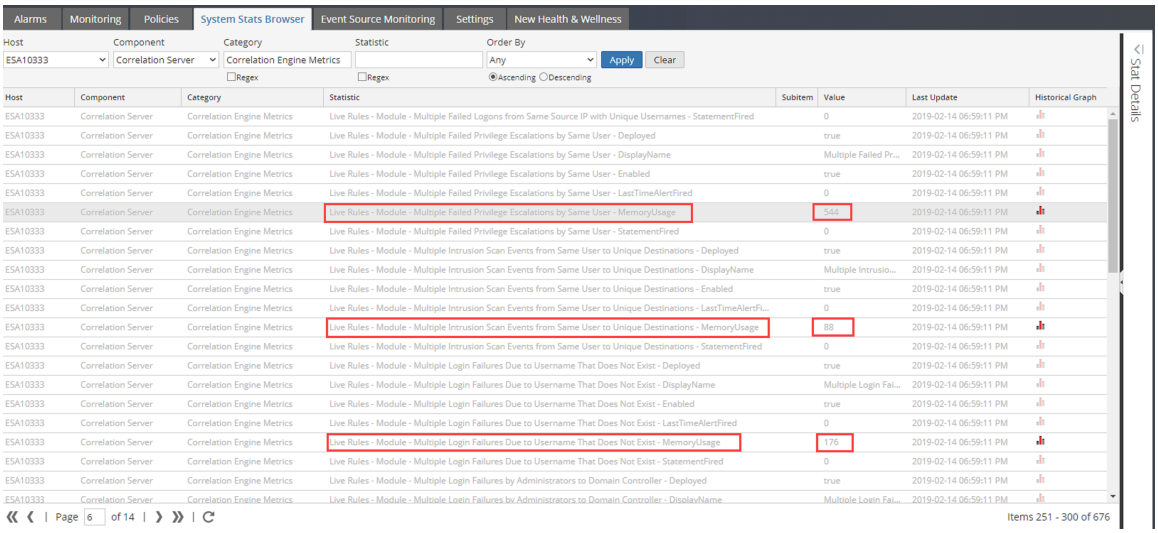
The **Memory Utilization** statistic shows the total memory in use by the ESA Correlation service.

3. To view the historical memory usage for the ESA Correlation service, click the **Historical Graph** icon.



4. To view the memory metrics for individual rules, in the **Category** field, enter **Correlation Engine Metrics** and click **Apply**.

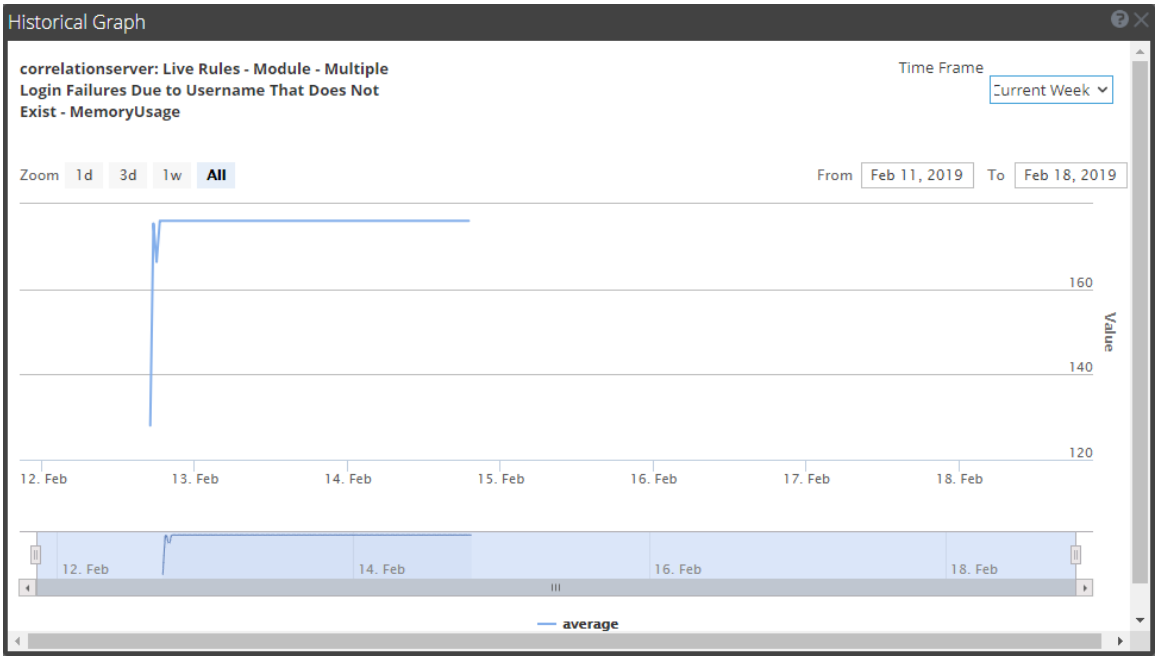
Host	Component	Category
<your host>	Correlation Server	Correlation Engine Metrics



The name of the rule is in the **Statistic** column appended with **MemoryUsage** and the memory usage in bytes is in the **Value** column.

Note: The **Last Update** field reflects when Health & Wellness polls ESA. However, the Memory Metrics is not synchronized with the Health & Wellness polling. For example, if the memory threshold is exceeded on 2/10/19 at 12 p.m., but Health & Wellness polls at 2/10/19 at 12:10 p.m., the **Last Update** field will display a timestamp of 2/10/19 12:10 p.m.

5. Click  to view a historical view of memory usage for the rule in the **Historical Graph** column.



Note: You can also view memory metrics for ESA rules in the  **(Configure) > ESA Rules > Services** tab. See View Stats for an ESA Service.

How ESA Handles Sensitive Data

This topic explains how ESA treats sensitive data, such as usernames or IP address, that it receives from Core services. The Data Privacy Officer (DPO) role can identify meta keys that contain sensitive data and should display obfuscated data. ESA does not display or store sensitive meta. Consequently, ESA will not pass sensitive data to NetWitness Respond.

Optionally, ESA can add an obfuscated version of the sensitive data to an event. For example, the DPO identifies `user_dst` as sensitive. ESA can add an obfuscated version, such as `user_dst_hash`, to an event. The obfuscated meta is not sensitive, so ESA will display and store it the same way as any other non-sensitive meta.

For more information on the strategy and benefits of obfuscating data, see the *Data Privacy Management Guide*.

This topic explains the following:

- How ESA treats sensitive data it receives from Core services
- How to prevent sensitive data leaks in an Advanced EPL rule
- How to remove sensitive meta keys from global alerts

How ESA Treats Sensitive Data from Core Services

When ESA receives sensitive data from Core services, ESA passes on only the obfuscated version of the data. ESA does not store or show sensitive data.

The following features are impacted:

- Outputs – ESA does not forward sensitive data to outputs, which include alerts, notifications, and MongoDB storage.
- Advanced EPL rules – If an EPL statement creates an alias for a sensitive meta key, sensitive data will leak. This topic illustrates how this happens so you can avoid it.
- Enrichments – If a sensitive meta key is used in the join condition, sensitive data will leak. This topic illustrates how this happens so you can avoid it.

Advanced EPL Rule

If an EPL query statement renames a sensitive meta key, the data will not be protected.

ESA identifies a sensitive meta key by the name:

- `ip_src` is the sensitive meta key.
- `ip_src_hash` is the non-sensitive, obfuscated version.

To support data privacy, the sensitive meta key must not be renamed in an EPL query. If a sensitive meta key is renamed, the data will no longer be protected.

For example, in a rule such as select `ip_src as ip_alias...`, `ip_alias` contains the sensitive data but it is not protected because ESA only knows about `ip_src`, not `ip_alias`. In this case, IP addresses would not be obfuscated. Real values would be displayed.

Enrichment Source

When a sensitive meta key is used in a join condition, sensitive data can be displayed.

The enrichment database, which is the other part of the join condition, has one column that matches the sensitive meta key. This cross reference is to actual values not obscured values. Consequently, actual values are displayed.

In the following example, both parts of the join condition are highlighted.

- `ip_src` contains sensitive data.
- `ipv4` will be added to the alert and exposed as non-sensitive data.

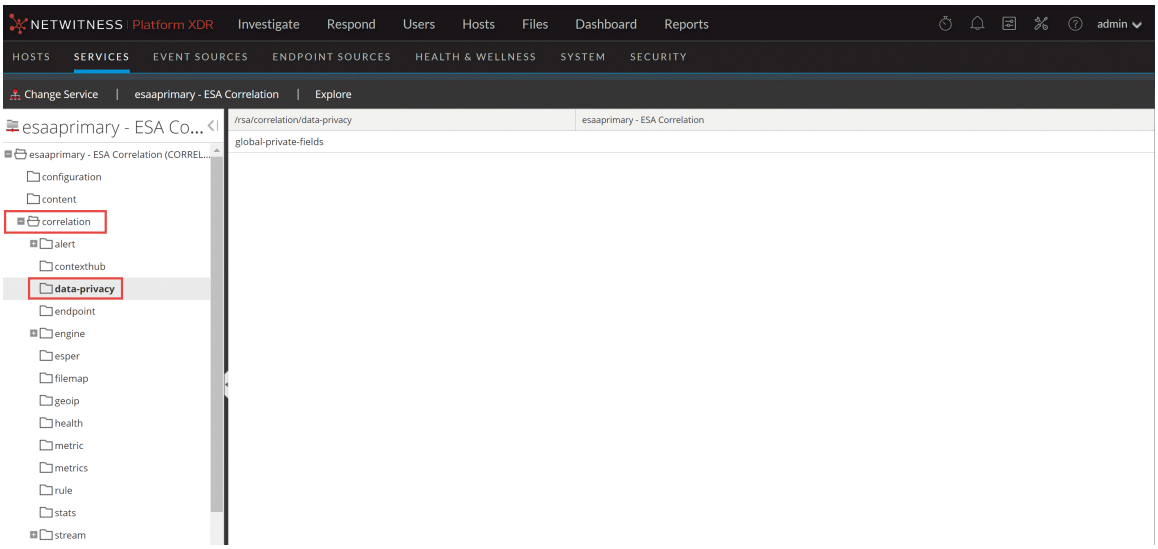
Because the `ipv4` value is the same as the `ip_src` value, `ipv4` contains and displays sensitive data.

How to Remove Sensitive Meta Keys Globally from All Alerts

Note: This procedure applies only to ESA Correlation Rules in NetWitness Platform 11.4 and later versions.

For data privacy reasons, it may be necessary to remove some sensitive meta keys from the alert output globally, regardless of the data source. In the ESA Correlation service, you can set the **`global-private-fields`** parameter to remove the meta keys from all alert output.

1. Go to **(Admin) > Services**, and in the Services view, select an ESA Correlation service and then select **> View > Explore**.
2. In the Explore view node list for the ESA Correlation service, select **correlation > data-privacy**.
3. In the **`global-private-fields`** parameter, add the sensitive meta keys that you want removed from all alerts.



The changes are effective immediately.

For more information on the strategy and benefits of obfuscating data, see the *Data Privacy Management Guide*.

ESA Rule Types

This topic describes each type of ESA rule, when to use them and the permissions each role has with them. The following table lists each type, describes it, and explains when to use it.

Rule Type	Description	When to Use
RSA Live ESA	RSA Live has a catalog of ESA rules that are available to modify to run in your network.	RSA Live ESA rules are available to leverage rules that are already built. Modify the configurable parameters to customize to meet your requirements.
Rule Builder	In the rule builder, you define rule criteria in an easy-to-use interface.	Use the rule builder to create your first rules. You choose many of the rule conditions from lists.
Advanced EPL	With the Event Processing Language (EPL), you define rule criteria by writing a query.	Use advanced EPL rules to define rule criteria in the EPL syntax.
Endpoint Rule Bundle	An Endpoint Risk Scoring Rules Bundle, which contains approximately 400 rules, comes with NetWitness 11.3 and later. The rules in this bundle only apply to NetWitness Endpoint.	If you have NetWitness Endpoint, you can configure risk scoring to identify suspicious files and hosts. To turn on risk scoring for NetWitness Endpoint, you must deploy endpoint risk scoring rules on ESA. For instructions, see "Deploy Endpoint Risk Scoring Rules on ESA" in the *ESA Configuration Guide*. For complete information on configuring NetWitness Endpoint, see the *NetWitness Endpoint Configuration Guide*.

Sample Rules

Sample Rule Builder rules come with NetWitness and appear in the Rule Library. Use sample rules to get comfortable working with rules before creating your own. You can safely edit and deploy these sample rules.

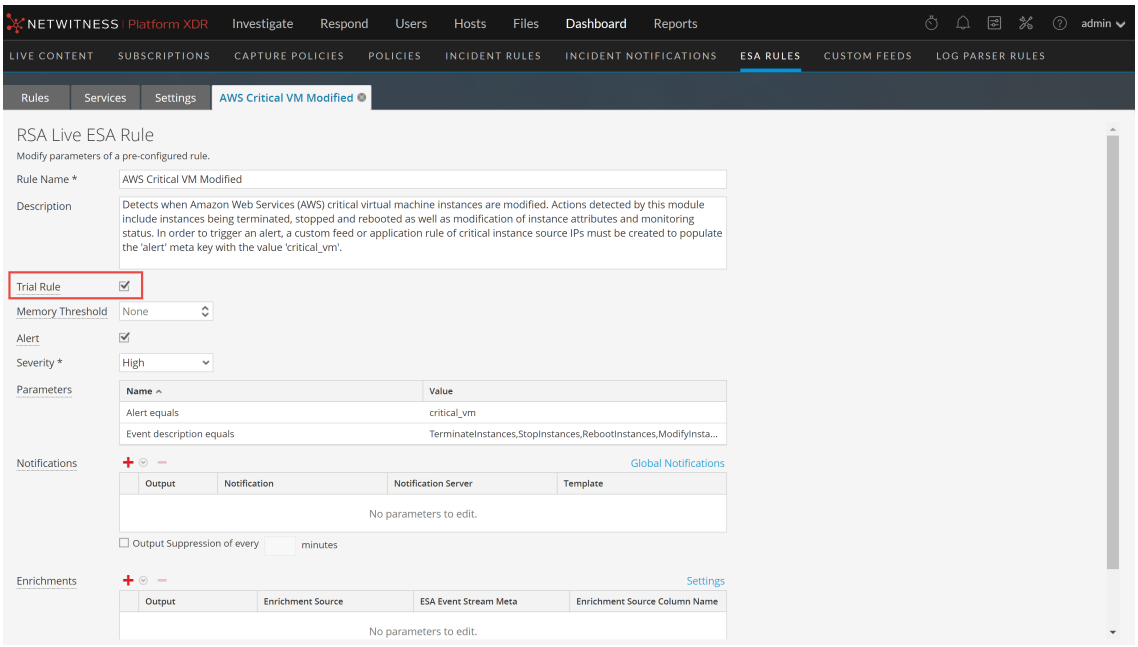
Endpoint Risk Scoring Rules Bundle

An Endpoint Risk Scoring Rules Bundle, which contains approximately 400 rules, comes with NetWitness 11.3 and later. These rules appear in the Rule Library with the sample rules. Endpoint risk scoring rules only apply to NetWitness Endpoint. You can add the Endpoint Risk Scoring Rules Bundle to an ESA rule deployment in the same way that you would add any ESA rule. However, you must specify endpoint data sources (Concentrators) in the ESA Rule Deployment.

Trial Rules Mode

For any type of rule, you can select the Trial Rule setting as an additional safeguard. Trial rules get disabled if they exceed a memory threshold set by the administrator. Run a rule in trial mode to monitor memory usage and to disable the rule automatically if it uses more memory than the threshold allows.

The following figure shows the Trial Rule setting in the Rule Builder.



ESA Permissions

This topic lists all ESA permissions and shows which permissions are assigned to each pre-configured NetWitness role. User access is restricted based on roles and permissions assigned to roles.

- Administrators
- Operators
- Analyst
- Security Operations Center (SOC) Managers
- Malware Analysts (MA)
- Data Privacy Officer

There are four permissions for ESA:

- **Access Alerting Module**: Is required for any permission
- **View Rules**: Allows view-only permission for rules in the Rule Library

- **View Alerts**: Allows view-only permission for alerts ESA generates
- **Manage Rules**:Allows you to view, create, edit, and delete rules

The following table lists permissions for ESA and the roles to which they are assigned. Use this table to see how each role can work with rules and alerts.

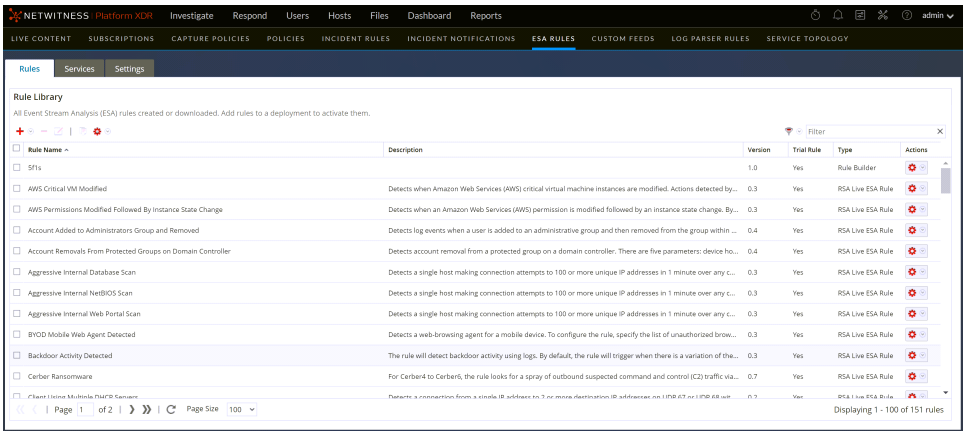
Permission	Administrators	Operators	Analysts	SOC Mgrs	MAs	DPOs
Access Alerting Module	Yes	Yes	Yes	Yes		Yes
View Rules	Yes	Yes		Yes		Yes
View Alerts	Yes		Yes	Yes		Yes
Manage Rules	Yes	Yes		Yes		Yes

For more information on roles and permissions, see the *System Security and User Management Guide*.

Practice with Sample Rules

NetWitness comes with sample rules so analysts can become familiar with how rules look before they create their own rules. Use the sample rules to become familiar with the Rule Builder and to practice editing and deploying a rule.

Sample rules are installed in the Rule Library, which contains every rule you download or create. The following figure shows sample rules in the Rule Library.



These are the available sample rules:

- SAMPLE - Blacklist - From inside countries that are not the US, Non SMTP Traffic on TCP Port 25 Containing Executable
- SAMPLE - Non SMTP Traffic on TCP Port 25 Containing Executable
- SAMPLE - P2P Software as Detected by an Intrusion Detection Device

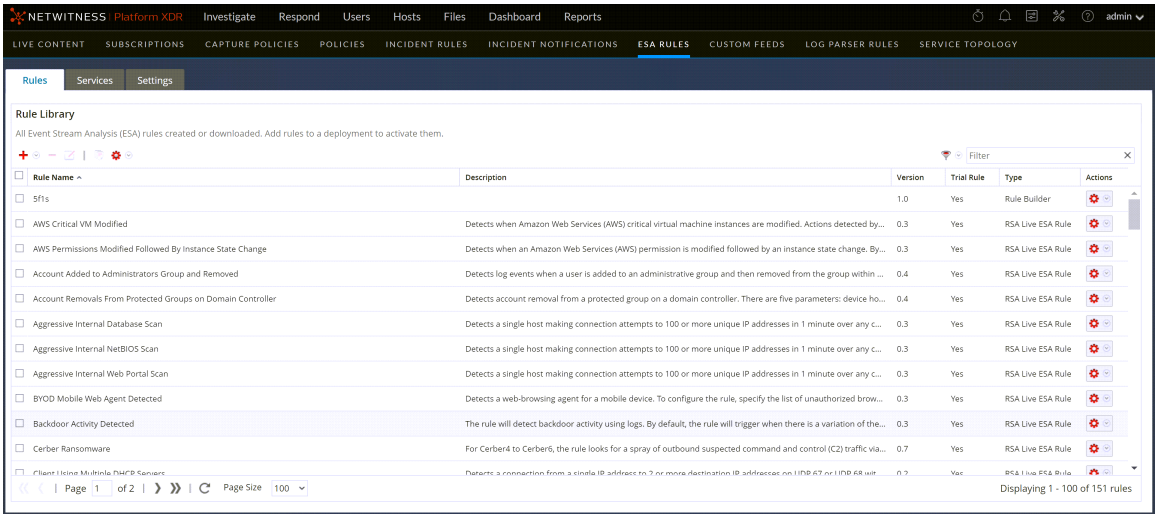
- SAMPLE - User Added to Admin Group Same User su Sudo
- SAMPLE - Whitelist - From outside of Germany, P2P Software as Detected by an Intrusion Detection Device.

Each name begins with SAMPLE to distinguish the rules that are installed with NetWitness from the rules you download and create.

Rule Library

The Rule Library shows the following information for a rule:

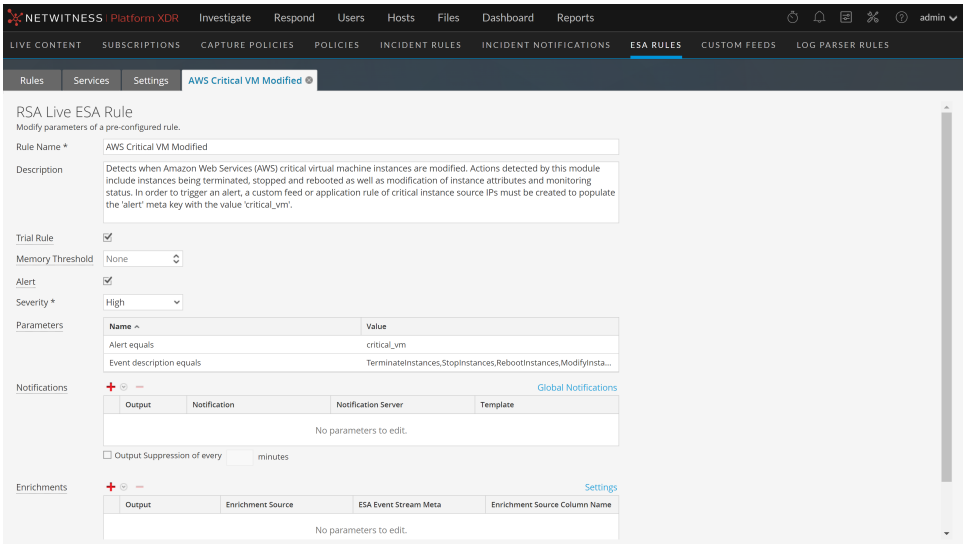
- **Name** summarizes the data or events the rule collects.
- **Description** explains the rule in more detail, although only the beginning shows in the Rule Library.
- **Trial Rule** indicates if trial mode is enabled or disabled for the rule.
- **Type** shows the origin of the rule, built in Rule Builder or Advanced EPL, downloaded from RSA Live, or Endpoint Rule Bundle.



Practice with Sample Rules

1. Go to **(Configure) > ESA Rules**.
   The ESA Rules view is displayed with the Rules tab open.

2. In the **Rule Library**, double-click a sample rule or select a sample rule and click . The rule is opened in Rule Builder.



3. To practice with a sample rule, refer to the following topics for detailed descriptions and procedures:
   - To familiarize yourself with the Rule Builder user interface, see Rule Builder Tab for a description of each field.
   - To learn how to edit a rule, see Add a Rule Builder Rule for a step-by-step procedure.
   - To deploy sample rule, see Deploy Rules to Run on ESA to learn how to associate the rule with an ESA service.

After you practice with sample rules, you will be able to download, create, and deploy your own rules.

Working with Trial Rules

The ESA Correlation service is capable of processing large volumes of disparate event data from Concentrators. However, when working with ESA Correlation rules, it is possible to create rules that use excessive memory. This can slow your ESA service or even cause it to shut down unexpectedly. To ensure that rules do not use excessive memory, you can enable them as trial rules. You should disable the trial rule setting only after testing the new rule in your environment during times of both normal and peak network traffic.

You can set a global threshold of the percentage of memory that trial rules may use. If that configured memory threshold is exceeded, all trial rules are disabled automatically. To configure the memory threshold, see "Change Memory Threshold for All Trial Rules" in the *ESA Configuration Guide*.

For suggestions on creating more efficient rules, see "Best Practices for Writing Rules" in Best Practices.

By default, new rules and RSA Live rules that you import are configured as trial rules. As a best practice, when you edit an existing rule, select the Trial Rule option, which allows you to deploy the rule with an added safeguard.

Note: Run a rule as a trial rule long enough to assess the performance during normal and peak network traffic.

Deploy Rules as Trial Rules

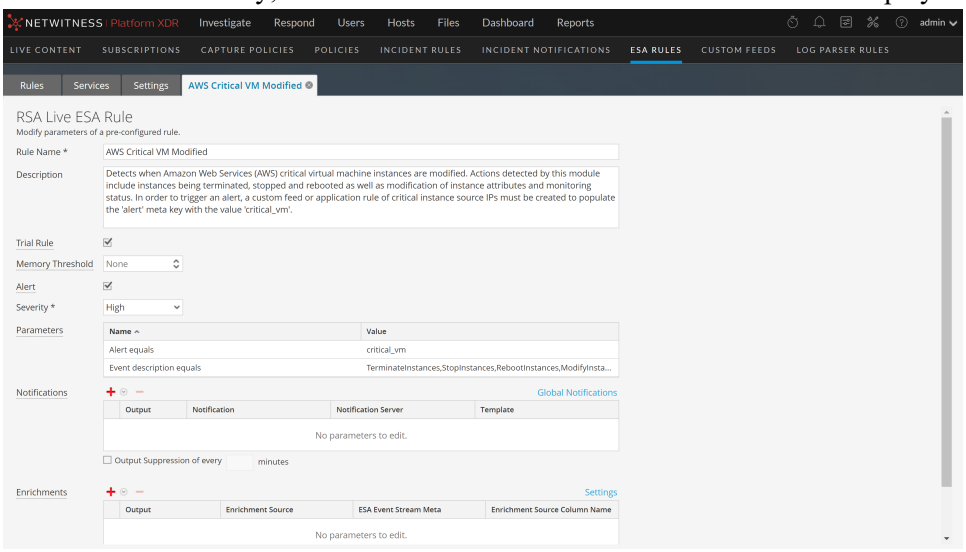
This topic explains to administrators how to enable trial rules when creating new rules or editing rules. Trial rules are automatically disabled if a specified total JVM memory utilization threshold is exceeded.

In NetWitness 11.4 and later, ESA trial rules no longer change status after an upgrade or deployment.

For example, if you change the status of a trial rule to disabled ((Configure) > ESA Rules > Services tab) and redeploy the ESA rule deployment ((Configure) > **ESA Rules** > **Rules** tab), the trial rule remains disabled.

1. Go to **(Configure) > ESA Rules**.
   The Configure ESA Rules view is displayed with the Rules tab open.

2. From the Rule Library, choose to add or edit a rule. The rule builder is displayed in a new tab.



3. To make a new or existing rule a trial rule, select the **Trial Rule** checkbox.
4. Add the rule conditions or modify the rule as needed. For instructions on editing rules, see Add Rules to the Rule Library.
5. Click **Save**.
6. Ensure that trial rules are enabled for your ESA and that you are satisfied with the thresholds configured for trial rules.
   The memory threshold is set in the configuration file. To configure it, see "Change Memory Threshold for All Trial Rules" in the *ESA Configuration Guide*.
   - The threshold is configured per ESA and is a percentage of Java Virtual Memory.
   - The configuration parameter, `fatal-percentage`, has a default value of 90.
7. Optionally, you can set up the policies in Health and Wellness to send you an email notification if the total JVM memory utilization threshold is exceeded.

The next time you deploy the rule, it runs in trial rule mode.

Note: If a trial rule is disabled, you will need to go to the **(Configure) > ESA Rules > Services** tab to re-enable the trial rules. For more instructions on re-enabling trial rules on a service, see View ESA Stats and Alerts.

Add Rules to the Rule Library

This topic explains how to add each type of rule to the rule library. You must add a rule to the Rule Library before you can deploy it. Permission to manage rules is required for all tasks in this section. To add rules, you can download them from ESA Live, create a rule via the Rule Builder, or write advanced EPL rules.

For more details on each of these procedures, see:

- Working with RSA Live ESA Rules
- Add a Rule Builder Rule
- Add an Advanced EPL Rule

In addition to deploying a rule, you can edit, duplicate, import, export, and remove a rule in the Rule Library. For details on these procedures, see Working with RulesWorking with Rules

Note: Analysts must have appropriate permissions to view the ESA rules under (**CONFIGURE**) > **ESA Rules** and (**CONFIGURE**) > **Policies** pages. For more information, see the **Source-server** section in the "Role Permissions" topic in the *System Security and User Management Guide*.

Working with RSA Live ESA Rules

This topic explains working with configurable RSA ESA rules from the NetWitness Live Content Management System so you can customize them to meet your needs.

RSA Live contains a catalog of rules. Each rule has configurable parameters so you can customize the rule for your environment. If RSA Live has a rule to detect events that you want to detect in your network, download the rule to save time. You can edit the configurable parameters and save the rule in your Rule Library. For detailed information about each rule, including whether the rule is for logs, packets, or both, see "RSA ESA Rules" at the following link:
https://community.netwitness.com/t5/netwitness-platform-threat/rules/ta-p/677884

This is an example of how each RSA Live ESA rule is described on RSA Live:

Rule Name	Description
Logins across Multiple Servers	Detects logins from the same user across 3 or more separate servers within 5 minutes. The time window and number of unique destinations are configurable.

As the name shows, the rule looks for logins across multiple servers. The description explains the rule criteria in more detail and specifies which parameters you modify.

Note: When a rule description includes a configurable parameter, the default setting for the parameter is used. In the example rule, the description states 5 minutes. However, the time window is configurable so 5 is the default number of minutes.

Prerequisites

These are the prerequisites for working with configurable Live ESA rules;

- Have permission to manage rules.
- Create a Live Account. See the *Live Services Management Guide* for details.
- Set up Live on NetWitness. See the *Live Services Management Guide* for details.
- Update your meta keys. See "Update Your ESA Rules for the Required Multi-Value and Single-Value Meta Keys" in the *ESA Configuration Guide*

Subscribe and Unsubscribe Live ESA Rules

You can subscribe and unsubscribe live ESA rules from the **Configure** > **Policies** page.

To Subscribe the Live ESA rules

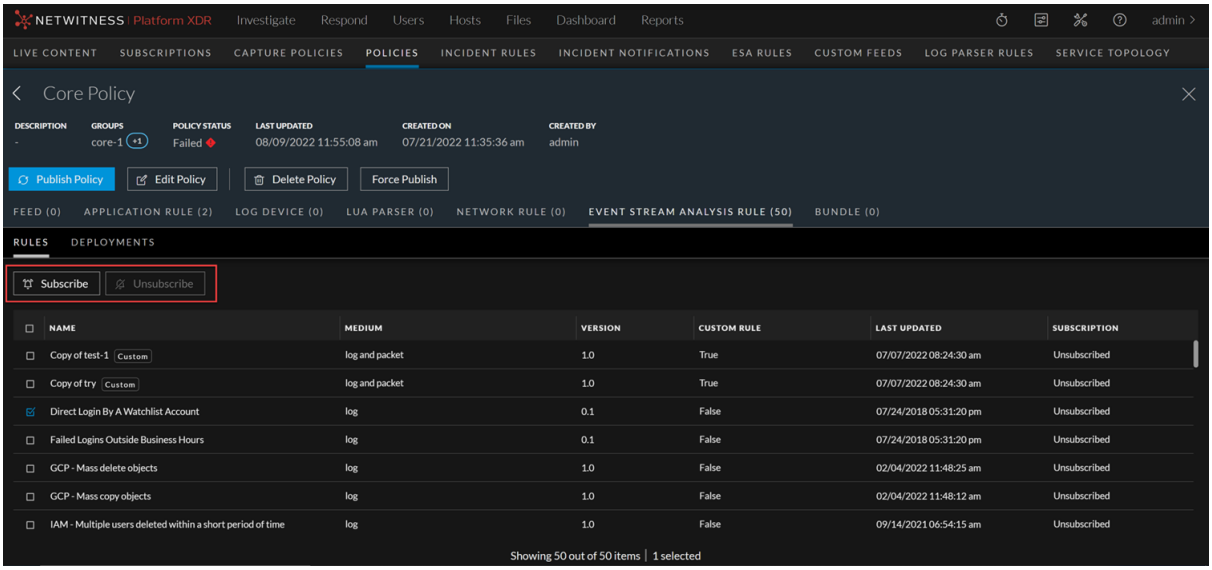
1. Go to **(CONFIGURE) > Policies**.
2. In the policies panel, click **Content**.
   The available policies are displayed.

3. Click a Policy.

   The selected policy view is displayed and by default **Application Rule** is selected.

4. Click **Event Stream Analysis Rule** > **Rules**.

5. Click one or more Live ESA rules and click **Subscribe**.

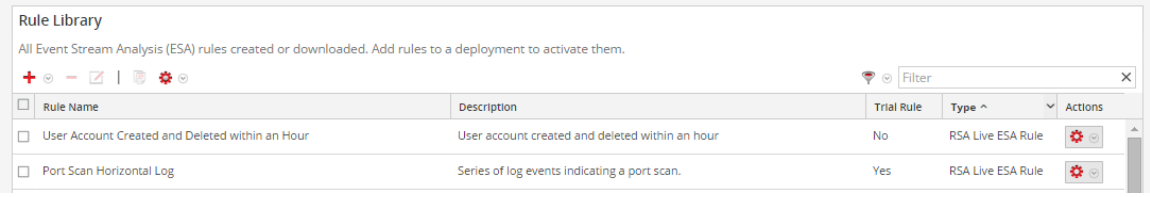


Note: ESA Rules cannot be manually deployed via Live Services. By default, all the ESA rules from Live are available in the ESA Rule library, if live is configured.

Customize an RSA Live ESA Rule

This topic explains how to configure parameters in an RSA Live ESA rule. When you download an RSA Live ESA rule, the rule appears in the Rule Library which includes the following columns:

- Rule Name
- Description
- Trial Rule
- Type
- Actions



The type is RSA Live ESA Rule.

Prerequisites

- Administrator, Operator, SOC Manager, or DPO role permissions are required.
- Rules must be downloaded to the Rule Library.

Configure Parameters for an RSA Live ESA Rule

1. Go to  **(Configure) > ESA Rules > Rules** tab.
2. In the **Rule Library**, double-click an RSA Live ESA Rule or select the rule and click  .
   The RSA Live ESA Rule tab is displayed.
3. (Optional) Change the following fields:
   - Rule Name
   - Description
   - Trial Rule (Enabled by default. NetWitness recommends you run a rule as a trial rule long enough to assess the performance during normal and peak network traffic.)
   - Alert (This option applies to 11.3 and later.) Select Alert to send an alert to Respond. Clear the checkbox if you do not want to send an alert to Respond. To turn alerts on or off for ALL rules, see the *ESA Configuration Guide*.
   - Severity
   - Notifications
   - Enrichments
4. To configure the rule for your environment, in the **Parameters** section replace the default in the **Value** Column.

5. Click **Save**.

Add a Rule Builder Rule

Each ESA rule is designed to detect something in your network and to generate an alert for it:

- User activity that is not allowed, such as attempting to download software that is not sanctioned
- Suspicious behavior, such as mass audit clearing
- Known malicious threats, such as worm propagation or a password-cracking tool

There are two methods to design a rule in ESA:

- **Rule Builder** is an easy-to-use interface. You provide a meta key and value, then select choices from lists to complete the criteria.
- **Advanced EPL** allows you to write queries in the Event Processing Language. You must know EPL syntax.

If you know EPL, you can use either method. If you do not know EPL, you should use Rule Builder. These topics explain the Rule Builder.

Step 1. Name and Describe the Rule

This topic provides instructions to identify a rule, indicate if it is a trial rule and assign a severity level. When you add a new rule, the first information to provide is a unique name and description of what the rule detects. After you save the rule, this information is displayed in the Rule Library.

