



**Security Threat Response Manager**

## **AQL Flow and Event Query CLI Guide**

***Release 2008.1***

**Juniper Networks, Inc.**

1194 North Mathilda Avenue

Sunnyvale, CA 94089

USA

408-745-2000

**[www.juniper.net](http://www.juniper.net)**

## Copyright Notice

Copyright © 2008 Juniper Networks, Inc. All rights reserved. Juniper Networks and the Juniper Networks logo are registered trademarks of Juniper Networks Inc. in the United States and other countries. All other trademarks, service marks, registered trademarks, or registered service marks in this document are the property of Juniper Networks or their respective owners. All specifications are subject to change without notice. Juniper Networks assumes no responsibility for any inaccuracies in this document or for any obligation to update information in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

## FCC Statement

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. The equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense. The following information is for FCC compliance of Class B devices: The equipment described in this manual generates and may radiate radio-frequency energy. If it is not installed in accordance with NetScreen's installation instructions, it may cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Reorient or relocate the receiving antenna. Increase the separation between the equipment and receiver. Consult the dealer or an experienced radio/TV technician for help. Connect the equipment to an outlet on a circuit different from that to which the receiver is connected.

Caution: Changes or modifications to this product could void the user's warranty and authority to operate this device.

## Disclaimer

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR JUNIPER NETWORKS REPRESENTATIVE FOR A COPY.

*STRM Adaptive Log Exporter*  
Release 2008.1

Copyright © 2008, Juniper Networks, Inc.

All rights reserved. Printed in USA.

Revision History

31 January 2008—Revision 1

The information in this document is current as of the date listed in the revision history.

# CONTENTS

---

## ABOUT THIS GUIDE

Documentation Feedback	3
Requesting Support	3

---

## 1 USING THE AQL QUERY CLI

About the AQL Query CLI	5
Accessing the AQL Query CLI	6
Using a Select Statement	7
Using Where Clauses	9
Using the Group By Clause	10
Using the Order By Clause	11
Using the Count(*) Clause	11
Using the Count (Distinct ...) Clause	11
Using the Materialize View Clause	12



# ABOUT THIS GUIDE

The *AQL Event and Flow Query CLI Guide* provides you with information for using the AQL CLI. This guide assumes you have advanced knowledge of Linux command line functionality.

- [Documentation Feedback](#)
- [Requesting Support](#)

---

## Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. Send your comments to [techpubs-comments@juniper.net](mailto:techpubs-comments@juniper.net), or fill out the documentation feedback form at <http://www.juniper.net/techpubs/docbug/docbugreport.html>. If you are using e-mail, be sure to include the following information with your comments:

- Document name
- Document part number
- Page number
- Software release version

---

## Requesting Support

Open a support case using the Case Management link at <http://www.juniper.net/support/> or call 1-888-314-JTAC (from the United States, Canada, or Mexico) or 1-408-745-9500 (from elsewhere).



# 1

## USING THE AQL QUERY CLI

You can use the AQL Event and Flow Query Command Line Interface (CLI) to access flows and events stored in the Ariel database. This document provides information on accessing and using the AQL query CLI including:

- [About the AQL Query CLI](#)
- [Accessing the AQL Query CLI](#)
- [Using a Select Statement](#)
- [Using Where Clauses](#)
- [Using the Group By Clause](#)
- [Using the Order By Clause](#)
- [Using the Count\(\\*\) Clause](#)
- [Using the Count \(Distinct ...\) Clause](#)
- [Using the Materialize View Clause](#)

---

### About the AQL Query CLI

The AQL event and flow query CLI allows you to access raw flows and events stored in the Ariel database. The AQL query CLI includes syntax that is a subset of the SQL92 standard and provides support for two tables: events and flows.



