Nagios and SEC
a happy (re-)union for advanced system monitoring

John Rouillard
Renesys Corporation

Presented at LISA 2006
December 8, 2006

Nagios

- A service monitoring program
- Plugins probe a device and assign severity to gathered data
- Implements basic service correlation based on severity or severity changes

Correlations include

- Topology (parent/child)
- Thresholds (count continuous errors only)
- Service (if other service in error don't poll/notify)
- Cluster (X of Y services must be ok)
- Flap detection (cycling up/down, problematic to implement)

Simple Event Correlator (SEC)

- Provide many types of correlations. Usually used in security event/log analysis monitoring.
- Can provide all of Nagios correlations except topology.
- When used with Nagios it adds:
- Counting ok states before rearming
- Different threshold triggers or polling interval on analysis of error not just non-ok severity.
- Changing trouble thresholds per time period

More SEC use cases

- Automatically disable notifications
- Implement acknowledgments that expire after some period of time.
- Max check attempts can change depending on the error.
- Automatic load redistribution to even load on polled clients
- Automatically schedule downtime

Time Based Correlation

- Nagios does support time periods, but in order to have different critical/warning levels at different times requires differently named service definitions.
- Integration with SEC allows different threshold levels based on time periods without having to create new service definitions.

Event Mapping

- SEC acts as an event mapping layer
  - Plugin talks to device and gathers data
  - SEC further analyzes data, determines proper severity, performs additional correlation, modifies plugin output
  - SEC notifies Nagios via passive check for notification, display

Implementation

- Uses Nagios 2.5 event broker
- Adds a callback to the core that allows event broker module to override the plugin assigned severity, replacing with current severity for that service.
- Module can write a Nagios PROCESS_SERVICE_CHECK_RESULT to a file for the external correlator (e.g. SEC) to analyze.
- Two new service object parameters *ec_active_action* and *ec_passive_action* control operating mode for each severity.

Operating modes

- The module operates in one of three modes:

1. Pass polled event to Nagios (no external correlation, current operation)
2. Pass unmodified polled event to Nagios and a copy to SEC
3. Pass modified polled event to Nagios changing severity to current severity and pass a copy to SEC

Current Testing

- Running with 1460 services. SEC uses 0.4% of cpu for processing 9 events/second. 11% of the events are sent back into Nagios. RTT (Nagios -> SEC -> Nagios) 2-10 seconds.
- SEC configuration has 8 rules that split apart every event line. Has 2 event mapping rules to provide more useful output messages and 1 rescheduling rule to distribute polls in time
- SEC also monitors Nagios log file for errors in addition to handling active event stream.

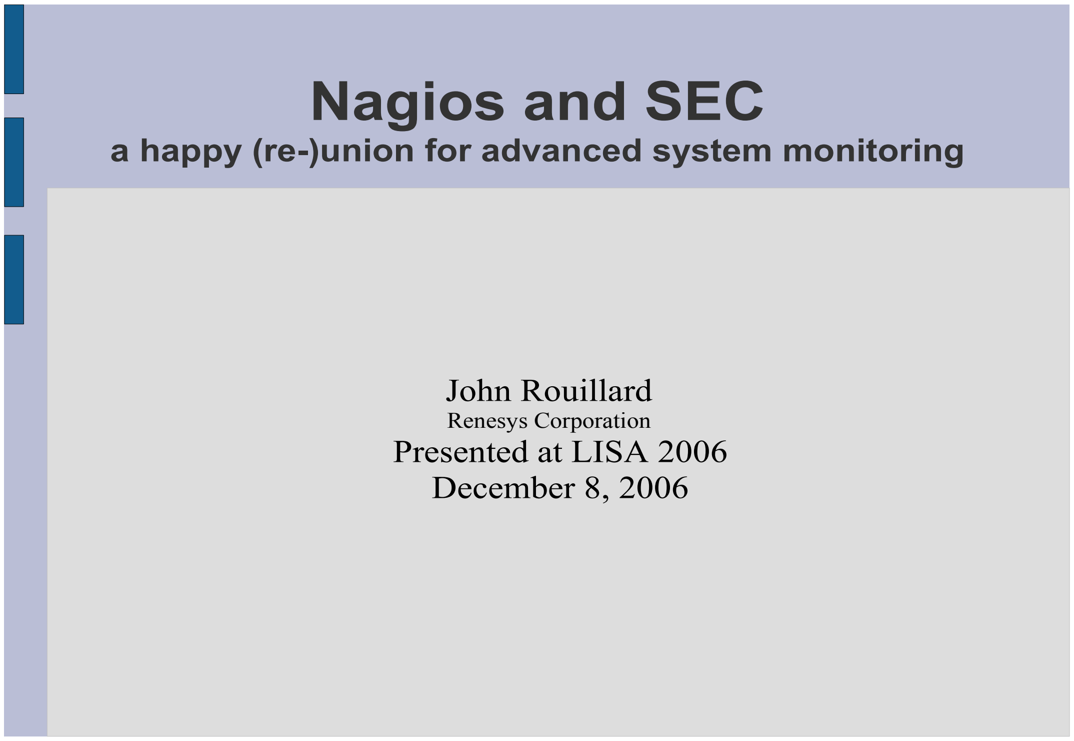
The Future

- **Need Beta Testers -** Beta release in early January 2007
- Stop losing events when writing to a FIFO.
- Command line parser to allow user to set items such as output mode at module load.
- Finalize annotation format for events
- Have module receive other data from Nagios (e.g. acknowledgments, host events)
- More stress testing
- Add a 4th operational mode to stop Nagios processing of active check result.