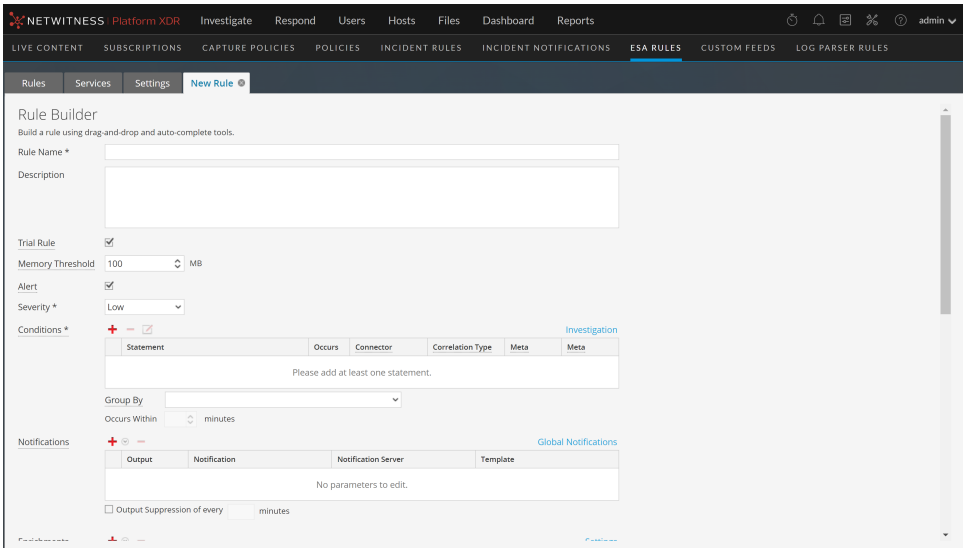
Prerequisites

You must have permission to manage rules. See ESA Permissions.

Name and Describe a Rule

1. Go to  **(Configure) > ESA Rules > Rules** tab.

2. In the **Rule Library**, select  **> Rule Builder**.
   The New Rule tab is displayed.



3. Type a unique, descriptive name in the **Rule Name** field.
   This name will appear in the Rule Library so be specific enough to distinguish the rule from others.
4. In the **Description** field, explain which events the rule detects.
   The beginning of this description will appear in the Rule Library.
5. By default, new rules are configured as a Trial Rule. A trial rule automatically disables the rule if all trial rules collectively exceed the memory threshold. If you are editing an existing rule, you can select **Trial Rule** to safely test the rule edits.
   Use trial rule mode as a safeguard to see if a rule runs efficiently and to prevent downtime caused by running out of memory. For more information, see Working with Trial Rules.
6. (This option applies to 11.5 and later.) Enter a **Memory Threshold** for a rule that uses memory, such as a rule that contains windows or pattern matching. If the configured memory threshold is exceeded, the rule gets disabled individually and an error is displayed for that rule on the  (Configure) > ESA Rules > Services tab. The Memory Threshold option works for trial rules and non-trial rules. New rules default to a 100 MB memory threshold. Rules that existed before version 11.5 do not have a default value and a memory threshold is not set.
7. (This option applies to 11.3 and later.) Select **Alert** to send an alert to Respond. Clear the checkbox if you do not want to send an alert to Respond. To turn alerts on or off for ALL rules, see the *ESA Configuration Guide*.
8. For **Severity**, classify the rule as Low, Medium, High or Critical.

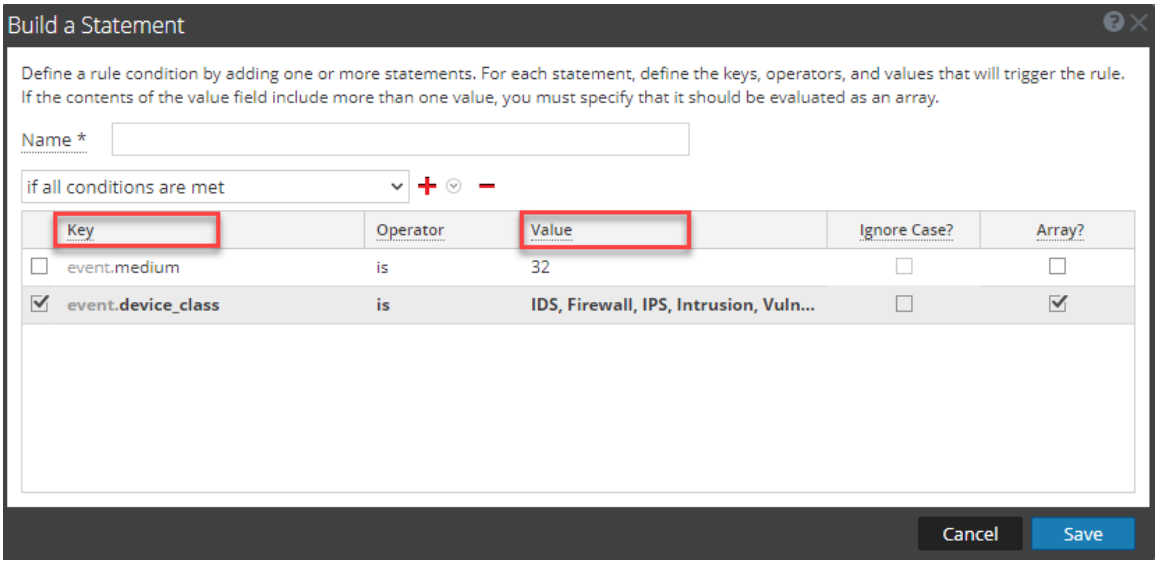
Step 2. Build a Rule Statement

This topic provides instructions to define rule criteria in Rule Builder by adding statements. A statement is a logical grouping of rule criteria in the Rule Builder. You add statements to define what a rule detects.

Example

The following graphic shows an example of a Rule Builder statement.

Every statement contains a key and value. Then, you build logic around the pair by selecting an option in each other field.



Prerequisites

To build a rule statement, you must know the meta key and the meta value.

For a complete list of meta keys, go to **(Configure) > ESA Rules > Settings > Meta Key References**.

Meta entities are not currently supported, such as:

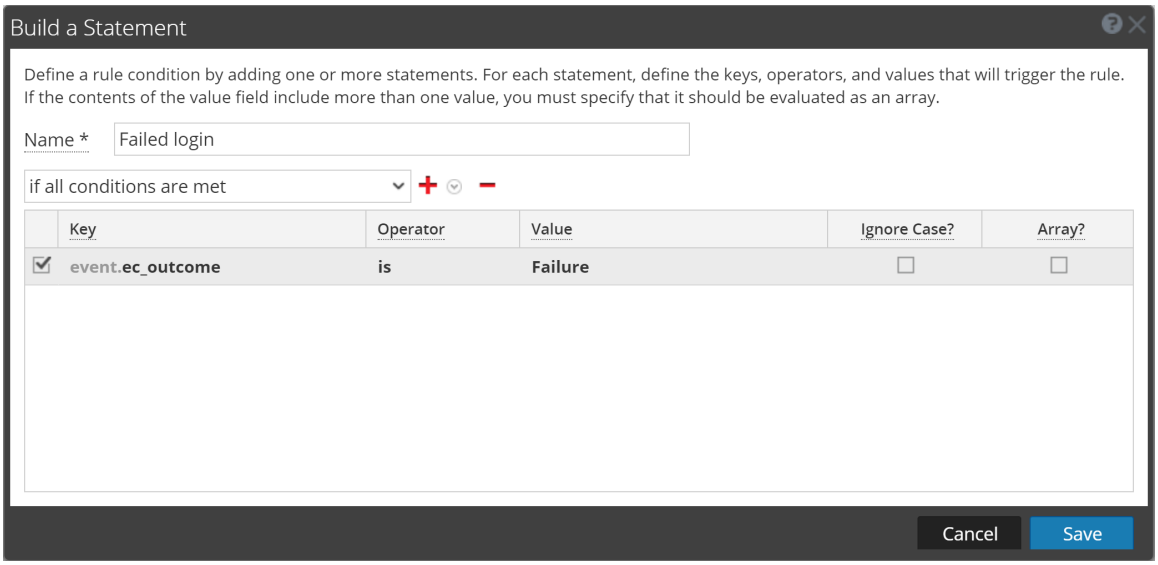
```
fullname.all
eth.all
ip.all
ipv6.all
port.src.all
port.dst.all
dir.path.all
org.all
geoip.all
port.all
```

```
domain.all
email.all
filename.all
directory.all
checksum.all
param.all
context.all
attack.all
analysis.all
compromise.all
inv.all
outcome.all
ec.all
user.all
host.all
client.all
```

Caution: If you add meta entities to your rule, they cannot get data from the data sources, so they do not trigger alerts.

Build a Rule Statement

1. Go to  **(Configure) > ESA Rules**.
   The Rules tab is displayed by default.
2. In the **Rule Library**, click  **> Rule Builder** or edit an existing Rule Builder rule.
   The Rule Builder view is displayed.
3. In the **Conditions** section, click  .
   The Build Statement dialog is displayed.



4. **Name** the statement. Be clear and specific. The statement name will appear in the Rule Builder.
5. From the drop-down list, select which circumstances the rule requires:
   - if **all conditions** are met
   - if **one of these conditions** are met
6. Specify the criteria for the statement:
   a. For **Key,** type the name of the **Meta Key**.
   b. For **Operator** specify the relationship between the meta key and the value you will provide for it. The operator that you use depends on the metadata type.
   The choices are: is, is not, is not null, is greater than (>), is greater than or equal to (>=), is less than (<), is less than or equal to (<=), is one of (For array type meta), is not one of (For array type meta), contains, not contains, begins with, ends with
   c. Type the **Value** for the meta key.
   Do not add quotes around a value. Separate multiple values with a comma.
   d. The **Ignore Case?** field is designed for use with string and string array values. By choosing the **Ignore Case** field, the query will treat all string text as a lowercase value. This ensures that a rule that searches for the user named Johnson would trigger if the event contains "johnson," "JOHNSON," or "JoHnSoN." (For best rule performance, only use the **Ignore Case** option when necessary.)
   e. The **Array?** field indicates if the contents of the Value field represent one or more than one value.
   Select the Array checkbox if you entered multiple, comma-separated values in the **Value** field. For example, "ec_activity is Logon, Logoff" requires you to select the Array checkbox.
7. To use another meta key in the statement, click  , select **Add Meta Condition** and repeat step 6.
8. To add a whitelist, click  and select **Add Whitelist Condition**.

9. To add a blacklist, click  and select **Add a Blacklist Condition**.
10. To save the statement, click **Save**.

To Add a Whitelist

You use a whitelist to ensure that specified entities are excluded from triggering the rule. Whitelists can be based on geographic location, in-memory enrichment, or Context Hub list sources. For example, if you want to create a rule that only triggers for IP addresses outside of the US, you can create a whitelist of US IP addresses.

1. After you add a meta condition, click  and select **Add Whitelist Condition**.
2. In the **EnterWhitelist Name** field, select an enrichment source. Any in-memory enrichment, Context Hub list, or a named window in Esper can be used as the source for a whitelist.
3. For the subcondition:
   a. If you used a GeoIP source for the whitelist, ipv4 is automatically entered for the subcondition. Enter the meta value for the corresponding value field. For example, enter *ipv4 is ip_src* to ensure the GeoIP records are selected based upon the ip_src being found in the GeoIP lookup database. In addition, if you used a GeoIP source for the whitelist, you might want to add a subcondition to specify the geographic region to exclude from the rule results. For example, to specify that the country code must be USA, enter "*CountryCode is US*".
   b. If you used a Context Hub list for the whitelist, select a column name from the list, then select an operator and enter the meta value for the corresponding value field.

Note: An *Entity* is a specified piece of meta, such as IP address, MAC address, user, host, domain, file name, or file hash.

To Add a Blacklist

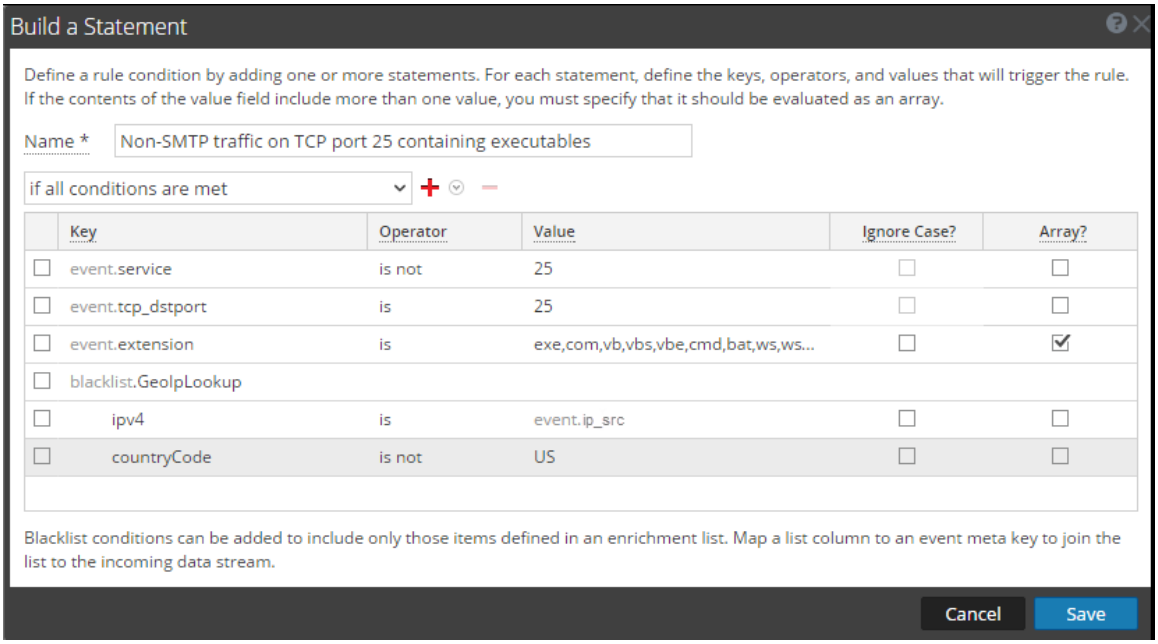
You use a blacklist to ensure that specified entities trigger the rule. Blacklists can be based on geographic location, in-memory enrichment, or Context Hub list sources. For example, you can specify that the rule only includes results from Germany.

1. After you add a meta condition, click  and select **Add Blacklist Condition**.
2. In the **Enter Blacklist Name** field, select an enrichment source. Any in-memory enrichment, Context Hub list, or a named window in Esper can be used as the source for a blacklist.
3. For the subcondition:
   a. If you used a GeoIP source for the blacklist, ipv4 is automatically entered for the subcondition. Enter the meta value for the corresponding value field. For example, enter ipv4 is ip_src to ensure the GeoIP records are selected based upon the ip_src being found in the GeoIP lookup database. In addition, if you used a GeoIP source for the blacklist, you might want to add a subcondition to specify the geographic region to include in the rule results. For example, to specify that the rule only includes results for Germany, enter "*CountryCode is DE*".

b. If you used a Context Hub list for the blacklist, select a column name from the list, then select an operator and enter the meta value for the corresponding value field.

Example: Blacklist

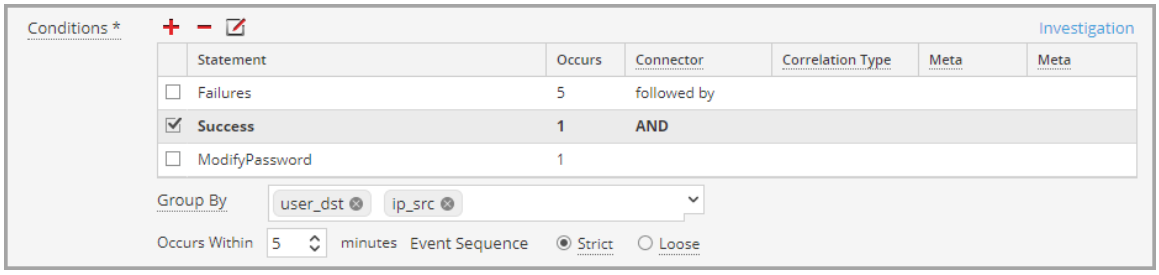
The following statement shows a blacklist statement for a rule that monitors for non-SMTP traffic on TCP destination port 25 containing an executable from countries that are outside of the United States.



Statement	Description
service is not 25	The traffic is not SMTP traffic.
tcp_dstport is 25	The traffic is running on TCP port 25.
extension is exe, com,vb,vbs,vbe,cmd,bat,ws,wsf,src,sh	The file extension is an executable.
GeoIpLookup	The blacklist is based on a GeoIPLookup source.
ipv4 is ip_src	The GeoIP records are selected based upon the ip_src being found in the GeoIP lookup database.
countryCode is not US	When looking up the IP address Event.ip_src in the GeoIP database, the record it returns does not contain "US" in the countryCode field.

Example: Strict Pattern Matching and Using the *Is Not Null* Operator

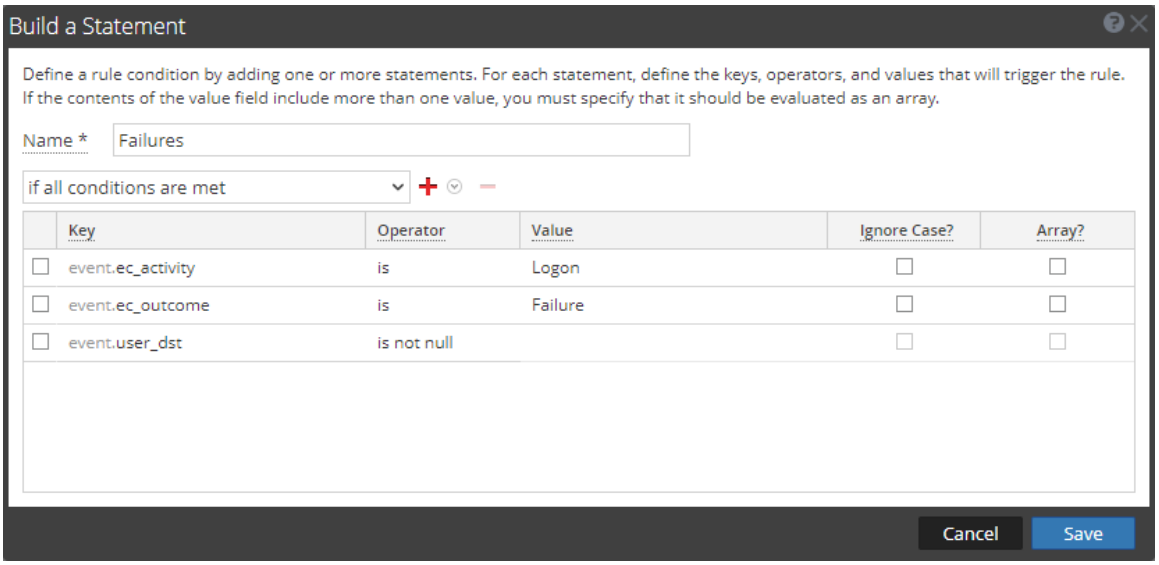
The following example uses the ability to exclude null values and create a strict pattern match to ensure that it returns the expected rule results. The following conditions make up the rule:



Rule Condition	Description
Failures	This condition searches for five failed logins with a "followed by" connector, meaning that the condition (Failures) must be followed by the next condition (Success).
Success	This condition searches for one successful login.
ModifyPassword	This condition searches for an instance where the password is modified.
GroupBy: user_dst, ip_src	The GroupBy field ensures that all the previous conditions are grouped by the user_dst meta (the user destination account) and ip_src. This is important to the construction of the rule because the rule attempts to find a case where a user has attempted to log into the same destination account multiple times, finally logged in successfully, and then changed the password. Grouping by ip_src ensures that the user logged in from the same machine attempted to log into an account multiple times. The rule may give unexpected results if you do not group the results.
Occurs within 5 minutes	The time window for the events to occur is five minutes. If the events occur outside of this time window, the rule does not trigger.
Event Sequence: Strict	The event sequence is configured for a strict pattern match. This means that the pattern must match exactly as it is specified with no intervening events. Strict pattern matching allows you to ensure that the Esper engine only generates alerts for rules that exactly match the pattern you want to find. For example, a common rule might be to search for five failed logins followed by a successful login. If you select a loose pattern match, this rule will trigger if there are any number of successful logins between the failed logins. Since the point of the rule is to find frequent *and* sequential login attempts, a strict match is required to ensure that you get the results you expect.

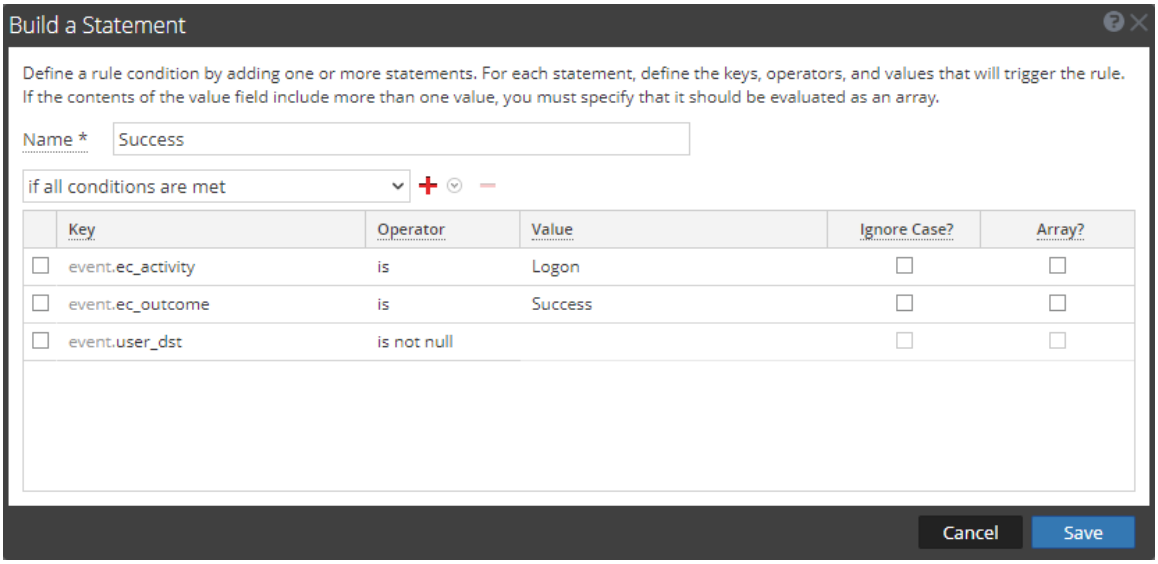
Note: Each of these conditions is explained in further detail in the sections below.

For each condition, a statement is built in the Rule Builder. The following statement makes up the Failures condition:



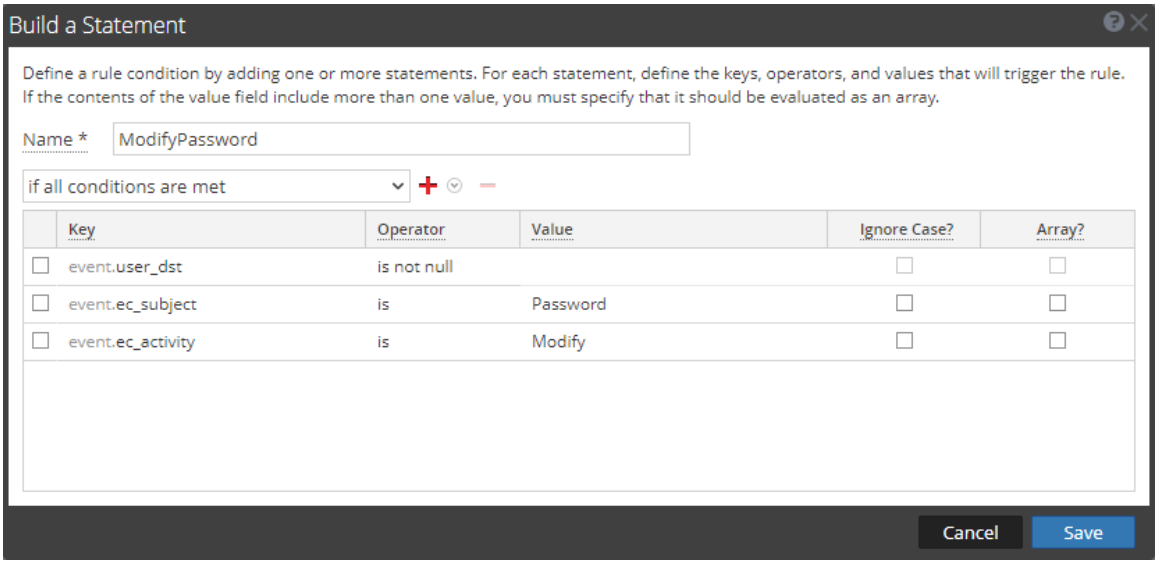
Rule Statement	Description
ec-activity is Logon	Identifies activity that attempts to log on to a system. The **Ignore Case** field is designed for use with string and string array values. By choosing the **Ignore Case** field, the query will treat all string text as a lowercase value. You may want to use this field if you are unsure what case may be used when logging a particular event. For best rule performance, only use the **Ignore Case** option when necessary.
ec_outcome is Failure	Identifies activity outcome logged as "failure."
user_dst is not null	Ensures that the condition is only true if user_dst is populated. The **is not null** operator allows you to ensure that a field returns a value. You may want to use this field when a rule depends on a particular field returning a value. For example, you want to create a rule that identifies the same user attempting to log into the same destination account multiple times (potentially a password-guessing attack). If the field that represents the user destination account is empty, you don't want the rule to trigger. To ensure the field contains a value, you use the **is not null** operator.

The following statement makes up the Success condition:



Rule Statement	Description
ec_activity is Logon	Identifies logon activity.
ec_outcome is Success	Identifies a logon that is successful.
user_dst is not null	Ensures that user destination account field must be populated for the condition to be true.

The following statement makes up the ModifyPassword condition:

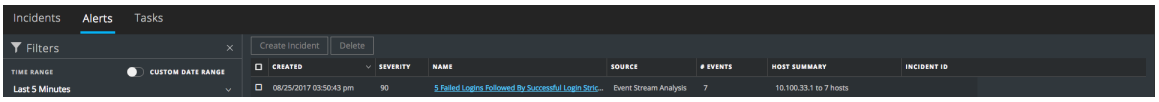


Rule Statement	Description
user_dst is not null	Ensures the user destination account field must be populated for the condition to be true.
ec_subject is Password	Identifies a subject of Password.
ec_activity is Modify	Identifies activity where the password was modified.

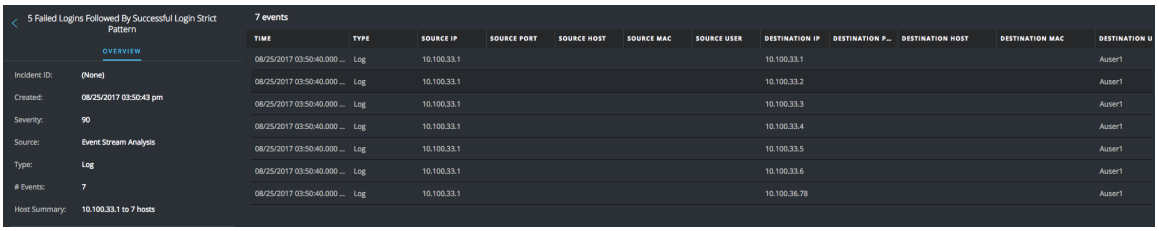
Example Results

When the alert fires for the above example rule, you can see that the rule triggered for seven events, and that each event contains a user. You can also see that the events follow a strict pattern: five failed login events, followed by a successful login event, followed by a modification to the account.

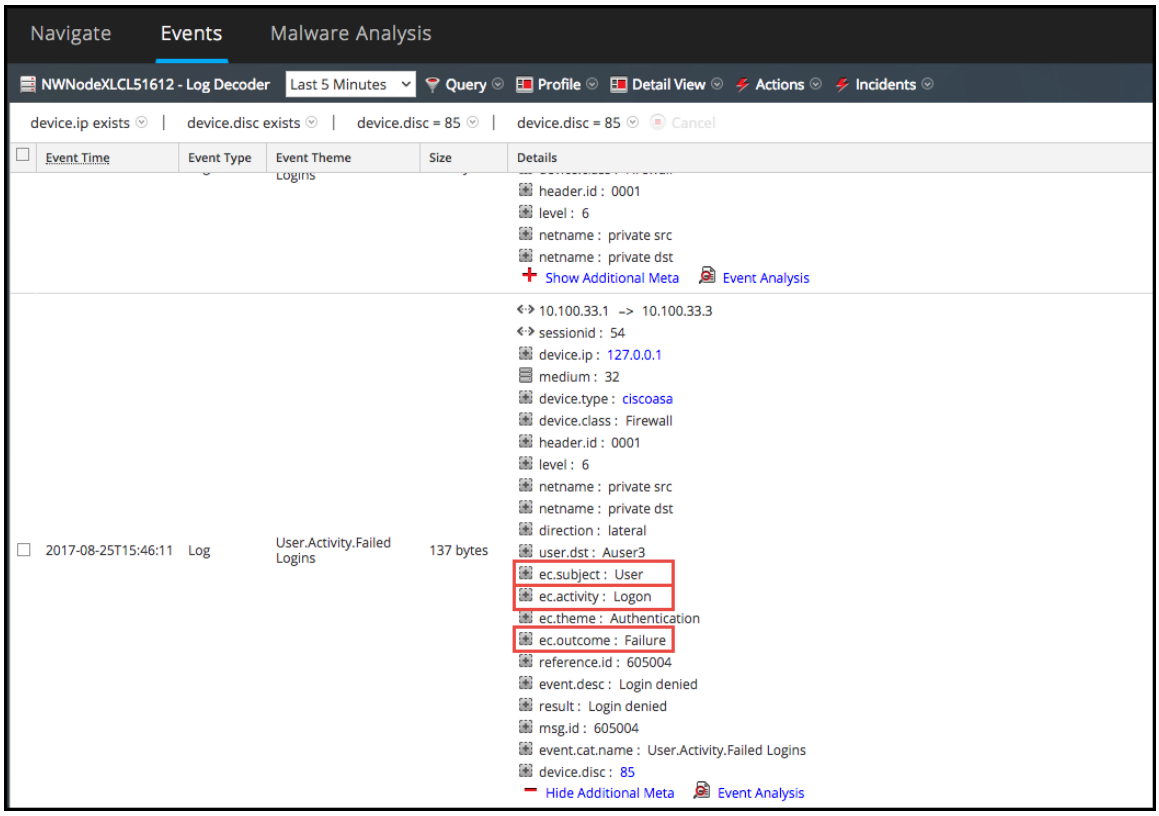
The following figure shows the alert in the Respond Alerts List view.



The next figure shows the events in the alert in the Respond Alert Details view.



Drilling down into the Investigation view by clicking on the source for one of the events, you can see the case for each of the string values.

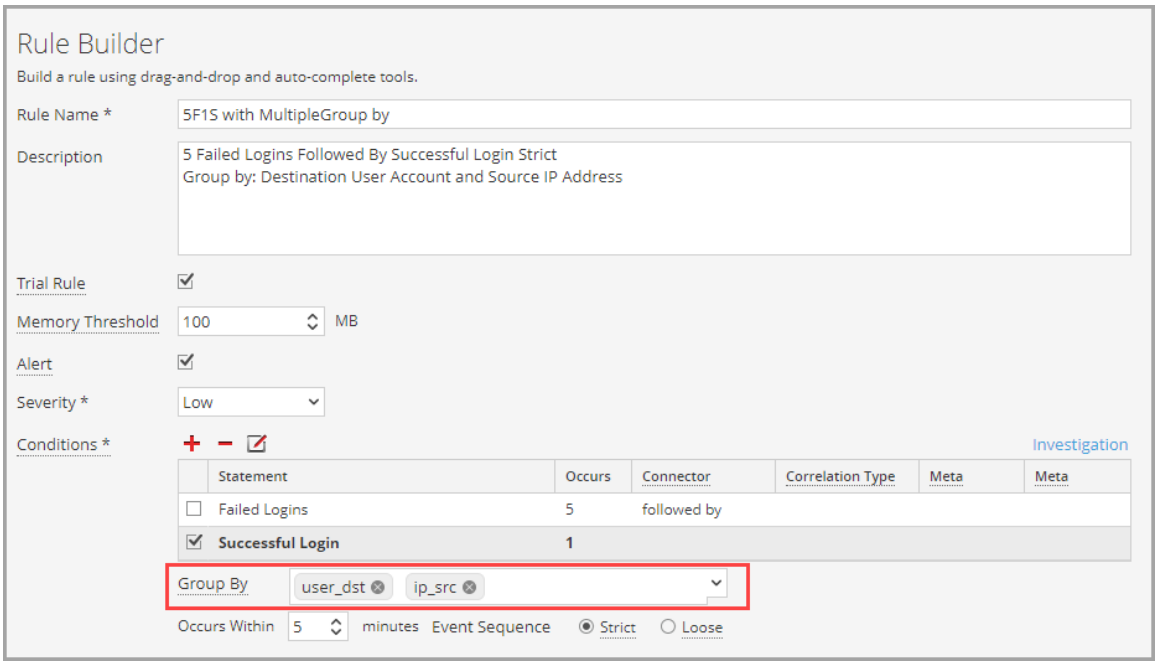


Example: Grouping the Rule Results

The **Group By** field allows you to group and filter rule results. For example, suppose that there are three user accounts; Joe, Jane, and John and you use the **Group By** meta, user_dst. The result will show events grouped under the accounts for Joe, Jane, and John.

You can also group by multiple keys, which can further filter rule results. For example, you might want to group by user destination account and machine to see if a user logged into the same destination account from the same machine attempts to log into an account multiple times. To do this, you might group by user_dst and ip_src.

The following example shows a rule grouped by user_dst and ip_src.

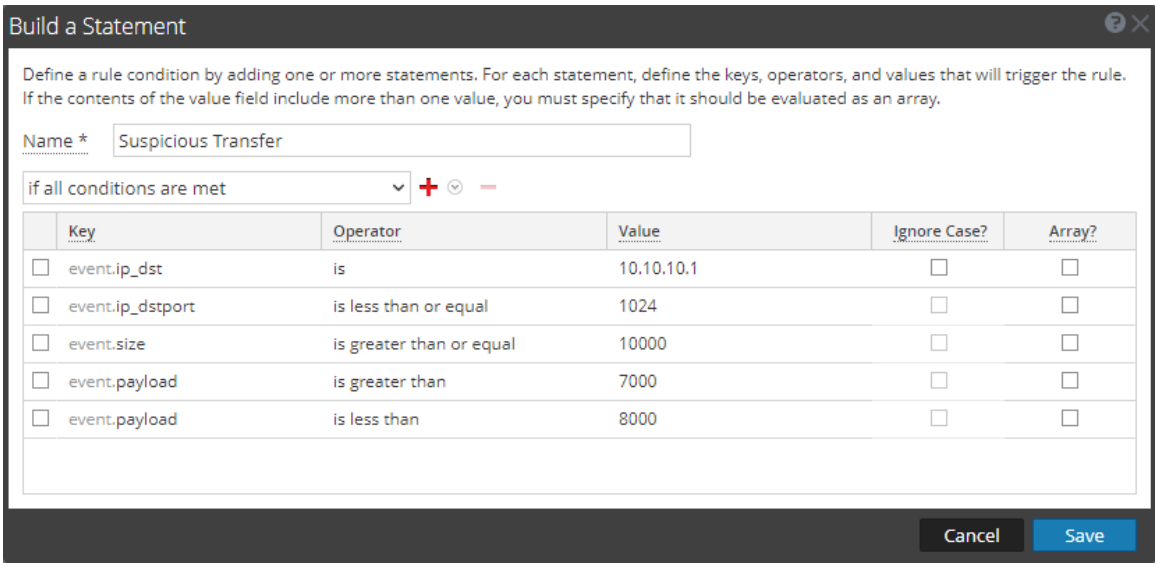


Rule Condition	Description
Failed Logins	Identifies five failed login attempts (must be followed by the next condition; that is, the five failed logins must be followed by a successful login).
Successful Login	Identifies one successful login.
Group By: user_dst and ip_src	Groups the rule results by user_dst (user destination account) and ip_src (IP address of the machine that the user is logging in from). This allows the rule to look for a user logged in from the same machine to the same destination account, resulting in a much more targeted rule result.
Occurs within 5 minutes with a strict pattern match	The events must occur within five minutes, and the pattern matching is strict, meaning it must follow the pattern exactly for the rule to trigger.

Example: Working with Numeric Operators

Numeric operators allow you to write rules against numeric values, such as specifying that a value is greater than, less than, or equal to a specific value. This is useful particularly for cases where you might want to specify a numeric threshold, that is, *payload is greater than 7000*.

The following example attempts to identify a data transfer to a particular destination through the common ports where the transfer size is high and the payload is in a suspicious range.



