Handling data delays in Splunk InfraMon

Metrics that are sent to Splunk Infrastructure Monitoring can sometimes be delayed. In other words, the time at which an actual measurement was taken may differ significantly - on the order of minutes - from the time when the data actually arrives at Splunk Infrastructure Monitoring. The reasons for delay are many and varied: there could be network congestion; the host from which the metric is being sent may be busy; is dependent on calculating the process or agent responsible for sending data may be improperly configured; or the API you access to gather the data may provide you with data in batches every 5 minutes, even though it takes measurements every 10 seconds.

The lack of timeliness has an impact on Splunk Infrastructure Monitoring detectors, as they are optimized to make use of streaming real-time data, sent with monotonically increasing timestamps. Consistently delayed data affects how quickly a detector can ‘react’ to the data, but has no impact on accuracy. Inconsistently delayed data, on the other hand, may impact the accuracy of detector computations.

For example, let’s say your detector condition is dependent on calculating the average value of CPU utilization from 10 servers, and the metrics from all of the servers consistently are 15 seconds ‘behind’. Because knows about that delay, it knows to calculate the average value after waiting 15 seconds for the data to arrive. Your alerts might only fire 15 seconds after the event, but they are accurately firing based on the average CPU utilization values for all 10 servers.

Let’s further say that one of the servers begins to arrive a full minute later than is expected, for a total delay of 1 minute and 15 seconds. In that case, the detector needs to decide whether it should wait that extra minute to calculate that average (i.e. wait for all the data to arrive), or if it should perform its calculation with all the data it has at a given point in time (i.e. potentially without the data from the 10th ‘late’ server).

It would be better to not have to make the choice: timely data yields timely and accurate alerts. For that reason, if some of the causes of delay (especially those that are volatile) are within your control, you should find ways to address them. Additionally, for the situations where the data is delayed and not easily correctable, you need to make sure that your detectors still perform with the level of accuracy that you need.

Configure your inputs to send data points in a timely manner

The amount of time between the logical time (the timestamp that accompanies the datapoint, i.e. when the measurement is actually taken) and wall time (the time at which the datapoint actually arrives at Splunk APM) should be as low as possible. That delta should be less than the interval between successive datapoints.

Use Network Time Protocol (NTP) to synchronize clocks

Clock skew causes delays in data transmission. Using NTP to synchronize the time of your clients or servers to another server or within a few milliseconds of Coordinated Universal Time (UTC) eliminates this potential delay.
Set the MaxDelay parameter

Every detector in Splunk Infrastructure Monitoring includes a parameter, MaxDelay, that sets a deadline for incoming data to be included in its evaluation set, and is expressed as a number of minutes or seconds up to a maximum value of 15 minutes. A detector waits up to the MaxDelay amount of time for expected datapoints to arrive; if they arrive after that, they will not be considered, even though they are persisted in the backend datastore and available for use by other charts or detectors.

MaxDelay is set automatically and dynamically based on observed latencies of incoming datapoints, but it can also be set manually to a fixed value if you have a good understanding of how your data arrives at Splunk Infrastructure Monitoring.

It is generally a good idea to leave MaxDelay in "auto" mode. However, if emitter lag is erratic, even dynamic MaxDelay may end up excluding some data. Specifying an explicit MaxDelay defines an upper limit to lag, which is useful if chart or detector timeliness is critical and you don't want the computation to lag by more than a specified limit.

Configure an extrapolation policy

If a datapoint is sufficiently delayed as to be excluded because it has exceeded MaxDelay, or if the datapoint does not exist because a measurement was never made or reported, then it will be considered a Null value for the purpose of the detector’s computations. Depending on the nature of your data, it may be appropriate to configure an extrapolation policy to compensate for potential Null values.

An extrapolation policy creates synthetic data points for missing data. The specific policy should be chosen to complement the metric and rollup type to lag, which is a counter metric with a sum rollup is probably best served with a zero extrapolation, whereas a last value extrapolation might be better for a gauge with a mean rollup.

The Max Extrapolations parameter indicates the number of consecutive data points for which the selected policy will apply. By default, extrapolation is set for infinity, but there are cases where this is inappropriate. For example, if your detector makes use of the count analytics function, then you will want to limit the number of times an extrapolated value is used, as it can make if the result inaccurate.

How can this affect your data? For example, a count analytics function on a metric can be used to create host up or down detectors. If the extrapolation policy is set to Zero or Last Value the synthetic value being created may hide the host being up or down leading to an inaccurate detector.

Note that a last value extrapolation will not yield a synthetic value prior to the first real value, and may therefore not synthesize any values for inactive or dead time series. In other words, a time series that never reports data cannot be made to report a value using extrapolation.

In the table below, we see a metric that reports values every 5 seconds, and it skips 2 intervals. The values that are output after applying different extrapolation policies are as follows:
<table>
<thead>
<tr>
<th>Time $t$ =</th>
<th>10:01:05 AM</th>
<th>10:01:10 AM</th>
<th>10:01:15 AM</th>
<th>10:01:20 AM</th>
<th>10:01:25 AM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Received datapoint value</td>
<td>10</td>
<td>15</td>
<td>null</td>
<td>null</td>
<td>5</td>
</tr>
<tr>
<td>Null extrapolation</td>
<td>10</td>
<td>15</td>
<td>null</td>
<td>null</td>
<td>5</td>
</tr>
<tr>
<td>Zero extrapolation</td>
<td>10</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Last value extrapolation</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>Linear extrapolation</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>5</td>
</tr>
</tbody>
</table>

You can see that:

- Null extrapolation policy does not alter the values at all.
- Zero extrapolation policy inserts zeros in place of null values. This is often used for counters that only report when there is a value, and where a null value is properly interpreted as a zero.
- Last value extrapolation policy uses the last value it received. This is most often used for cumulative counters and gauges, where a null value is usually interpreted as no change in value.
- Linear extrapolation policy uses the last two data points received to determine what the following values would have been.

**Next steps**

These additional Splunk resources might help you understand and implement these recommendations:

- Splunk Docs: [Detector options: Max delay](#)