
Micro Focus Security ArcSight ESM

Software Version: 7.6

ESM Best Practices: Trends

Document Release Date: December 2021

Software Release Date: December 2021



Legal Notices

Copyright Notice

© Copyright 2001-2021 Micro Focus or one of its affiliates

Confidential computer software. Valid license from Micro Focus required for possession, use or copying. The information contained herein is subject to change without notice.

The only warranties for Micro Focus products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein.

No portion of this product's documentation may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's internal use, without the express written permission of Micro Focus.

Notwithstanding anything to the contrary in your license agreement for Micro Focus ArcSight software, you may reverse engineer and modify certain open source components of the software in accordance with the license terms for those particular components. See below for the applicable terms.

U.S. Governmental Rights. For purposes of your license to Micro Focus ArcSight software, "commercial computer software" is defined at FAR 2.101. If acquired by or on behalf of a civilian agency, the U.S. Government acquires this commercial computer software and/or commercial computer software documentation and other technical data subject to the terms of the Agreement as specified in 48 C.F.R. 12.212 (Computer Software) and 12.211 (Technical Data) of the Federal Acquisition Regulation ("FAR") and its successors. If acquired by or on behalf of any agency within the Department of Defense ("DOD"), the U.S. Government acquires this commercial computer software and/or commercial computer software documentation subject to the terms of the Agreement as specified in 48 C.F.R. 227.7202-3 of the DOD FAR Supplement ("DFARS") and its successors. This U.S. Government Rights Section 18.11 is in lieu of, and supersedes, any other FAR, DFARS, or other clause or provision that addresses government rights in computer software or technical data.

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

Support

Contact Information

Phone	A list of phone numbers is available on the Technical Support Page: https://softwaresupport.softwaregrp.com/support-contact-information
Support Web Site	https://softwaresupport.softwaregrp.com/
ArcSight Product Documentation	https://community.softwaregrp.com/t5/ArcSight-Product-Documentation/ct-p/productdocs

Contents

- Purpose of This Document 4
- Why Use Trends? 5
- Definition of Terms 6
- Building Blocks 7
 - Queries 7
 - Variables 8
- Types of Trends 9
 - Interval Trends 9
 - Daily or Hourly Trends? 9
 - It's a Matter of Timing 9
 - Trend Refreshes 13
 - Data from the Past: Backfilling 13
 - Snapshot Trends 15
- Trend Use Case Using Advanced Examples 16
 - Query (Q2) forTrend (T2) 18
- More Examples: Sharing Trend Data with Reports 22
- Debugging Your Trend: Where to Look 24
- Send Documentation Feedback 26

Purpose of This Document

This Best Practices document is written for ESM content authors responsible for creating ESM trends. If you are tasked to configure trends, this document will give you suggestions on trend settings that will make your trends work efficiently. If you have previously deployed ESM trends, refer to this document for any fine tuning techniques. If you are new to ESM trends, refer to this document to get ideas on trend configurations that will best work for you.

References

The Best Practices series supplements, but not replaces, the main ESM guides that contain complete step-by-step instructions and details. Refer to the following documentation:

- *ESM 101*, the topic "Trends" in the Reporting and Incident Analysis section
- *ArcSight Console User's Guide*, the "Building Trends" section and the topic "Trends" in the Reference section
- *ESM Administrator's Guide*, the topic "Query and Trend Performance Tuning" in the Troubleshooting section

Prerequisites

To understand how to work properly with trends, you must have experience in ESM queries that collect the data for trends. To view the output data from trends, you must have experience in ESM reports. While trends can also feed into query viewers, this document uses reports as examples.

Why Use Trends?

Trends help you understand long term changes in your environment. You can keep track of the prevailing behaviors and tendencies, for example, frequency of worm outbreaks or status of operating systems in your assets. Trends are ESM resources that collect such information. With trends, you define your query and set a schedule for collection. ESM manages the database with trend tables.

Trends can make reports and query viewers run faster because they can query the trend table that has fewer rows and columns. You can schedule trends to run at the optimum time when they don't conflict with other consumers of system resources.

Trends allow you to persist specific data for reporting beyond the ESM retention period. For example, if your ESM retention period is 15 days but you need to perform monthly reporting on detected viruses, a trend would allow you to persist event information around virus infections without being affected by the retention period.



Caution: Keep disk space availability in mind. Trends generally grow, especially when the trend does not designate an end date for data collection.

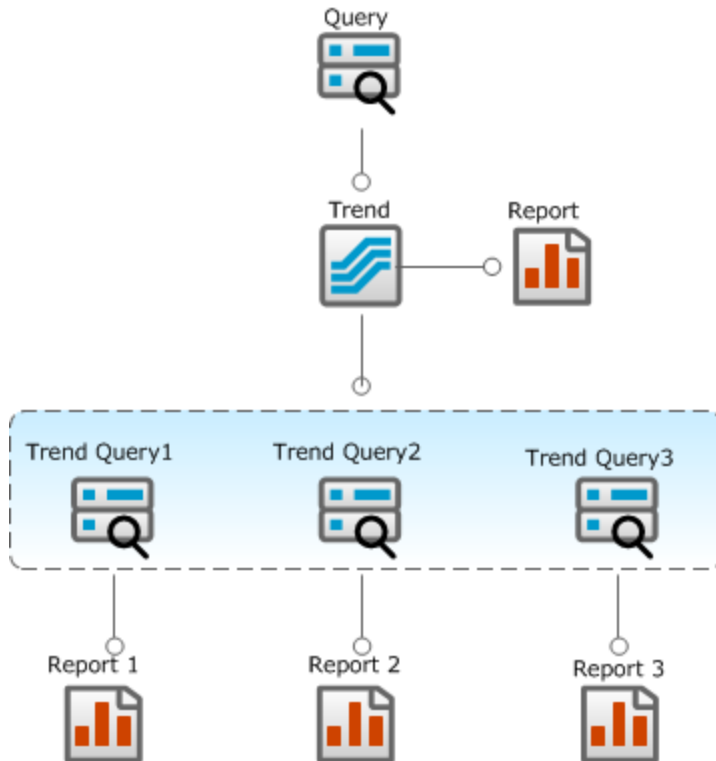
Definition of Terms

The following table describes relevant terms used in this and other ESM documents.