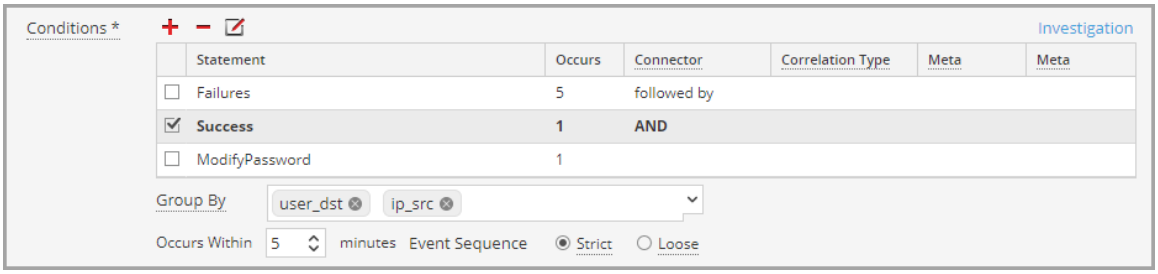
Rule Statement	Description
ip_dst is 10.10.10.1	The destination port is 10.10.10.1.
ip_dstport is greater than or equal to 1024	The destination port is in a commonly used port range, 1024 or greater.
size is greater than or equal to 10000	The size of the transfer is 10000 or greater, which is a suspiciously large transfer.
payload is greater than 7000	The payload is between 7000 and 8000, which is a suspiciously large payload.
payload is less than 8000	The payload is between 7000 and 8000, which is a suspiciously large payload.

Step 3. Add Conditions to a Rule Statement

This topic provides instructions to add conditions, such as specifying a certain time frame, to a rule statement. When you build a statement, you specify what a rule detects. You add conditions to make further stipulations, such as how many times or when the criteria must occur.

Example

The following graphic shows an example of the conditions for Rule Builder statements. Combined, the statements and conditions comprise the rule criteria.



This rule detects 5 failed logon attempts followed by one successful logon, which could be the sign that someone has hacked into user account. This is the criteria for the rule:

A. 5 failed logons are required.

B. 1 successful logon must follow the failures

C. A password was changed.

D. All events must occur within 5 minutes.

E. Group alerts by user (user_dst), because steps A and B must be performed on the same user destination account. Also, group by machine (ip_src) to ensure that the user logged in from the same machine attempts to log into an account multiple times.

F. The match is a strict pattern, meaning that the pattern must match exactly with no intervening events.

Add Conditions to a Rule Statement

1. In the **Conditions** section, select a statement and click  .
2. For **Occurs**, enter a value to specify how many occurrences are required to meet the rule criteria.
3. If you have multiple statements, in the **Connector** field select a logical operator to join one statement to another:
   - followed by
   - not followed by

- AND
- OR

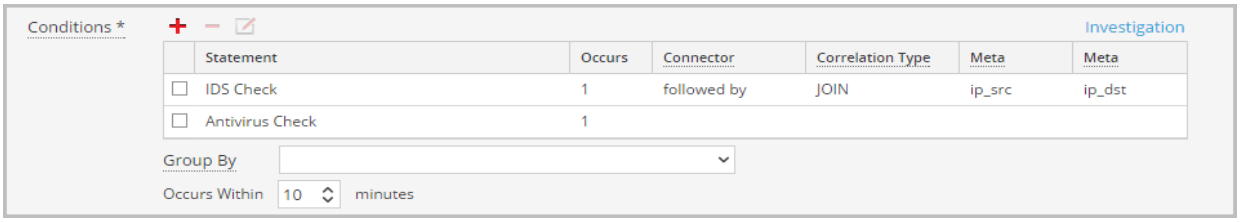
4. **Correlation Type** applies only to **followed by** and **not followed by**. If you choose a correlation type of SAME, select one meta to correlate on, and if you choose a correlation type of JOIN, select two meta to correlate on. You may want to use JOIN if you are trying to correlate on meta from two different data sources. For example, say you want to correlate an AV alert with an IDS alert. See the examples below for a use case where two meta from different sources are joined.
5. If events must happen within a specific timeframe, enter a number of minutes in the **Occurs Within** field.
6. Choose whether the pattern must follow a **Strict** match or a **Loose** match. If you specify a strict match, this means that the pattern must occur in the exact sequence you specified with no additional events occurring in between.

   For example, if the sequence specifies five failed logins (F) followed by a successful login (S), this pattern will only match if the user executes the following sequence: F,F,F,F,F,S. If you specify a loose match, this means that other events may occur within the sequence, but the rule will still trigger if all of the specified events also occur. For example, five failed login attempts (F), followed by any number of intervening successful login attempts (S), followed by a successful login attempt might create the following pattern: F,S,F,S,F,S,F,S,F,S which would trigger the rule despite the intervening successful logins.
7. Choose the fields to group by from the dropdown list. The **Group by** field allows you to group and evaluate the incoming events.
   For example, in the rule that detects 5 failed logon attempts followed by 1 successful attempt, the user must be the same, so user_dst is the **Group By** meta key. You can also group by multiple keys. Using the previous example, you might want to group by user and machine to ensure that the same user logged in from the same machine attempts to log in to an account multiple times. To do this, you might group by user_dst and ip_src.

Example

The following graphic shows an example of the conditions for a rule that allow you to evaluate the same entities across multiple devices so you can accomplish complex use cases. For example, you can create a rule that triggers if an IDS (Intrusion Detection System) alert is followed by an AV(Anti-virus) alert for the same workstation. The work station key is not the same between the two (IDS & AV) sources, so you can perform a JOIN in order to evaluate the different entities.

In the IDS alert, the workstation is identified by the source IP address from the IDS alert, and would be compared to the destination IP address from the AV alert.



This is the criteria for the rule:

A. An IDS alert occurs.

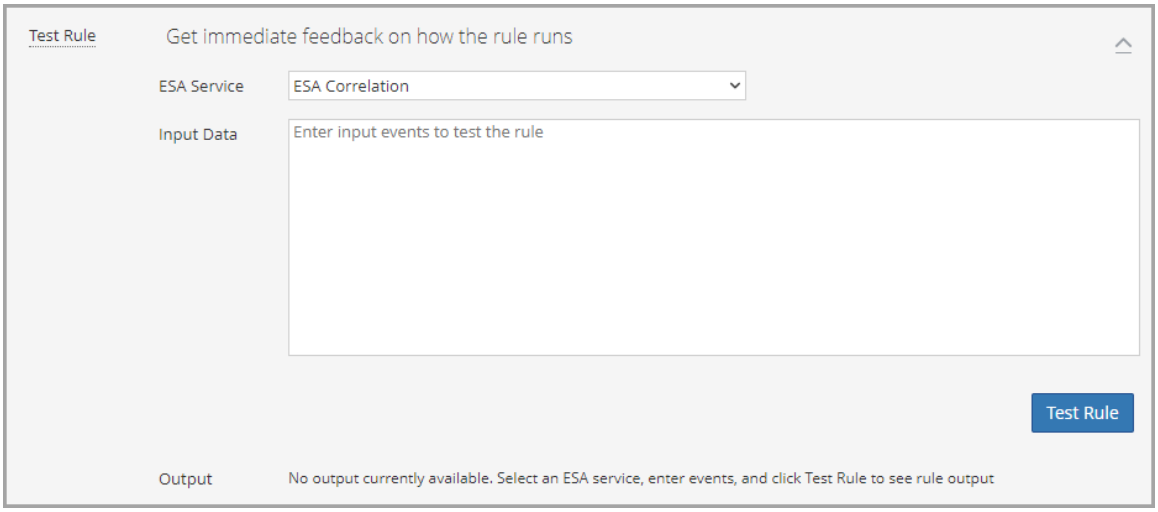
B. The destination IP address from the AV alert and source IP address for the workstation from the IDS alert are joined to allow you to view the same entities across different sources.

C. An Antivirus alert follows the IDS alert.

Validate an ESA Rule

You can confirm that an ESA rule generates the expected alerts by testing the rule logic using JSON input data. You can view the alerts in the output, but this test does not send any alert notifications.

1. If you are not already in the rule, go to **(Configure) > ESA Rules > Rules** tab and in the **Rule Library**, open the ESA rule that you want to test.
2. Scroll down to the **Test Rule** section.



3. In the **ESA Service** field, select the ESA Correlation service to process the rule. Use the same ESA Correlation service that you plan to use in the ESA rule deployment that contains the rule.
4. In the **Input Data** field, enter the input events to test the rule. Download the events from the Investigate view in JSON format, copy the events, and paste them in this field. You can do this from the Investigate > Navigate view or the Investigate > Events view.

   To download the events from the Investigate > Navigate view:

   a. In the main menu, go to **Investigate > Navigate** in a new tab, select a data source, and click **Navigate**.

   b. In the Navigate view, click **Load Values** and click a meta value to filter the events.

c. Save the events as meta in the JSON file format [Save Events > Meta > (name the file) > Export Meta Format: choose JSON].

d. In the toolbar click the **(Jobs)** icon and then click **View Your Jobs**.

e. In the Jobs panel, download your extracted meta, for example: **`investigation-2020-May-19-08-30-20.json`**.

f. Go to back to the **(Configure) > ESA Rules** tab opened previously and copy the contents of the JSON file into the **Input Data** field in your ESA rule.

To download the events from the Investigate > Events view:

a. In the main menu, go to **Investigate > Events** in a new tab.

b. In the Events view, enter a query for the ESA rule test.

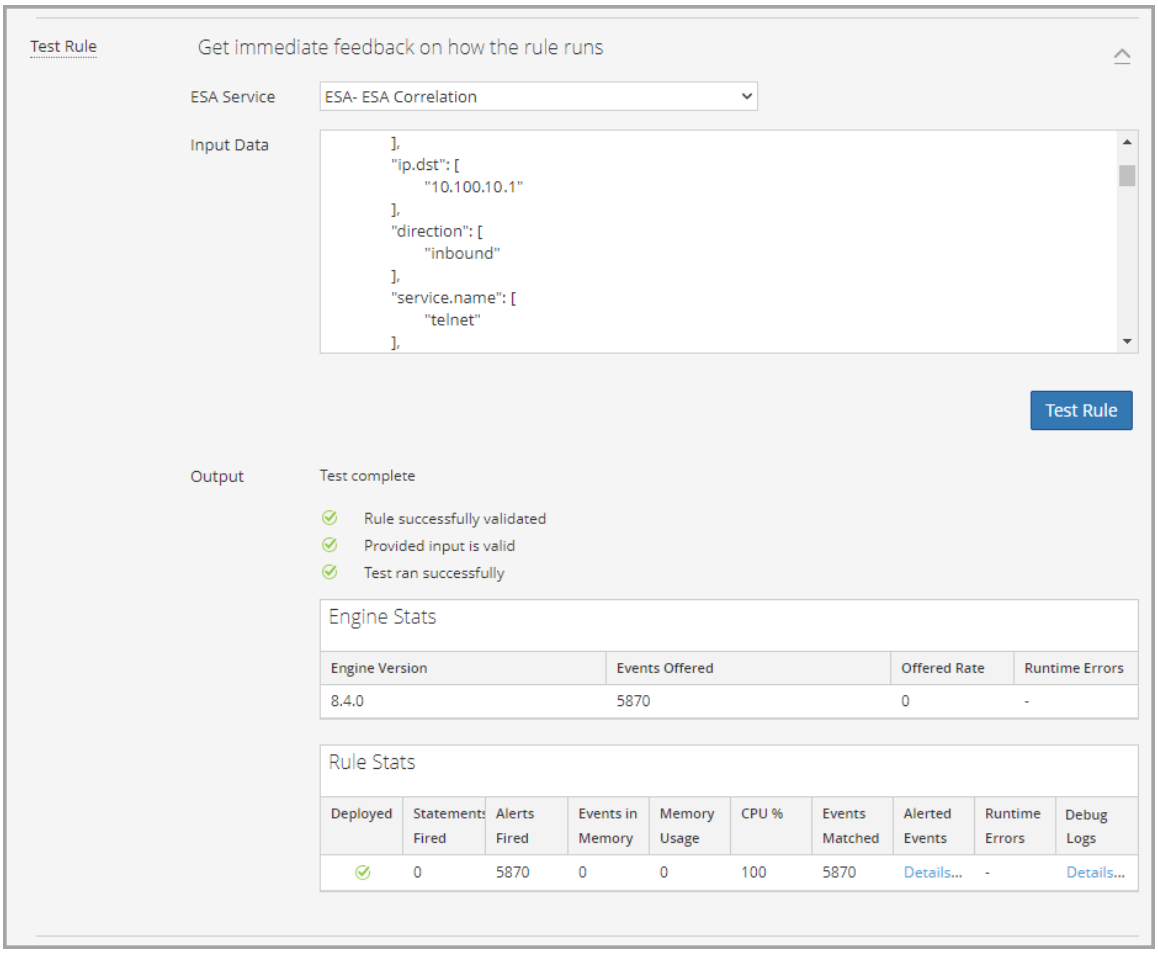
c. Select the events to use and in the **Download** or **Download All** menu, select **Visible Meta as JSON** or **All Meta as JSON**, depending on the size of your selection.

d. In the main menu, go to **Dashboards** and in the toolbar click the **(Jobs)** icon and then click **View Your Jobs**.

e. In the Jobs panel, download your extracted meta, for example: **Concentrator_ALL_EVENTS_ALL_META.json**.

f. Go to back to the **(Configure) > ESA Rules** tab opened previously and copy the contents of the JSON file into the **Input Data** field in your ESA rule.

5. Click **Test Rule**. The **Output** field shows the output of your rule and you can determine if the results meet your requirements.



The following table describes the test rule output **Engine Stats**.

Field	Description
Engine Version	Esper version running on the ESA service
Events Offered	Number of events processed by the ESA service since the last service start
Offered Rate	The rate that the ESA service processes current events / The maximum rate that the ESA service processed events
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the ESA rule deployment.

The following table describes the test rule output **Rule Stats**.

Field	Description
Deployed	A green checkmark indicates that the rule is deployed on the selected ESA service.

Field	Description
Statements Fired	The number of statements that fired the alerts
Alerts Fired	The number of alerts generated from the test data
Events in Memory	The number of events placed in memory by the rule
Memory Usage	The total amount of memory used by the rule
CPU %	The percentage of the deployment CPU used by the rule. For example, a deployment with 1 rule shows 100% CPU usage for that rule and a deployment with two equally CPU heavy rules show 50% each.
Events Matched	The number of events that matched the rule
Alerted Events	If applicable, this field can contain a link to events that caused an alert.
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the rule.
Debug Logs	This field contains a link to Esper debug (audit) logs.

Working with Rules

This topic discusses additional procedures you can perform on rules. You may want to perform any of the following procedures:

- Edit, Duplicate or Delete a Rule
- Filter or Search for Rules
- Import or Export Rules

Edit, Duplicate or Delete a Rule

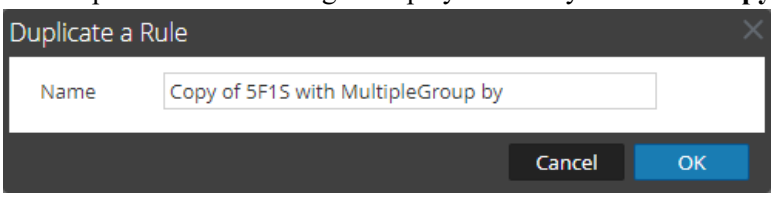
This topic provides instructions to edit, duplicate, or delete an Event Stream Analysis (ESA) rule. When you edit a rule, ESA applies the updated criteria going forward. No changes are made to previously generated alerts.

Edit a Rule

1. Go to **(Configure) > ESA Rules > Rules** tab.
   The Rules tab is displayed.
2. In the **Rule Library**, select the rule you want to edit and click .
   Depending on the rule type, the respective rule tab is displayed.
3. Modify the required parameters.
4. Click **Save**.

Duplicate a Rule

1. In the **Rule Library**, select the rule you want to duplicate and click .
2. The Duplicate a Rule dialog is displayed. The system adds **Copy of** in front of the rule name.

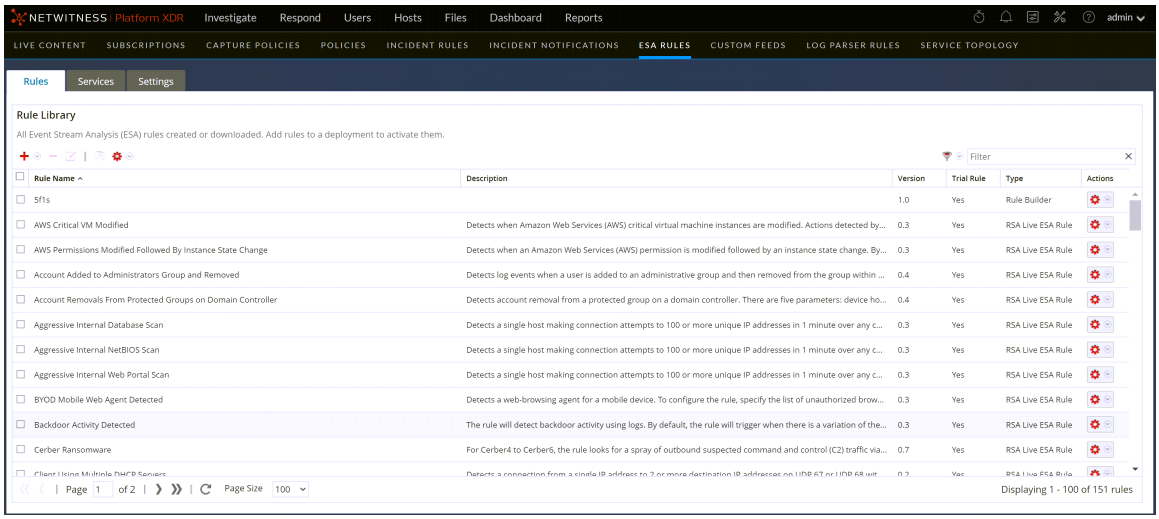


3. In the **Name** field, type a unique name for the duplicate rule and click **OK**.

A duplicate rule with the new name is added to the Rule Library.

Delete a Rule

1. Go to **(Configure) > ESA Rules > Rules**.
   The Rules tab is displayed.



2. In the Rule Library, select one or more rules and click .
   A warning dialog is displayed.
3. Click **Yes**.
   A confirmation message that the rule is deleted successfully is displayed and the selected rule is deleted from the Rule Library.

Filter or Search for Rules

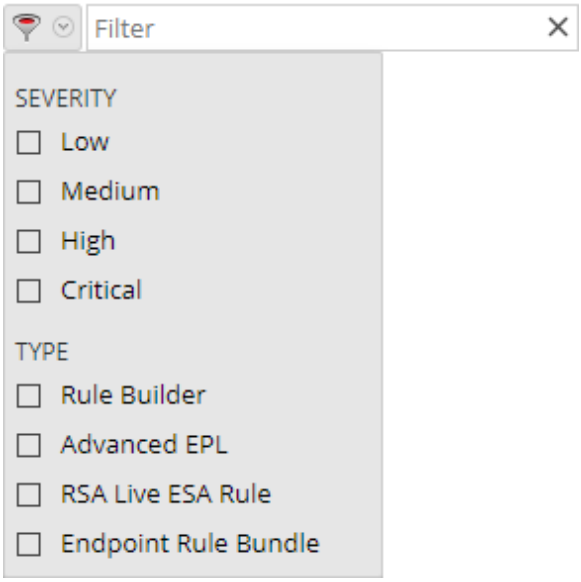
This topic shows analysts how to specify the type of rules that display in the Rule Library.

Prerequisites

Make sure that you understand the Rule Library view components. For more information, see Rule Library Panel.

Filter Rules

1. Go to **(Configure) > ESA Rules**.
   The Rules tab is displayed by default.
2. In the **Rule Library** panel toolbar, click and select the severity and type of rules that you would like to appear in the Rule Library list. The following figure shows the Filter drop-down list.



The selected rule types appear in the list.

Search for Rules

1. Go to **(Configure) > ESA Rules**.
   The Rules tab is displayed by default.
2. In the **Rule Library** panel toolbar, type a rule name in the Filter field.
   The Rule Library panel lists the rules that match the names entered in the Filter field.

Import or Export Rules

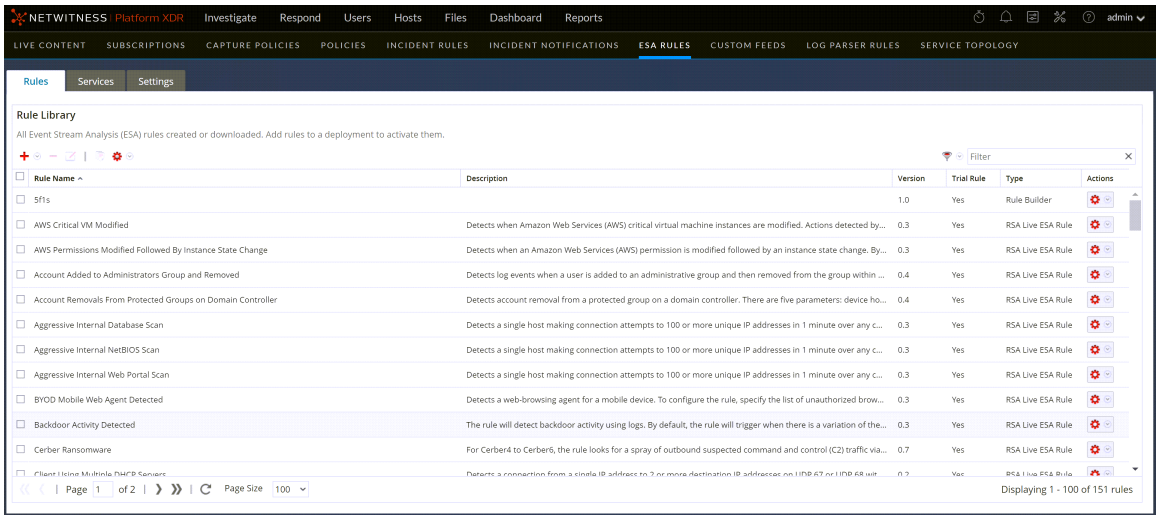
The topic provides instructions to import ESA rules from a NetWitness instance and to export ESA rules to your hard drive so you can keep a local copy.

If you exported a rule in an earlier version of NetWitness, the following conditions apply when you import the rule in version 10.5 or later:

- Exported in version 10.3 – You cannot import rules to version 10.5 or later.
- Exported in version 10.4 – You can import rules to version 10.5 or later.

Import ESA Rules

1. Go to **(Configure) > ESA Rules > Rules** tab.
   The Rules tab is displayed.



2. In the **Rules Library** toolbar, select > **Import**.
   The Import ESA Rules dialog is displayed.

3. Click **Browse** to browse and select the file containing the ESA rules.
4. Click **Import**.

Export ESA Rules

1. Select an ESA rule or multiple rules and select > **Export** in the Rule Library toolbar.
   A warning dialog is displayed.
2. Click **Yes**.
   The Export Rules dialog is displayed.
3. In the **Enter File Name** field, type a filename for the file with the ESA rules and click **Export**.
   The file is exported as a binary file to your machine.

Note: The binary file cannot be edited.

Choose How to be Notified of Alerts

This topic explains the different notification methods and how to add a notification method to a rule. Administrator, SOC Manager or DPO role permissions are required for all tasks in this section.

When a rule triggers an alert, ESA can send a notification in the following ways:

- Email
- Syslog
- Script

To configure a notification, you configure these components:

- **Notification Server**: The notification server is the source of the notifications. After you configure a notification server, you can add it to a rule. When the rule triggers an alert, the rule will use that server to send alert notifications.
- **Notifications**: These are the outputs (destinations) of the notifications, which can be email, script, and Syslog. When you design a rule, you can specify the notification for an alert.
- **Templates**: The message format of an alert notification is defined in a template.

If you use an ESA rule that has an enrichment, such as a Context Hub list, you must create a custom template. You can duplicate a default template and adjust it for your enrichment. For more information, see Troubleshoot ESA Rules. For information on creating a custom template, see see "Configure Meta Keys as Arrays in ESA Correlation Rules" in the *System Configuration Guide*.

Go to the NetWitness All Versions Documents page and find NetWitness Platform guides to troubleshoot issues.

Note: ESA SNMP notifications are not supported for NetWitness 11.3 and later.

Alert suppression and alert rate regulation are two features that Event Stream Analysis provides. Alert suppression ensures that multiple emails are not sent out for the same alert. For example, consider a rule to detect failed user logins. If you set the alert suppression to three minutes, you will see only the alerts generated in that time frame. This is fewer than the number of alerts you would see without alert suppression. Some alerts can be duplicates. With alert suppression, emails are not sent for duplicate alerts. This ensures the inbox is not flooded with redundant alert notifications.

Alert rate regulation is a preventive measure to ensure that alerts from misconstrued rules do not flood the system. This ensures that ESA does not send more than the configured limit of emails within one minute.

Notification servers, notifications, and templates are configured in the Administration System view. For more information, see "Configure Notification Servers", "Configure Notification Outputs", and "Configure Templates for Notifications" in the *System Configuration Guide.*

Notification Methods

When a rule triggers an alert, ESA can send a notification in the following ways:

- Email
- Syslog
- Script

Note: ESA SNMP notifications are not supported for NetWitness 11.3 and later.

Email Notifications

ESA Correlation can send notifications to users through email about various system events.

To configure these email notifications, you need to:

- Configure the SMTP email server as an output provider. For instructions, see "Configure the Email Settings as Notification Server" in the *System Configuration Guide*.
- Set up an email account to receive notifications. For instructions, see "Configure Email as a Notification" in the *System Configuration Guide*.
- Configure a template for email notification. For instructions, see "Configure Global Notifications Templates" in the *System Configuration Guide*.

Syslog

Event Stream Analysis can send events and consolidate logs in Syslog format to a Syslog server.

To configure these Syslog notifications, you need to:

- Configure Syslog server settings as an output provider. For instructions, see "Configure a Syslog Notification Server" in the *System Configuration Guide*.
- Configure Syslog message format as an output action. For instructions, see "Configure Syslog as a Notification" in the *System Configuration Guide.*
- Configure a template for Syslog. For instructions, see "Configure Global Notifications Templates" in the *System Configuration Guide*.

Script Alerter

Apart from the alert notifications ESA allows users to run scripts in response to ESA alerts.

Scripts enable you to do custom integration with applications that exist in your environment. For example, if you want to open an incident ticket from an application when a specific alert is triggered, Script Alerter lets you write a script that calls the application API and has ESA invoke it when the specific ESA rule is triggered. You can configure a FreeMarker template to define what details you want to extract from the output of the ESA rule and pass it as command line arguments to the script.

To use the Script Alert, you need to:

- Configure the user identity and other details that are required to execute the script. For instructions, see "Configure Script as a Notification Server" in the *System Configuration Guide*.

- Define the Script. For instructions, see "Configure Script as a Notification" in the *System Configuration Guide*.
- Configure a template for the script. For instructions, see "Configure Global Notifications Templates" in the *System Configuration Guide*.

Add Notification Method to a Rule

This topic tells administrators how to add a notification, such as email, to a rule. ESA uses the notification method when it generates an alert for an event that meets rule criteria.

You add a notification to a rule so ESA can let you know when a rule triggers an alert. Although the notification fields are not required, it is a best practice to add a notification to a rule.

When you add a notification method to a rule, you select the following information:

- Output
- Notification
- Notification Server
- Template

Prerequisites

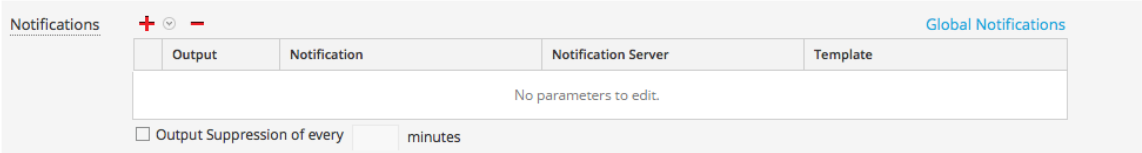
- Your role must have permission to manage rules.
- The rule must exist.
- The notification method must be configured with a supported server and template:

  Go to **(Admin) > System > Global Notifications**.

  For detailed procedures, see the *System Configuration Guide.*

Add a Notification Method to a Rule

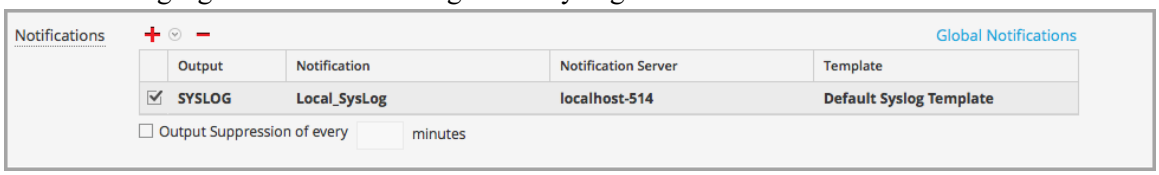
1. Go to **(Configure) > ESA Rules > Rules** tab.
2. In the **Rule Library**, click to add a new rule or select an existing rule and click .
   Depending on the rule type, the Rule Builder or Advanced EPL tab is displayed.
   The Notifications section is the same for both tabs.



3. Click and select the **Output** for the alert:

- Email
- SNMP (This option is not supported in NetWitness 11.3 and later.)
- Syslog
- Script

4. Double-click the **Notification** field and select the name of a previously configured output.
   For example, Level 1 Analyst could be the name of an email notification that goes to the L1-Analysts email distribution group.
5. Double-click the **Notification Server** field and select the server that sends the notification.
6. Double-click the **Template** field and select a format for the alert.
   The following figure shows the settings for a Syslog notification.

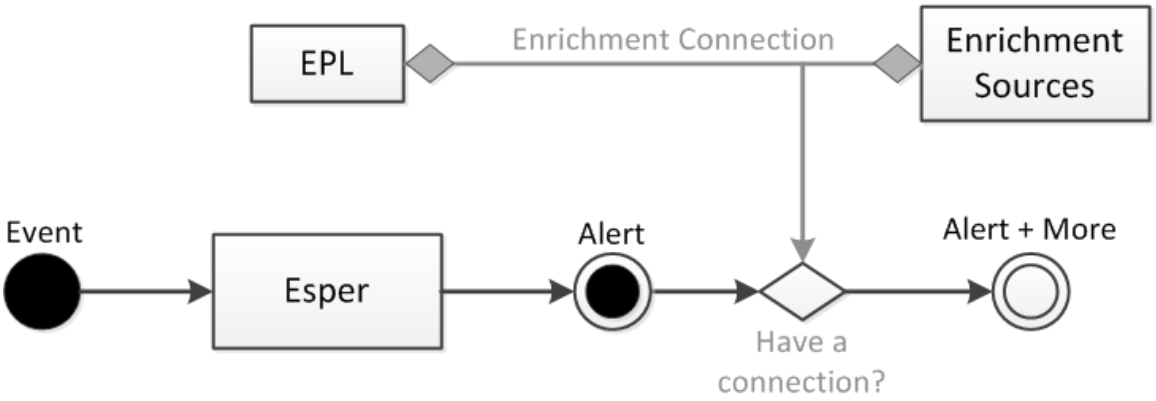


7. If you want to specify frequency, select **Output Suppression**, then enter the number of **minutes**.
8. If you want to add another notification, repeat steps 3-7.
9. Click **Save**.
   When ESA generates an alert for an event that matches the rule criteria, you will be notified of the alert via each notification method added to the rule.

Add a Data Enrichment Source

This topic tells how to add a previously configured enrichment source to a rule. When ESA creates an alert, information from the source gets included in it.

Enrichments provide the ability to include contextual information into correlation logic and alert output. Without enrichments, all information included in an ESA alert is from a Core service. With enrichments, you can request for look ups into a variety of sources and include the results into the outgoing alerts. The following figure illustrates the enrichment feature.



Enrichment configuration is made up of two logical units:

- Enrichment Sources – These are data stores of contextual information.
- Enrichment Connections – These act as connectors between alert meta and source columns.

ESA allows you to make connections between Event Processing Language (EPL) statements and enrichment sources. Once the connections are established, the system joins the selected fields from the alert output with the information in the sources and uses the matching data to enrich the alert that is sent out. ESA can connect with the following sources:

- Esper Named Windows
- MaxMindGeoIP Database

Note: The geoIP enrichment source can neither be created nor deleted. It is provided out of the box to the user.

Example Rule with Enrichments

The following example rule illustrates how ESA enrichments can enhance alerts.

```
@RSAAlert @Name("simple") SELECT * FROM Event(ec_theme='Login Failure')
```

This rule generates an alert for every logon failure and thus if the following (simplified) event stream is received at ESA:

sessionid	ec_ theme	username	ip_src	ip_dst	host_dst
1	Login Success	dshrute	23.xx.23x.16		
2	Login Failure	jhalpert	23.xx.23x.16	31.1x.x9.1x8	www.facebook.com

An alert without an enrichment with the following constituent `events` might be generated in response to the second session:

```
{
     "events": [
          {
               "username": "jhalpert",
               "host_dst": "www.facebook.com",
               "ip_dst": "31.1x.x9.1x8",
               "sessionid": 2,
               "ec_theme": "Login Failure",
               "esa_time": 1406148964130,
               "ip_src": "23.xx.23x.16"
          }
     ]
 }
```

The JSON output shows all the information available for inclusion into an ESA notification using an appropriate FreeMarker template. For instance, the template expression `${events[0].username}` would evalute to `jhalpert`.

With enrichments, the same deployment, with the same event stream, can generate the alert shown below.

```
{"events": [
     {
          "username": "jhalpert",
          "host_dst": "www.facebook.com",
          "GeoIpLookup": [
          {
               "city": "Cambridge",
               "longitude": -71,
               "countryCode": "US",
               "areaCode": 617,
               "metroCode": 506,
               "region": "MA",
               "dmaCode": 506,
               "ipv4Obj": "/23.xx.23x.16",
               "countryName": "United States",
               "postalCode": "02142",
               "ipv4": "23.xx.23x.16",
               "latitude": 42,
               "organization": "Verizon Business"
          }
     ],
     "orgchart": [
          {
               "supervisor": "mscott",
               "name": "James Halpert",
               "extension": 3692,
               "location": "Scranton",
```

```
            "department": "Sales",
            "id": "jhalpert"
        }
    ],
    "ip_dst": "31.1x.x9.1x8",
    "sessionid": 2,
    "LoginRegister": [
        {
            "username": "dshrute",
            "ip_src": "23.xx.23x.16"
        }
    ],
    "ec_theme": "Login Failure",
    "esa_time": 1406155218912,
    "ip_src": "23.xx.23x.16"
    }
]}
```


The system pulls contextual data to make the alert more meaningful.

To include the name of the supervisor and the name of the user with the last successful login in the ESA notification, this example includes the following template expressions: `${events[0]["orgchart"][0].supervisor}` gives the name of the supervisor of the employee in the alert and `${events[0]["LoginRegister"][0].username}` gives the name of the user with the last successful logon from the same `ip_src` (using a stream based Named Window).

Enrichment Sources

This topic explains options for adding an external data source to provide additional information in alerts. Enrichment sources provide additional information in alerts. For example, an in-memory table can provide a full name, title, office location, and employee number if a user matches rule criteria. The following types of enrichment sources are available:

- Context Hub List (Preferred)
- In-Memory Table (Ad hoc only)
- GeoIP

Note: Database, Database Connection, Warehouse Analytics, and Recurring In-Memory Tables as enrichment sources are not supported for the ESA Correlation service in NetWitness 11.3 and later.

It is preferable to use Context Hub List enrichment sources instead of In-Memory Table enrichment sources. You can share Context Hub List enrichment sources across the NetWitness. You can only use the In-Memory Table with ESA. Recurring In-Memory Tables are no longer supported; use Content Hub Lists as enrichment sources.

Note: The geoIP enrichment source can neither be created nor deleted. It is provided out of the box to the user.

Configure a Context Hub List as an Enrichment Source

This topic provides instructions on how to configure a Context Hub list as an enrichment source for ESA. Once a Context Hub list is added as an enrichment source, analysts can use the configured list as a statement condition when creating an ESA rule. Any changes made to the list from within Context Hub are automatically reflected in the enrichment source in real-time. For example, you could create a list of IP addresses in Context Hub and then use that list as either a blacklist or whitelist as part of a correlation rule condition. Any subsequent changes made to the IP list in Context Hub will be reflected in the enrichment source in real-time, to ensure the correlation rule operates with a constantly updating set of information.

Prerequisites

Before configuring a Context Hub list as an enrichment source, the list must first be created as a data source in Context Hub. Any list created in Context Hub is supported and the lists may contain string or numeric values, including IP addresses. For information on creating a list as a data source in Context Hub, see the *NetWitness Context Hub Configuration Guide*.