Questions

- Left as an exercise for the viewer.

- Presented by:
  John Rouillard (Renesys Corp)
  - rouilj@renesys.com
  - rouilj+ns@cs.umb.edu



Nagios and SEC

a happy (re-)union for advanced system monitoring

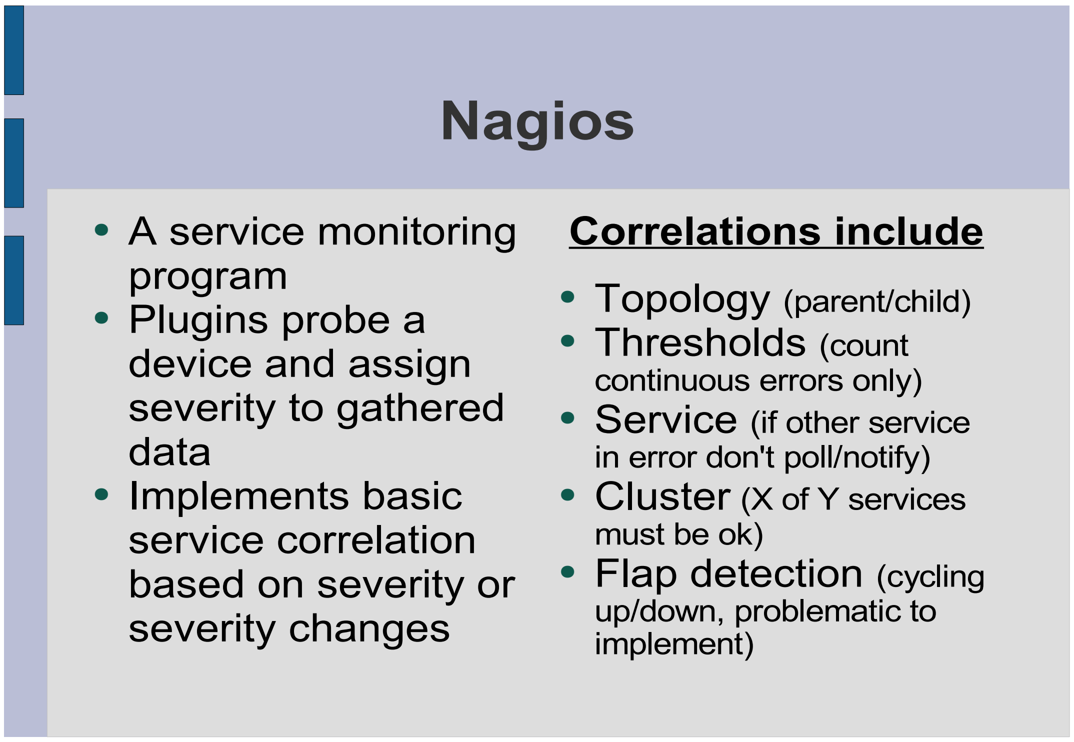
John Rouillard
Renesys Corporation

Presented at LISA 2006
December 8, 2006

This work was supported by Renesys Corporation <http://www.renesys.com>.

John Rouillard is a senior systems administrator for Renesys Corporation.

This talk discusses using SEC to correlate/analyze actively gathered information in nagios. Using SEC to analyze passivly gathered data and sending into nagios is currently easily done.