Term	Definition
Base query	The query specified on the trend definition. This query collects all data from the event table in the ArcSight database as required by the trend, and stores the data in a separate trend table.
Trend query	Also referred to as query on the trend. This query uses trend data as its data source.

Building Blocks

A query is the fundamental building block for a trend because that query collects the data for the trend table. A single trend can be the data source for one or more reports. You can create more granular queries on that trend (trend queries, a collection of queries on a trend). From these trend queries, you can create smaller, more granular reports.



Queries

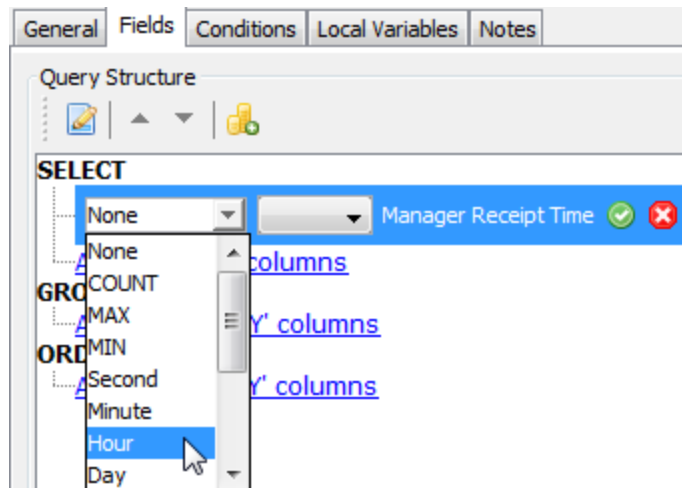
A trend uses a **base query** to collect all data required by the trend. A **trend query** uses the trend as the data source.

Time ranges and row limits

Queries include timestamp and row limit options. A trend can have its own settings for these options that may be different from the query's. The following options on the query **will** be overwritten by the trend to match the trend's scheduling parameters:

- Start Time and End Time (applies to interval trends, not snapshot trends)
- Row Limit

Be careful to avoid aggregation with the raw value of a time field (for example, Manager Receipt Time or End Time) for interval trends; these trends can become potentially large because each unique time will now have its own unique row. Instead, use the Hour or Day function in your query, as in the following example:




The values from these functions are sortable as text, for example:

- Day outputs as something like 2015-07-15.
- Hour outputs as something like 2015-07-15 18.

Query conditions

A base query, especially one with conditions set in the Conditions tab, will typically consider every event in the specified time range. That may take a long time.

 **Tip:** Entries on the query editor's Conditions tab represent the WHERE clause of a query statement.

Variables

Choose variables for your trend queries carefully.

- If you must use variables in your query, use them in trend queries against the trend table instead of in the base query against the event table.
- Group and Category Model variable functions (for example, FormatGroupsOf for assets and network zones, HasRelationship for actor models) are expensive to evaluate. These examples of functions are applicable to snapshot queries.
- Save zone and asset IDs in the trend table, then resolve names, URIs, and so on later, through variables.

Types of Trends

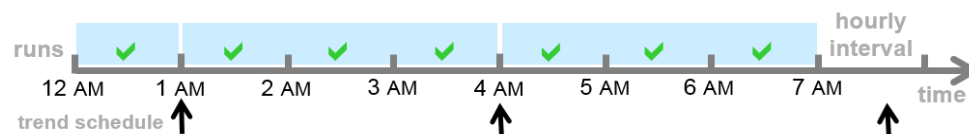
ESM supports two types of trends:

- **Interval Trends**
- **Snapshot Trends**

Interval Trends

Interval trends are for querying events. Interval trends have a rolling time window, for example, a steadily advancing hourly trend. Data collection time determines the oldest data to collect, such as 2016-01-01. You specify the trend's interval, and the query is executed at the start times of each interval. If the data at the requested time does not exist in the system, the query will still run, but will return NULL results.

The following example shows a trend that runs a query at hourly intervals with a three-hour schedule.



Daily or Hourly Trends?

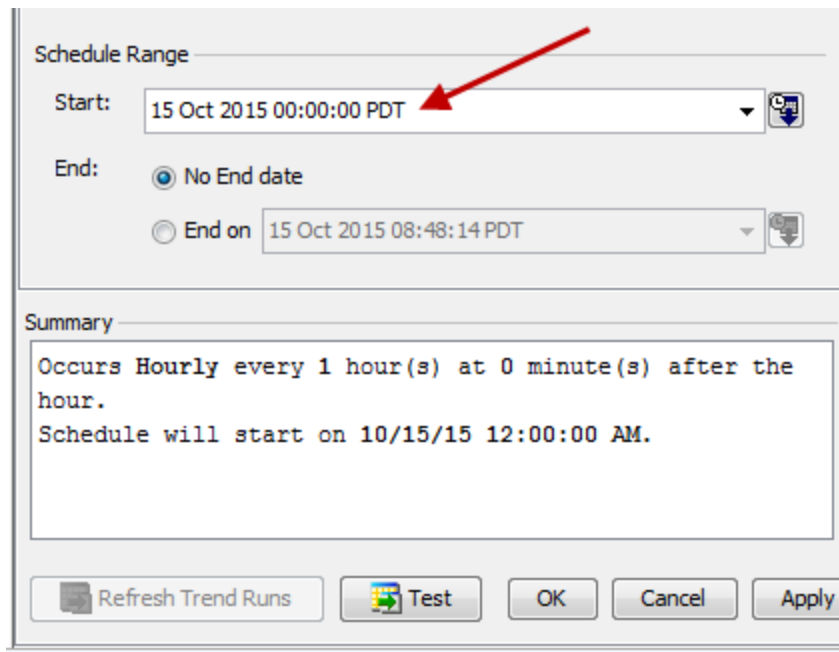
Consider using hourly intervals. With daily intervals, a combination of these factors might contribute to problems: high EPS rates (for example, greater than 15K); trends containing `ORDER BY` and `GROUP BY` clauses (very common for trends); and long-running trends returning huge sets of data.