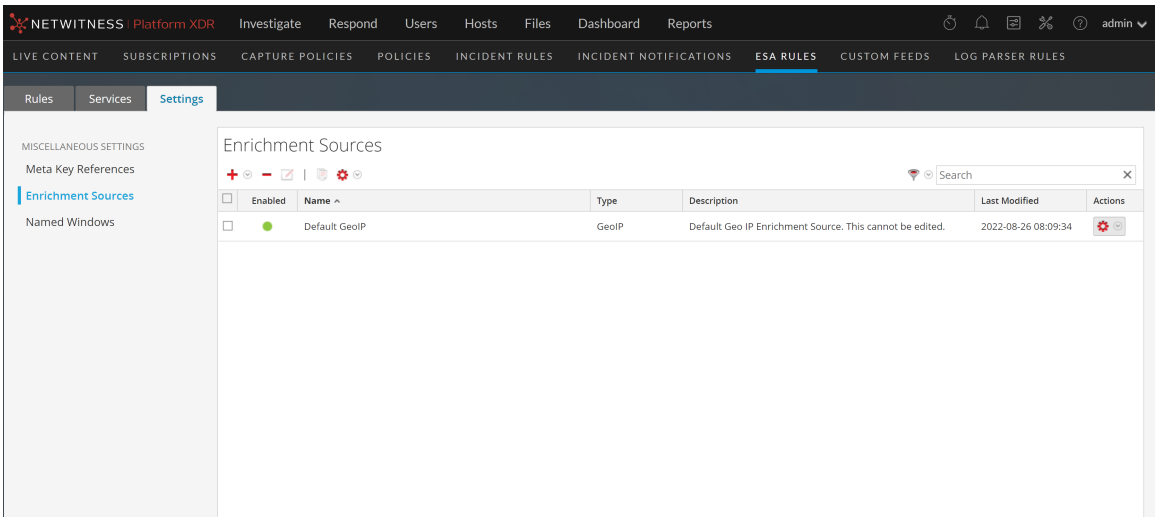
Caution: When creating a Context Hub list for use as an enrichment source, the list name and its field names cannot include any spaces or special characters, or start with a number. If you do not follow this naming convention, when you attempt to add the list as an enrichment source in ESA, an error message will be displayed and you will not be allowed to add the list.

IMPORTANT: If you rename a Context Hub list or recreate the Context Hub list with the same name, update the ESA rules that use that Context Hub list, and then redeploy the ESA rule deployments that contain those rules.

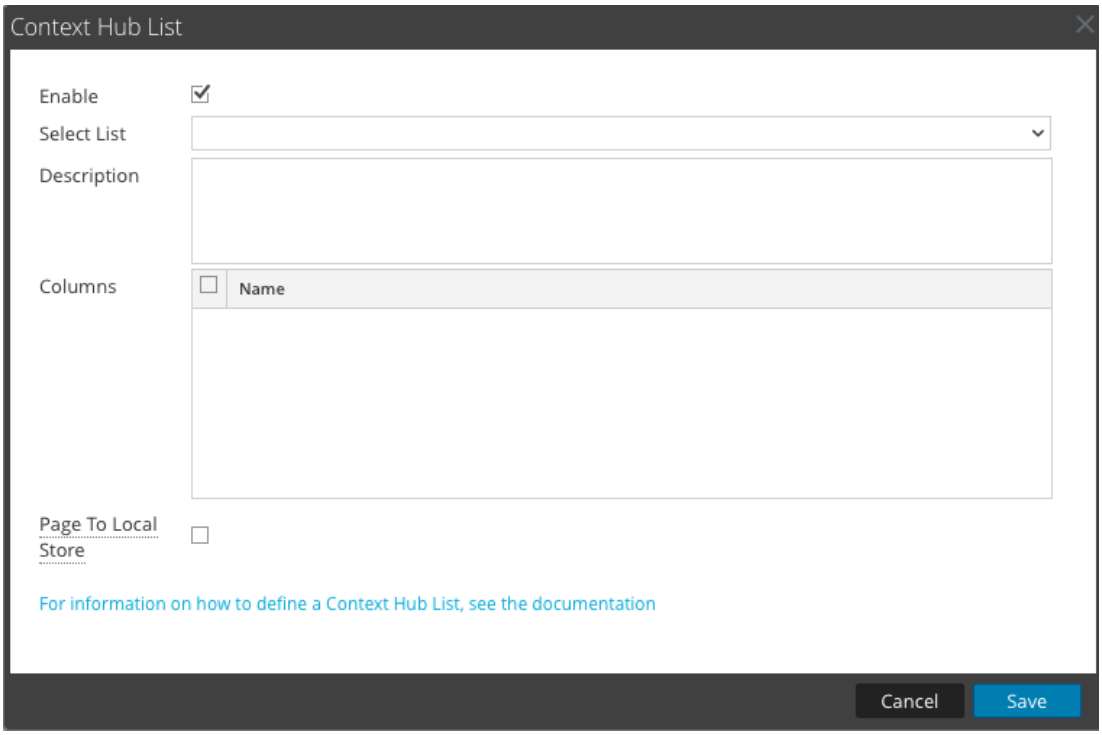
Configure a Context Hub List as an Enrichment Source

1. Go to  **(Configure) > ESA Rules > Settings** tab.
2. In the options panel, select **Enrichment Sources**.

   The Enrichment Sources panel is displayed.



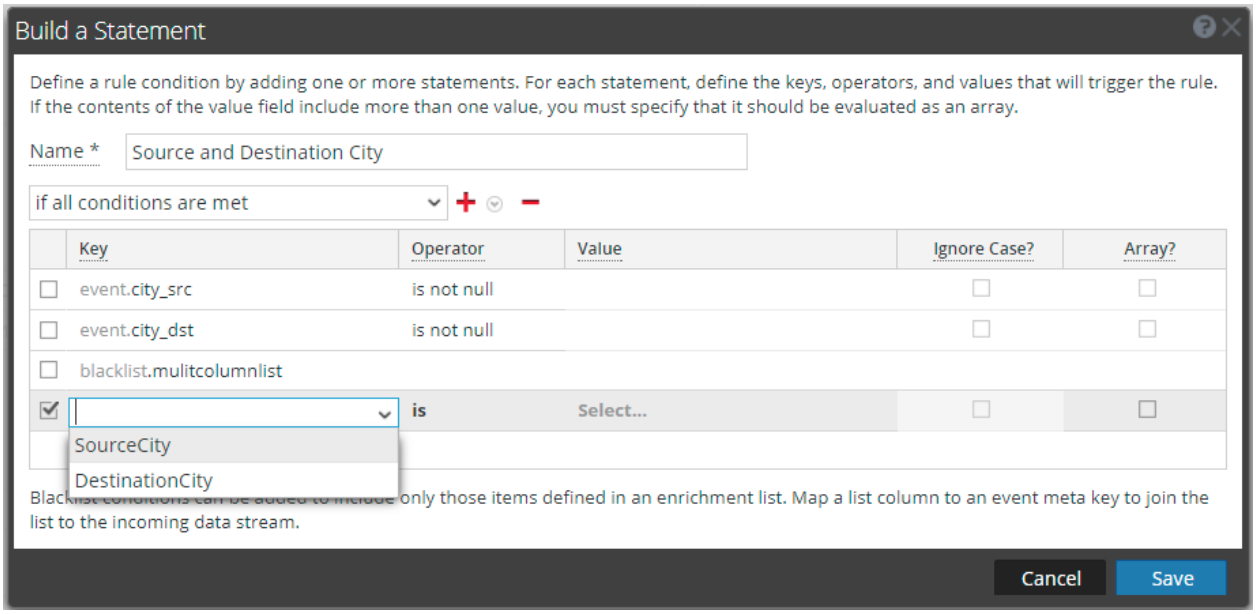
3. From the  drop-down menu, select **Context Hub**.



4. Select **Enable** to enrich alerts with a Context Hub list. This is selected by default. If disabled, the alerts will not be enriched with the configured Context Hub list.
5. Select the desired Context Hub list from the **Select List** drop-down menu of pre-configured lists.
6. (Optional) In the **Description** field, type a brief description about the selected Context Hub list. The text entered here is displayed on the Enrichment Sources panel.
7. In the **Columns** field, all columns included in the selected Context Hub list are listed. Click to enable or disable the columns in the list that you wish to include when using this list as an enrichment source in an alert.

8. (Optional) Click to enable the **Page To Local Store** option. This option is useful if you have a very large list and performance is affected. If this is the case, enabling this option will write a copy of the Context Hub list to the local disk to improve performance.
9. Click **Save**.
   The Context Hub list is configured. You can now add it to an ESA rule as part of a condition statement as either a blacklist or a whitelist condition.

The following figure illustrates adding a Context Hub list as part of a condition statement. In this example, a context Hub list named "multicolumnlist" was added as a blacklist condition. The list contains two columns, SourceCity and DestinationCity. The next step would be to select one of the column names as the subcondition and then specify the operator and enter the meta value for the corresponding value field.



For complete details for adding a whitelist or blacklist to a condition statement, see Step 2. Build a Rule Statement.

To add a Context Hub list as a condition to an existing rule, select to edit the desired rule in the Rule Library, then add a condition in the Conditions section and select to add a whitelist or blacklist condition to the new condition statement.

Configure an In-Memory Table as an Enrichment Source

This topic provides instructions on how to configure an in-memory table. When you configure an in-memory table, you upload a .CSV file as an input to the table. You can associate this table with a rule as an enrichment source. When the associated rule generates an alert, ESA will enrich the alert with relevant information from the in-memory table.

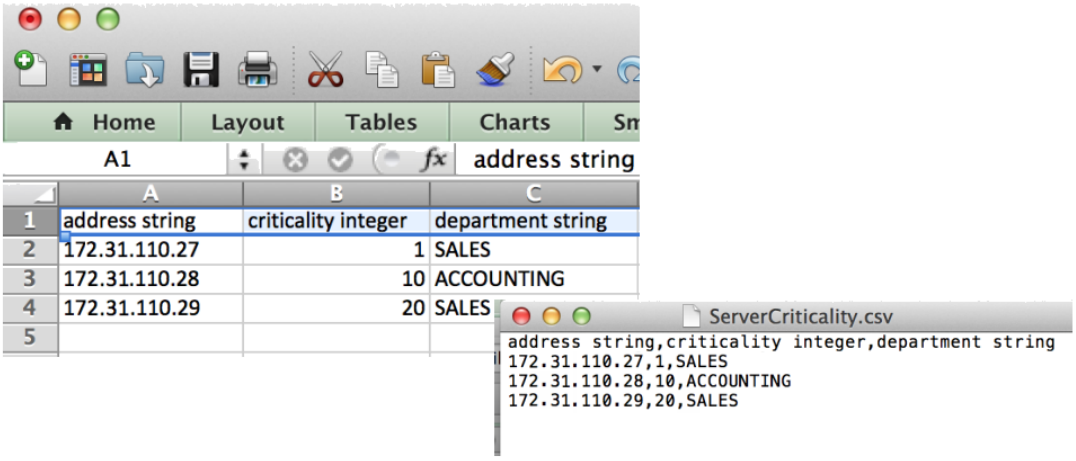
For example, a rule could be configured to detect when a user tries to download freeware and to identify the person by user ID in the alert. The alert could be enriched with additional information from an in-memory table that contains details such as full name, title, office location and employee number.

Note: It is preferable to use Context Hub List enrichment sources instead of In-Memory Table enrichment sources for rules. Recurring In-Memory Tables are no longer supported; use Content Hub Lists as enrichment sources. For more information, see Configure a Context Hub List as an Enrichment Source.

Prerequisites

- The column name in the .CSV file cannot have whitespace characters.
  For example *Last_Name* is correct, and *Last Name* is incorrect.
- The .CSV file must begin with a header line that defines fields and types.
  For example, *address string* would define the header field as *address*, and the type as *string*.

The following shows a valid .CSV file represented as a .CSV and as a table.

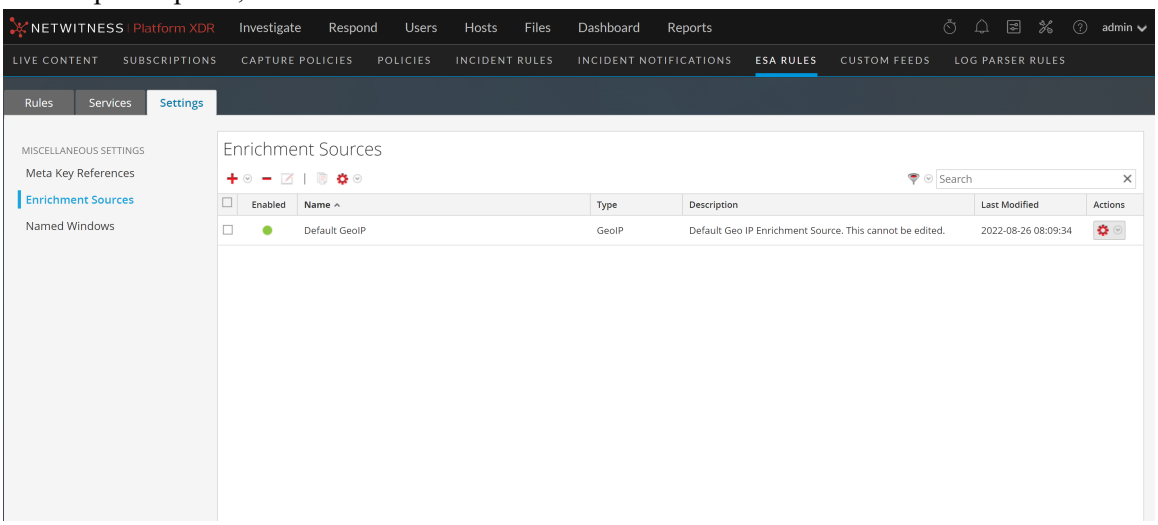


Configure an Ad hoc In-Memory Table

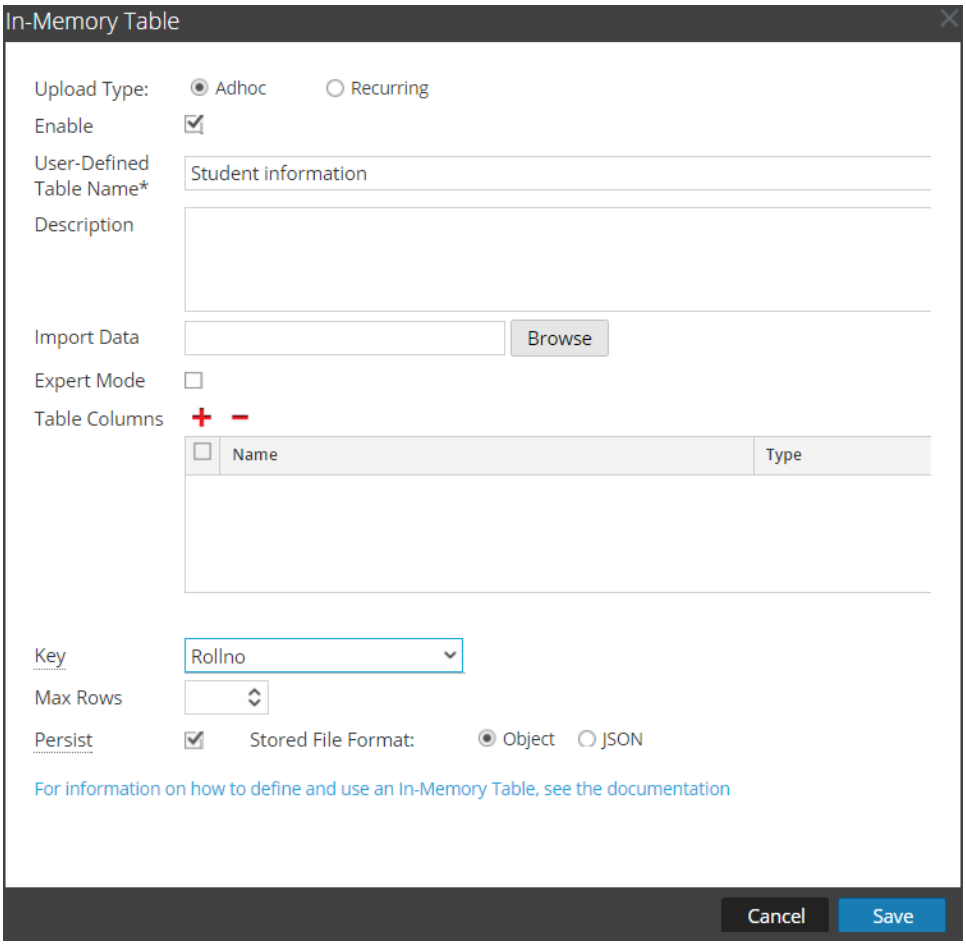
Note: It is preferable to use Context Hub List enrichment sources instead of In-Memory Table enrichment sources for rules. Recurring In-Memory Tables are no longer supported; use Content Hub Lists as enrichment sources. For more information, see Configure a Context Hub List as an Enrichment Source.

1. Go to **(Configure) > ESA Rules**.
   The Configure view is displayed with the ESA Rules tab open.
2. Click the **Settings** tab.

3. In the options panel, select **Enrichment Sources**.



4. In the **Enrichment Sources** section, click  > **In-Memory Table**.



5. Describe the in-memory table:
   a. Select **Ad hoc**.
   b. By default, **Enable** is selected. When you add the in-memory table to a rule, alerts will be enriched with data from it.
   If you add an in-memory table to a rule but do not want alerts to be enriched, deselect the checkbox.
   c. In the **User-Defined Table Name** field, type a name, such as Student Information, for the in-memory table configuration.

   Note: Do not use any Esper keyword as **User-Defined Table Name** since this causes an error while using this enrichment in the ESA Rule. For Esper keywords, see Reserved keywords.

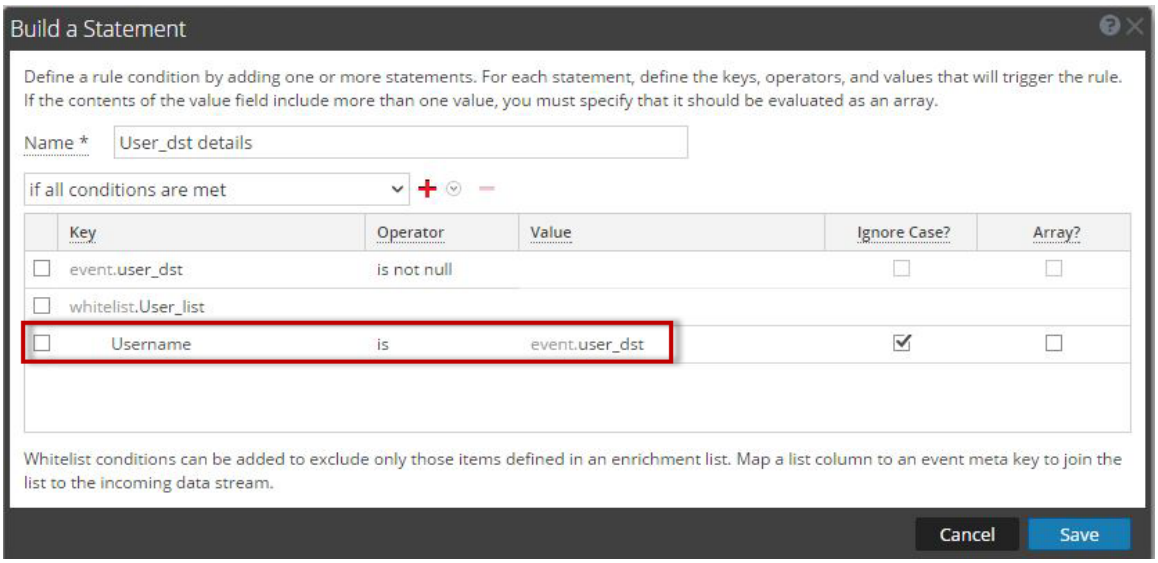
   d. If you want to explain what the enrichment adds to an alert, type a **Description** such as:
   When an alert is grouped by Rollno, this enrichment adds student information, such as name and marks.
6. In the **Import Data** field, select the .CSV file that will feed data to the in-memory table.
7. If you want to write an EPL query to define an advanced in-memory table configuration, select **Expert Mode**.
   The Table Columns are replaced by a **Query** field.
8. In the **Table Columns** section, click + to add columns to the in-memory table.
9. If a valid file is selected in the Import Data field, the columns populate automatically.

Note: If you selected Expert mode, a Query field is displayed instead of Table Columns.

10. In the **Key** drop-down menu, select the field to use as the default key to join incoming events with the in-memory table when using a CSV-based in-memory table as an enrichment. By default, the first column is selected. You can also later modify the key when you open the in-memory table in enrichment sources.
11. In **Max Rows** drop-down menu, select the number of maximum number of rows that can reside in the in-memory table at a particular instance.
12. Select **Persist** to preserve the in-memory table on disk when the ESA service stops and to re-populate the table when the service restarts.
13. In **Stored File Format** field, do one of the following:
    - Select **Object**, if you want to store the file in a binary format.
    - Select **JSON**, if you want to store the file in a text format.
    By default, **Object** is selected.
14. Click **Save**.
    The adhoc in-memory table is configured. You can add it to a rule as an enrichment or part of the rule condition. See Add an Enrichment to a Rule.

When you add an in-memory table, you can add it to a rule as an enrichment or as a part of the rule condition. For example, the following rule uses an in-memory table as a part of the rule condition to create a whitelist, and it also uses an in-memory table of details in the user_dst file to enrich the alert that is displayed.

The rule shows the in-memory table as a whitelist rule condition:



Next, the alert is enriched with the User_list in-memory table:

Therefore, the user_dst in-memory table is used to create a whitelist, and it is also used to enrich the data in the alert if the alert is triggered.

Add a Recurring In-Memory Table

Recurring In-Memory Tables are no longer supported; use Content Hub Lists as enrichment sources. For more information, see Configure a Context Hub List as an Enrichment Source.

It is preferable to use Context Hub List enrichment sources for ESA rules instead of In-Memory Table enrichment sources. You can share Context Hub List enrichment sources across the NetWitness Platform. You can only use the In-Memory Table with ESA.

Note: Database, Database Connection, Warehouse Analytics, and Recurring In-Memory Tables as enrichment sources are not supported for the ESA Correlation service in NetWitness 11.3 and later.

Add an Enrichment to a Rule

This topic tells how to add a previously configured enrichment source to a rule. When ESA creates an alert, information from the source gets included in it.

Adding an enrichment to a rule allows you to request for look ups into a variety of sources and include the results in the outgoing alerts, giving you a more detailed alert. This procedure requires role permissions for Administrator, DPO, and SOC Manager.

Note: This procedure does not apply to adding a Context Hub list as an enrichment to a condition statement in an existing rule. For information see Configure a Context Hub List as an Enrichment Source.

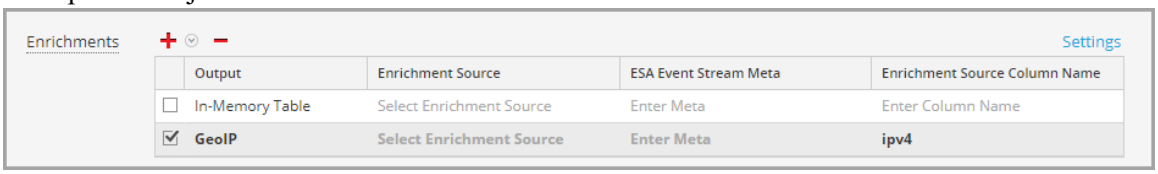
To add an enrichment to a rule:

1. Go to **(Configure) > ESA Rules**.
2. In the **Rule Library** view, do one of the following:
   - Double-click a rule.
   - Select a rule and click in the **Rule Library** toolbar.

   The Rule Builder panel is displayed in a new NetWitness tab.
3. In the **Enrichments** section, click and select any of the following enrichment types:
   - In-Memory Table
   - GeoIP

   Note: If you use a GeoIP source, ipv4 is automatically populated, and is not editable.

   The enrichment types that you have selected are displayed in the table.
4. For the added enrichment type, perform the following:
   - In the **Output** column, select the type that you have configured.
   - In the **Enrichment Source** drop-down list, select the enrichment source defined.
   - In the **ESA Event Stream Meta** field, type the event stream meta key whose value will be used as one operand of join condition.

   - In the **Enrichment Source Column Name** field, type the enrichment source column name whose value will be used as another operand of the join condition.
5. Select **Debug**. This adds an @Audit('stream') annotation to the rule. This is useful when debugging the Esper rules.
6. Click **Show Syntax** to test if the defined ESA rule is valid.
7. Click **Save**.

For details on parameters and their descriptions, see Rule Builder Tab.

Deploy Rules to Run on ESA

This section explains how an ESA Rule Deployment works and how to set up a deployment to run a group of ESA rules. Administrator, SOC Manager, or Data Privacy Officer role permissions are required for all procedures in this section.

To create an ESA rule deployment, you need to perform the steps described in Managing ESA Rules, Data Sources and Deployments

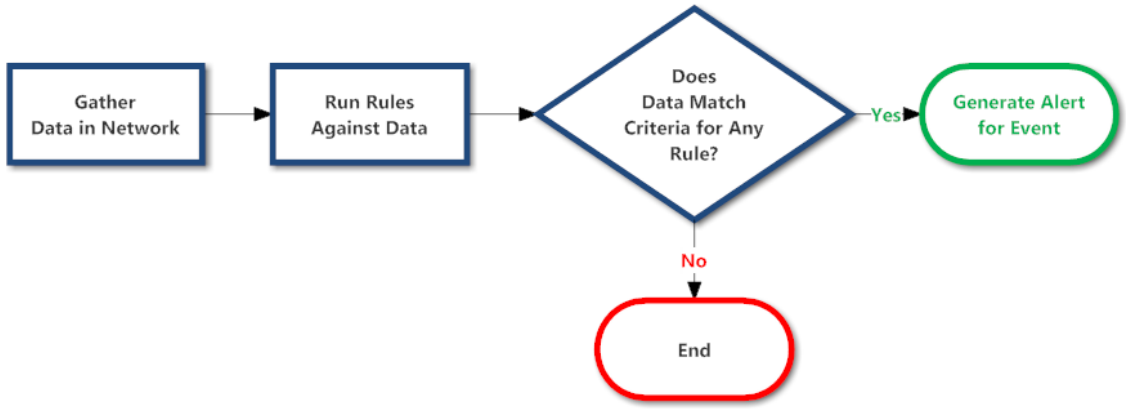
How an ESA Rule Deployment Works

An ESA rule deployment consists of an ESA service, one or more data sources, and a set of ESA rules. When you deploy rules, the ESA service runs them to detect suspicious or undesirable activity in your network. Each ESA rule detects a different event, such as when a user account is created and deleted within one hour.

The ESA service performs the following functions:

1. Gathers **data** in your network
2. Runs ESA **rules** against the data
3. Applies rule **criteria** to data
4. Generates an **alert** for the captured event

The following graphic shows this workflow:



In addition, you may want to perform other steps on your deployment, such as replacing an ESA service, changing a data source, editing or deleting a rule from the deployment, renaming or deleting the deployment, or showing updates to the deployment. For descriptions of these procedures, Managing ESA Rules, Data Sources and Deployments.

Managing ESA Rules, Data Sources and Deployments

From 12.1 and later versions, the ESA deployments are managed by policies and groups on the **(Configure)** > **Policies** page.

If you want to create any new ESA rules, add new data sources or perform any new ESA deployments, see the following sections in the Live Services Management guide:

- To view, add, edit, or delete data sources, see section "Manage ESA Datasources".
- To add, edit or delete an ESA rule, see the following topics, "Create an ESA Rule", "Edit, Duplicate or Delete a Rule", and "Delete an ESA Rule".
- To view, create, edit, remove, deploy or stop a deployment, see section "Manage Deployments".

View ESA Stats and Alerts

When ESA generates alerts, you can view details about how the rules performed, such as statistics on the engine, rule, and alert, and you can also view information on which rules are enabled or disabled. For instructions on viewing ESA stats, see View Stats for an ESA Service

When your ESA generates alerts, you can view the results in the Respond Alerts List view. This enables you to see trends and understand both the volume and frequency of alerts. For instructions on viewing alerts, see View a Summary of Alerts

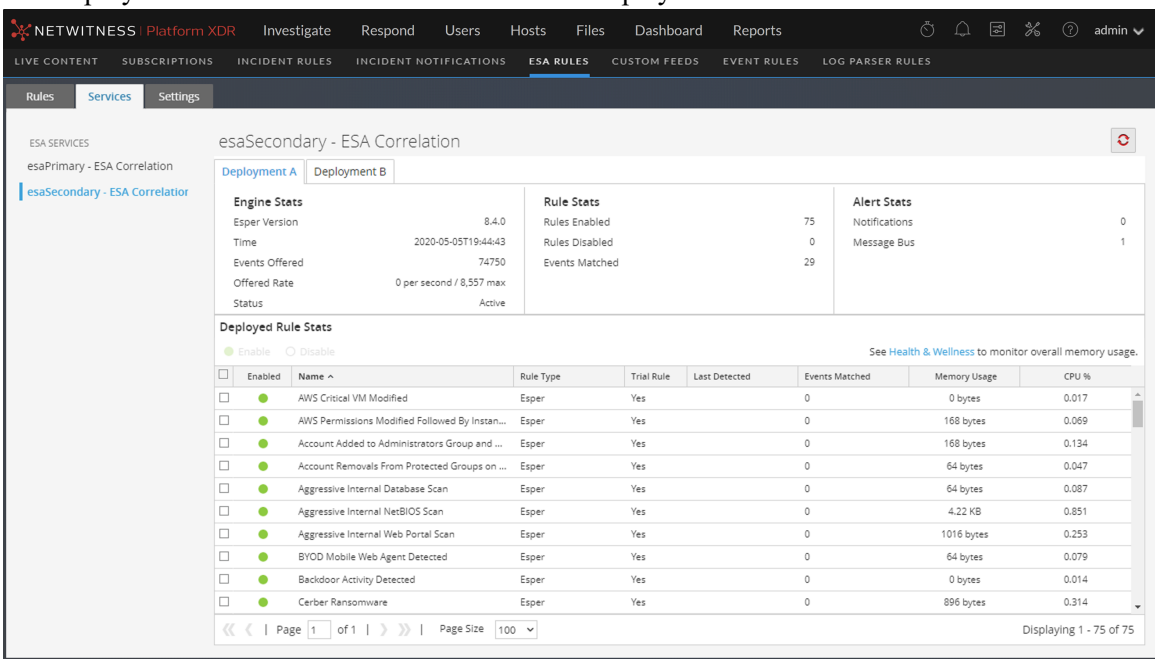
View Stats for an ESA Service

This topic describes how to view the deployment statistics (stats) for an ESA Correlation service. This procedure is useful when you are attempting to determine the effectiveness of a rule or troubleshoot an ESA rule deployment.

Caution: When you modify and re-deploy an ESA rule deployment, all of the stats are removed from that deployment. The generated alerts are not removed from NetWitness Respond.

View ESA Stats

1. Go to **(Configure) > ESA Rules > Services** tab.
2. From the **ESA Services** list on the left, select a service.
   The deployment stats for the selected service are displayed.



3. (This option applies to NetWitness version 11.3 and later.) In the Deployment view under the ESA Correlation service name, select the tab of the deployment you would like to view. For example,

select the Deployment A tab to view the stats for deployment A. Select the Deployment B tab to view the status for deployment B.

4. Review the following sections of ESA stats.
   For a complete description of each statistic in each section, see Services Tab.
   - **Engine Stats**
   - **Rule Stats**
   - **Alert Stats**
5. In the **Deployed Rule Stats**, review details about the rules deployed on the ESA.
   For a complete description of each column in each section, see Services Tab.
   - If the rule is enabled or disabled
   - What the rule name is
   - The type of rule
   - If the rule is running in Trial Rule mode
   - Last detected
   - Events matched
   - The amount of memory used by the rule
   - The percentage of the deployment CPU used by the rule (available in NetWitness version 11.5 and later)
6. To monitor overall memory usage and health of your ESA Correlation service, click **Health & Wellness**.

Enable or Disable Rules

1. In the **Deployed Rule Stats** panel, select a rule from the grid.
2. Click Enable to enable the rule, or click Disable to disable the rule.
   The Services tab is refreshed to show the changes, which take effect immediately.

Refresh the Statistics

The Services tab does not update statistics automatically unless you enable or disable a rule. To ensure you view current statistics:

1. Click in the upper right corner to refresh the information.
2. View the updated information.

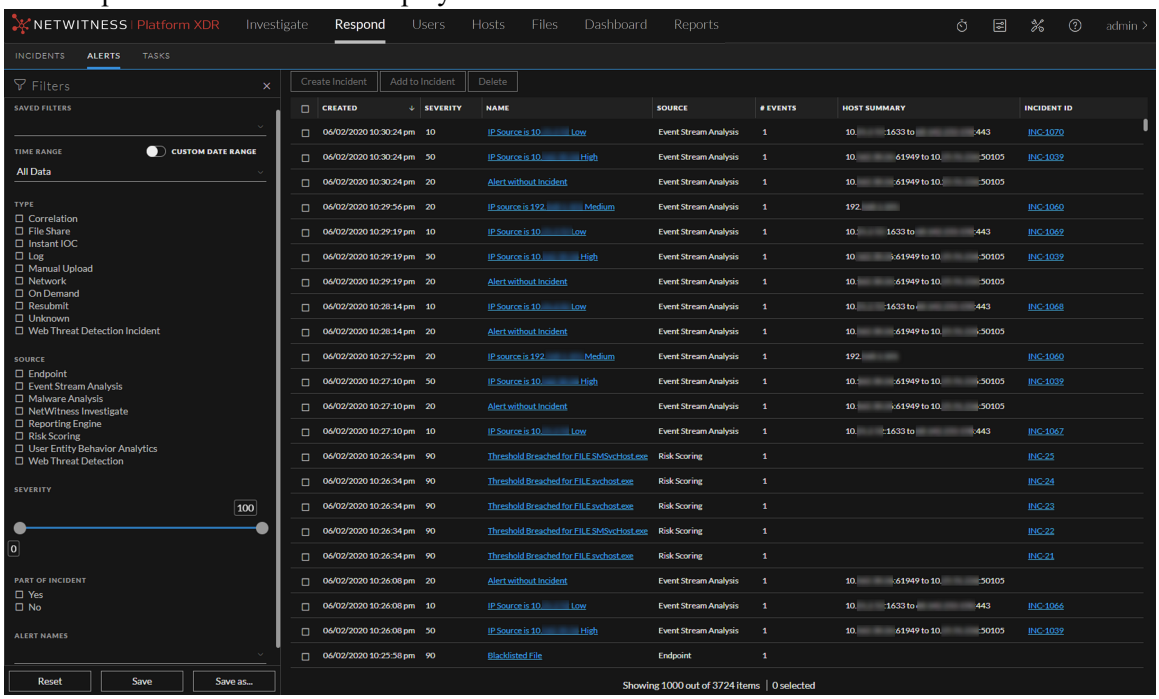
View a Summary of Alerts

In the Repond view, you can browse through various alerts from multiple sources. You can filter the alerts list to show only alerts of interest, such as by Alert Name, alert source, and a specific time range from the following sources:

1. Detect AI
2. Endpoint
3. Event Stream Analysis
4. Malware Analysis
5. NetWitness Investigate
6. Reporting Engine
7. Risk Scoring
8. User Entity Behavior Analysis
9. Web Threat Detection

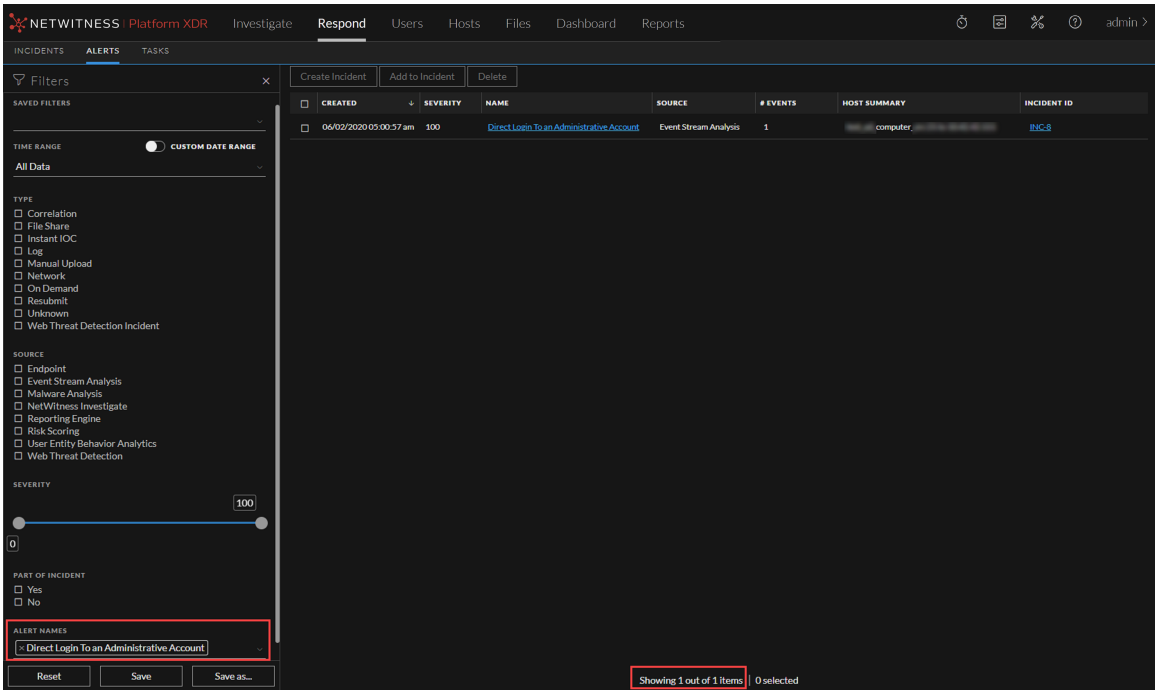
Perform the following steps to use the functionalities provided in the Respond view.

