NetWitness® Platform

Version 12.5.1

MetaExport Installation and Configuration Guide

NETWITNESS
Platform

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

RSA and other trademarks are trademarks of RSA Security LLC or its affiliates ("RSA"). For a list of RSA trademarks, go to https://www.rsa.com/en-us/company/rsa-trademarks. Other trademarks are trademarks of their respective owners.

License Agreement

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

No title to or ownership of the software or documentation or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software and the documentation may be subject to civil and/or criminal liability. This software is subject to change without notice and should not be construed as a commitment by RSA.

It is advised not to deploy third-party repos or perform any change to the underlying NetWitness Operating System that is not part of the supported NetWitness version. Any such change outside of the NetWitness approved image may result in a service or functionality conflict and require a reimage of the NetWitness system to bring NetWitness back to an optimized functional state. In the event a third-party repo is deployed, or other non-supported change is made by the customer without NetWitness approval, the customer takes full responsibility for any system malfunction until the issue can be remediated through troubleshooting efforts or a reimage of the service.

Third-Party Licenses

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

Note on Encryption Technologies

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

Distribution

Use, copying, and distribution of any RSA Security LLC or its affiliates ("RSA") software described in this publication requires an applicable software license.

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

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." RSA MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Miscellaneous

This product, this software, the associated documentations as well as the contents are subject to NetWitness' standard Terms and Conditions in effect as of the issuance date of this documentation and which can be found at https://www.netwitness.com/standard-form-agreements/.

November, 2024

Contents

Overview

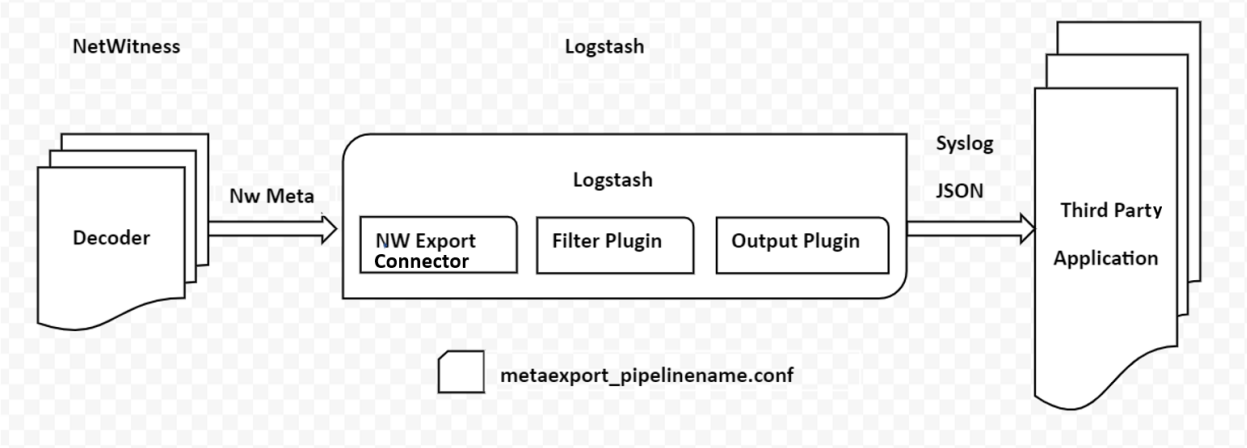
MetaExport uses the Netwitness Export Connector. The NetWitness Export Connector is an input plugin for Logstash, used to export NetWitness Platform meta and routes the data where you want, all in continuous, streaming fashion. Giving you the flexibility to unlock a variety of downstream use cases.

This plugin is installed on Logstash and integrates with NetWitness Platform Decoders. This plugin aggregates meta data from the Decoder and converts it to Logstash JSON object, which can easily integrate with numerous consumers such as Kafka, AWS S3, TCP, Elastic, Syslog receivers and others.