It's a Matter of Timing

Data collection occurs on the time intervals (for example, once every hour) and at the scheduled trend run time. Spread out the trend run times, if you have one or more trends. Do this to avoid overloading the system with too many concurrently running queries.

Start schedule

On the trend's Schedule tab, the trend schedule's Start value indicates the oldest data to collect. By default, the value is set to midnight of the trend's creation date, as displayed on the trend editor's Summary field. For example, if you created the trend at 10:00 a.m. on October 15, 2015, the start time shows as midnight of October 15, 2015 (10 hours ago).



The screenshot shows the 'Schedule Range' section of a trend editor. The 'Start' field is a dropdown menu set to '15 Oct 2015 00:00:00 PDT', with a red arrow pointing to it. The 'End' section has two options: 'No End date' (selected) and 'End on' (set to '15 Oct 2015 08:48:14 PDT'). Below this is the 'Summary' field, which contains the text: 'Occurs Hourly every 1 hour(s) at 0 minute(s) after the hour. Schedule will start on 10/15/15 12:00:00 AM.' At the bottom are buttons for 'Refresh Trend Runs', 'Test', 'OK', 'Cancel', and 'Apply'.



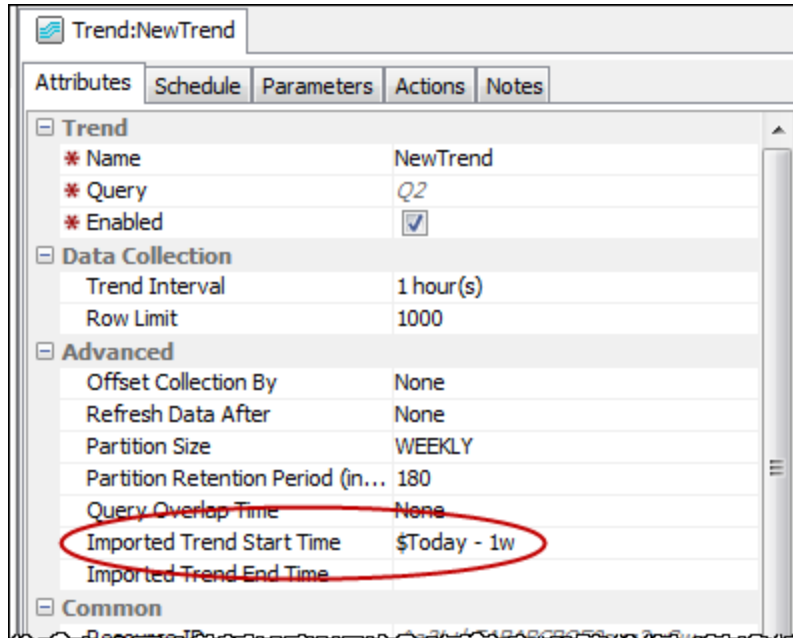
Note: Regarding timezones

- The trend will consider midnight to be 00:00:00 on the Manager's timezone, not the Console's.
- If you are creating a trend in an ArcSight Console with a different timezone from the Manager, the confirmation in the Summary field will display the Manager's time that will not match your original entry in the trend definition.

You can change the Start value, which can be in the past. For example, if you specified yesterday's date (relative to October 15 in the example), the query will collect matching events that were stored yesterday. Be mindful of the oldest event you want to collect. If you go beyond the installation or the trend's retention time, you will not get data back.

Start schedule for imported trends

If you are creating a trend to be imported into another ESM instance, specify a dynamic start time like `$Today - 1w` as in the following example:



This ensures that when the trend is imported and there is a huge gap between its creation and import (for example six months or longer), the trend does not use your old start date.

For details about exporting and importing ESM resources such as trends, refer to the topic, "Creating or Editing Packages" topic in the *ArcSight Console User's Guide*. That topic describes a package attribute, Format = **Export**, specifically for transporting resources from one ESM system to another.

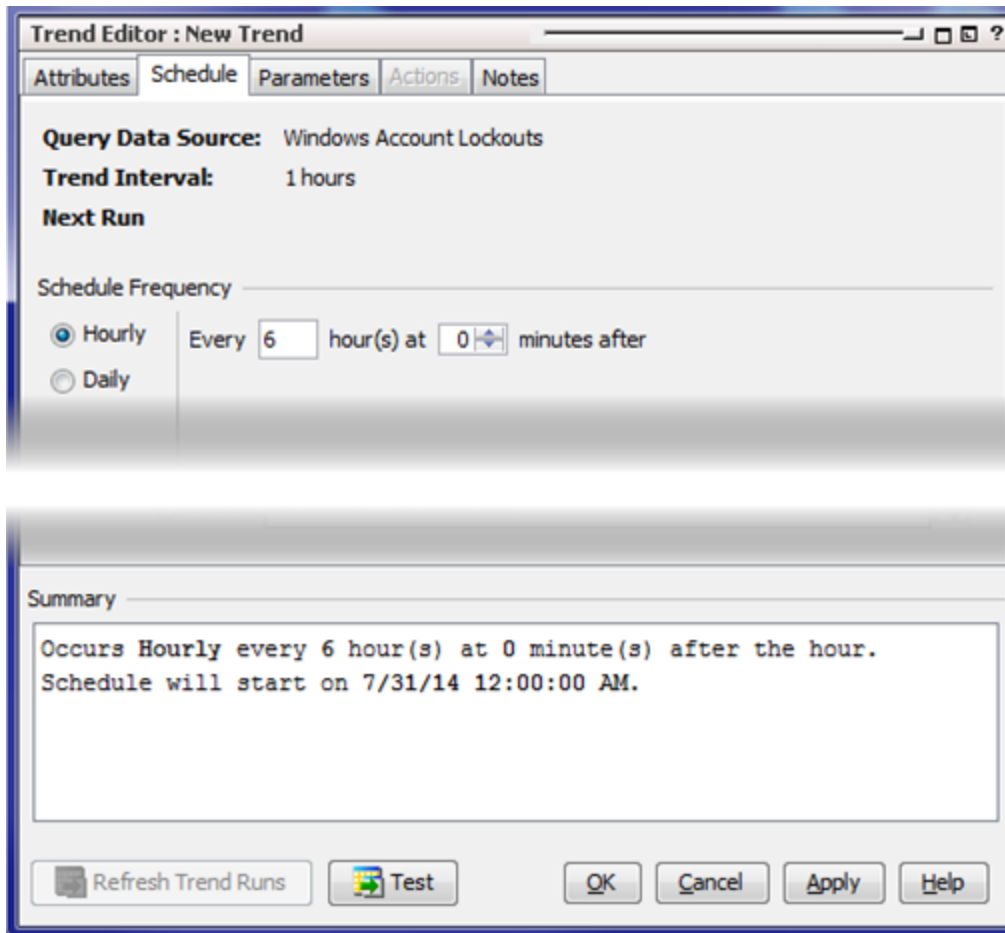
Refer to "[Data from the Past: Backfilling](#)" on page 13 that includes discussions about imported trends.