1. Go to **Respond > Alerts**.
   The Respond Alerts List view displays a list of all NetWitness alerts.



2. In the **Filters** panel on the left, you can filter the alerts list to view specific alerts for a specific time frame. For example, in the Alert Names section, you can select an alert for an ESA rule, such as

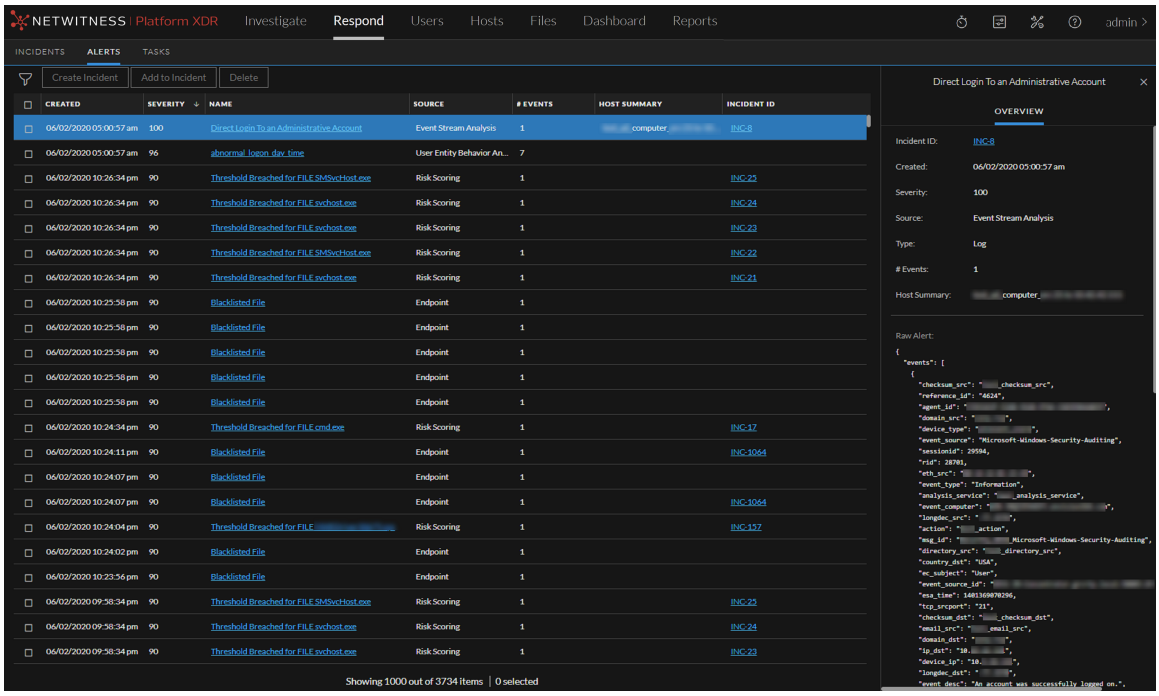
Direct Login to an Administrative Account, and leave the Time Frame set to Last Hour.
The alerts list to the right shows a list of alerts that match your filter selection along with a count of the alerts at the bottom of the alerts list.



The alerts list shows information about each of the alerts.

- **Created**: Displays the date and time when the alert was created in the source system.
- **Severity**: Displays the level of severity of the alert. The values are from 1 to 100.
- **Name**: Displays a basic description of the alert.
- **Source**: Displays the original source of the alert.
- **# of Events**: Indicates the number of events contained within an alert.
- **Host Summary**: Displays details of the host, like the host name from where the alert was triggered.
- **Incident ID**: Shows the incident ID of the alert. If there is no incident ID, the alert does not belong to an incident.

3. You can click an alert in the list to open an **Overview** panel on the right where you can view raw alert metadata.



For more information about filtering alerts and viewing alert details, see the *NetWitness Respond User Guide*.

Add an Advanced EPL Rule

This topic provides instructions to define rule criteria by writing an EPL query. EPL is a declarative language for handling high-frequency time-based event data. It is used to express filtering, aggregation, and joins over possibly sliding windows of multiple event streams. EPL also includes pattern semantics to express complex temporal causality among events.

Write an advanced EPL rule when rule criteria is more complex than what you can specify in Rule Builder.

It is outside the scope of this guide to explain EPL syntax.

- For EPL Documentation, see http://www.espertech.com/esper/esper-documentation/
- For the EPL Online Tool, see http://esper-epl-tryout.appspot.com/epltryout/mainform.html

For best practices on writing advanced EPL rules, see ESA Rule Writing Best Practices.

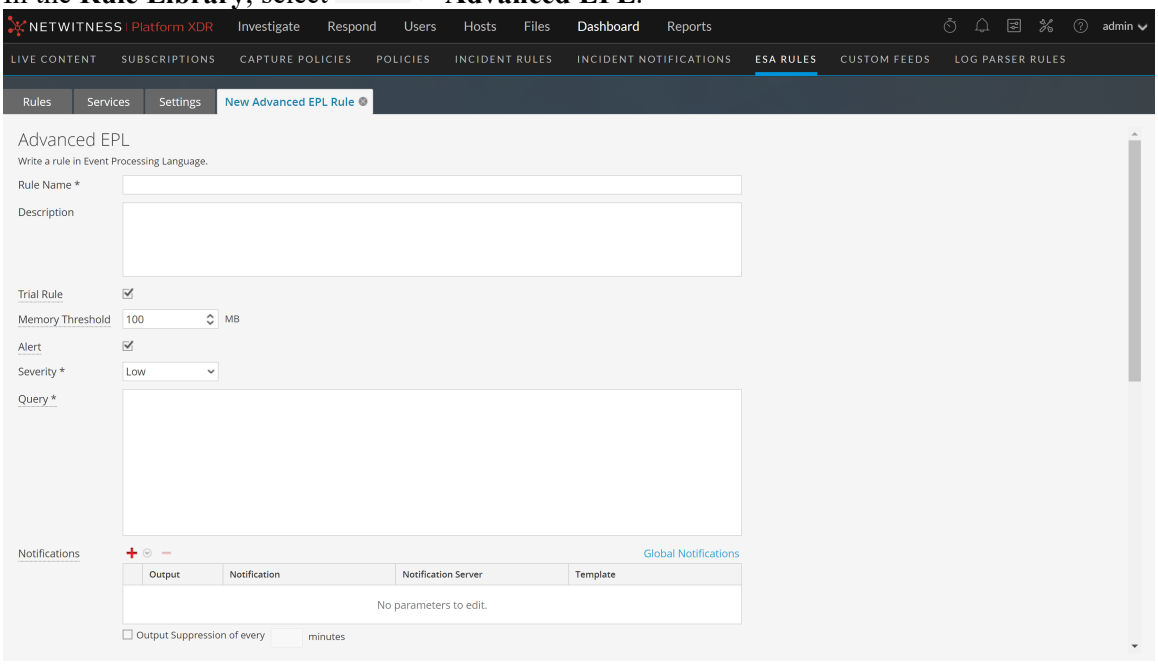
Prerequisites

The following are prerequisites for adding an advanced rule:

- You must know Event Processing Language (EPL).
- You must understand ESA Annotations to mark which EPL statements are linked to generating alerts.

Add an Advanced EPL Rule

1. Go to **(Configure) > ESA Rules**.
2. In the **Rule Library**, select **> Advanced EPL**.



3. Type a unique, descriptive name in the **Rule Name** field.

   This name will appear in the Rule Library so be specific enough to distinguish the rule from others.

4. In the **Description** field, explain which events the rule detects.

   The beginning of this description will appear in the Rule Library

5. Select **Trial Rule** to automatically disable the rule if all trial rules collectively exceed the memory threshold.

   Use trial rule mode as a safeguard to see if a rule runs efficiently and to prevent downtime caused by running out of memory. For more information, see Working with Trial Rules.

6. (This option applies to 11.5 and later.) Enter a **Memory Threshold** for a rule that uses memory, such as a rule that contains windows or pattern matching. If the configured memory threshold is exceeded, the rule gets disabled individually and an error is displayed for that rule on the (Configure) > ESA Rules > Services tab. The Memory Threshold option works for trial rules and non-trial rules. New rules default to a 100 MB memory threshold. Rules that existed before version 11.5 do not have a default value and a memory threshold is not set.

7. (This option applies to 11.3 and later.) Select **Alert** to send an alert to Respond. Clear the checkbox if you do not want to send an alert to Respond. To turn alerts on or off for ALL rules, see the *ESA Configuration Guide*.

8. For **Severity**, classify the rule as Low, Medium, High or Critical.

9. To define rule criteria, write a **Query** in EPL.

   Note: For all meta key names, use an underscore not a period. For example, `ec_outcome` is correct but `ec.outcome` is not.

   Supported meta entities:

S. No	Supported Meta Entities	Description
1	fullname_all	
2	eth_all	
3	ip_all	Combines all the IPv4 meta keys.
4	ipv6_all	Combines all the IPv6 meta keys.
5	port_src_all	
6	port_dst_all	
7	dir_path_all	
8	org_all	
9	geoip_all	
10	port_all	
11	domain_all	
12	email_all	
13	filename_all	
14	directory_all	
15	checksum_all	
16	param_all	
17	context_all	
18	attack_all	
19	analysis_all	
20	compromise_all	
21	inv_all	
22	outcome_all	
23	ec_all	
24	user_all	
25	host_all	

26	client_all

10. For dynamic statement name generation in ESA, you must enclose the meta keys in curly brackets and include this annotation in the syntax:

    ```
    @Name("RIG {ip_src} {alias_host} {ec_activity}")
    ```

    where,

    - RIG is the static part of the statement name
    - {ip_src}, {alias_host}, {ec_activity} is the dynamic part of the statement name

    Note: If any of the metas in the dynamic part of the statement name has a null value, it is displayed as a static text.

    If a rule should generate an alert, include this ESA annotation in the syntax:

    ```
    @RSAAlert
    ```

    For more information on ESA Annotations, see ESA Annotations.

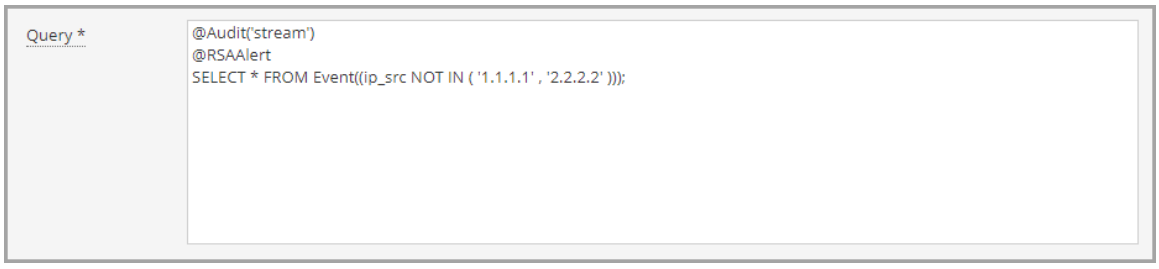
Validate an Advanced EPL Rule

You can confirm that an ESA rule generates the expected alerts by testing the rule logic using JSON input data. You can view the alerts in the output, but this test does not send any alert notifications.

If you want to view all of the debug information, include an `@Audit('stream')` annotation to your rule query and view the Debug Log in the test output. To enable auditing you require to add `@Audit` to the rule and set the logging level for the Esper audit package to INFO. This can be done by creating `correlation-server.yml` file under `/etc/netwitness/correlation-server` with this content and restarting the correlation-server service with systemctl restart rsa-nw-correlation-server:

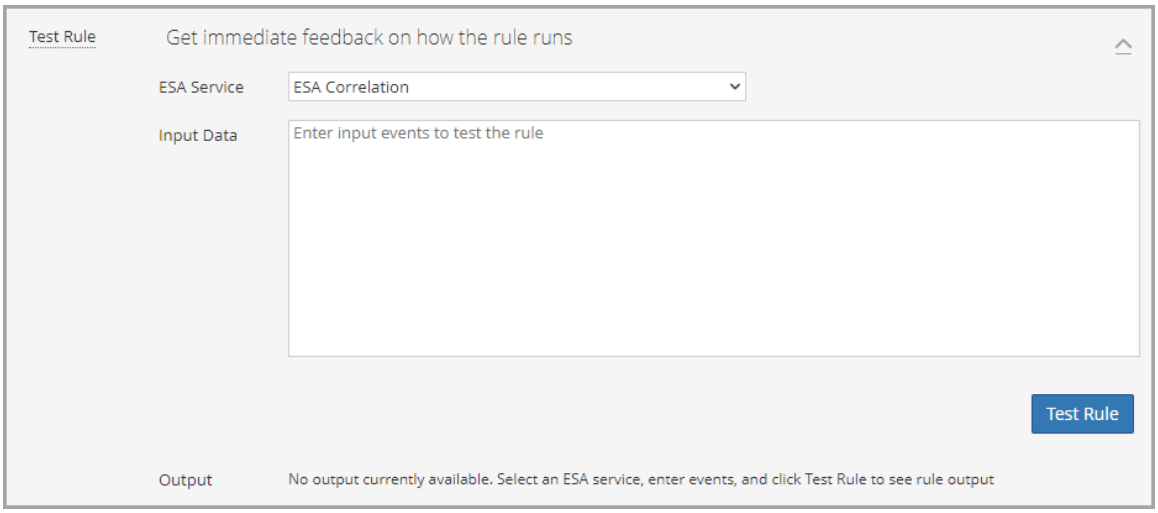
```
logging:
   level:
     com.espertech.esper.audit: INFO
```

The following basic example query contains the `@Audit('stream')` annotation and queries for events that do not have a source IP of 1.1.1.1 or 2.2.2.2.



1. If you are not already in the rule, go to **(Configure) > ESA Rules > Rules** tab and in the **Rule Library**, open the ESA rule that you want to test.

2. Scroll down to the **Test Rule** section.



3. In the **ESA Service** field, select the ESA Correlation service to process the rule. Use the same ESA Correlation service that you plan to use in the ESA rule deployment that contains the rule.
4. In the **Input Data** field, enter the input events to test the rule. Download the events from the Investigate view in JSON format, copy the events, and paste them in this field. You can do this from the Investigate > Navigate view or the Investigate > Events view.

   To download the events from the Investigate > Navigate view:

   a. In the main menu, go to **Investigate > Navigate** in a new tab, select a data source, and click **Navigate**.
   b. In the Navigate view, click **Load Values** and click a meta value to filter the events.
   c. Save the events as meta in the JSON file format [Save Events > Meta > (name the file) > Export Meta Format: choose JSON].
   d. In the toolbar click the **(Jobs)** icon and then click **View Your Jobs**.
   e. In the Jobs panel, download your extracted meta, for example: **`investigation-2020-May-19-08-30-20.json`**.
   f. Go to back to the **(Configure) > ESA Rules** tab opened previously and copy the contents of the JSON file into the **Input Data** field in your ESA rule.

   To download the events from the Investigate > Events view:

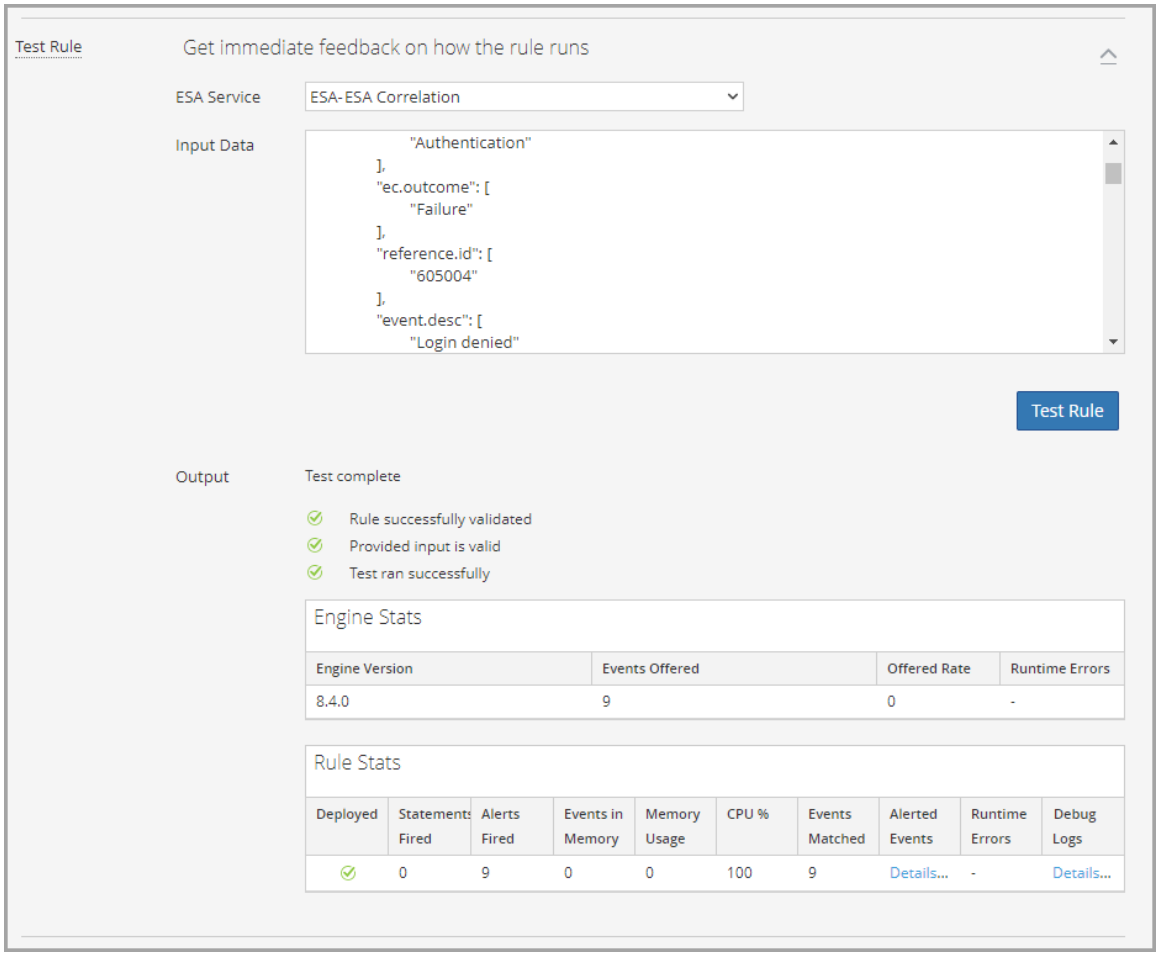
   a. In the main menu, go to **Investigate > Events** in a new tab.
   b. In the Events view, enter a query for the ESA rule test.
   c. Select the events to use and in the **Download** or **Download All** menu, select **Visible Meta as JSON** or **All Meta as JSON**, depending on the size of your selection.

d. In the main menu, go to **Dashboards** and in the toolbar click the **(Jobs)** icon and then click **View Your Jobs**.

e. In the Jobs panel, download your extracted meta, for example: **Concentrator_ALL_EVENTS_ALL_META.json**.

f. Go to back to the **(Configure) > ESA Rules** tab opened previously and copy the contents of the JSON file into the **Input Data** field in your ESA rule.

5. Click **Test Rule**. The **Output** field shows the output of your rule and you can determine if the results meet your requirements.



Note: If you are testing any Rule that has meta key defined as type 'short', the Test Rule will not generate alert for the event.

The following table describes the test rule output **Engine Stats**.

Field	Description
Engine Version	Esper version running on the ESA service

Field	Description
Events Offered	Number of events processed by the ESA service since the last service start
Offered Rate	The rate that the ESA service processes current events / The maximum rate that the ESA service processed events
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the ESA rule deployment.

The following table describes the test rule output **Rule Stats**.

Field	Description
Deployed	A green checkmark indicates that the rule is deployed on the selected ESA service.
Statements Fired	The number of statements that fired the alerts
Alerts Fired	The number of alerts generated from the test data
Events in Memory	The number of events placed in memory by the rule
Memory Usage	The total amount of memory used by the rule
CPU %	The percentage of the deployment CPU used by the rule. For example, a deployment with 1 rule shows 100% CPU usage for that rule and a deployment with two equally CPU heavy rules show 50% each.
Events Matched	The number of events that matched the rule
Alerted Events	If applicable, this field can contain a link to events that caused an alert.
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the rule.
Debug Logs	This field contains a link to Esper debug (audit) logs.

Event Processing Language (EPL)

This topic describes Event Processing Language (EPL), a declarative language for dealing with high frequency time-based event data. ESA uses Event Processing Language (EPL), a declarative language for dealing with high frequency time-based event data. It is used for express filtering, aggregation, and joins over possibly sliding windows of multiple event streams. EPL also includes pattern semantics to express complex temporal causality among events. It can perform, but is not limited to, the following functions:

- Filter Event
- Alert Suppression
- Compute percentages or rations
- Average, count, min and max for a given time window
- Correlate events arriving in multiple stream
- Correlate events that arrive out of order
- On-Off Windows
- Followed-by and Not Followed-by support
- Regex filter support

Databases require explicit querying to return meaningful data and are not suited to push data as it changes. The developer must implement the temporal and aggregation logic himself. By contrast, the EPL engine provides a higher abstraction and intelligence and can be thought of as a database turned upside-down. Instead of storing the data and running queries against stored data, EPL allows applications to store queries and continuously run the data through. Response from the EPL engine is real-time when conditions occur that match user defined queries.

For best practices on writing advanced EPL rules, see ESA Rule Writing Best Practices.

Advanced ESA rules require correct character case, but in the Investigate Navigate view all characters are converted to lowercase. However, the meta may not be lowercase despite appearances in the Investigate Navigate view. To ensure you are using the correct case, you can make a strict pattern match for better performance.

Strict Pattern Match Example

```
@RSAAlert(oneInSeconds=0)
SELECT * FROM Event(
     (medium IN ( 1 ) AND
     filetype IN ( 'pdf' , 'windows_executable' , 'x86 pe' , 'windows
executable' ))
     ).win:time(5 Minutes)
MATCH_RECOGNIZE (
     MEASURES E1 as e1_data , E2 as e2_data
     PATTERN (E1 E2)
     DEFINE
          E1 as (E1.filetype IN ('pdf')),
          E2 as (E2.filetype IN ( 'pdf' , 'windows_executable' , 'x86 pe' ,
'windows executable' ))
```

```
);
```

Caution: Care should be taken to only add the case-insensitive *toLowerCase()* function on meta keys as needed. The *toLowerCase()* function can cause significant performance decreases. Consider checking the Investigate Events view or the Event Analysis view to see the real character case for meta fields and avoid unnecessary usage of the function.

For the purposes of online help, basic statements are used to illustrate how to set up ESA; however, for more information about writing EPL statements, the http://www.espertech.com site provides tutorials and examples.

Note: In NetWitness version 11.5, ESA Correlation supports Esper version 8.4.0. In version 11.4, ESA Correlation supports Esper version 8.2.0 and in version 11.3, ESA Correlation supports Esper version 7.1.0.

ESA Annotations

This topic describes annotations that NetWitness provides to use in advanced EPL rules.

For best practices on writing advanced EPL rules, see ESA Rule Writing Best Practices.

@RSAContext Annotation (11.5 and later)

The @RSAContext annotation can be used in advanced rules to dynamically add or remove data from a Context Hub list after the rule fires. For example, you can create a rule that automatically adds an IP address to a blacklist and removes it from a whitelist.

You can update a single-column or a multi-column Context Hub list. The @RSAContext annotation also performs error handling when the Context Hub list cannot be reached.

Prerequisites

Before deploying a rule using the @RSAContext annotation, the list must exist in Context Hub. For information on creating a Context Hub list, see the *Context Hub Configuration Guide*. Go to the NetWitness All Versions Documents page and find NetWitness Platform guides to troubleshoot issues.

IMPORTANT: If you rename a Context Hub list or recreate the Context Hub list with the same name, update the ESA rules that use that Context Hub list, and then redeploy the ESA rule deployments that contain those rules.

Single-Column Context Hub Lists

The @RSAContext annotation uses the following format for a single column Context Hub list:

```
@RSAContext (list="<single_column_list>", action=<DELETE_ENTRY or ADD_ENTRY>,
onError = <STOP_ALL_RULE_PROCESSING_AND_WAIT or IGNORE_ERROR_AND_CONTINUE>,
fields={"LIST=<meta_key>"})
```

The @RSA Context parameters for a **single-column** Context Hub List are described in the following table.

Parameter	Description
`list`	Where `<single_column_list>` is the name of the single-column Context Hub list (whitelist or blacklist).
`action`	You can add an entry to or remove an entry from the list: • `DELETE_ENTRY` - Removes an entry from the list. • `ADD_ENTRY` - Adds an entry to the list. You can only have one action per `@RSAContext` entry.

Parameter	Description
onError	Identifies how to handle errors, for example, when the Context Hub list is not available or is full. You can choose one of the following options: • STOP_ALL_RULE_PROCESSING_AND_WAIT: If any error occurs with the selected actions on the Context Hub list, stop all rule processing including data aggregation and retry indefinitely until the actions are successfully executed. Halting processing for one Context Hub list results in halting processing for all Context Hub lists. Select this option if it is important to update a Context Hub list before processing any more rules. For example, you may have a blacklist that must be updated before continuing rule processing. • IGNORE_ERROR_AND_CONTINUE: If any error occurs with the selected actions on the Context Hub list, ignore the error and the @RSAContext entry, and continue processing. If it is not a Context Hub error, assume that the error occurs before reaching Context Hub, and retry until the actions are successfully executed. Select this option if continuing rule processing is more important than updating the Context Hub list.
fields	Maps a meta key value to the column in the Context Hub list. Single-column lists always show LIST= in capital letters.

The first example deletes a source IP address from an IP whitelist. However, if the list is not available, the source IP address is not removed from the list and processing is continued.

```
@RSAContext (list="ip_whitelist", action=DELETE_ENTRY, onError = IGNORE_ERROR_
AND_CONTINUE, fields={"LIST=ip_src"})
```

The second example adds a source IP address to an IP blacklist. However, if the list is not available, all rule processing stops including data aggregation, and the ESA Correlation service retries indefinitely until the entry is added to the specified Context Hub list.

```
@RSAContext (list="ip_blacklist", action=ADD_ENTRY, onError = STOP_ALL_RULE_
PROCESSING_AND_WAIT, fields={"LIST=ip_src"})
```

Multi-Column Context Hub Lists

The @RSAContext annotation uses the following format for a multi-column Context Hub list:

```
@RSAContext (list="<multi_column_list>", action=<DELETE_ENTRY or ADD_ENTRY>,
onError = <STOP_ALL_RULE_PROCESSING_AND_WAIT or IGNORE_ERROR_AND_CONTINUE>,
fields={"<column1>=<meta_key1 or constant value1>",<column2>=<meta_key2 or
constant value2>",<column n>=<meta_key n or constant value n>",})
```

The @RSA Context parameters for a **multi-column** Context Hub List are described in the following table.

Parameter	Description
list	Where <multi_column_list> is the name of the multi-column Context Hub list.
action	You can add an entry to or remove an entry from the list: • DELETE_ENTRY - Removes an entry from the list. • ADD_ENTRY - Adds an entry to the list. You can only have one action per @RSAContext entry.
onError	Identifies how to handle errors, for example, when the Context Hub list is not available or is full. You can choose one of the following options: • STOP_ALL_RULE_PROCESSING_AND_WAIT: If any error occurs with the selected actions on the Context Hub list, stop all rule processing including data aggregation and retry indefinitely until the actions are successfully executed. Halting processing for one Context Hub list results in halting processing for all Context Hub lists. Select this option if it is important to update a Context Hub list before processing any more rules. For example, you may have a blacklist that must be updated before continuing rule processing. • IGNORE_ERROR_AND_CONTINUE: If any error occurs with the selected actions on the Context Hub list, ignore the error and the @RSAContext entry, and continue processing. If it is not a Context Hub error, assume that the error occurs before reaching Context Hub, and retry until the actions are successfully executed. Select this option if continuing rule processing is more important than updating the Context Hub list.
fields	Maps a meta key values or constant values to the columns in the Context Hub list. Multi-column lists show the names of each column in the fields parameter.*

The first example deletes a user and the associated source and destination IP addresses from a multi-column IP whitelist. However, if the list is not available, the specified fields are not removed from the list and processing is continued.

```
@RSAContext (list="MultColumn_whitelist", action=DELETE_ENTRY, onError =
IGNORE_ERROR_AND_CONTINUE, fields={"source=ip_src","destination=ip_
dst","user='smith'"})
```

The second example adds a user and the associated source and destination IP addresses to a multi-column IP blacklist. However, if the list is not available, all rule processing stops including data aggregation, and the ESA Correlation service retries indefinitely until the entry is added to the specified Context Hub list.

```
@RSAContext (list="MultColumn_blacklist", action=ADD_ENTRY, onError = STOP_
ALL_RULE_PROCESSING_AND_WAIT, fields={"source=ip_src","destination=ip_
dst","user='smith'"})
```

Automatic Context Hub List Updates

When using the @RSAContext annotation in your rules, string array meta keys (shown as `string[]` in the Meta Key References on the ESA Rules > Settings tab) are inserted and removed as separate lines in the Context Hub list. This enables lookups in the Investigate and Respond views.

Single Column Context Hub List Update Example

A single column Context Hub list has one column with the string array meta key `alias_host`. If the rule using that list fires and `alias_host` has three values (Google, Yahoo, and Dell), ESA Correlation adds three rows to the Context Hub list:

Google
Yahoo
Dell

Multi-Column Context Hub List Update Example

A multi-column Context Hub list has three columns. If the rule using that list fires and the meta keys have the following values:

- `username` (string array) with values Aimee and Chris
- `ip_dst` (string) with value 10.10.10.10
- `alias_host` (string array) with values Dell, Google, and Yahoo

ESA Correlation adds six rows to the Context Hub list:

Aimee, 10.10.10.10, Dell
Aimee, 10.10.10.10, Google
Aimee, 10.10.10.10, Yahoo
Chris, 10.10.10.10, Dell
Chris, 10.10.10.10, Google
Chris, 10.10.10.10, Yahoo

@RSAAlert Annotation

The @RSAAlert annotation is used to mark which EPL statements are linked to generating alert notifications. It is designed to work with the alert notification suppression feature in the Rule Builder user interface.

The @RSAAlert annotation can be useful when working with alert notifications, especially if you want to filter notifications, such as sending one notification for each user that triggers an alert.

For example, suppose you want to generate alert notifications for login failures. You could add the following statement:

*@RSAAlert select * from event(msg_id="login_fail")*

Event number	Message ID	username	src_IP	Time
1	login_fail	alice	1.2.3.4	10:00
2	login_fail	alice	1.2.3.4	10:01
3	login_fail	alice	6.7.8.9	10:01
4	login_fail	bob	1.2.3.4	10:01
5	login_fail	alice	1.2.3.4	10:03

For the above statement, five alert notifications are generated.

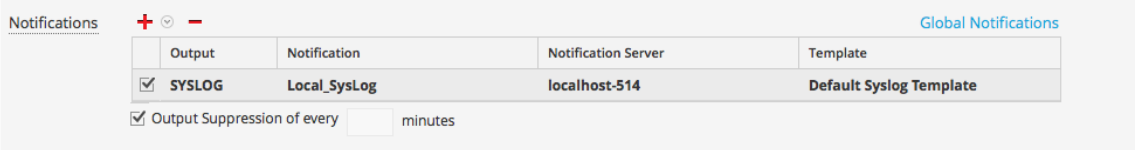
However, suppose you wanted to modify the statement to generate one alert for each separate username. You can use the *identifier* attribute. For example, the statement *@RSAAlert(identifier={"username"}) SELECT* FROM Event(msg_id="login_fail")* generates one notification for the first alert for "bob" and one for the first alert for "alice." Subsequent alerts for "bob" and "alice" are ignored.

You can further distinguish the users by adding details via the identifier variable. For example, you can distinguish by user and IP address using the following statement: *@RSAAlert(identifier={"username", "src_ip"}) SELECT* FROM Event(msg_id="login_fail")*. Then, you would see notifications generated by user name and IP address (one alert for "alice" at 1.2.3.4, another alert for "alice" at 6.7.8.9, and an alert for "bob" at 1.2.3.4).

To use identifiers with Alert Notification Suppression:

The @RSAAlert annotation is designed to work with the alert notification suppression feature in the Rule Builder user interface. To do this:

1. Create a rule in the Rule Builder user interface, and select the alert suppression feature when configuring notifications.



2. Copy the code from the Rule Builder rule into a new advanced rule.
3. Configure the advanced rule to include identifiers (as described above) and save the advanced rule.
4. Delete the original rule builder rule.

@RSAPersist Annotation

The @RSAPersist annotation is used to mark a named window as an ESA managed window for persistence. By marking the named window as an ESA managed window, ESA periodically writes the contents of the window to disk and restores them back if the window is undeployed and redeployed. The systems take a snapshot just before the ESA rule deployment is undeployed and the window is removed. Conversely, it restores the window contents from the snapshot just after the deployment is redeployed. This ensures that the contents of the window are not lost if the deployment state is altered or if the ESA service goes down.

For example, consider a named window, `DHCPTracker` that holds a mapping from IP addresses to each assigned hostname.You can annotate the statement with the @RSAPersist annotation as:

```
@RSAPersist
 create window DHCPTracker.std:unique(ip_src) as (ip_src string, alias_
host string);
 insert into DHCPTracker select IP as ip_src, HostName as alias_host from
DHCPAssignment(ID=32);
```

Note: All windows definitions are not suitable for persistence. `@RSAPersist` annotation must be used with care. If the window has timed-records or if it depends on time based constraints it is very likely that the reverted snapshots will not restore it to the correct state. Also, any changes to the window definition will invalidate the snapshots and reset the window to a blank state. The system does not do any semantic analysis to determine if the changes to the window definition are conflicting or not. Note that other parts of a deployment (that is, other than the particular CREATE WINDOW call that defines the window) may change, without invalidating the snapshots.

Caution: (This caution applies only to NetWitness Platform versions 11.3.x, 11.4.x, and 11.5.0.0.) To avoid data being overwritten, if you have a rule with a named window, do not disable and re-enable it. Instead, undeploy and redeploy the ESA rule deployment that contains the rule.

IMPORTANT: The @RSAPersist annotation suggests avoiding the use of 'Event' as a data type when keeping all windows, as 'Event' is already declared as the schema. Please refer to examples of valid and invalid syntax scenarios.

Invalid Examples:

1. `@RSAAlert`
2. `@RSAPersist`
3. `create window datas#keepall(evt Event);`
4.
5. `Insert into datas SELECT * FROM Event;`

1. `@RSAAlert`
2. `@RSAPersist`
3. `CREATE WINDOW demo_keepall.win:keepall (evt1 Event);`

4. `INSERT INTO demo_keepall`
5. `SELECT * FROM Event`
6. `WHERE user_dst = ANY('User35','User31');`
7. `SELECT * FROM demo_keepall;`

Valid Examples:

1. `@RSAAlert`
2. `@RSAPersist`
3. `create window datatest#keepall(evt String);`
4. `Insert into datatest SELECT * FROM Event;`

1. `@RSAAlert`
2. `@RSAPersist`
3. `CREATE WINDOW demo_keepall.win:keepall (evt1 java.util.Collection);`
4. `INSERT INTO demo_keepall`
5. `SELECT * FROM Event`
6. `WHERE user_dst = ANY('User35','User31');`
7. `SELECT * FROM demo_keepall;`

Note: When using rules with named windows and enrichment windows, it is essential to ensure no differing or missing column types in the window definition. It is also advisable to avoid using keywords like 'Event' and 'Schema' as column names in the window definition. Additionally, if a rule attempts to define the same window twice, it is expected that runtime exceptions will occur.