**Note:** *The AQL CLI does not provide support for joining tables.*

The AQL Event and Flow Query CLI functions in the following modes:

- **Interactive mode** - Using a simple shell, you can enter queries interactively and view the results in a standard output. At the query prompt, any valid AQL statement is accepted. If time is not specified (using `-start` and `-end` options), the last minute is assumed as the time range. You can also access previous commands by using your up arrow. This is the default mode.
- **Non-interactive mode** - You can enter the non-interactive mode by adding the `-execute <AQL query>` parameter to the command. The `-execute` command must be followed by a valid AQL query surrounded by double quotes. The non-interactive mode does not include a prompt allowing you to redirect the output to a file with a regular UNIX pipe syntax. By default, the results are sent to a standard output.

---

## Accessing the AQL Query CLI

To access the AQL query CLI:

- Step 1** Log in to STRM, as root.
- Step 2** Enter the following command:
- ```
/opt/qradar/bin/arielClient
```
- The Query prompt appears.

**CLI Options** [Table 1-1](#) lists the supported CLI options:

**Table 1-1** AQL CLI Options

| Option                                                       | Description                                                                                                                                                                                                                                                                                                                                         |
|--------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-range &lt;first record&gt; &lt;last record&gt;</code> | Limits the number of records sent to the output within the specified range. This is useful for viewing a selection of records generated by an ordered query. For example, if you wish to view the first ten records, you must specify <code>-range 1 10</code> .                                                                                    |
| <code>-debug</code>                                          | Generates debugging output during execution.                                                                                                                                                                                                                                                                                                        |
| <code>-start &lt;time&gt;, -end &lt;time&gt;</code>          | Specifies the start and end time of the query. Where <code>&lt;time&gt;</code> specifies the time. You must specify the time as either a UNIX timestamp or a date using the following format: <code>yyyy/mm/dd-hh:mm:ss</code> .<br><br>For example:<br><pre>/opt/qradar/bin/arielClient - start 2007/08/11-01:15:00 -end 2007/08/11-01:17:00</pre> |
| <code>-exectime &lt;time limit&gt;</code>                    | Specifies the maximum period of time, in seconds, a single query may continue processing.                                                                                                                                                                                                                                                           |
| <code>-execute &lt;AQL query&gt;</code>                      | Allows you to enter non-interactive mode that allows you to process a query that is sent to standard output. If you do not include this option, the command is entered in interactive mode. You must include your query in double quotes.                                                                                                           |
| <code>-f &lt;output format&gt;</code>                        | Allows you to specify the output format for the query results. The table format is an ASCII drawing of a multi-column table while the CSV format provides a comma separated list.<br><br>Where <code>&lt;output format&gt;</code> indicates the output format. The options are <code>table</code> or <code>csv</code> .                             |
| <code>-remote &lt;host:port&gt;</code>                       | Specifies that you wish to connect to a specific Ariel query host and port.                                                                                                                                                                                                                                                                         |

For example:

If you wish to enter a command in interactive mode:

```
/opt/qradar/bin/arielClient -start 2007/08/11-01:15:00 -end 2007/08/11-01:17:00 -exectime 60
/opt/qradar/bin/arielClient
```

```
/opt/gradar/bin/arielClient -start 2007/08/11-01:15:00 -end
2007/08/11-01:17:00 -f csv
```

If you wish to enter a command in non-interactive mode:

```
/opt/gradar/bin/arielClient -start 2007/08/11-01:15:00 -end
2007/08/11-01:17:00 -exectime 60 -execute "select * from flows
where sourceIP = '231.12.37.17' and protocol != 'TCP.tcp_ip'"
```

---

## Using a Select Statement

You can use a select statement that includes one or more fields of a flow or event. You can also use an asterisk (\*) to denote all columns. All field names are case sensitive, however, the terms `select` and `from` are not case sensitive. The supported fields include:

**Table 1-2** Supported Fields

| Table       | Supported Statement    |
|-------------|------------------------|
| <b>Flow</b> | application            |
|             | destinationASN         |
|             | destinationBytes       |
|             | destinationByteRatio   |
|             | destinationDSCP        |
|             | destinationFlags       |
|             | destinationIP          |
|             | destinationIfIndex     |
|             | destinationNetwork     |
|             | destinationPackets     |
|             | destinationPacketRatio |
|             | destinationPayload     |
|             | destinationPort        |
|             | destinationPrecedence  |
|             | firstPacketTime        |
|             | flowDirection          |
|             | flowSource             |
|             | flowType               |
|             | geographic             |
|             | icmpType               |
|             | interface              |
|             | lastPacketTime         |
|             | packetsOut             |
|             | protocol               |
|             | remoteNet              |

**Table 1-2** Supported Fields (continued)

| <b>Table</b>  | <b>Supported Statement</b> |
|---------------|----------------------------|
|               | remoteServices             |
|               | sourceASN                  |
|               | sourceBytes                |
|               | sourceByteRatio            |
|               | sourceDSCP                 |
|               | sourceFlags                |
|               | sourceIP                   |
|               | sourceIfIndex              |
|               | sourceNetwork              |
|               | sourcePackets              |
|               | sourcePacketRatio          |
|               | sourcePort                 |
|               | sourcePrecedence           |
|               | sourcePayload              |
|               | totalBytes                 |
| <b>Events</b> | category                   |
|               | credibility                |
|               | creEventList               |
|               | destinationIP              |
|               | destinationMAC             |
|               | destinationNetwork         |
|               | destinationPort            |
|               | device                     |
|               | deviceGroup                |
|               | deviceType                 |
|               | duration                   |
|               | eventCount                 |
|               | hasOffense                 |
|               | magnitude                  |
|               | postNatDestinationIP       |
|               | postNatDestinationPort     |
|               | postNatSourceIP            |
|               | postNatSourcePort          |
|               | preNatDestinationIP        |
|               | preNatDestinationPort      |
|               | preNatSourceIP             |

**Table 1-2** Supported Fields (continued)

| Table | Supported Statement |
|-------|---------------------|
|       | preNatSourcePort    |
|       | protocol            |
|       | qid                 |
|       | relevance           |
|       | severity            |
|       | sourceIP            |
|       | sourceMAC           |
|       | sourceNetwork       |
|       | sourcePort          |
|       | startTime           |
|       | unparsed            |
|       | userName            |

For example:

```
select sourceIP, destinationIP, application from flows where
protocol = 'TCP.tcp_ip'
select category, credibility from events where severity > 8
select * from events where credibility >=9
```

---

## Using Where Clauses

You can restrict your AQL queries using **where** clauses. The supported logical operators in the clause include **and**, **or**, and parentheses. Also, the supported comparison operators include: **=**, **<**, **>**, **>=**, **<=**, and **!=**

For example,

```
select sourceIP, category, credibility from events where
severity > 9 and category = 5013
select sourceIP, category, credibility from events where
(severity > 9 and category = 5013) or (severity < 5 and
credibility > 8)
```

The **where** clause also supports the **arieltime** variable, which overrides the time settings passed to the AQL CLI. The **arieltime** variable must be used with the **between** keyword to specify the start and end time bounds of the query. All time constraints must be entered as either UNIX timestamps or formatted date/time strings.

You can only use the **arieltime** variable once in a single query. Therefore, you can only query a continuous span of time in a single AQL command.

The logical operator for the `arietime` variable and the remainder of the `where` clause should be the `and` operator. We recommend that you use the `arietime` variable as the last constraint of the query and the `and` operator between the `arietime` variable and the rest of the `where` clause.