Intervals

Intervals define the query's time range, say, every hour; and the number of query intervals to run is determined by your value for Schedule Frequency. By default, trends are scheduled to run every hour. If you change the schedule frequency to 1 day, this will result in the trend running the same query a total of 24 times (the hourly interval) with these coverages:

- First run: Midnight to 01:00:00
- Second run: 01:00:00 to 02:00:00
- ... and so on for 24 query runs over one day (serially, not concurrently)

Following is an example of a trend with an hourly interval and a schedule frequency of 6 hours:



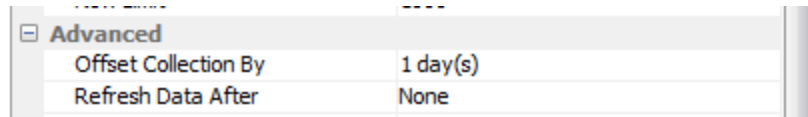
With this frequency, the trend will run four times a day over a period of 24 hours (6x4=24) as shown in the following table. Each trend run will run the query 6 times in 1-hour intervals:

Intervals	Run 1 at 00:00	Run 2 at 06:00	Run 3 at 12:00	Run 4 at 18:00
1st	00:00 - 01:00	06:00 - 07:00	12:00 - 13:00	18:00 - 19:00
2nd	01:00 - 02:00	07:00 - 08:00	13:00 - 14:00	19:00 - 20:00
3rd	02:00 - 03:00	08:00 - 09:00	14:00 - 15:00	20:00 - 21:00
4th	03:00 - 04:00	09:00 - 10:00	15:00 - 16:00	21:00 - 22:00
5th	04:00 - 05:00	10:00 - 11:00	16:00 - 17:00	22:00 - 23:00
6th	05:00 - 06:00	11:00 - 12:00	17:00 - 18:00	23:00 - 00:00

Collection offset time

Do not match your schedule with trend intervals exactly. Specify an offset to delay data collection and therefore handle late data arrivals for various reasons, for example, network delays, connector caching, or batching modes

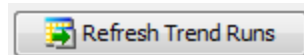
of different devices or connectors. The offset specifies how much time to delay before starting to collect data. The offset can be in minutes or hours.



You can also use offsets to avoid piling up any trend runs at the same hour. For example, schedule a trend run for midnight but give it a 3-hour offset so that it runs at 3:00 a.m.

Trend Refreshes

While ESM supports automatic refreshes, this is costly. Consider manually refreshing your trends for a special reason, such as a system downtime that may have caused trends not to run according to schedule. Manually refresh a trend by clicking the **Refresh Trend Run** button on the trend's Attributes tab:



Click **Cancel** after the refresh.

Data from the Past: Backfilling

A trend can have a start time that is in the past. When the trend is created, that trend will start its query from a start date such as "now," and move along time intervals in both directions: from now towards a past date; and from now moving forward. Collecting data from past dates is called backfilling, and this is done until the trend has caught up to "now." Consider a trend with a start time of $\$Today - 1d$. If today = March 29th, the trend will collect data from March 28th.

Be careful about extending the oldest range too far back into the past, because the system may not have data that old. The trend cannot tell this; you will simply not get data back.

Consider another trend that has an actual start time of March 28th and it is disabled; but before it is disabled, it has already collected some data. If it is re-enabled, the trend will try to catch up to "now" by filling in the gaps. Here are the events if the trend is enabled on April 3rd at 2:47 PM:

- The trend's first run will be scheduled to run at 12:00:00 AM on April 4th.
- The trend will start backfilling at 12:00:00 AM on April 4th.
- The **Refresh Trend Runs** button will not be available until the first run has completed.

- When you check the trend data on April 4th, you should see backfilled data from April 2nd and April 3rd. When you check the trend data on April 5th, you should see backfilled data from April 1st to April 4th.
- The trend will continue to backfill until it has collected data from the past all the way back to the start date.
- Always check if your trend continues to run after several days of backfilling. If necessary, disable then re-enable the trend.

Because the trend in this example was disabled for a time and then re-activated, the trend will try fill the gap for uncollected data during its disabled state, until the data has caught up to "now."

In summary, backfilling can take a long time, depending on how far back the trend needs to go and how much data is collected. If the start date is beyond what your storage retention period policy states, the trend will not return any data from old dates because data is no longer available. You should keep this in mind especially for a long-disabled trend that is now being re-activated, or an imported trend with an old start date.

Imported Trends

For imported trends, keep in mind that you have two Managers: the source Manager (call this the development system) where the trend was created and packaged, and the destination Manager (call this is the production system) into which the packaged trend is imported. When creating a trend package in a source Manager for export purposes, make sure the package's **Format > Export** option is specified (this is not the default setting for package formats). This sets the trend attribute's **Imported Trend Start Time** to `$Today - 1d` (the Manager's timezone is in consideration, not the Console's. Both can possibly be in different timezones). In this case, if the packaged trend was created 6 months ago and you import into another Manager today, the query will collect data from yesterday's date of the destination Manager instead of from 6 months ago, when the trend was created from the source Manager. See the topic, "Managing Packages," in the *ArcSight Console User's Guide*.