@UsesEnrichment (10.6.1.1 and later)

The @UsesEnrichment can be used in advanced EPL rules to reference enrichments. In order to synchronize enrichments with ESA, all enrichment dependencies in EPL rules must be referenced with the @UsesEnrichment annotation.

The @UsesEnrichment annotation uses the following format:

```
@UsesEnrichment(name= '<enrichment_name>')
```

For example, the following EPL references a whitelist enrichment:

```
@UsesEnrichment(name = 'Whitelist')
@RSAAlert
SELECT * FROM Event(ip_src NOT IN (SELECT ip_address FROM Whitelist))
```

@Name

The @Name is the statement name defined in ESA advanced rules. It is used to dynamically generate statement names in ESA alerts. The statement name of only an alert triggering statement is displayed. This annotation has meta keys enclosed in curly brackets.

Note: Dynamic-name-enabled option allows you to disable dynamic alert name generation.

The @Name annotation uses the following format:

```
@Name("<static_part_of_statement_name> {meta_key1} {meta_key2}…")
```

For example, the following EPL references meta keys *ip_src* and *user_name* whose values will be dynamically generated.

@Name(“Login Event to {ip_src} by {user_name}”)

Note: You can specify any number of meta keys in the statement for dynamic statement name generation.
The length of individual meta key is limited to 64, after which the value is truncated and appended with “…”.
The length of the dynamic generation of statement name is limited to 128, after which the value is truncated to 128 and appended with “…”. All the remaining values post truncation will be treated as static values.

@Audit Annotation

Add the `@Audit('stream')` annotation to your ESA rules to print alerts to the ESA logs for troubleshooting. This is useful when debugging the Esper rules. The @Audit('stream') annotation provides debug information for the next statement in the rule. For example:

```
@Audit('stream')
@RSAAlert
SELECT * FROM Event((ip_src NOT IN ( '1.1.1.1' , '2.2.2.2' )));
```

In the above example, `@RSAAlert` is only necessary if the statement needs to send an alert to the Respond view.

In NetWitness Platform 11.5 and later, you can test rules in the rule builders. If you add the `@Audit ('stream')` annotation to an advanced EPL rule, you can view the Debug Log in the test output. For more information, see Validate an Advanced EPL Rule .

Example Advanced EPL Rules

Following are the examples of Advanced ESA rules. Each example has multiple ways of implementing the same use-case.

For best practices on writing advanced EPL rules, see ESA Rule Writing Best Practices.

Example #1:

Create a user account and delete the same user account in 300s. User information is stored in user_src meta.

EPL #1:

Rule Name	CreateAndDelete Useraccount1
Rule Description	Create a user account followed by an action to delete the same user account in 300 seconds.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * FROM Event(ec_subject='User'`
	`  AND ec_outcome='Success'`
	`  AND user_src is NOT NULL`
	`  AND ec_activity IN ('Create', 'Delete')`
	`  ).win:time(300 seconds)`
	`match_recognize (partition by user_src`
	`   measures C as c, D as d`
	`   pattern (C D)`
	`  define`
	`   C as C.ec_activity='Create' ,`
	`   D as D.ec_activity='Delete');`

Note	• Filter events needed for pattern in given time frame. Filter conditions should be such that only required events are passed to match recognize function. In this case, they are create and delete user account Events. That is, Event(ec_subject='User' AND ec_outcome='Success' AND user_src is NOT NULL AND ec_activity IN ('Create', 'Delete') • Partition by creates buckets. In this case, Esper creates buckets per value of user_src. And hence value of user_src is common between both events. • Define the pattern you want. Right now it is set to Create Followed by Delete. You can do multiple creates followed by delete (C+ D). Pattern is very similar to regular expression. • Most efficient use case. • The ‘loose’ pattern match of (C+ D) will result in decreased performance. Unless you need to include all C events within the generated alert, keep the strict pattern match of (C D). See the Esper documentation for more details.

EPL #2:

Rule Name	CreateAndDeleteUseraccount2
Rule Description	Create a user account followed by an action to delete the same user account in 300 seconds.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * from pattern[every (a= Event(ec_subject='User' AND ec_outcome='Success' AND user_dst is NOT NULL AND ec_activity IN ('Create'))`
	`->`
	`( Event(ec_subject='User' AND ec_outcome='Success' AND user_dst is NOT NULL AND ec_activity IN ('Create') AND user_src = a.user_src)`
	`) )where timer:within(300 Sec) ];`

Note	• Lets say same user is created twice and deleted once in that order. Then the above pattern will fire 2 alerts. • A thread is created for every User creation. • There is no way to control threads. It is important to have time bounds and preferably small intervals. • If you do not need every first event to start a new thread and match with the subsequent second event, then add suppression syntax of @SuppressOverlappingMatches after the pattern keyword. See the Esper documentation for more details.

Example #2:

Detect pattern where user created followed by login by same user and user is deleted in end. In case of windows logs user info is stored in either user_dst or user_src depending on event.

user_src(create) = user_dst(Login) = user_src(Delete)

EPL #3:

Rule Name	CreateUserLoginandDeleteUser
Rule Description	Detect a pattern where a user creates a User account followed by login by the same user followed by deletion of the User account.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * FROM Event(ec_subject='User'`
	`and ec_activity in ('Create','Logon','Delete')`
	`and ec_theme in ('UserGroup', 'Authentication')`
	`and ec_outcome='Success'`
	`).win:time(300 seconds)`
	`match_recognize (measures C as c, L as l, D as d`
	`pattern (C L D)`
	`define`
	`C as C.ec_activity = 'Create',`

	L as L.ec_activity = 'Logon' AND L.user_dst = C.user_src, D as D.ec_activity = 'Delete' AND D.user_src = C.user_src);
Note	• Since user_src/user_dst is not common across all events we can't use partition. It will be 1 single bucket running 1 pattern at a time. For example, for user 1 and 2 if the stream of events are C1C2L1D1, C1L1C2D1, there will be no alert because C1 thread got reset by C2. Alert will be fired only if C1L1D1 are in order and no other event either from same user or other user falls in between. • Another solution would be to use Named Window and merge user_dst and user_src into single column and then run match recognize. (EPL #3). • Pattern can also be used. You might get more alerts than expected. (EPL #4).

EPL #4: Using NamedWindows and match recognize

Rule Name	CreateUserLoginandDeleteUser
Rule Description	Detect a pattern where a user creates a User account followed by login by the same user followed by deletion of the User account.
Rule Code	@Name('NormalizedWindow') create window FilteredEvents.win:time (300 sec) (user String, ecactivity string, sessionid Long); @Name('UsersrcEvents') Insert into FilteredEvents select user_src as user, ec_activity as ecactivity, sessionid from Event(ec_subject='User' and ec_activity in ('Create','Delete') and ec_theme in ('UserGroup', 'Authentication') and ec_outcome='Success' and user_src is not null); @Name('UsrdstEvents') Insert into FilteredEvents select user_dst as user, ec_activity as ecactivity, sessionid from Event(ec_subject='User' and ec_activity in (Logon') and ec_theme in ('UserGroup', 'Authentication') and ec_outcome='Success' and user_dst is not null); @Name('Pattern') @RSAAlert(oneInSeconds=0, identifiers={"user"}) select * from FilteredEvents match_recognize (partition by user measures C as c, L as l, D as d pattern (C L+D) define C as C.ecactivity= 'Create', L as L.ecactivity= 'Logon', D as D.ecactivity='Delete');

EPL #5: Using Every @RSAAlert(identifiers={"user_src"})

Rule Name	CreateUserLoginandDeleteUser

Rule Description	Detect a pattern where a user creates a User account followed by login by the same user followed by deletion of the User account.
Rule Code	`SELECT a.time as time,a.ip_src as ip_src,a.user_dst as user_dst,a.ip_dst as ip_dst,a.alias_host as alias_host from pattern [every (a=Event (ec_subject='User' and ec_activity='Create' and ec_theme='UserGroup' and ec_outcome='Success') -> (Event(ec_subject='User' and ec_activity='Logon' and ec_theme='Authentication' and user_src=a.user_dst) -> b=Event(ec_subject='User' and ec_activity='Delete' and ec_theme='UserGroup' and user_dst=a.user_dst))) where timer:within(300 sec)];`

Example #3:

Excessive login failures from same sourceIP.

EPL #6: @RSAAlert(identifiers={"ip_src"})

Rule Name	ExcessLoginFailure
Rule Description	The same user tried logging in from the same Source IP and faced login failures.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * FROM Event ( ip_src IS NOT NULL AND ec_activity ='Logon'`
	`AND ec_outcome = 'Failure').win:time_batch(300 seconds) GROUP BY`
	`ip_src HAVING COUNT(*) = 10;`

Note	• Uses time _batch: Looks at events in batches(tumbling window). Every event matching the filter criteria will be kept for the specified time window. • "GROUP BY" clause aggregates events within the data window by ip_src and HAVING clause instructs a count of 10 events with the same ip_src must occur within the time window. • One of the issues with tumbling windows is that events occurring towards the end of the batch might not lead to an alert. In the below sequence of events at t=301 even though 10 login failures occurred for the same login in the last 300 secs, there will be no alert because the batch of events was dropped at t=300.

Time t	Login Failures for Specific Users	Alert	Time Batch
0	0	0	1
295	6	0	1
299	3	0	1
301	1	0	2
420	6	0	2
550	3	0	2
600	0	0	3

	720 6 0 3 850 3 0 3 900 1 1 3 ends and 4 begins • Above problem can be resolved using win:time windows (EPL#7)instead of win:time_length_batch windows. • Outer group by is to control events when time elapses. Say you have 9 events at end of 60 secs, Esper engine will push those 9 events to listener. Group by and count will restrict it since count is not equal to 10. • Time and count can be modified as needed.

EPL #7: @RSAAlert(identifiers={"ip_src"})

Rule Name	ExcessLoginFailure
Rule Description	The same user tried logging in from the same Source IP and faced login failures.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * FROM Event ( ip_src IS NOT NULL AND ec_activity ='Logon'`
	`AND ec_outcome = 'Failure').win:time(300 seconds) GROUP BY ip_src HAVING COUNT(*) = 10;`

Note	• This is a sliding window and hence after an alert is fired for a set of events they can be used for another alert as well until time has passed. • If 10 events were involved in causing the alert only the last event will appear. • Events are not removed from the time window. You could use output rate limiting. See the Esper documentation for more details.

Example #4:

Multiple failed logins from multiple different users from same source to same destination, a single user from multiple different sources to same destination.

EPL #8: using time_batch

Rule Name	MultiplefailedLogins
Rule Description	There are multiple failed logins for the following cases: - From multiple users from same source to same destination. - Single user from multiple sources to the same destination.

Rule Code	`@RSAAlert(oneInSeconds=0)`
	`SELECT * FROM Event (`
	`ec_activity='Logon'`
	`AND ec_outcome='Failure'`

	AND ip_src IS NOT NULL AND ip_dst IS NOT NULL AND user_dst IS NOT NULL) .win:time_batch(300 seconds) group by ip_src,ip_dst having count(distinct user_dst) >= 5;
Note	• ip.dst and ip.src are common across all events. • user_dst is unique for all events. • Alert is fired when there are at least 5 different users try to login from same ip.src and ip.dst combination.

Example #5:

No Log traffic from a device in a given timeframe.

EPL #9: using timer:interval

Rule Name	NoLogTraffic
Rule Description	There is no log traffic observed from a device in a given time frame.
Rule Code	SELECT * FROM pattern [every a = Event(device_ip IN ('10.0.0.0','10.0.0.1') AND medium = 32) -> (timer:interval (3600 seconds) AND NOT Event(device_ip = a.device_ip AND device_type = a.device_type AND medium = 32))];
Note	• Rule only detects sudden loss of traffic. It won't alert if there is no traffic to begin with. You need at least 1 event for rule to alert. • List of device ip address or device hostnames as input. Only these systems will be tracked. • Time input is required. Alert is fired when time interval between events exceeds input time.

Example #6:

Multiple Failed Logins NOT followed by a Lockout event by the same user.

EPL #10: using timer and Lockout

Rule Name	FailedloginswoLockout
Rule	There are multiple failed logins that are not followed by Lockout event by the same

Description	user.

Rule Code	`SELECT * FROM pattern [every-distinct(a.user_dst, a.device_ip, 1 msec) (a= Event(ec_activity='Logon' and ec_outcome='Failure' and user_dst IS NOT NULL) -> [2]( Event ( device_ip =a.device_ip and ec_activity='Logon' and ec_outcome='Failure' and user_dst=a.user_dst)`
	`AND NOT Event( ( ec_activity='Logon' and ec_outcome='Success' and device_ip = a.device_ip and user_dst=a.user_dst) or (ec_activity='Lockout' and device_ip = a.device_ip and user_dst=a.user_dst))))`
	`where timer:within(60 seconds) -> (timer:interval(30 seconds) and not Event(device_ip=a.device_ip and user_dst=a.user_dst and ec_activity='Lockout')) ];`

Note	• Above query detects the absence of a Lockout Event after the occurrence of 2 failed logins from same user. • The occurrence of the multiple failed logins are timed and are assumed to occur within a certain period of time. Also, in-practice the Lockout event is assumed to occur within a short time after the occurrence of the last failed login event because the threshold value of Failed logins per user is set in a given domain. • In current query, every distinct will suppress new thread for combination of user and device for 1 millisec. • Time allowed for 3 failed logins is 60 secs since 1st failed attempt. Wait period for lockout event to occur is 30 secs

Example #7:

Custom functions to perform LIKE and REGEX operations for ARRAY elements.

EPL #11: @RSAAlert(oneInSeconds=0)

Rule Name	MatchLikeRegex
Rule Description	There are custom functions to perform LIKE and REGEX comparisons of array meta keys.

Rule Code	`SELECT * FROM pattern[`
	`e1=Event(matchLike(alias_host, "10.0.0.%")) AND`
	`e2=Event(matchRegex(alias_host, "10\.0\.0\.1[0-9][0-9]"))`
	`where timer:within(5 Minutes)];`

Note:
1. "**.**" in meta keys should be replaced with ("_").
2. All patterns should be time bound.
3. Use appropriate tags in front of statements, for example:

```
@RSAPersist:
@RSAAlert:
```

For additional details you can refer to:

- EPL Documentation: http://www.espertech.com/esper/esper-documentation/
- EPL Online Tool: http://esper-epl-tryout.appspot.com/epltryout/mainform.html

Configure an In-Memory Table Using an EPL Query

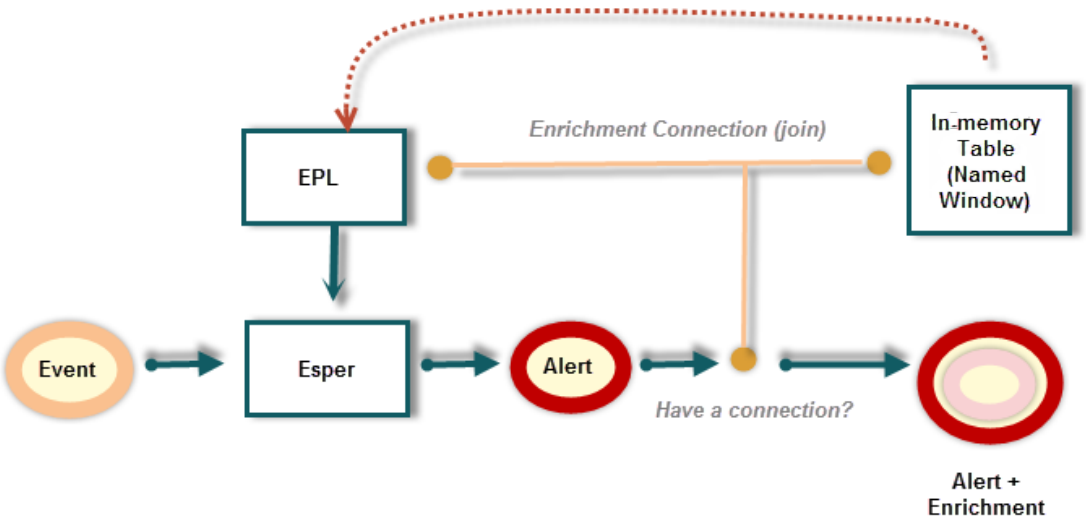
Note: It is preferable to use Context Hub List enrichment sources instead of In-Memory Table enrichment sources for rules. Recurring In-Memory Tables are no longer supported; use Content Hub Lists as enrichment sources. For more information, see Configure a Context Hub List as an Enrichment Source.

When you use an In-Memory Table configuration in expert mode, you can create an enrichment source or named window based on an Esper query. This allows you to have more control over the content and create more dynamic content. When you do this, an EPL query constructs the named window to capture interesting states from the event stream.

Workflow

The following shows the workflow for creating a query using a named window:

1. The event is sent to the Esper Engine.
2. An EPL query is generated.
3. An alert is triggered.
4. The query checks to see if there is a connection between the event and the Named Window.
5. If there is a connection, the query that populates the Named Window is run and populated.
6. The content from the Named Window is added to the alert content and sent or displayed (depending on your settings).



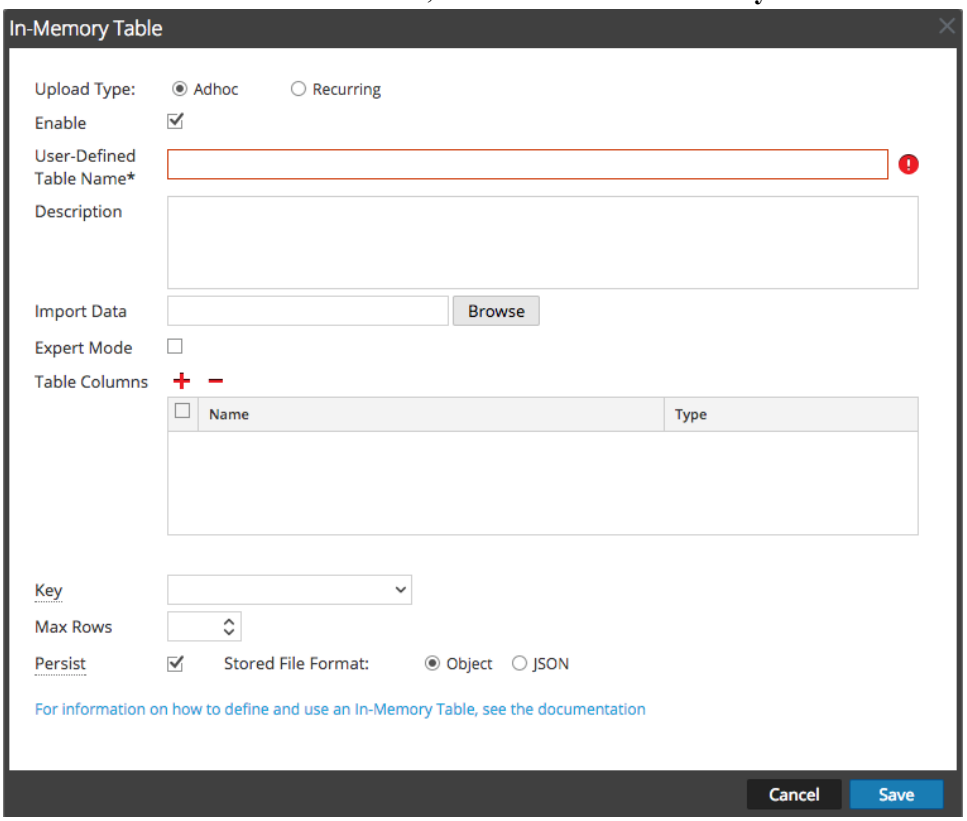
Prerequisites

- The meta used in the EPL statement must exist in the data.
- You must create well-formed EPL statements.

Procedure

Note: It is preferable to use Context Hub List enrichment sources instead of In-Memory Table enrichment sources for rules.

1. Go to **(Configure) > ESA Rules**.
   The Configure view is displayed with the Rules tab open.
2. Click the **Settings** tab.
3. In the options panel, select **Enrichment Sources**.
4. In the **Enrichment Sources** section, click > **In-Memory Table**.



5. Select **Adhoc**.
   By default, Enable is selected. When you add the in-memory table to a rule, alerts will be enriched with data from it.
6. In the **User-Defined Table Name** field, type a descriptive name to describe the in-memory table.
7. If you want to explain what the enrichment adds to an alert, enter information in the **Description** field.
   This description displays when you view the list of enrichments from the Enrichment Sources view, so it's a good idea to enter a thorough description as a best practice. Doing this allows other users to understand the content of the enrichment without opening it to examine its contents.
8. Select **Expert Mode** to define an advanced in-memory table configuration by writing an EPL query. The Table Columns are replaced by a **Query** field.
9. Select **Persist** to preserve the in-memory table on disk when the ESA service stops and to re-populate the table when the service restarts.
10. Enter the EPL query in the **Query** field. The query should be well-formed, and it's a good idea to test it before entering it in the field.
11. Click **Save**.

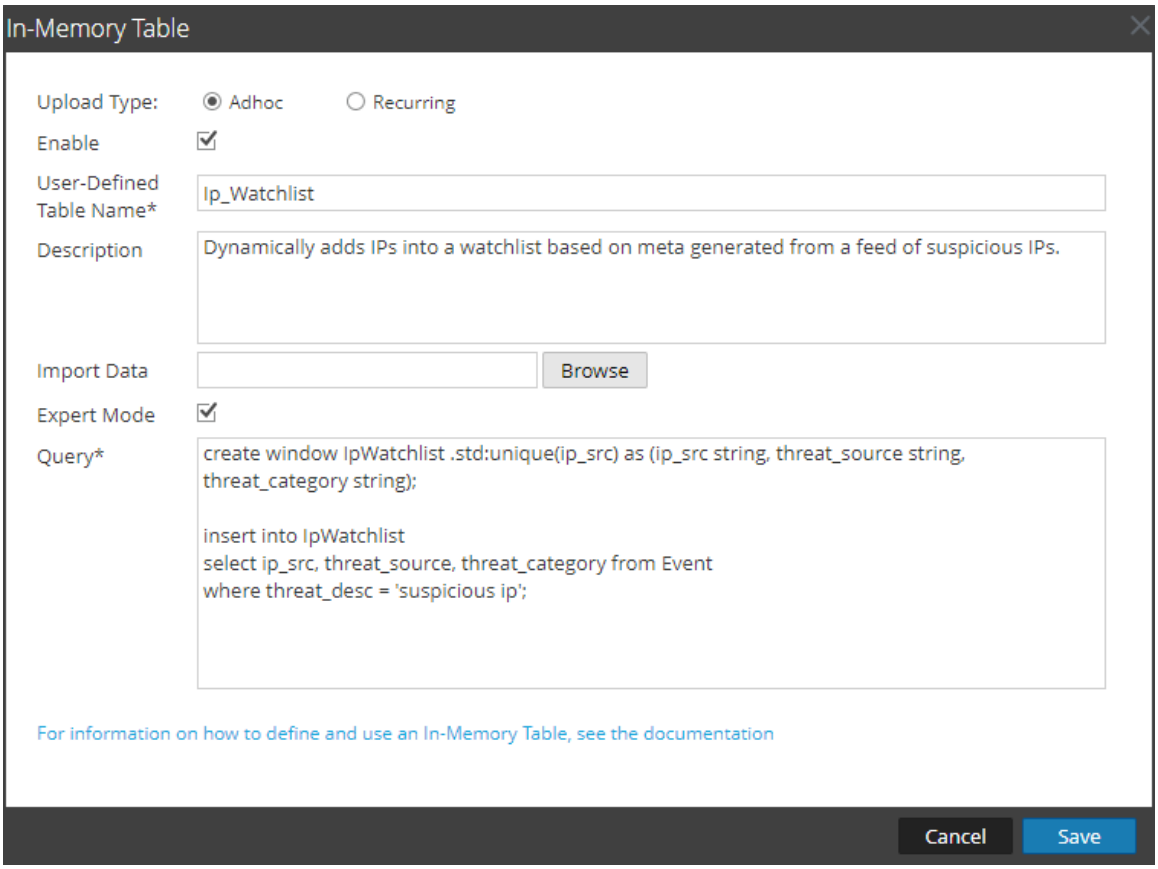
Example

For example, you want to know when an IPS or IDS is giving five or more inbound events with an event identified with malicious code. Additionally, you would like to know when the source IP of those events has been identified as suspicious by other sources. This information helps to more quickly triage the event and determine whether the alert is a true positive.

Step 1: Create the Enrichment

In this example, this enrichment is a watchlist of IPs that have been identified as suspicious by third party sources or by internal staff. The meta of `threat_desc equal to 'suspicious ip'` is generated when a match to a feed occurs. This meta can be matched and output based on a log, packet, or endpoint event.

The enrichment should look like the following:



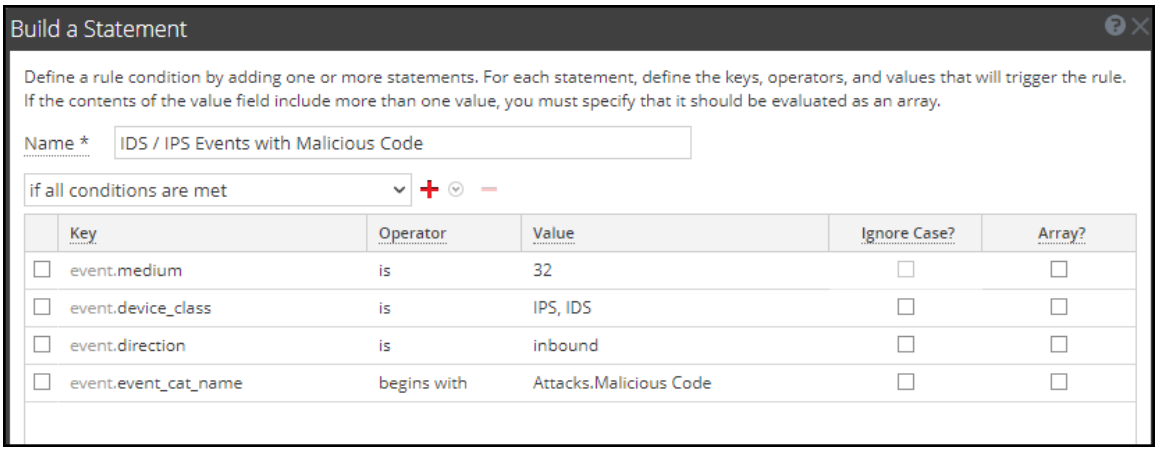
Parameters	Description
Upload Type	Adhoc
IP_Watchlist	IP_Watchlist
Description	Dynamically populated whitelist based on a feed of IPs that are considered suspicious.
Expert Mode	Selected

Query	`create window IpWatchlist .std:unique(ip_src) as (ip_src string, threat_source string, threat_category string);`
	`insert into IpWatchlist`
	`select ip_src, threat_source, threat_category from Event`
	`where threat_desc = 'suspicious ip';`

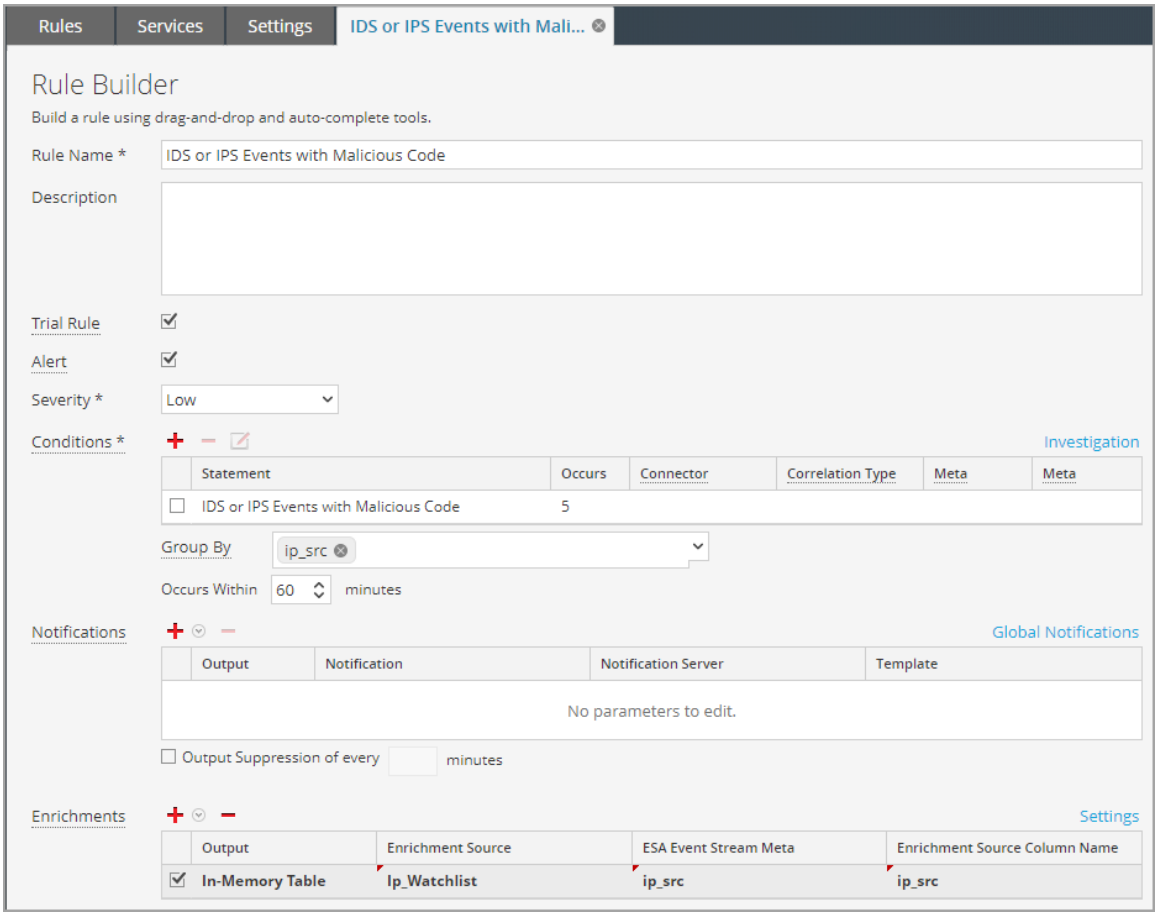
Step 2: Create Your Rule

First, you need to create your ESA Correlation rule. This example rule looks for inbound IPS or IDS log events with the `event_cat_name` beginning with `Attacks.Malicious Code`. If five or more events for the same `ip_src` occur within 60 minutes, then an alert will be triggered. If an `ip_src` from the Enrichment equals the `ip_src` from the alert, then that alert will be enriched with additional meta. In this case, the analyst would see the values for `threat_source` and `threat_category` in the raw alert. `Threat_category` would indicate the type of malware and `threat_source` would indicate the entity that has reported the ip as suspicious. The analyst could use this information to do additional research or escalate to the next tier for creation of a possible incident.

Rule Statement



Rule Logic with Enrichment Added



ESA Alert References

In Event Stream Analysis (ESA), you configure and deploy ESA rules to get alerted about potential network threats.

These topics explain the user interface for ESA Correlation rules.

Rules Tab

The Rules tab enables you to view and configure ESA rules.

What do you want to do?

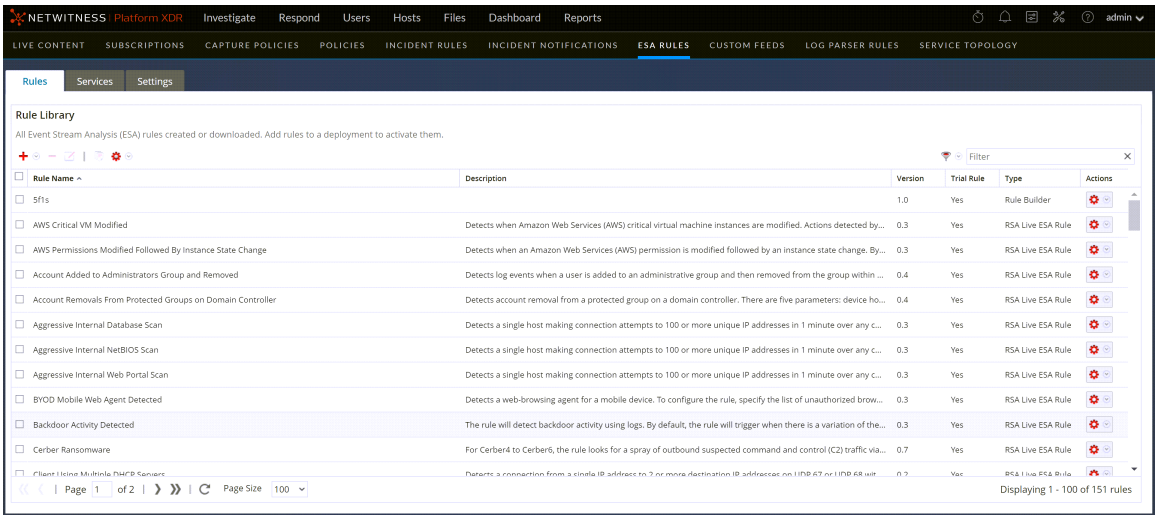
Role	I want to ...	Show me how
Content Expert	View types of rules	ESA Rule Types
Content Expert	Deploy Trial Rules	Working with Trial Rules
Content Expert	Create a rule	Add Rules to the Rule Library

Related Topics

Getting Started with ESA

Quick Look

The Rules tab is displayed when you go to **(Configure) > ESA Rules**.
The following figure shows the Rules tab.



The Rules tab is divided into two sections:

- Rules Tab Options Panel
- Rule Library Panel

Rule Library Panel

The Rule Library panel allows you to manage rules.

What do you want to do?

Role	I want to ...	Show me how
Content Expert	Add an ESA rule.	Add a Rule Builder Rule
Content Expert	Edit, duplicate, or delete an ESA rule.	Edit, Duplicate or Delete a Rule
Content Expert	Import or export ESA rules.	Import or Export Rules
Content Expert	Filter the ESA rules list.	Filter or Search for Rules

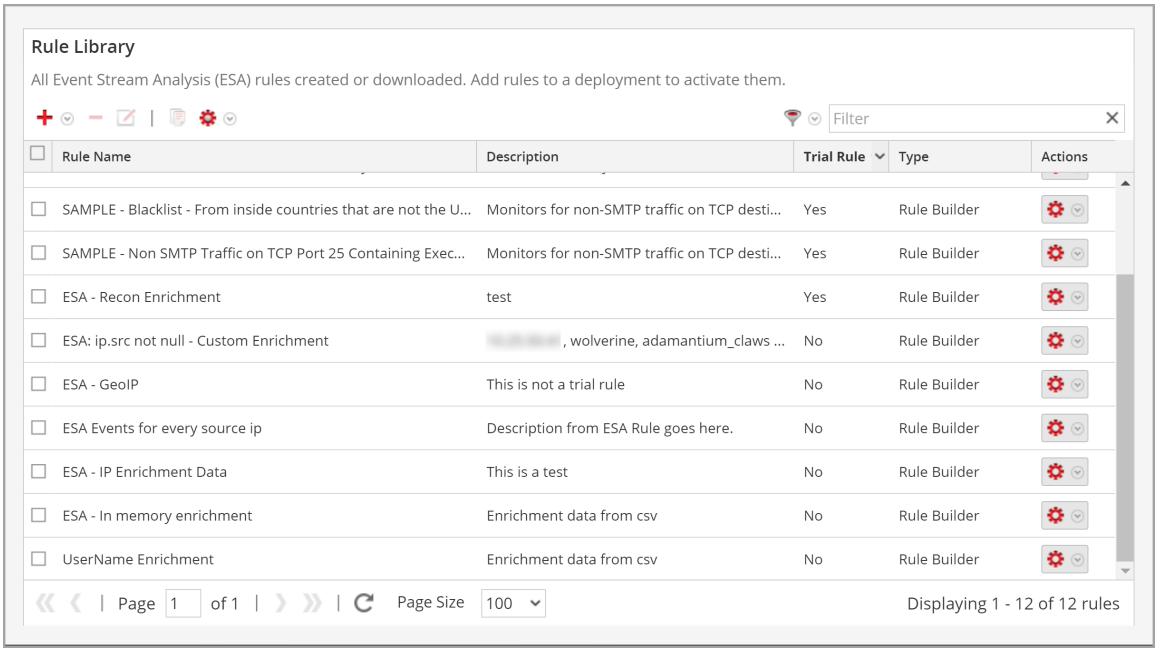
Related Topics

- Add an Advanced EPL Rule

Quick Look

To access this view, go to **(Configure) > ESA Rules**. The Rules tab is displayed and the Rule Library panel is on the right.

The following figure shows the Rule Library panel.



