opEvents - Solution Guide - Event Consolidation Based on Location

Purpose

Demonstrate the practical application of event consolidation based on location, the principles here could be applied to other shared properties of events and nodes, like Business Service, Application, Customer, Group, Interface, and many others. Combinations can also be made, especially useful for nodes in Data Centres.

Related Pages

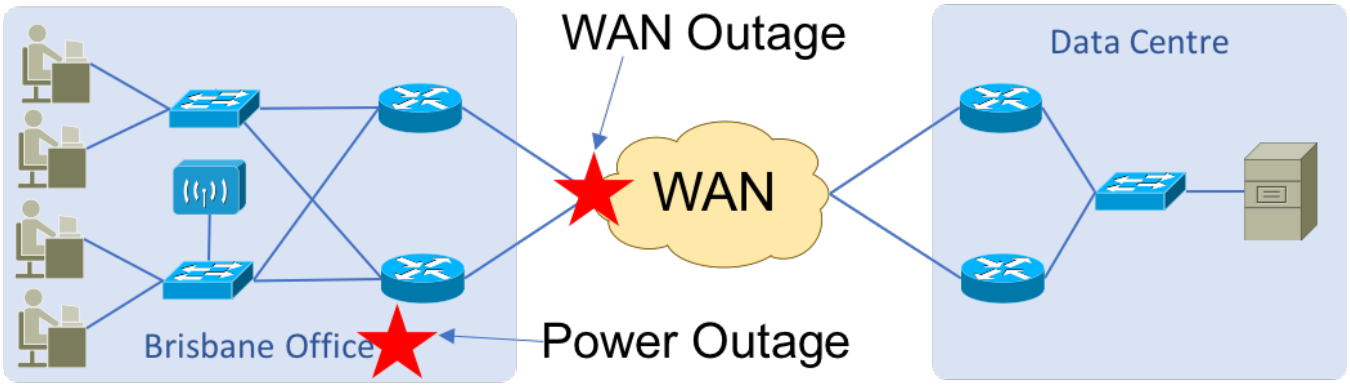
Event Correlation - Highly recommended, a must read!

Deduplication and storm control in opEvents

Scenario

Tasked with managing a large network that is either geographically seperated or the topology is such that 'fault domains' are easy to recognize, we would like to consolidate events to prevent network management noise and reduce troubleshooting time. With this in mind it would be desirable to have a single alert that notifies us that site "X" is experiencing a problem, versus many (10 ~ 500+) alerts from individual nodes. This not only cuts down on the noise, it also automates a component of the troubleshooting process, enabling operations to vector in on a common symptom in order to crush the problem.

A simple example would be a remote office, with several managed nodes, any problem with the WAN or with power would result in many events being seen in opEvents, enabling this feature would reduce that to a single event.



What is a synthetic event?

For our example this single event that consolidates many events is referred to as a 'synthetic event'. To further define a synthetic event we can say that it is not created from a log file or API input, it is derived from other existing events.

Prerequisites

There needs to be a common way to identify nodes such as location, group, business service, etc. The common attribute is assigned when the node is provisioned in NMIS. For example, if it's determined that all the nodes at the San Jose data center can be grouped into a single fault domain, then they should all have the same location attribute of 'San_Jose_Data_Center'. This gives opEvents something to grip onto for correlation.

Configuration

Defining Synthetic Events

The configuration of this feature is done in EventRules.nmis which is found in /usr/local/omk/conf. Here is an example of event consolidation based on location. It is recommended to create synthetic events in 'matched pairs' if possible, this can provide the synthetic with a dimension of 'state'.

/usr/local/omk/conf/EventRules.nmis

```
%hash = (
        'rules' => {
                '1' => {
                    name => 'Location Outage',
                    events => ["Node Down","Service Down","SNMP Down"],
                    window => 30,
                    inhibit => 30,
                    count => 5,
                    groupby => ['node.location'], # this time count location \cross group independently
                    enrich => {priority=>8, stateful => "Location Outage", state => "down"},
                    copy_first => [ qr// ],
                    #delayedaction => 70, # optional action delay delay, set for the underlying events
                    autoacknowledge => 1,
                },
                '2' => {
                    name => 'Location Up',
                    events => ["Node Up","Service Up","SNMP Up"],
                    window => 30,
                    inhibit => 30,
                    count => 5,
                    groupby => ['node.location'], # this time count location \cross group independently
                    enrich => {priority=>2, stateful => "Location Outage", state => "up"},
                    #delayedaction => 70, # optional action delay delay, set for the underlying events
                    copy_first => [ qr// ],
                    autoacknowledge => 1,
                },
```

Rule Attributes

Name

This is the event name that will appear if the event consolidation rule is triggered.

Events

Event names that should be considered for consolidation. This is where were we define what specific events should be part of the fault domain. For something network centric 'Node Up/Down' is a logical choice.

Window

This value is in seconds. This defines a window of time in which the conditions must be met in order for the rule to fire.

Inhibit

This value is in seconds. Once all conditions of the rules match, the rule fires. The inhibit value is the time in seconds that must elapse before the rule is eligible to be fired again. This prevents excessive 'synthetic events' in the event of a major outage. Matching individual node events that are received during this inhibit period will be associated with the previously fired synthetic event and shall be suppressed.

Count

This value represents the minimum number of matching events that must be received in order for the rule to fire.

Groupby

This along with the 'Events' defined above makes the hash for consolidation. In this example we are focusing on location, so the value will be node.location. This location value is assigned when the node is provisioned in NMIS, and important field to populate in order for this feature to work properly. Please reference the related links above in order to better understand all the functionality provided by this value.

Enrich

Here we can set specific event properties for the synthetic event. If the property is not set the synthetic event will inherit the properties based on the 'copy_first, copy_last, copy_highest, copy_present, or copy_groupby directives. In this example we are using 'copy_first'.

Delayedaction

This example does not utilize this attribute. Please refer to the related links above to discover how this can be useful in preventing the 'water hammer' effect.

Copy_first

In this example we are stating that all attributes of the first matching rule will be cloned into the synthetic event unless the enrich directive states otherwise. Please refer to the related links for more information about the 'copy_*' options.

Autoacknowledge

This value relates to the matching individual node event. If this value is 'true' or '1', then it will automatically acknowledge the matching individual event causing processing to stop for it. If autoacknowledge is not set in this manner then the individual matching events will not be suppressed.