When the imported trend "wakes up" (is activated) in the destination, the trend will run additional queries to cover for any gaps in the requested time range. The trend will preferentially concentrate on keeping the data up to the current trend, then will fill in any gaps in the backward direction as time permits. To fill in the gap, the trend will run the query to up to 48 times by default, for example, two days backward as the trend goes forward for its hourly run.

Re-activated Trends

Before re-activating a de-activated trend, check its start date. If necessary, change the start date to focus the range that will cover recent, available data.

Interval Trends with Overlapping Queries

By default, queries for interval trends have no gaps and overlaps. You can include a `Query Overlap Time` value (an optional setting for interval trends) if you want the trend to support trailing N -day moving averages. Overlaps ensure your moving averages are what you expect. If you are only interested in counts, overlaps are not required. Refer to the *ArcSight Console User's Guide* on the `Query Overlap Time` trend setting.

Queries on the event table should not run longer than a day; using hourly intervals is preferred. If you have a 10-day query with a 9-day overlap, it is better to run the query on a trend table so that the query finishes in a reasonable amount of time. See "[Trend Use Case Using Advanced Examples](#)" on the next page for specific examples on how overlapping queries work for interval trends.



Note: `Query Overlap Time` does not apply to, and is therefore disabled for, snapshot trends.

Snapshot Trends

A snapshot trend is based on a query of other ESM resources such as assets, cases, notifications, and so on. Snapshots operate on a fixed moment in time. For example, snapshot trends on assets can reveal information such as total number; and of these, how many are with a particular OS; and also how many contain certain vulnerabilities. Snapshots only include data from the current moment in time, and onward; for example, a month's worth of data from now until 30 days after. Unlike interval trends, snapshots cannot run back in time.

Because resources do not have the same kind of timestamp information as do events (except for the Case resource which has a create date), pay attention to how you set up your snapshot trends. If you want the trend to capture assets that were created yesterday, you can set up your base query to collect the assets with a create date between `$Today - 1D` and `$Today`. For a snapshot query that only needs to run once a day, this query is fine. If your organization has multiple assets that are created daily, you can have the trend run multiple times per day. With thousands of assets, query performance may slow down. While you can change the schedule to run more frequently, the create date parameters for the query will not change. This means you must manually change the create date parameters for the query.

Trend Use Case Using Advanced Examples

As discussed elsewhere in this guide, a trend is one of the building blocks for creating a report. The *ArcSight Console User's Guide* shows a basic, end-to-end example of creating a base query to populate a trend table, creating three trend queries, and generating three distinct reports. Refer to the "Building Reports" section in that guide for those examples.

This use case shows an example of how to use advanced features on an interval trend that will capture logins in 15-day intervals. The data will be used to calculate a 15-day moving average for reports and rules. We are interested in generating a report on successful logins to monitor system capacity and we also want to track failed logins for possible attacks.

This example use case implements the following techniques:

- Creation of the base query for a daily-interval trend
- Creation of a query based on the daily-interval trend to create a 15-day interval trend with a 14-day overlap

The Base Query: a Start

The base query for the daily trend will get a count of all logins, regardless of outcome: successful and failed .

The screenshot displays two panels from a query configuration tool. The left panel, titled 'Query Fields:', contains a SQL query snippet: `SELECT Count(Event ID) COUNT, Category Outcome`, followed by a link to 'Add 'SELECT' columns'. Below this is the `GROUP BY` clause: `Category Outcome`, with a link to 'Add 'GROUP BY' columns'. The right panel, titled 'Query Conditions:', shows a tree view starting with 'Event conditions' and 'Event'. It includes an 'AND' operator with a condition 'Category Behavior = /Authentication/Verify'. Below this is an 'OR' operator with two conditions: 'Category Outcome = /Success' and 'Category Outcome = /Failure'.



Tip: The example query conditions does not show it, but you can consider including asset categories as one of the condition statements. Also consider using SUM(Aggregated Event Count).

Daily Trend (T1)

Initially, to fulfill our goal of capturing logins every 15 days, the instinct would be to create a 15-day interval trend that would run every 15th day to collect 15 days worth of data (provides answers to "Give me the last 15 days of logins ..."). However, this is not a good use of the trend. Instead, run the trend at the first scheduled time and search the event table for the last 15 days of login data. Based on the trend's start date therefore, the trend would collect that day, plus data from the previous 14 days.

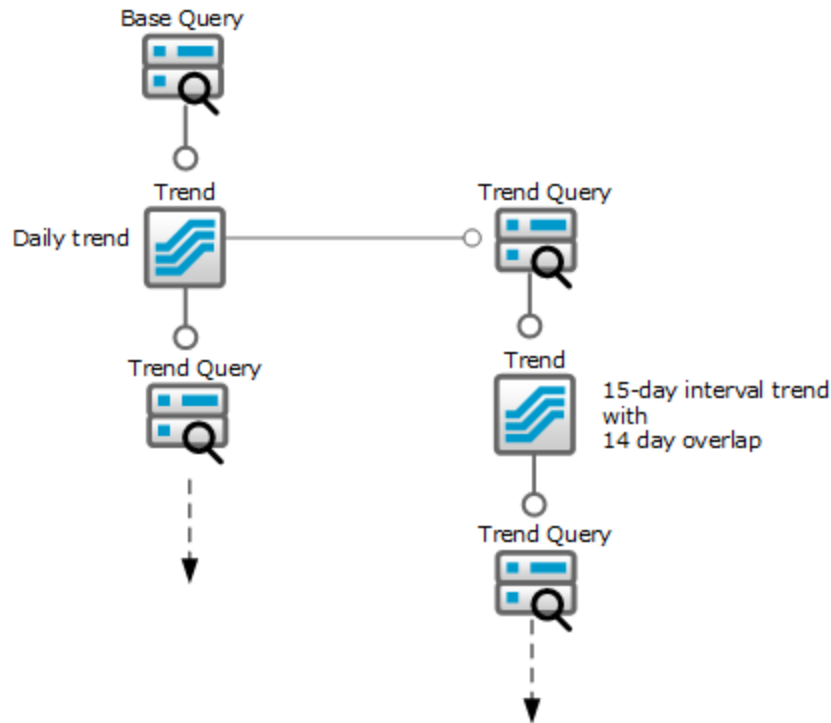
For you to get the desired information from this trend, you would wait for the completion of the 15th day. This is not the ideal way to go. Because the goal is to capture a moving average in reports and rules, we want to build in an overlap to make sure we don't have incomplete data.