The Rule Library panel includes the following components:

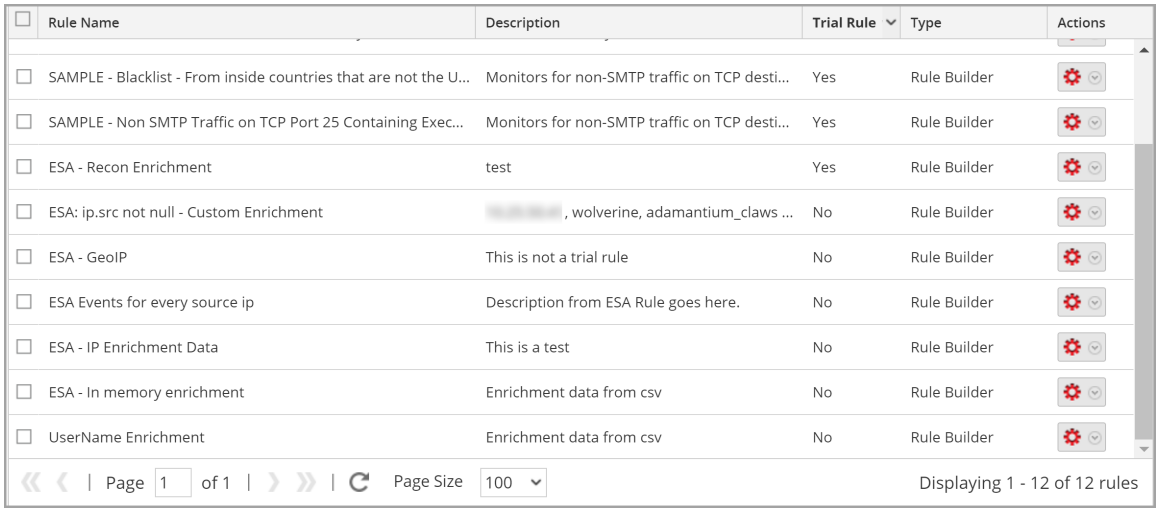
- Rule Library toolbar
- Rule Library list

Rule Library Toolbar

The Rule Library toolbar allows you to add, delete, edit, duplicate, filter, export, and import ESA rules. The following figure shows the icons for these actions.

Rule Library List

The following figure shows the Rule Library list.

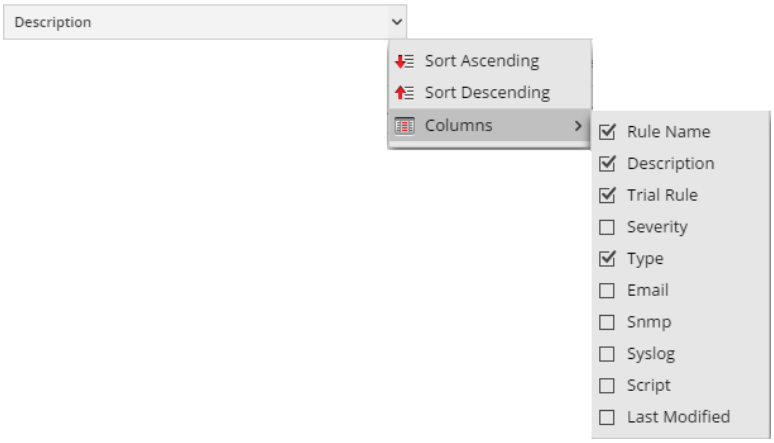


The Rule Library list shows all of the ESA rules. The following table lists the columns in the Rule Library list and their description.

Column	Description
Rule Name	Purpose of the ESA rule.
Description	Summary of what the ESA rule detects.
Trial Rule	Deployment mode to see if the rule runs efficiently.
Type	The type of rule. For more information, see ESA Rule Types.
Actions ( )	Menu to delete, edit, duplicate, or export the selected rule.
Severity	Threat level of alert triggered by the rule.
Email	Indicates whether an alert notification for the rule is sent by email. This column is not visible by default.

Column	Description
SNMP	Indicates whether an alert notification for the rule is sent using SNMP. This column is not visible by default. (ESA SNMP notifications are not supported in NetWitness version 11.3 and later.)
Syslog	Indicates whether an alert notification for the rule is sent using Syslog. This column is not visible by default.
Script	Indicates whether an alert notification for the rule executes a script. This column is not visible by default.
Last Modified	The date and time when the ESA rule was last modified. This column is not visible by default.

To display columns which aren't visible by default, hover over the title of a column and click the **v** on the right. This opens a drop-down menu in which you can sort the contents of the column or choose which columns you want to see in the Rule Library list.



Rule Builder Tab

The Rule Builder tab enables you to define a Rule Builder rule.

What do you want to do?

Role	I want to ...	Show me how
Content Expert	Define a Rule Builder rule.	Add a Rule Builder RuleStep 1. Name and Describe the Rule
Content Expert	Define rule criteria.	Step 2. Build a Rule Statement
Content Expert	Add conditions to the rule.	Step 3. Add Conditions to a Rule Statement
Content Expert	Test the ESA rule logic.	Validate an ESA Rule

Related Topics

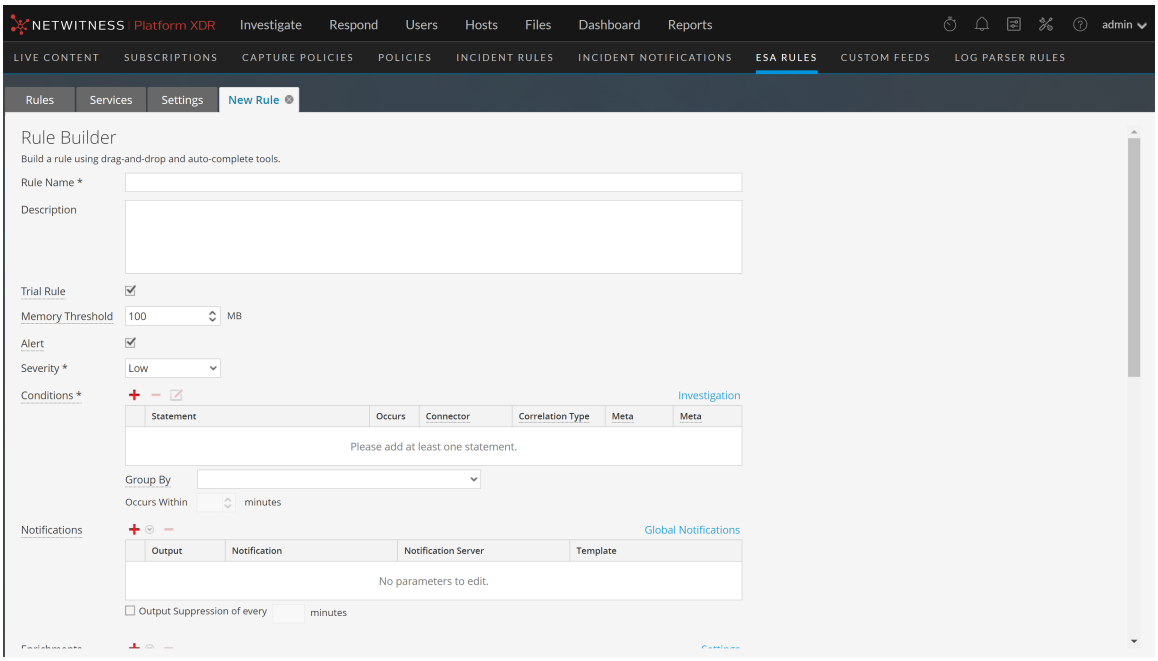
- Add an Advanced EPL Rule

Quick Look

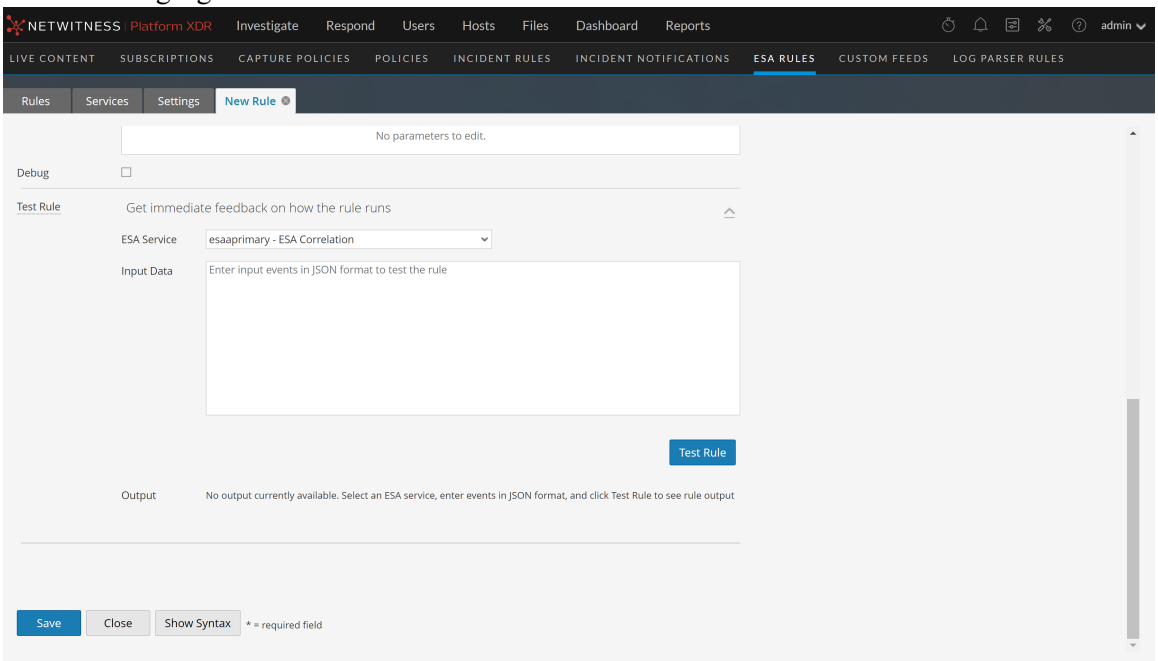
To access the Rule Builder tab:

1. Go to  **(Configure) > ESA Rules**.
   The Rules tab opens by default.
2. In the **Rule Library** toolbar, select  **> Rule Builder**.
   The Rule Builder tab is displayed.

The following figure shows the Rule Builder tab.



The following figure shows the Rule Builder tab scrolled down with the Test Rule section in view.



The following table lists the parameters in the Rule Builder tab.

Field	Description
Rule Name	Purpose of the ESA rule.
Description	Summary of what the ESA rule detects.
Trial Rule	Deployment mode to see if the rule runs efficiently.

Field	Description
Memory Threshold	(This option applies to version 11.5 and later.) The maximum memory usage allowed for this rule in MB. Add Memory Thresholds to ESA rules that use memory. For example, if a rule contains windows or pattern matching, configure a memory threshold for that rule. If the configured memory threshold is exceeded, it gets disabled individually and an error is displayed for that rule on the (Configure) > ESA Rules > Services tab. New rules default to a 100 MB memory threshold. Rules that existed before version 11.5 do not have a default value and a memory threshold is not set.
Alert	(This option applies to version 11.3 and later.) When selected, the alert is sent to Respond. If the checkbox is cleared, an alert will not be sent to Respond. To turn alerts on or off for ALL rules, see the *ESA Configuration Guide*.
Severity	Threat level of alert triggered by the rule.

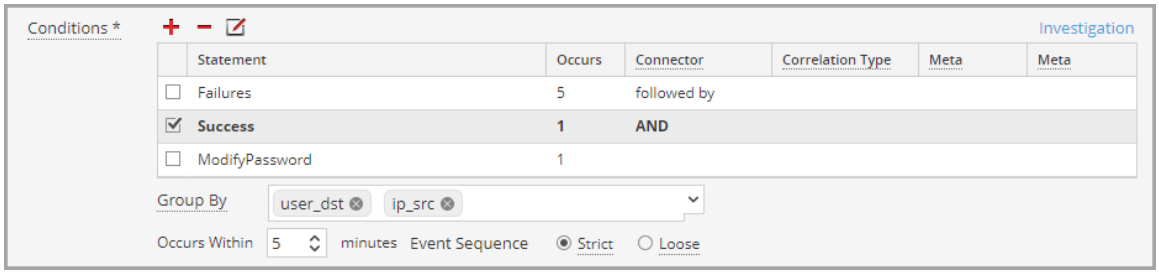
The Rule Builder includes the following components:

- Conditions Section
- Notifications Section
- Enrichments Section
- Debug Option
- Test Rule Section

Conditions Section

In the Conditions section of the Rule Builder tab, you define what the rule detects.

The following figure shows the Conditions section.



The following table lists the parameters of the Conditions section.

Parameter	Description
+	Add a statement.
−	Remove selected statement.
	Edit selected statement.

Parameter	Description
Statement	Logical group of conditions for one operation.
Occurs	Alert frequency if the condition is met. This specifies that there must be at least that many events that satisfy the criteria in order to trigger an alert. The time window in minutes binds the Occurs count.
Connector	Options to specify relationship among the statements: • followed by • not followed by • AND • OR The Connector joins two statements with AND, OR, followed by, or not followed by. When followed by is used, it specifies that there is a sequencing of those events. AND and OR build one large criteria. The followed by creates distinct criteria that occurs in sequence.
Correlation Type	**Correlation Type** applies only to **followed by** and **not followed by**. If you choose a correlation type of SAME, select one meta to correlate on, and if you choose a correlation type of JOIN, select two meta to correlate on. You may want to use JOIN if you are trying to correlate on meta from two different data sources. For example, say you want to correlate an AV alert with an IDS alert.
Meta	Enter the meta condition if choosing a correlation type of SAME or JOIN (as described above).
Meta	Enter the second meta condition if choosing a correlation type of JOIN (as described above). For example, The destination IP address from the AV alert and source IP address for the workstation from the IDS alert are joined to allow you to view the same entities across different sources.
occurs within minutes	Time window within which the conditions must occur.
Event Sequence	Choose whether the pattern must follow a *strict* match or a *loose* match. If you specify a strict match, this means that the pattern must occur in the *exact* sequence you specified with no additional events occurring in between. For example, if the sequence specifies five failed logins (F) followed by a successful login (S), this pattern will only match if the user executes the following sequence: F,F,F,F,F,S. If you specify a loose match, this means that other events may occur within the sequence, but the rule will still trigger if all of the specified events also occur. For example, five failed login attempts (F), followed by any number of intervening successful login attempts (S), followed by a successful login attempt might create the following pattern: F,S,F,S,F,S,F,S,F,S which would trigger the rule despite the intervening successful logins.

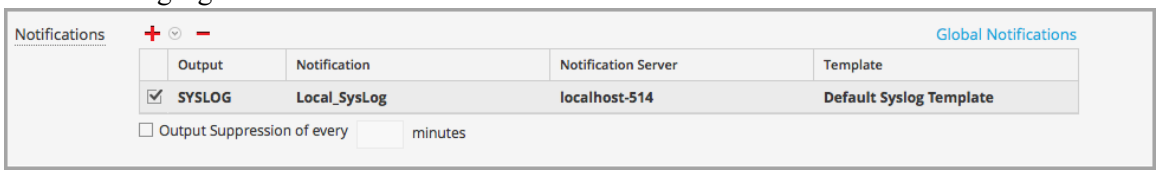
Parameter	Description
Group By	Select the meta key by which to group results from the dropdown list. For example, suppose that there are three users; Joe, Jane, and John and you use the Group By meta, **user_dst** (user_dst is the meta field for the user destination account). The result will show events grouped under the user destination accounts, Joe, Jane, and John. You can also group by multiple keys. For example, you might want to group by user and machine to see if a user logged in from the same machine attempts to log into an account multiple times. To do this, you might group by user_dst and ip_src.

Notifications Section

In the Notifications section, you can choose how to be notified when ESA generates an alert for the rule.

For more information on the alert notifications, see Add Notification Method to a Rule.

The following figure shows the Notifications section.



Parameter	Description
+	To add an alert notification type.
−	To delete the selected alert notification.
Output	Alert notification type. Options are: • Email • SNMP (This option is not supported in NetWitness version 11.3 and later.) • Syslog • Script
Notification	Name of previously configured output, such as an email distribution list.
Notification Server	Name of server that sends the output.
Template	Name of template for the alert notification.
Output Suppression of every	Option to specify alert frequency.
Minutes	Alert frequency in minutes.

Enrichments Section

In the Enrichments section, you can add a data enrichment source to a rule.

For more information on the enrichments, see Add an Enrichment to a Rule.

The following figure shows the Enrichments section.

Parameter	Description
+	To add an enrichment.
−	To delete the selected enrichment.
Output	Enrichment source type. Options are: • In-Memory Table (Ad hoc only - Recurring In-Memory Tables are no longer supported in version 11.3 and later.) • GeoIP
Enrichment Source	Name of previously configured enrichment source, such as a .CSV filename for an In-Memory Table.
ESA Event Stream Meta	ESA meta key whose value will be used as one operand of join condition.
Enrichment Source Column Name	Enrichment source column name whose value will be used as the other operand of the join condition. For an in-memory table, If you configured a key when creating a .CSV-based enrichment, this column automatically populates with the selected key. However, you can change it if you like. For a GeoIP enrichment source, ipv4 is automatically selected.

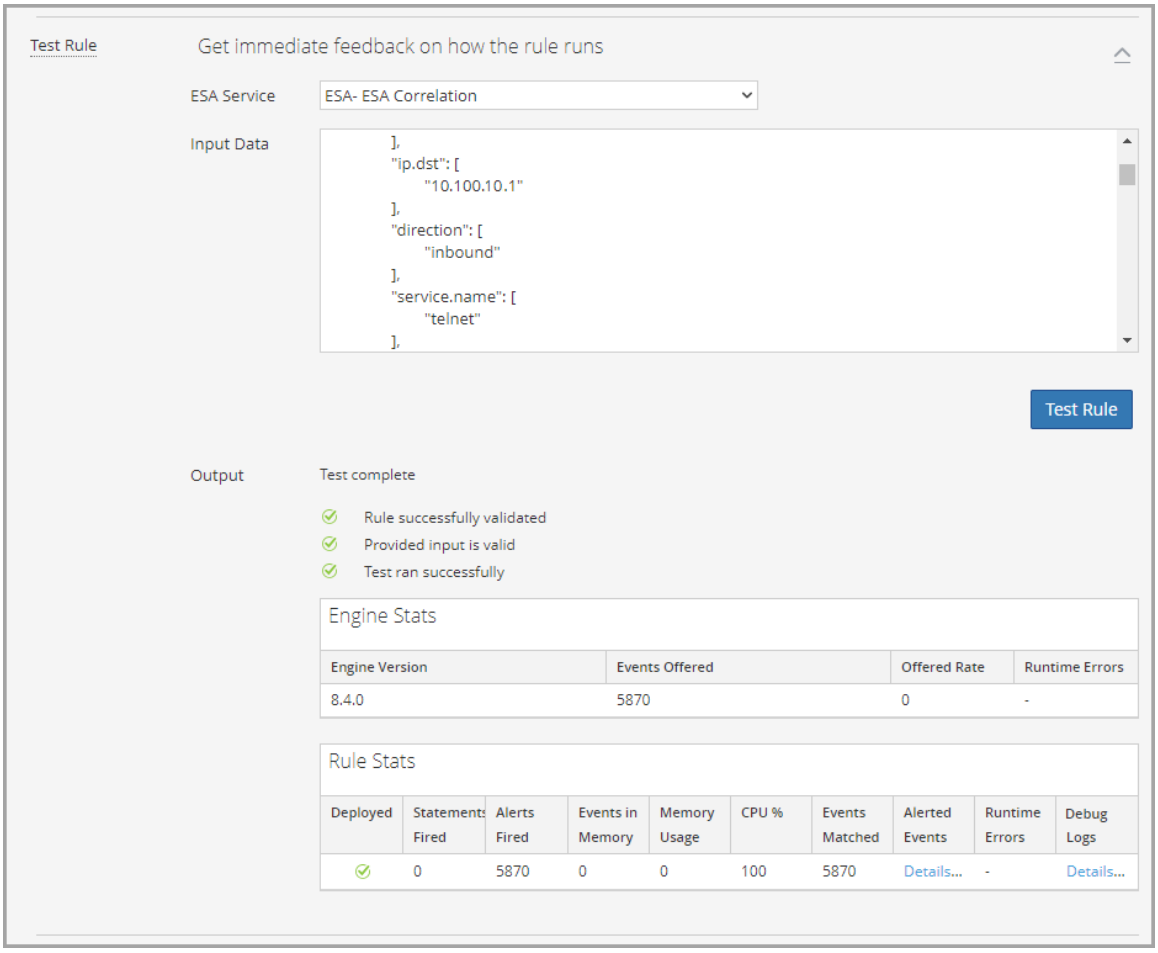
Debug Option

Select the Debug option to print alerts to the ESA logs for troubleshooting. This adds an @Audit ('stream') annotation to the rule. This is useful when debugging the Esper rules.

Test Rule Section

Note: The Test Rule section is available in NetWitness Platform 11.5 and later.

In the Test Rule section, you can validate your ESA rule to determine if the rule logic is working as expected before deploying the rule.



Field	Description
ESA Service	Select the ESA Correlation service to process the rule.
Input Data	Enter the input events to test the rule. Download the events from the Investigate view in JSON format, copy the events, and paste them in this field.
Output Data	After you select an ESA Correlation service, input data, and click the Test Rule button, you can view the output of the rule here and verify that the rule is working according to your requirements. You can view the alerts in the output, but this test does not send any alert notifications.

The following table describes the test rule output **Engine Stats**.

Field	Description
Engine Version	Esper version running on the ESA service
Events Offered	Number of events processed by the ESA service since the last service start
Offered Rate	The rate that the ESA service processes current events / The maximum rate that the ESA service processed events
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the ESA rule deployment.

The following table describes the test rule output **Rule Stats**.

Field	Description
Deployed	A green checkmark indicates that the rule is deployed on the selected ESA service.
Statements Fired	The number of statements that fired the alerts
Alerts Fired	The number of alerts generated from the test data
Events in Memory	The number of events placed in memory by the rule
Memory Usage	The total amount of memory used by the rule
CPU %	The percentage of the deployment CPU used by the rule. For example, a deployment with 1 rule shows 100% CPU usage for that rule and a deployment with two equally CPU heavy rules show 50% each.
Events Matched	The number of events that matched the rule
Alerted Events	If applicable, this field can contain a link to events that caused an alert.
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the rule.
Debug Logs	This field contains a link to Esper debug (audit) logs.

Syntax

Click **Show Syntax** to view the EPL syntax of conditions, statements, and debugging parameters. It also provides a warning when the syntax is invalid. For more information, see Rule Syntax Dialog.

Build a Statement Dialog

The Build a Statement dialog allows you to construct a condition statement when creating a new Rule Builder rule.

What do you want to do?

Role	I want to ...	Show me how
Content Expert	Configure a rule statement.	Step 2. Build a Rule Statement
Content Expert	Add conditions to the rule.	Step 3. Add Conditions to a Rule Statement

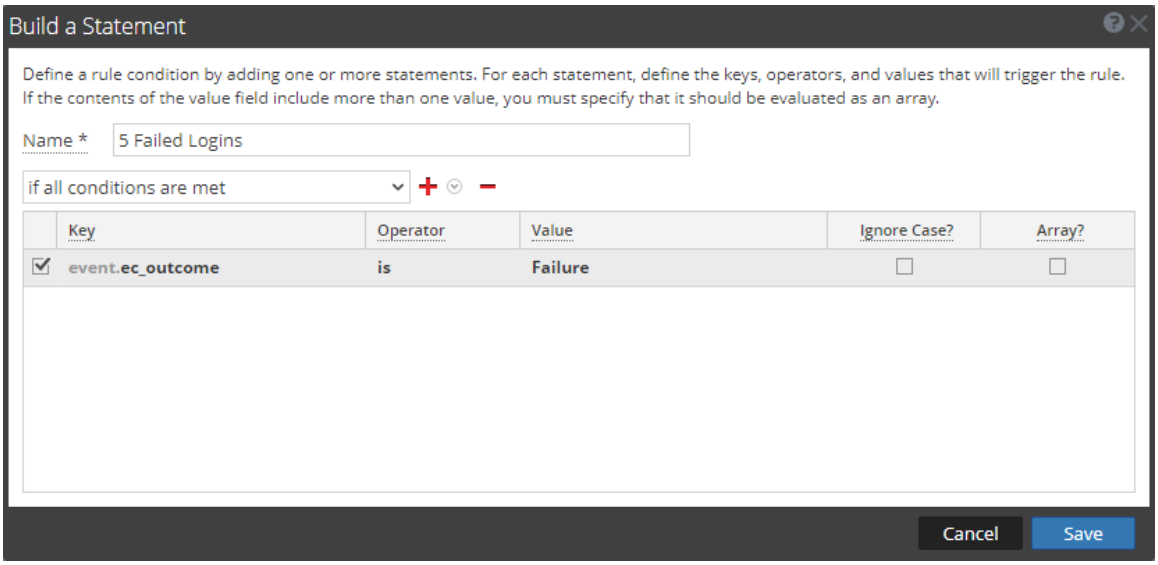
Related Topics

- Add a Rule Builder Rule

Quick Look

To access the Build a Statement dialog:

1. Go to  **(Configure) > ESA Rules**.
   The Configure ESA Rules view is displayed with the Rules tab open.
2. In the **Rule Library** toolbar, select  **> Rule Builder**.
   A New Rule tab is displayed..
3. In the **Conditions** section, click  .
   The Build a Statement dialog is displayed.



The following table describes the parameters in the Build a Statement dialog.

Parameter	Description
Name	Purpose of the statement.
Select	Conditions the rule requires. There are two options: • If all conditions are met • If any of these conditions are met
Key	Key for ESA to check in the rule statement.

Parameter	Description
Operator	Relationship between the meta key and value for the key: • is • is not • is not null • is greater than (>) • is greater than or equal to (>=) • is less than (<) • is less than or equal to (<=) • is one of (For array type meta) • is not one of (For array type meta) • contains • not contains • begins with • ends with
Value	Value for ESA to look for in the key.
Ignore Case?	This field is designed for use with string and array of string values. By choosing the **Ignore Case** field, the query will treat all string text as a lowercase value. This ensures that a rule that searches for the user named Johnson would trigger if the event contains "johnson," "JOHNSON," or "JoHnSoN."
Array?	Choice to indicate if contents of Value field represent one value or multiple values: • Select the box to indicate multiple values. • Clear the box to indicate one value.
	Add a statement. You can add a meta condition, whitelist condition, or blacklist condition.
	Delete selected statement.
Save	Add statement to the Conditions section of the Rule Builder tab.

The following table shows the operators you can use in the Rule Builder:

Operator	Required Value	Usage	Example	Meaning
is	Singular string value	The meta key is equal to the *value* field.	***user_dst*** is John Doe.	***user_dst*** is equal to the string *"John Doe"*.

Operator	Required Value	Usage	Example	Meaning
is	Array string value	The meta key is equal to one of the elements of the *value* field.	***user_dst*** is John, Doe, Smith.	***user_dst*** is equal either to the string *"John"* or to the string *"Doe"* or to the string *"Smith"* (Note, the spaces are stripped.).
is not	Singular string value	The meta key is not equal to the *value* field.	***size*** is not 200.	***size*** is not equal to the number 200 (size is a numeric value).
is not	Array string value	The meta key is not equal to any of the elements of the *value* field.	***size*** is not 200, 300, 400.	***size*** is equal neither to *200* nor to *300* nor to *400*.
is not null	N/A (looks for any value)	The meta key value is not null.	***user_dst*** is not null.	***user_dst*** is a meta that contains a value.
is greater than (>)	Number	The numeric value of the meta key is greater than the number in the *value* field.	***payload*** is greater than 7000.	***payload*** is a numeric value that is greater than 7000.
is greater than or equal to (>=)	Number	The numeric value of the meta key is greater than or equal to the number in the *value* field.	***payload*** is greater than or equal to 7000.	***payload*** is a numeric value that is greater than or equal to 7000.
is less than (<)	Number	The numeric value of the meta key is less than the number in the *value* field.	***ip_dstport*** is less than 1024.	***ip_dstport*** is a numeric value that is less than the numeric value 1024.
is less than or equal to (<=)	Number	The numeric value of the meta key is less than or equal to the number in the *value* field.	***ip_dstport*** is less than or equal to 1024.	***ip_dstport*** is a numeric value that is less than or equal to numeric value 1024.
is one of	Array string value	The meta key is one of the array string values in the *value* field.	***alias_host*** is one of Facebook, UTube, Instagram.	***alias_host*** is one of the array string values *Facebook, UTube, Instagram*.
is not one of	Array string value	The meta key is not one of the array string values in the *value* field.	***alias_host*** is not one of Facebook, UTube, Instagram.	***alias_host*** is not one of the array string values *Facebook, UTube, Instagram*.

Operator	Required Value	Usage	Example	Meaning
contains	String	The *value* field is a substring of the meta key. (This operator is only available for a string-valued meta key).	***ec_outcome*** contains failure.	***ec_outcome*** is a string that contains the substring *"failure"*.
not contains	String	The *value* field is not a substring of the meta key (This operator is only available for a string-valued meta key).	***ec_outcome*** not contains failure.	***ec_outcome*** is a string that does not contain the substring*"failure"*.
begins with	String	The *value* field is the beginning of the meta key (This operator is only available for a string-valued meta key).	***ip_dst*** begins with 127.0.	***ip_dst*** is a string that starts with *"127.0"*.
ends with	String	The *value* field is the end of the meta key (This operator is only available for a string-valued meta key).	***user_dst*** ends with son.	***user_dst*** is a string that ends in*"son"*.

Note: Terms in ***bold italics*** are Meta that may not exist in all customer environments.

Advanced EPL Rule Tab

The Advanced EPL Rule tab enables you to define rule criteria with an Event Processing Language (EPL) query.

What do you want to do?

Role	I want to ...	Show me how
Content Expert	Define an Advanced EPL rule.	Add an Advanced EPL Rule
Content Expert	Test the Advanced EPL rule logic.	Validate an Advanced EPL Rule
Content Expert	See examples of an Advanced EPL Rule.	Example Advanced EPL Rules
Content Expert	See best practices for writing Advanced EPL Rules.	ESA Rule Writing Best Practices

Related Topics

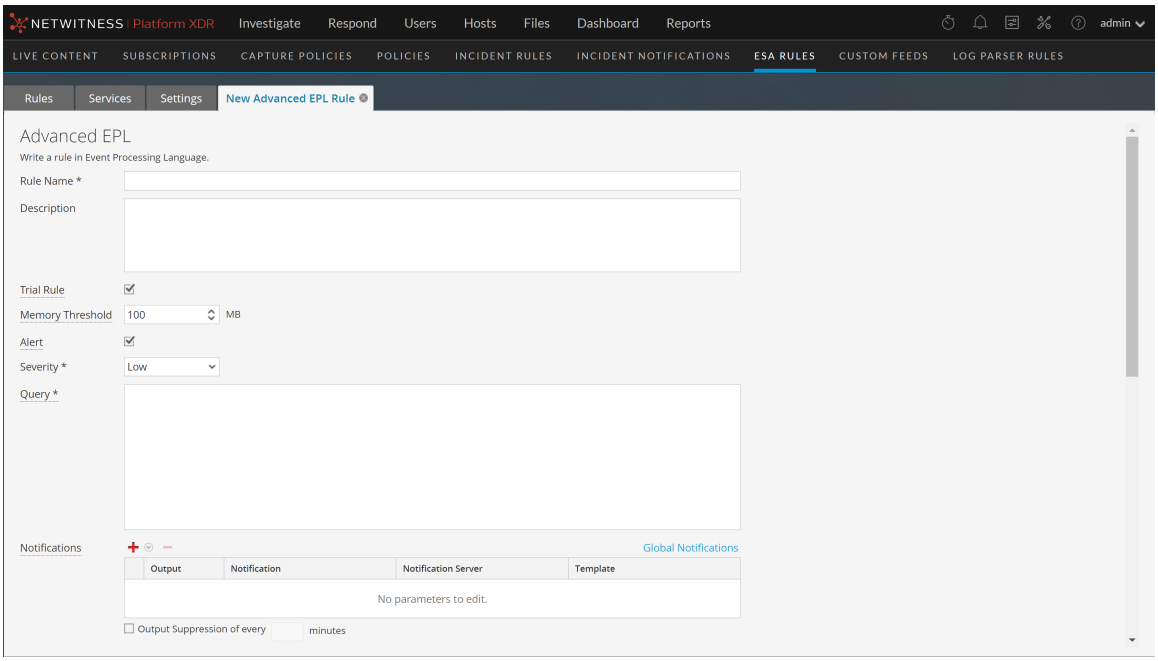
- Add a Rule Builder Rule
- Enrichment Sources

Quick Look

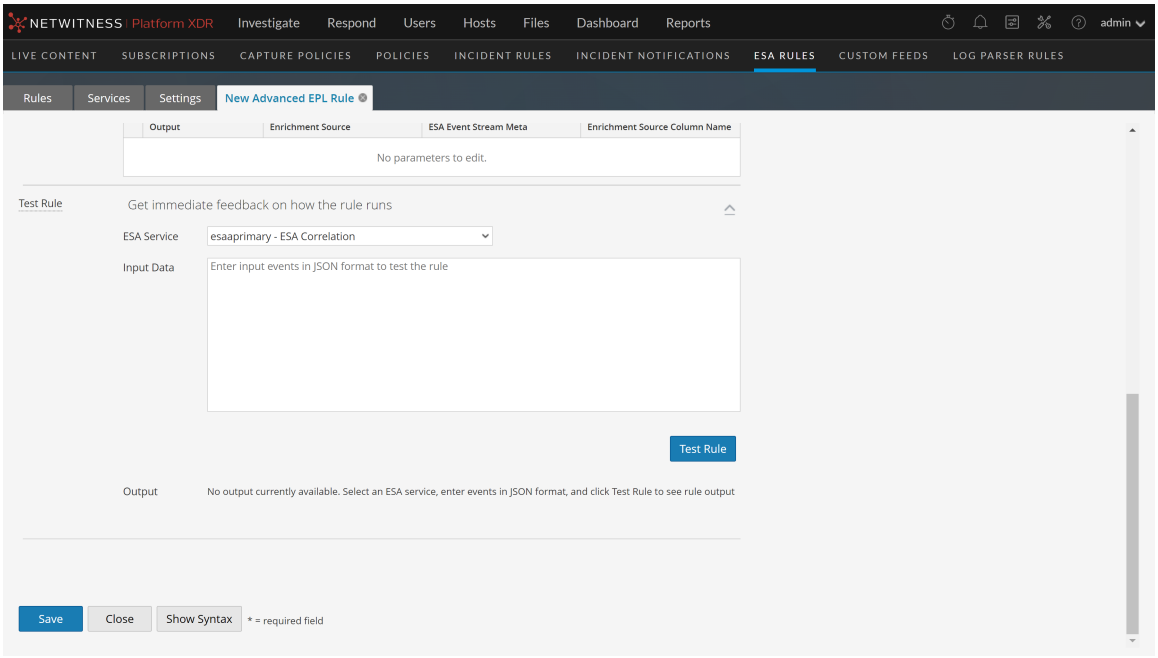
To access the Advanced EPL Rule tab:

1. Go to **(Configure) > ESA Rules**.
   The Configure view is displayed with the Rules tab open by default.
2. In the **Rule Library** toolbar, select **> Advanced EPL**.
   The Advanced EPL Rule tab is displayed.

The following figure shows the Advanced EPL Rule tab.



The following figure shows the Advanced EPL Rule tab scrolled down with the Test Rule section in view.



The following table lists the parameters in the Advanced EPL Rule tab.

Parameters	Description
Rule Name	Purpose of the ESA rule.
Description	Summary of what the ESA rule detects.

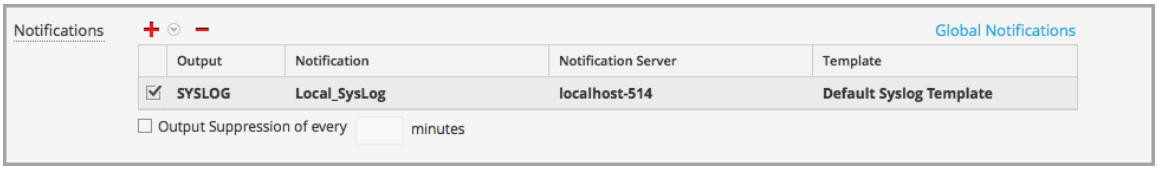
Parameters	Description
Trial Rule	Deployment mode to see if the rule runs efficiently.
Memory Threshold	(This option applies to version 11.5 and later.) The maximum memory usage allowed for this rule in MB. Add Memory Thresholds to ESA rules that use memory. For example, if a rule contains windows or pattern matching, configure a memory threshold for that rule. If the configured memory threshold is exceeded, it gets disabled individually and an error is displayed for that rule on the (Configure) > ESA Rules > Services tab. New rules default to a 100 MB memory threshold. Rules that existed before version 11.5 do not have a default value and a memory threshold is not set.
Alert	(This option applies to version 11.3 and Later.) When selected, the alert is sent to Respond. If the checkbox is cleared, an alert will not be sent to Respond. To turn alerts on or off for ALL rules, see the *ESA Configuration Guide*.
Severity	Threat level of alert triggered by the rule.
Query	EPL query that defines rule criteria.