MetaExport uses logstash syslog output plugin to send events to a syslog server. You can send messages compliant with RFC3164 or RFC5424 using either UDP or TCP as the transport protocol.

Workflow of NetWitness MetaExport

Following diagram shows how NetWitness MetaExport works:

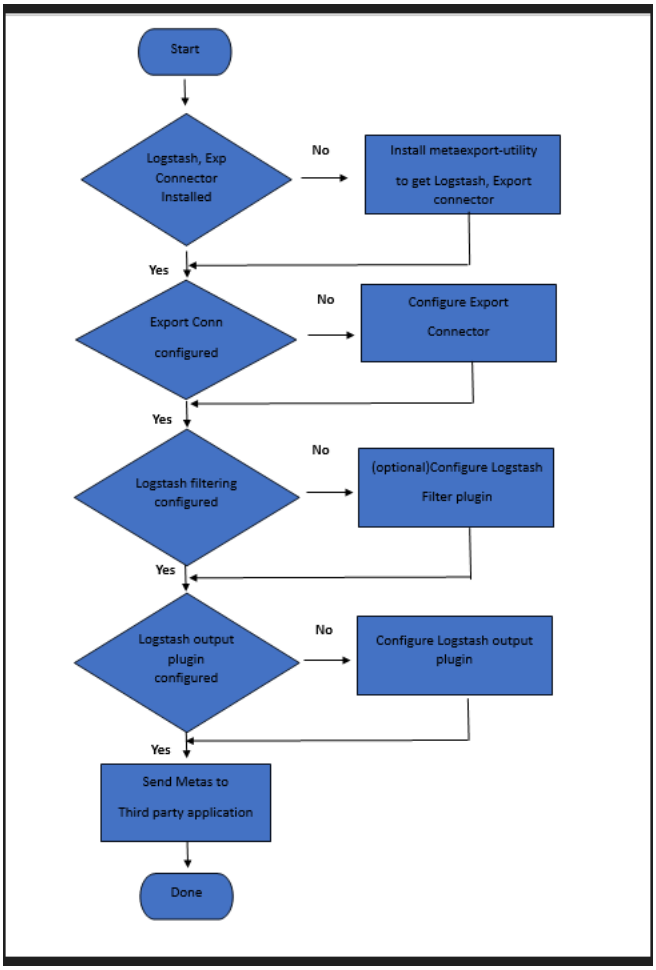


There are of three plugins available that helps with MetaExport.

- **Input plugin**: The Input plugin collects the events from the event sources. The NetWitness Export Connector uses NetWitness API that collects the meta data from the decoder. The data is then forwarded to the **Filter plugin**.
- **Filter plugin (optional)**: The Filter plugin adds, removes, or modifies the received data and forwards it to the **Output plugin**. You can use the standard Logstash filter plugins to add, remove, or modify the data.
- **Output plugin**: The syslog Output plugin sends the processed event data to the third-party application where the Syslog receiver is configured.

Configuration Process

The following flowchart describes the steps to configure NetWitness MetaExport.



VM Sizing Recommendations

For optimal performance, it is recommended to set the JVM Xms and Xmx to 1g and 8g respectively, when running the MetaExport directly on the Decoder. You can update the JVM settings for Logstash in the */etc/logstash/jvm.options* file. For more information, refer JVM settings.

MetaExport Running on a Decoder:

	EPS	VCPUs	JVM Memory
Without SSL	6.3 Gbps (10G Decoder)	32	8GB
With SSL	6.4 Gbps (10G Decoder)	32	8GB

MetaExport Running on a VLC:

	EPS	VCPUs	JVM Memory
Without SSL	6.2 Gbps (10G Decoder)	16	8GB
With SSL	6.3 Gbps (10G Decoder)	16	8GB

Install NetWitness MetaExport

Do the following steps to install Netwitness MetaExport.

1. Run the following command on the Decoder.

   ```
   yum install rsa-nw-metaexport-utility
   ```

2. The utility internally will install the open source version of logstash (OSS) as a dependency to the metaexport-utility. *For more information on released versions of Logstash, refer Logstash Reference*.