T1 will use the base query defined above and build upon the data.

Trend2-on-Trend1: An Improvement

To avoid unnecessary queries off of our 15-day trend T1, we will create another trend, T2, that queries on T1. T2 will be a 15-day interval trend with a 14 day overlap. The reason for the overlap is to ensure that you don't lose a day's data as you go. Everyday, you can get data on the last 15 days. Without the overlap, you have to wait till the end of the 15th day to get 15 days worth of data.

T2 will provide data for the moving average report.



Query (Q2) for Trend (T2)

The purpose of the second level of query-on-trend is to perform calculations on data from T1 (which gets a count of all logins).

Q2 Attributes:

Query	
Name	Q2
Query On	Trend
Query On Resource	T1
Start Time	\$Now - 1d
End Time	\$Now
Use as Timestamp	TimeStamp

Columns for Q2:

```

SELECT
  AVG(Count(Event ID)) AVG
  Category Outcome
  Add 'SELECT' columns
GROUP BY
  Category Outcome
  Add 'GROUP BY' columns
ORDER BY
  Add 'ORDER BY' columns
    
```



Note: For this example, query conditions are not necessary, although you can use something like Category Outcome = /Success; or both /Success and /Fail, and so on.

T2 Setup

T2 Attributes:

Trend	
* Name	T2
* Query	Q2
* Enabled	<input checked="" type="checkbox"/>
Data Collection	
Trend Interval	15 day(s)
Row Limit	1000
Advanced	
Offset Collection By	1 day(s)
Refresh Data After	None
Partition Size	WEEKLY
Partition Retention Period (in days)	180
Query Overlap Time	14 day(s)
Imported Trend Start Time	

- **Schedule with care.** To make sure you are not missing data, schedule T2 after T1 completes a run.
- **T2 works off of data from T1.** If T2 is meant to get something like a **Top N** count, T1 data should get a count of all (considered as Top All). If T1 uses COUNT, T2 needs to do a SUM or AVG of the total count. If T1 uses SUM, T2 must also do a SUM or AVG.

Trend Data

A view of T1's trend data would be something like this on the Trend Details viewer:

T Details
Name: T1 85 shown / 85 matches
Start Time: 16 Jul 2014 00:00:00 PDT
End Time: 17 Jul 2014 16:31:52 PDT
Filter: No Filter

TimeStamp	Count(Event ID)	Category Outcome
12 Jul 2014 00:00:00 PDT	30771	/Success
12 Jul 2014 00:00:00 PDT	204	/Failure
11 Jul 2014 00:00:00 PDT	30101	/Success
11 Jul 2014 00:00:00 PDT	200	/Failure
10 Jul 2014 00:00:00 PDT	196	/Failure
10 Jul 2014 00:00:00 PDT	29420	/Success
9 Jul 2014 00:00:00 PDT	28914	/Success
9 Jul 2014 00:00:00 PDT	193	/Failure
8 Jul 2014 00:00:00 PDT	189	/Failure
8 Jul 2014 00:00:00 PDT	28300	/Success
7 Jul 2014 00:00:00 PDT	27606	/Success
7 Jul 2014 00:00:00 PDT	185	/Failure
6 Jul 2014 00:00:00 PDT	26943	/Success
6 Jul 2014 00:00:00 PDT	181	/Failure
5 Jul 2014 00:00:00 PDT	26582	/Success
5 Jul 2014 00:00:00 PDT	180	/Failure
4 Jul 2014 00:00:00 PDT	179	/Failure
4 Jul 2014 00:00:00 PDT	26242	/Success
3 Jul 2014 00:00:00 PDT	175	/Failure
3 Jul 2014 00:00:00 PDT	25580	/Success

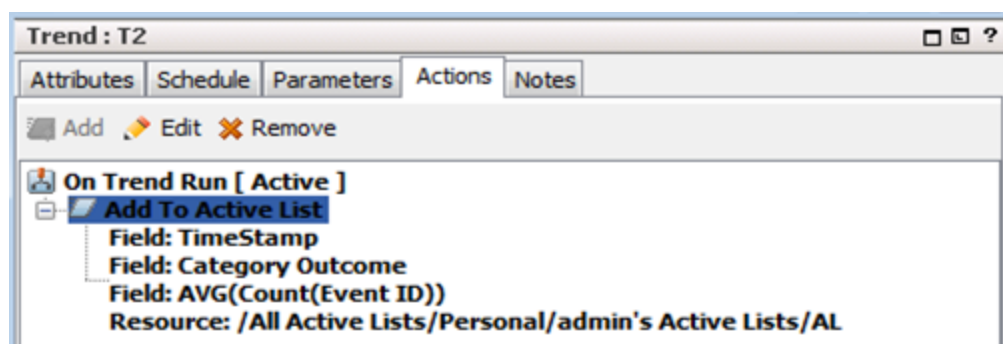
Trend Action: Add to Active List

A trend action consists of inserting new or updating existing entries into a **field-based** active list. Resources like rules, filters, reports, query viewers, and so on, make use of data from active lists. For example, a rule can use active lists that have been updated by trends.