Notifications Section

In the Notifications section, you can choose how to be notified when ESA generates an alert for the rule.

For more information on the alert notifications, see Add Notification Method to a Rule.

The following figure shows the Notifications section.



Parameter	Description
+	To add an alert notification type.
−	To delete the selected alert notification type.
Output	Alert notification type. Options are: • Email • SNMP (This option is not supported in NetWitness version 11.3 and later.) • Syslog • Script
Notification	Name of previously configured output, such as an email distribution list.

Parameter	Description
Notification Server	Name of server that sends the output.
Template	Name of template for the alert notification.
Output Suppression of every	Option to specify alert frequency.
Minutes	Alert frequency in minutes.

Enrichments Section

In the Enrichments section, you can add a data enrichment source to a rule.

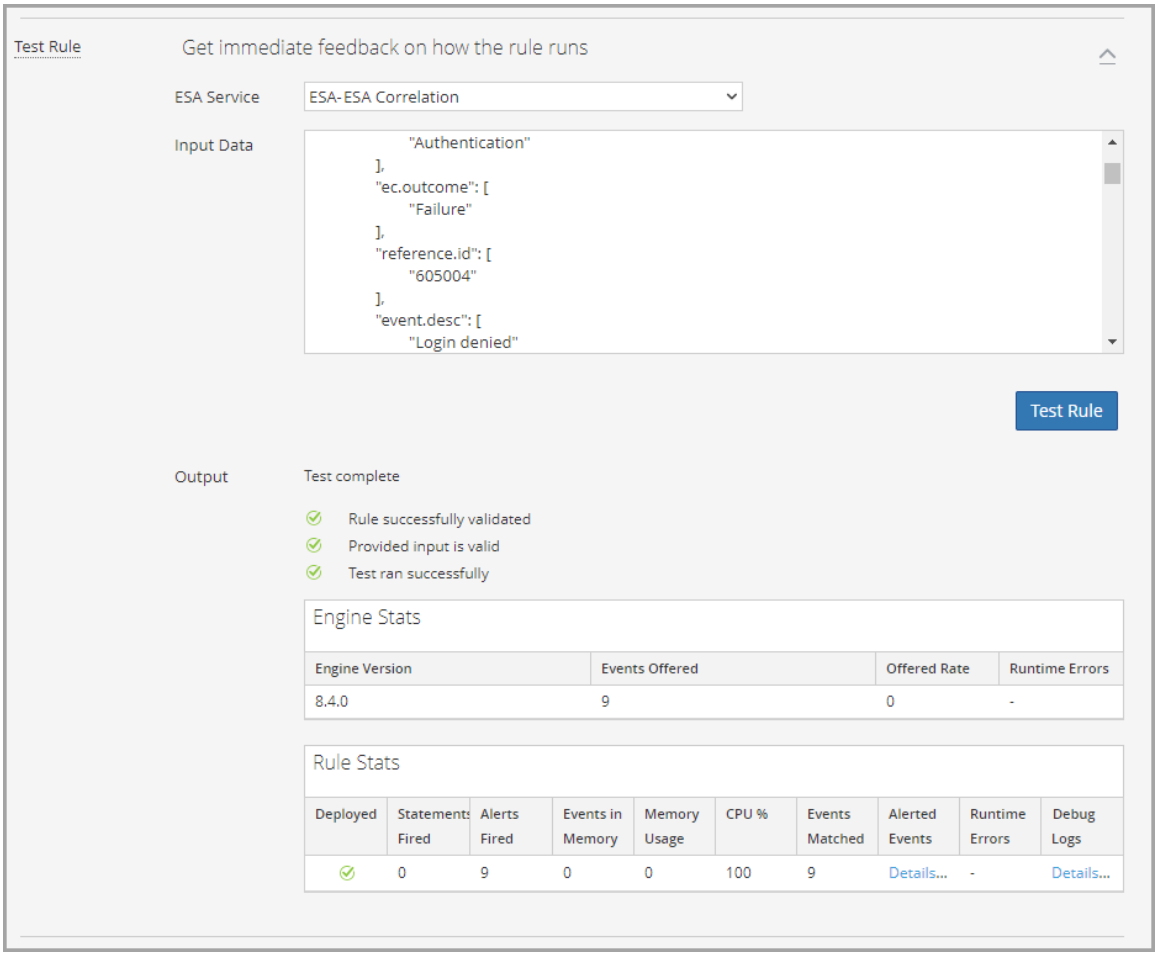
For more information on the enrichments, see Add an Enrichment to a Rule.
The following figure shows the Enrichments section.

Parameter	Description
	To add an enrichment.
	To delete the selected enrichment.
Output	Enrichment source type. Options are: • In-Memory Table (Ad hoc only - Recurring In-Memory Tables are no longer supported in version 11.3 and later.) • GeoIP
Enrichment Source	Name of previously configured enrichment source, such as a .CSV filename for an In-Memory Table.
ESA Event Stream Meta	ESA meta key whose value will be used as one operand of join condition.
Enrichment Source Column Name	Enrichment source column name whose value will be used as the other operand of the join condition.

Test Rule Section

Note: The Test Rule section is available in NetWitness Platform 11.5 and later.

In the Test Rule section, you can validate your ESA rule to determine if the rule logic is working as expected before deploying the rule.



Field	Description
ESA Service	Select the ESA Correlation service to process the rule.
Input Data	Enter the input events to test the rule. You can download the events from the Investigate view in JSON format, copy the events, and paste them in this field.
Output Data	After you select an ESA Correlation service, input data, and click the Test Rule button, you can view the output of the rule here and verify that the rule is working according to your requirements. You can view the alerts in the output, but this test does not send any alert notifications. If you want to view all of the debug information for the test, include an `@Audit('stream')` annotation to your rule query.

The following table describes the test rule output **Engine Stats**.

Field	Description
Engine Version	Esper version running on the ESA service
Events Offered	Number of events processed by the ESA service since the last service start
Offered Rate	The rate that the ESA service processes current events / The maximum rate that the ESA service processed events
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the ESA rule deployment.

The following table describes the test rule output **Rule Stats**.

Field	Description
Deployed	A green checkmark indicates that the rule is deployed on the selected ESA service.
Statements Fired	The number of statements that fired the alerts
Alerts Fired	The number of alerts generated from the test data
Events in Memory	The number of events placed in memory by the rule
Memory Usage	The total amount of memory used by the rule
CPU %	The percentage of the deployment CPU used by the rule. For example, a deployment with 1 rule shows 100% CPU usage for that rule and a deployment with two equally CPU heavy rules show 50% each.
Events Matched	The number of events that matched the rule
Alerted Events	If applicable, this field can contain a link to events that caused an alert.
Runtime Errors	If applicable, this field can contain a link to runtime error messages related to the rule.
Debug Logs	This field contains a link to Esper debug (audit) logs.

Syntax

Click **Show Syntax** to view the EPL syntax of conditions, statements, and debugging parameters. It also provides a warning when the syntax is invalid. For more information, see Rule Syntax Dialog.

Rule Syntax Dialog

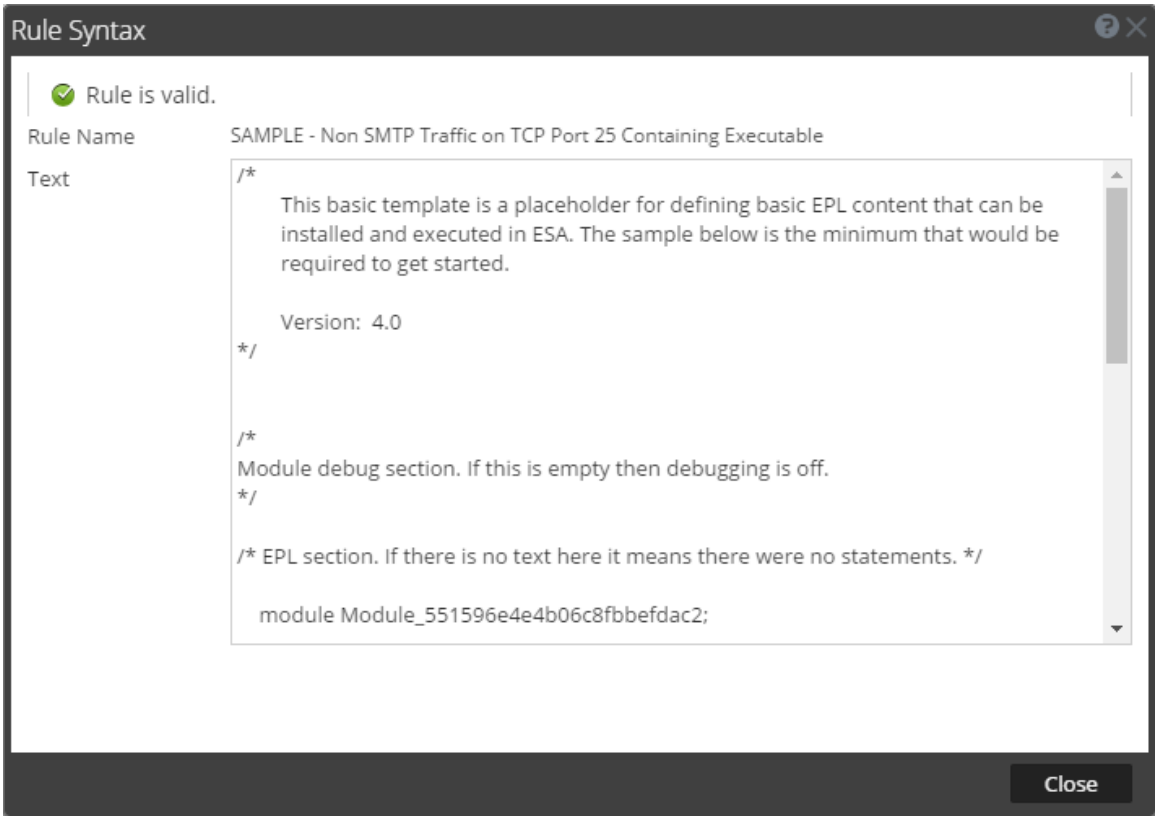
This topic describes the features of the Rule Syntax dialog. The Rule Syntax dialog displays the EPL syntax of conditions, statements, and debugging parameters, and provides a warning when the syntax is invalid.

Quick Look

To access this dialog:

1. Go to **(Configure) > ESA Rules**.
2. In the **Rule Library** view, do one of the following:
   a. Click and select **Advanced EPL** or **Rule Builder**.
   b. Double-click an existing rule.
   c. Select an existing rule and click in the **Rule Library** toolbar.
   d. In the row of an existing rule, select **> Edit**.
   The new or existing rule is displayed in a new tab, available to edit.
3. Click **Show Syntax** at the bottom of the tab.

The following figure shows an example of the Rule Syntax dialog showing a valid rule.



The following table describes the Rule Syntax dialog parameters.

Parameters	Description
Rule is valid or **Validation error in rule**	Indicates whether the rule syntax is valid or needs to be changed.
Rule Name	Displays the name of the rule.
Text	Displays the EPL syntax of conditions, statements, and debugging parameters if the rule is valid.

Services Tab

This topic provides an overview of the **(Configure) > ESA Rules > Services** tab. The Services tab shows the status of the deployments on each ESA service.

What do you want to do?

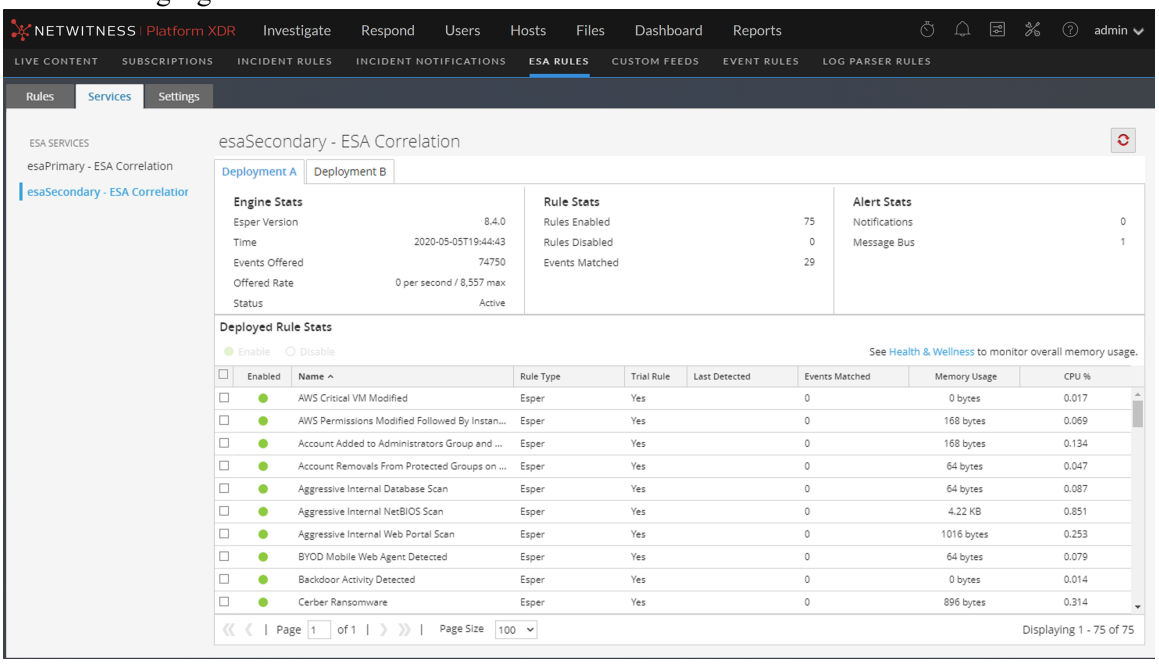
Role	I want to ...	Show me how
Content Expert	Troubleshoot Services Tab.	Troubleshoot ESA
Content Expert	View deployment Stats for an ESA Service.	View Stats for an ESA Service

Related Topics

- View a Summary of Alerts

Quick Look

The following figure shows the Services tab:



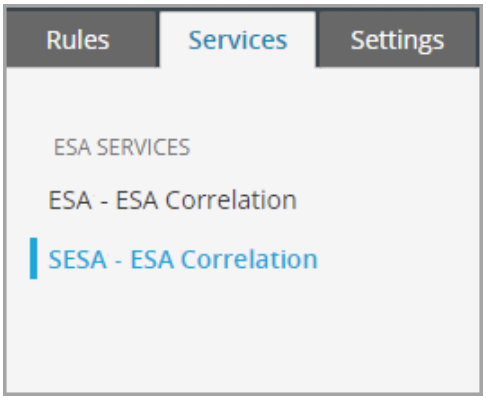
(This option is available in NetWitness version 11.3 and later.) If an ESA Correlation service has multiple deployments, under the service name, you will see a tab for each deployment. In the above example, there are two deployment tabs, Deployment A and Deployment B. Each tab displays information specific to that deployment.

The Services tab has the following sections:

- ESA Services panel (on the left)
- General Stats panel (top right)
- Deployed Rule Stats panel (bottom right)

ESA Services Panel

The ESA Services panel lists the name of each ESA service added to NetWitness.



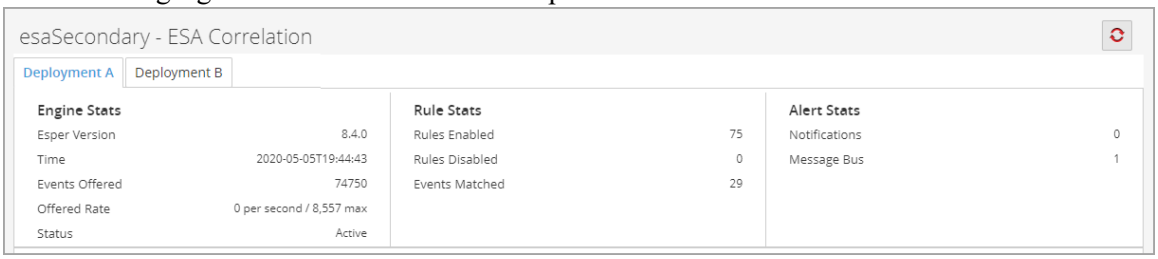
General Stats Panel

The General Stats panel provides information on the Esper engine, rules, and alerts.

The General Stats panel contains the following sections:

- Engine Stats
- Rule Stats
- Alert Stats

The following figure shows the General Stats panel.



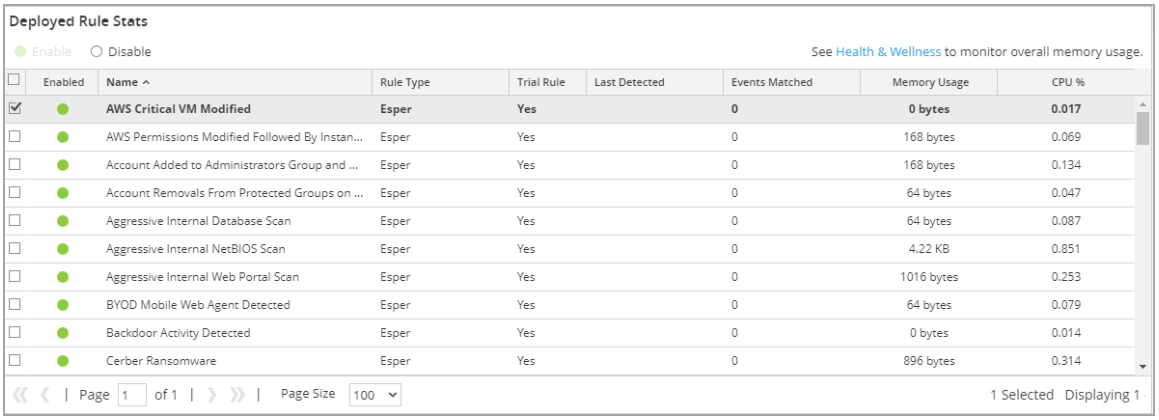
The following table lists and describes the parameters in each section.

Sections	Parameter	Description
Engine Stats	Esper Version	Esper version running on the ESA service
	Time	Time when the last event was sent to Esper Engine
	Events Offered	Number of events processed by the ESA service since the last service start
	Offered Rate	The rate that the ESA service processes current events / The maximum rate that the ESA service processed events.
	Status	Shows the status of the deployment. A status of **Active** means that the deployment is active. A status of **Inactive** means that there was probably an error starting the deployment. Check the error log file for more information: `/var/log/netwitness/correlation-server/correlation-server.log`.
Rule Stats	Rules Enabled	Number of rules enabled
	Rules Disabled	Number of rules disabled
	Events Matched	Total number of events matched to all rules on the ESA service
Alert Stats	Notifications	The total number of notifications sent by email, SNMP, syslog, or script for the deployment. (ESA SNMP notifications are not supported in NetWitness Platform version 11.3 and later.)
	Message Bus	The total number of alerts sent to Respond for the deployment

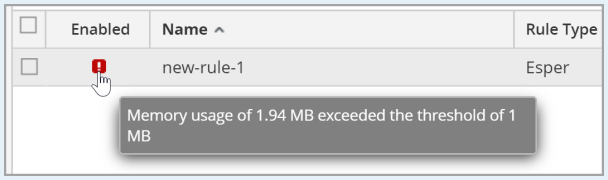
Deployed Rule Stats Panel

The Deployed Rule Stats panel provides details on the rules that are deployed on the ESA service.

The following figure shows the Deployed Rule Stats panel.



The table lists the various parameters in the view and their description.

Parameters	Description
Enable	Enables a rule that was disabled.
Disable	Disables a rule that was enabled.
Health & Wellness link	Enables you to monitor overall memory usage and health of your ESA Correlation service.
Enabled	Indicates whether the rule is enabled or disabled. A green circle icon indicates that the rule is enabled. A white circle icon indicates that the rule is disabled. If a disabled rule has an error message, it shows in the Enabled field. Hover over the icon to view the error message tooltip. The following example shows that the rule was disabled because it exceeded the configured memory threshold for that rule. 
Name	Name of the ESA rule.
Rule Type	(This field applies to version 11.3 and later.) **Endpoint** indicates a rule from the Endpoint Risk Scoring Bundle and **Esper** indicates Esper-specific rules, such as Rule Builder and Advanced EPL rules.
Trial Rule	Indicates if the rule is running in trial rule mode.
Last Detected	The last time alert was triggered for the rule.
Events Matched	The total number of events that matched the rule.
Memory Usage	The total amount of memory used by the rule. **Note:** The Endpoint Risk Scoring Rules Bundle rules do not show memory usage.
CPU %	The percentage of the deployment CPU used by the rule. For example, a deployment with 1 rule shows 100% CPU usage for that rule and a deployment with two equally CPU heavy rules show 50% each. (This field is available in version 11.5 and later.) **Note:** The Endpoint Risk Scoring Rules Bundle rules do not show CPU usage.

Settings Tab

This topic describes the components of the **(Configure) > ESA Rules > Settings** tab. In the Settings tab, you can perform the following tasks:

- MetaKey References
- Viewing the List of Meta Entities
- Enabling Meta Entity in the ESA Correlation Server
- Building Rules with Custom Meta Entities
- Configure a data enrichment source
- View Named Windows

What do you want to do?

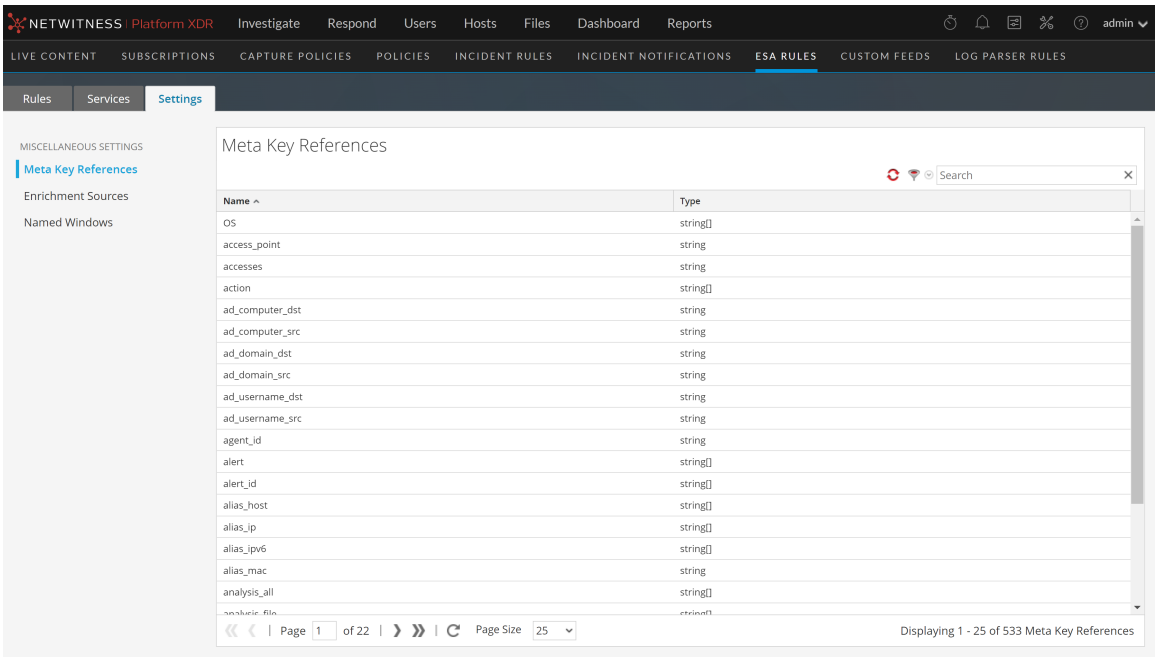
Role	I want to ...	Show me how
Content Expert	Configure an in-memory table as an enrichment source. (Recurring In-Memory Tables are no longer supported in version 11.3 and later.)	Configure an In-Memory Table as an Enrichment Source
Content Expert	Configure a Context Hub list as an enrichment source.	Configure a Context Hub List as an Enrichment Source

Related Topics

- Add a Data Enrichment Source
- Rule Builder Tab

Quick Look

The following figure shows the Meta Key References section in the Settings tab.



Meta Key References

The Meta Key References section lists each Meta key used by the ESA Correlation server and the type of value the key requires.

Meta key entities are configured to be a part of Event Schema. After you upgrade from NetWitness® Platform 11.5, or previous releases to NetWitness® Platform 11.6, you can enable the string [] meta keys entities. You can create rules and configure alerts based on the meta key entities that you select. You can also add meta entities to create rules. The meta entities retrieves data from the data sources to trigger alerts.

Note: Only the meta keys comprising of string [] are supported.

Viewing the List of Meta Entities

Perform the following steps to view the list of meta entities that are available in the system:

1. Go to (Configure) > **ESA RULES** > **Settings** > **Meta Key References**.

The Meta Key References page displays all the meta keys that are enabled.

2. Enter the name of the meta key in the Search text box to view specific meta entities.

Enabling Meta Entity in the ESA Correlation Server

After an upgrade to NetWitness 11.6 or later, you need to enable the meta entities in the ESA Correlation server to create rules and configure alerts based on the meta key entities.

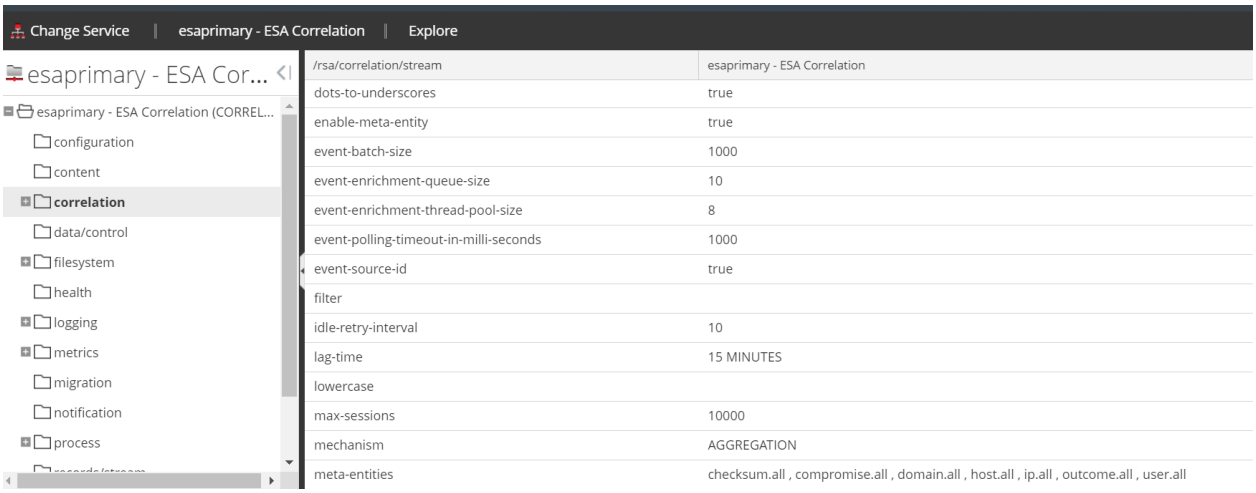
To enable meta entities in the ESA Correlation server:

1. Click (Admin) **Admin** > **Services**.

2. In the Services view, select the ESA Correlation service.

3. Click the settings icon and go to **View** > **Explore**.

The ESA Correlation Configuration page lists all the services.



4. Click **enable-meta-entity** and set it to **true**.

Note: By default, **enable-meta-entity** is set to **false**. Ensure that you enable only the specific meta entitiy using which you want to build a rule. Enabling multiple meta entities affects the system performance.

5. Add custom meta entities by appending or editing the meta-entities field values under explore view.

In the Explore view node list for the ESA Correlation service, select **Correlation** > **Stream**.All the stream settings are listed. You can add custom meta entities to the list as per the requirement. Once you enable custom meta entities, you can view them under Meta Key References. Once you enable the custom meta key entities, the rules will start evaluating the meta key entities and start to evaluate and trigger alerts as per the configured conditions. For more information, see Viewing the List of Meta Entities

The following meta entities are listed in the ESA Correlation Configuration page:

- checksum.all
- compromise.all
- domain.all
- host.all
- ip.all
- outcome.all
- user.all

Meta Entity Usage Example

ip_all, and *host_all* are some of the examples of meta entities.

For example, the *ip_all* meta entity can be used to check for any malicious IPs. It combines all the IPv4 meta keys (*ip_src*,*alias_ip*, *ip_dst*, and *forward_ip*) and creates a string array to retrieve information and trigger alerts.

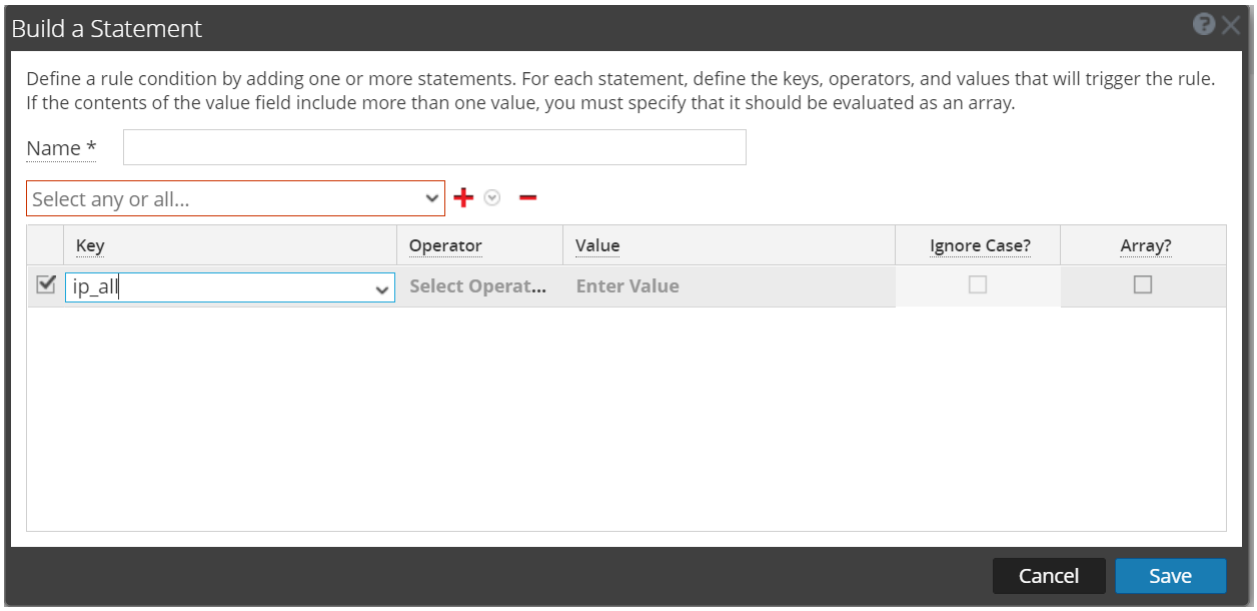
Note: You can add and enable custom meta entities (string arrays) to the list.

Building Rules with Custom Meta Entities

This topic provides instructions to define rule criteria in Rule Builder by adding statements. A statement is a logical grouping of rule criteria in the Rule Builder. You add statements to define what a rule detects.

The following graphic shows an example of a Rule Builder statement using meta entities.

Every statement contains a key and value. Then, you build logic around the pair by selecting an option in each other field.



Prerequisites

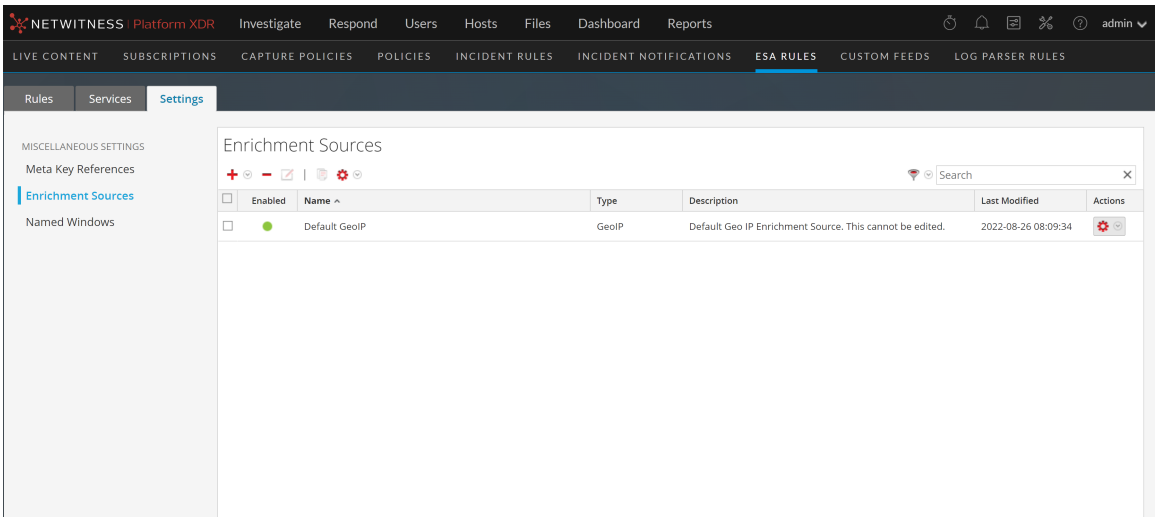
To build a rule statement, you must know the meta key and the meta value. For a complete list of meta keys, go to (Configure) > **ESA Rules** > **Settings** > **Meta Key References**.For more information about building rules with meta entities, see Rule Builder Tab.

Enrichment Sources

In the Enrichment Sources section, you can use the following external data sources:

- GeoIP
- In-Memory Table (Add hoc only - Recurring In-Memory Tables are no longer supported in version 11.3 and later.)
- Context Hub

The following figure shows the Enrichment Sources section in the Settings tab.

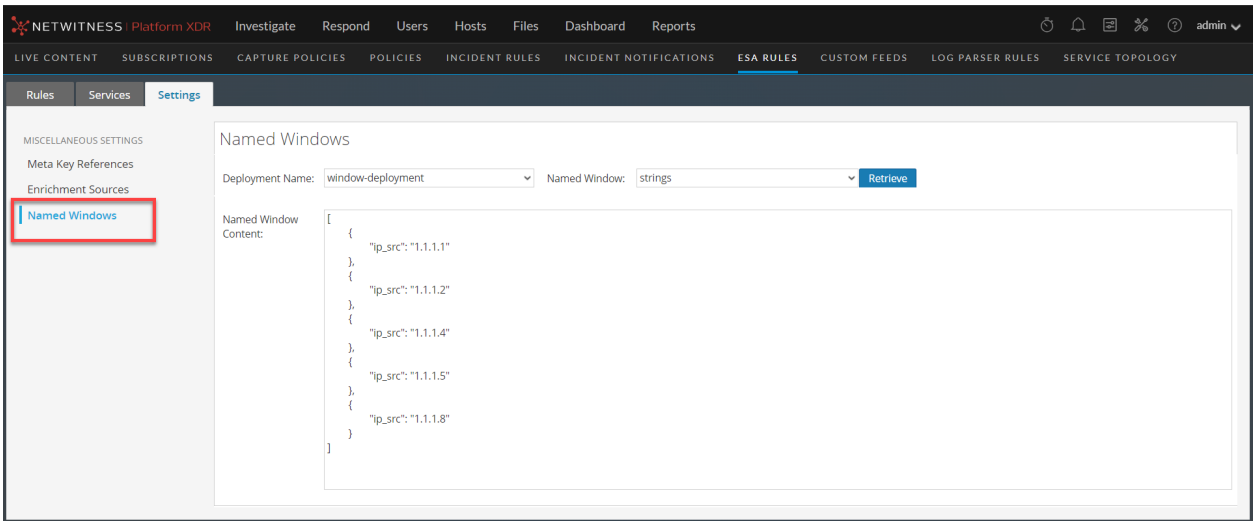


View Named Windows

The ESA Esper processing model is continuous, helps in storing the data from the consumed sessions in the form of named windows based on the deployed ESA rule.

From 12.0 and later, you can view the contents of the windows of a particular named window using the named window settings tab.

The following figure shows the Named Windows section in the Settings tab.



Perform the following steps to view the list of named windows that are available in the system:

1. Go to **Configure** > **ESA RULES** > **Settings** > **Named Windows**.
2. Select a deployment name from the **Deployment Name** drop-down list.
3. Select a named window from the **Named Window** drop-down list.
4. Click **Retrieve**.