## Using the Group By Clause

You can use the `group by` clause to aggregate your data. Typically, data aggregation is combined with arithmetic functions on remaining columns to provide meaningful results of the aggregation. For example, to enter a query to investigate the IP addresses that sent more than 1 million bytes within all flows in a specific time frame, you must enter:

```
select sourceIP, SUM(sourceBytes) from flows where sourceBytes >
1000000 group by sourceIP
```

The output includes:

```
-----
| sourceIP          | SUM_sourceBytes |
-----
64.124.201.151	4282590.0
10.105.2.10	4902509.0
10.103.70.243	2802715.0
10.103.77.143	3313370.0
10.105.32.29	2467183.0
10.105.96.148	8325356.0
10.103.73.206	1629768.0
-----
```

However, if you compare this information to a non-aggregate query, the output displays all the IP addresses that are unique:

```
select sourceIP, sourceBytes from flows where sourceBytes >
1000000
```

```
-----
| sourceIP          | sourceBytes      |
-----
64.124.201.151	1448629
10.105.2.10	2412426
10.103.70.243	1793095
10.103.77.143	1449148
10.105.32.29	1097523
10.105.96.148	4096834
64.124.201.151	2833961
10.105.2.10	2490083
10.103.73.206	1629768
10.103.70.243	1009620
10.105.32.29	1369660
10.103.77.143	1864222
-----
```

```
| 10.105.96.148 | 4228522 |
-----
```

In addition to the SUM operator, the MIN, MAX, and AVG arithmetic aggregation functions are also supported.

---

### Using the Order By Clause

You can add a single `order by` clause to the end of your AQL CLI query. Only one field can be used in the `order by` clause. Also, sorting can be switched between ascending or descending by appending the `asc` or `desc` keyword to the `order by` clause, respectively. By default, the query returns results in descending order.

For example:

```
select sourceBytes, sourceIP from flows where sourceBytes >
1000000 order by sourceBytes
```

Or, if you wish to display results in ascending order:

```
select sourceBytes, sourceIP from flows where sourceBytes >
1000000 order by sourceBytes asc
```

Combining the `group by` and the `order by` clauses in a single query is useful for creating data, such as, TopN lists to determine the most abnormal events or the most bandwidth intensive IP addresses. For example, the following query displays the top traffic intensive IP address in a descending order:

```
select sourceIP, sum(sourceBytes) from flows group by sourceIP
order by sum(sourceBytes) desc
```

---

### Using the Count(\*) Clause

You can use the `count(*)` clause to count the number of records matching your query. For example, if you wish to count all events with credibility equal to or greater than 9:

```
select count(*) from events where credibility >= 9
```

---

### Using the Count (Distinct ...) Clause

You can use the standard SQL `Count(Distinct ...)` clause to obtain unique counts. Using the AQL CLI, you can only use one field. For example, if you wish to view all the IP addresses that are connected to a specific IP address over time:

```
select count(distinct sourceIP) from flows where destinationIP =
'192.168.61.71'
```

Or, if you wish to view the number of unique source IP addresses communicating with a particular destination IP address:

```
select destinationIP, count(distinct sourceIP) from flows group
by destinationIP
```



**Note:** Using this clause may require additional system resources. Therefore, depending on the query, the amount of time to return results may vary.

## Using the Materialize View Clause

The `materialize view` clause allows you to produce query results as a static view and run subsequent queries against the view. You can also specify the period of time that the `materialized view` is accessible.

The syntax for the `materialized view` includes:

```
materialize view <time> NameOfView as select <statement>
```

Where:

- `<time>` specifies the time you wish the `materialized view` to be accessible.
- `<statement>` specifies a valid select statement.

For example, if you wish to create a `materialized view` containing flows with more than 1,000,000 source bytes, enter the following:

```
materialize view LargeSourceBytesFlows as select * from flows
where sourceBytes >1000000
```

To select from this view, enter the select statement as you would a valid table:

```
select * from LargeSourceBytesFlows
```

You can also use an aggregation statement on a materialized view:

```
select sourceIP, sum(sourceBytes) from LargeSourceBytesFlows
group by sourceIP
```



**Note:** You cannot create a `materialized view statement` based on a previously created `materialized view`.

If you wish to create a `materialized view` to select from a record set with ambiguous column names, you can define aliases for all computed columns. For example:

```
materialize view MyView as select sourceIP, sum(sourceBytes) as
srcBytesSum from flows group by sourceIP
```

Then you can refer to the alias in a subsequent query against `MyView`:

```
select * from MyView orderBy srcBytesSum
```