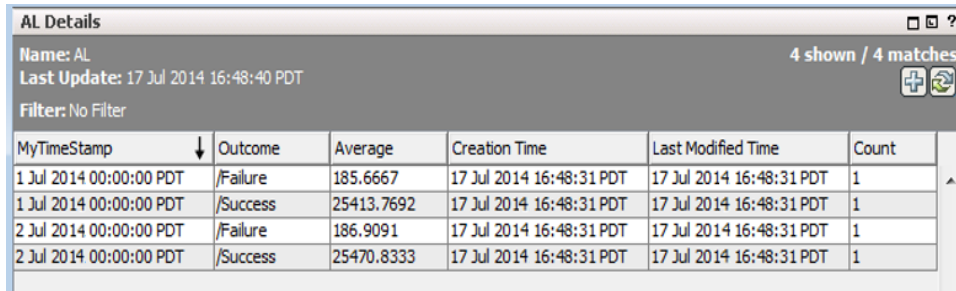
Requirements

- Make sure you have previously defined a field-based active list containing the fields in the query for the trend. Refer to the "Creating an Active List" topic in the *ArcSight Console Use's Guide* for instructions.
- Save the trend. The Actions tab is disabled until then.

Following is an example of T2's Actions tab:



Following is an example of how the resulting active list from T2's action will look like:



MyTimeStamp	Outcome	Average	Creation Time	Last Modified Time	Count
1 Jul 2014 00:00:00 PDT	/Failure	185.6667	17 Jul 2014 16:48:31 PDT	17 Jul 2014 16:48:31 PDT	1
1 Jul 2014 00:00:00 PDT	/Success	25413.7692	17 Jul 2014 16:48:31 PDT	17 Jul 2014 16:48:31 PDT	1
2 Jul 2014 00:00:00 PDT	/Failure	186.9091	17 Jul 2014 16:48:31 PDT	17 Jul 2014 16:48:31 PDT	1
2 Jul 2014 00:00:00 PDT	/Success	25470.8333	17 Jul 2014 16:48:31 PDT	17 Jul 2014 16:48:31 PDT	1

Tips and Best Practices for Trend Actions

- For each row returned from the query, the action either inserts a new entry into the active list or replaces specified values of an existing entry.
- The trend action cannot delete active list entries, unlike a rule's active list action. In this case, use the active list's Time-to-Live (TTL) option to age out entries; or overwrite to replace old entries. For example, based on T2's actions and resulting active list entries:
 - If the key is Outcome, use overwrite.
 - If the key is timestamp, use TTL
- The action only adds the newest data from the trend.
- A single trend can update multiple active lists, and multiple trends can update one active list. Use this technique to combine several queries into one table for reporting purposes.
- If you are building your active list for the trend to act on, know the data types of the data returned by the query and set up your active list entries accordingly. Understand the difference between integer, long, and double. For example, you cannot write a long value into an active list column of double type.

More Examples: Sharing Trend Data with Reports

You can apply different techniques to create similar reports with different data.

Multiple Reports, Same Report Columns, Different Filters

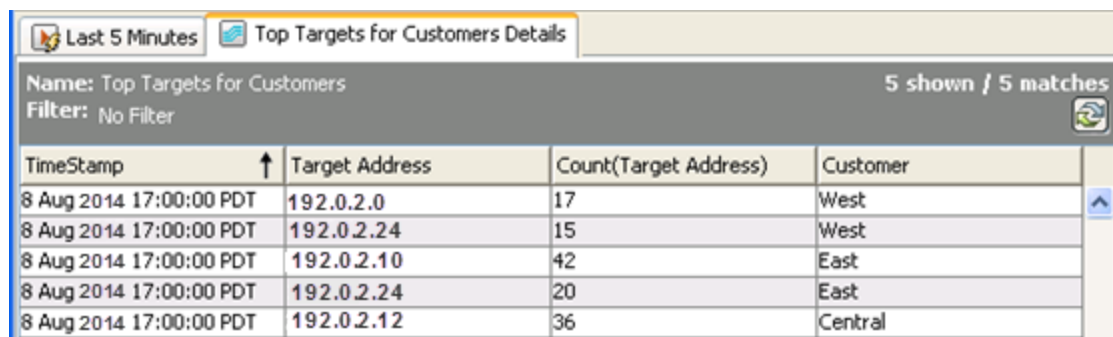
This technique can apply to MSSPs or different divisions within a company.

You are interested in creating similar reports for different situations:

- Report 1: Top target addresses for Customer1
- Report 2: Top target addresses for Customer2

The reports will use a trend query plus a filter on the distinguishing column, for example, Customer.

To achieve these goals, set up a single trend that will fetch the data to include the distinguishing column, for example, Customer. The data will look something like this:



The screenshot shows a web-based interface for a trend report. At the top, there are two tabs: "Last 5 Minutes" and "Top Targets for Customers Details". Below the tabs, the report title is "Name: Top Targets for Customers" and it indicates "5 shown / 5 matches". The filter is set to "No Filter". The main content is a table with the following columns: TimeStamp, Target Address, Count(Target Address), and Customer. The data rows are as follows:

TimeStamp	Target Address	Count(Target Address)	Customer
8 Aug 2014 17:00:00 PDT	192.0.2.0	17	West
8 Aug 2014 17:00:00 PDT	192.0.2.24	15	West
8 Aug 2014 17:00:00 PDT	192.0.2.10	42	East
8 Aug 2014 17:00:00 PDT	192.0.2.24	20	East
8 Aug 2014 17:00:00 PDT	192.0.2.12	36	Central

The reports will use a trend query plus a filter on the distinguishing column, for example, Report1 is about Customer = "West" and Report2, Customer ="East".