3. After completing the logstash installation, the utility will proceed to install the Export connector input plugin.

4. The utility will also install the logstash syslog output plugin.

   Note: NetWitness MetaExport uses logstash syslog output plugin to ship the metas to any third party application. Hence, you should create the syslog receiver on the third party application side to process the incoming metas.

Configure LogStash Keystore

When you configure Logstash, you might need to specify sensitive settings or configuration, such as passwords. Rather than relying on file system permissions to protect these values, you can use the Logstash keystore to securely store secret values for use in configuration file. In such a scenario, you should incorporate the decoder password into the Logstash Keystore.

Run the following commands to create the Logstash Keystore.

1. `vi /etc/sysconfig/logstash`

   ```
   # Add environment variable to the /etc/sysconfig/logstash file. For
   example, the expected format of /etc/sysconfig/logstash is LOGSTASH_
   KEYSTORE_PASS=<keystorepassword>, with one entry per line.
   ```

2. `chmod 600 /etc/sysconfig/logstash`
3. `source /etc/sysconfig/logstash`
4. `systemctl restart logstash.service`
5. `set +o history`
6. `export LOGSTASH_KEYSTORE_PASS=<keystorepassword>`

   ```
   # Replace <keystorepassword> with the password set in
   /etc/sysconfig/logstash
   ```

7. `set -o history`
8. `sudo -E /usr/share/logstash/bin/logstash-keystore --path.settings /etc/logstash create`
9. `sudo -E /usr/share/logstash/bin/logstash-keystore --path.settings /etc/logstash add KEY`

   ```
   # Give decoder password during the addition of the KEY

   # Make sure that the key got generated by running the following command
   ```

10. `sudo -E /usr/share/logstash/bin/logstash-keystore --path.settings /etc/logstash list`

For more information on LogStash keystore, refer Secret Keystore for Secure Settings.

Configure NetWitness MetaExport

You must configure the Logstash configuration file to process the NetWitness Platform events.

1. Run the following commands

   ```
   cd /etc/logstash/conf.d/metaexport/
   ```

2. Modify **metaexport_pipelinename.conf** file according to the need.

3. A Logstash configuration file can have three separate sections for each type of plugin that you want to add to the event processing pipeline. The first section is for Input plugin (NetWitness Export Connector), the second section is for Filter plugin (optional) and the third section is for syslog Output plugin. To configure the NetWitness Export Connector plugin, add the parameter settings in the first section of the Logstash configuration file.

The following is an example of NetWitness Export Connector with block of required and optional parameter settings which fetches data from a decoder.

```
input
{
netwitness_export_connector
{
# Mandatory Fields.
name => "<Pipelinename>"
host => "<IP/Host>"
username => "<Username>"
password => "${KEY}"
decoder_type => "decoder"

# Optional Fields.
meta_include => ""
meta_exclude => ""
query => ""
minutes_back => 0
multi_value => ""
start_session => 0
# SSL Configuration.
# ssl_enable => true
# ssl_certificate_path => "/etc/pki/nw/trust/truststore.pem"
# ssl_client_certificate_path => "/etc/pki/nw/node/node-cert.pem"
# ssl_key_path => "/etc/pki/nw/node/node-key.pem"
}
}
```

Following are the parameters accepted by NetWitness Export Connector.