Multiple Reports with Similar Conditions

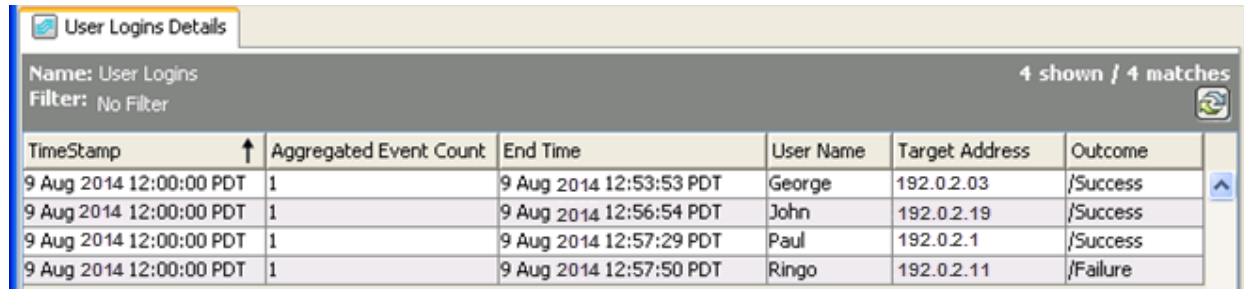
In this scenario, you are interested in these reports:

- Report 1: Daily count of user login activities (attempts)
- Report 2: Daily count of user login failures

To achieve these goals, you will set up a single trend that has:

- A table of all login attempts
- Add column to trend for login successes and failures (/Success, /Failure)

The data will look something like this:



The screenshot shows a table titled 'User Logins Details' with the following data:

TimeStamp	Aggregated Event Count	End Time	User Name	Target Address	Outcome
9 Aug 2014 12:00:00 PDT	1	9 Aug 2014 12:53:53 PDT	George	192.0.2.03	/Success
9 Aug 2014 12:00:00 PDT	1	9 Aug 2014 12:56:54 PDT	John	192.0.2.19	/Success
9 Aug 2014 12:00:00 PDT	1	9 Aug 2014 12:57:29 PDT	Paul	192.0.2.1	/Success
9 Aug 2014 12:00:00 PDT	1	9 Aug 2014 12:57:50 PDT	Ringo	192.0.2.11	/Failure

For Report 1, you will use trend data directly and ignore login outcome.

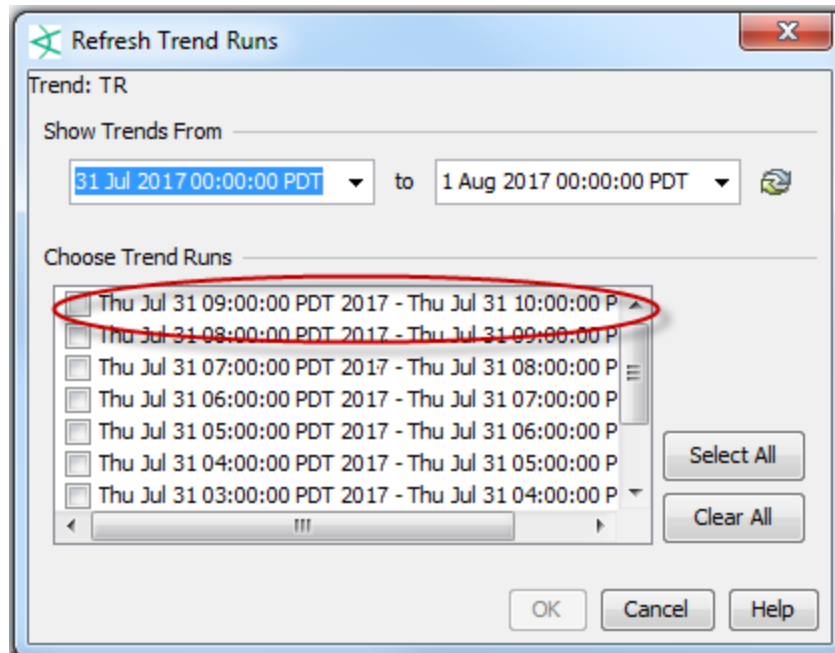
For Report 2, you will use the trend data, then filter on outcome="/Failure".

Debugging Your Trend: Where to Look

As your trends run over time, you will begin to observe some performance issues. For example, trends are taking too long, you are not getting the expected data, or the expected action was not executed. This topic describes areas that you can monitor. From the data, investigate further. Then you can decide if you need to fine tune any trend settings.

- Open the dashboard for trend status (/All Dashboards/ArcSight Administration/ESM/SystemHealth/Resources/Reporting/Trends Details). This dashboard provides information on longest running trend queries, trend queries that are running but not returning any results, any trend queries that are currently running, and any trend queries that failed in the last 24 hours.
- Open the channel for trend status (/All Active Channels/ArcSight Administration/ESM/System Health/Resources/Trends Status). Look for Device Event Class ID = trend:102 that indicates a trend run failure.
- Run system health reports related to trends (/All Reports/ArcSight Administration/ESM/System Health/Resources/Reporting/Longest Trend Query). Note that this report does not provide information about why the trend is not returning data.
- If the trend cannot collect any data, look at your schedule range and the start time of the first interval to see if the range covers the time when you expect data.
- Check the query interval and the scheduled run time. The Scheduler may not have run the trend, or the trend has not yet finished.
- To see what has been already run on interval trends, click the **Refresh Trend Runs** button at the bottom of the interval trend's Trend Editor panel (make sure to click **Cancel** when you're done). The Choose Trend Runs section on the panel should provide a list of successful trend runs, as in the following example. If the trend did not run at all, you will

instead see No matching runs found.



- View the trend's data through the Data Viewer. Do you see the data you expect in your fields? How old is the most recent data? Are you getting the number of rows you expect? Are you missing certain field values where you expect the values to be populated? How about the number of rows - is it correct, or close to what you expect? Or is the number of rows way off?
- Try to use the trend's query in a Query Viewer (bypass the trend). If the Query Viewer does not return data, the trend does not.
- Check the logs for exceptions on the trend (look for exceptions on the trend's resource ID) and exceptions on the base queries.

Send Documentation Feedback

If you have comments about this document, you can [contact the documentation team](#) by email. If an email client is configured on this computer, click the link above and an email window opens with the following information in the subject line:

Feedback on ESM Best Practices: Trends (ESM 7.6)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to Documentation-Feedback@microfocus.com.

We appreciate your feedback!