Parameter	Description	Parameter Type	Default Value
name	A unique name to identify the logstash export connector pipeline	String	N/A
host	IP address or hostname of the Decoder (mandatory)	String	N/A
username	Username used to access the Decoder (mandatory)	String	N/A
password	LogStash KeyStore key for accessing Decoder (mandatory).	String	${KEY}
decoder_type	Accepts only 'decoder' (mandatory)	String	decoder
meta_include	Aggregates only the meta keys that are added in this parameter setting. Accepts comma separated values (csv) format	String	NIL
meta_exclude	Excludes the meta keys that are added in this parameter setting from aggregation. Accepts comma separated values (csv) format	String	NIL
query	Takes any NetWitness Platform query as Input. **Note:** Only Indexed meta key must be the part of the query. For example, `select * where country.src = 'Country Name'`	String	Select *
minutes_back	How far back in time should we go for a fresh start. This field accepts the values in minutes.	Number	0
multi_value	Metas to be treated as multiple values. For example, action, alias.host, alias.ip, alias.ipv6, email and username.	String	N/A
start_session	Session from which the aggregation starts. Setting the value to 0 starts the aggregation from `last.session.id` in the Decoder	Number	0

Filter Configuration (Optional):

You can configure the Logstash Filter plugin to add, remove, or modify the specific input events from the Decoder. To configure the Filter plugin, add the Filter plugin parameter settings in the second section of the Logstash configuration file . This plugin modifies the events based on the parameter settings. You can use the existing standard Logstash filter plugins for adding the parameter settings to the configuration file.

For more information on existing Logstash standard filter plugins, see Filter Plugins.

The configuration of the plugin must consist of the plugin name followed by a block of parameter settings for that plugin.

Example:

```
# Example
filter {
```

```
if [country_src] == "country name" {
# If this condition holds true, the entire session will be dropped.
drop {}
}
}
```

Output Configuration:

You must configure the Logstash syslog Output plugin to send the input events to a third party application.

The syslog Output plugin sends the processed event data to the third-party application where the Syslog receiver is configured.

For more information on existing Logstash standard output plugins, see Output Plugins.

Example:

```
output
{
syslog
{
id => "plugin_id"
host => "<server_IP>"
port => 514
sourcehost => “%{did}”
protocol => "tcp"
rfc => "rfc5424"
codec => "json"
#ssl_cert => “/path/to/server.crt”
#ssl_key => “path/to/server.key”
#ssl_cacert => “path/to/ca.crt”
}
}
```

Parameter	Description
id	Add a unique ID to the plugin configuration. If no ID is specified, Logstash will generate one. It is strongly recommended to set this ID in your configuration. This is particularly useful when you have two or more plugins of the same type.
host	Syslog server address to connect to the third party application such as splunk, palo alto cortex and so on.
port	Syslog server port to connect to the third party app. Use 6154 for the ssl connection.
sourcehost	Source host for syslog message. The default value for this is the Decoder ID.
protocol	Syslog server protocol. Use ssl-tcp for the ssl connection.
rfc	Syslog message format. You can choose between rfc3164 or rfc5424.

Parameter	Description
codec	Encodes the data to the json format if set to json.
ssl_cacert	The SSL CA certificate, chainfile or CA path. The system CA path is automatically included.
ssl_cert	The SSL certificate path.
ssl_key	There is no default value for this setting.

Configure Multiple Pipelines:

Follow these steps to configure multiple pipelines.

1. In order to create the multiple pipelines, Copy the file with the name **metaexport_ pipelinename.conf** inside the `/etc/logstash/ to /etc/logstash/conf.d/metaexport` with different name.The name of the configuration files should be unique.

   If you need to run more than one pipeline in the same process, Logstash provides a way to do this through a configuration file called pipelines.yml. This file must be placed in the `/etc/logstash/` folder and follows this structure:

   ```
   - pipeline.id: pipeline-1
   path.config: "/etc/logstash/conf.d/metaexport/metaexport_pipelinename.conf"
   ```

2. Add one more configuration block to the same **pipelnes.yml** file with unique pipeline ID and the path to the newly created configuration file.

   Example:

   ```
   - pipeline.id: pipeline-1
   path.config: "/etc/logstash/conf.d/metaexport/metaexport_pipelinename.conf"
   - pipeline.id: pipeline-2
   path.config: "/etc/logstash/conf.d/metaexport/metaexport_
   pipelinename1.conf"
   ```

   For more information on multiple pipelines, See Multiple Pipelines.