Nagios

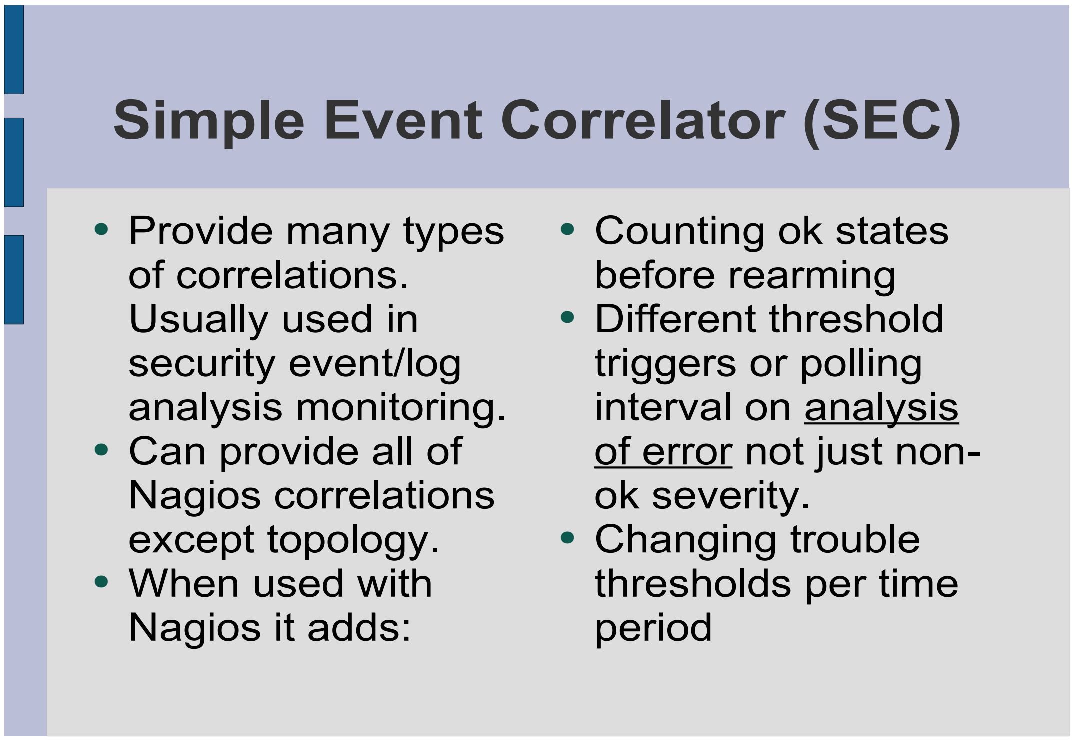
- A service monitoring program
- Plugins probe a device and assign severity to gathered data
- Implements basic service correlation based on severity or severity changes

Correlations include

- Topology (parent/child)
- Thresholds (count continuous errors only)
- Service (if other service in error don't poll/notify)
- Cluster (X of Y services must be ok)
- Flap detection (cycling up/down, problematic to implement)

I have had a lot of issues configuring flap detection to work as I wanted. What I really wanted was something similar to HPOV NNM rearm capability. X number of successive ok events before determining that the service is ok.

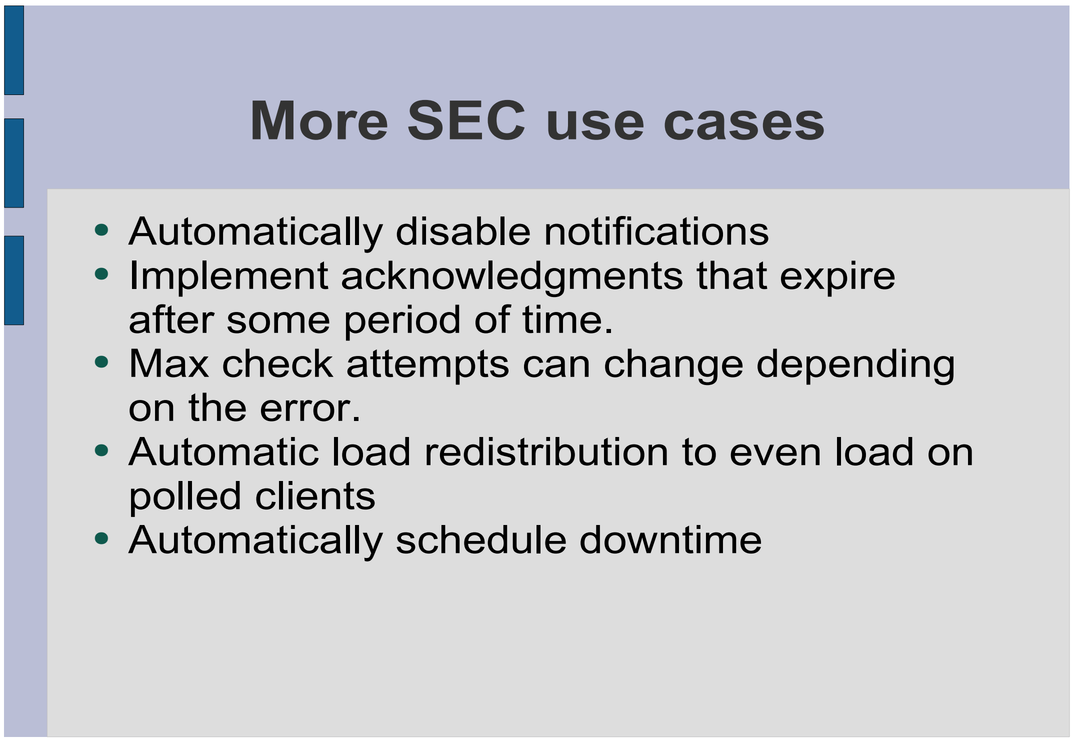
For non-polled host flap detection it just doesn't work as there never seem to be enough OK states to get it to reset.



Simple Event Correlator (SEC)

- Provide many types of correlations. Usually used in security event/log analysis monitoring.
- Can provide all of Nagios correlations except topology.
- When used with Nagios it adds:
- Counting ok states before rearming
- Different threshold triggers or polling interval on <u>analysis of error</u> not just non-ok severity.
- Changing trouble thresholds per time period

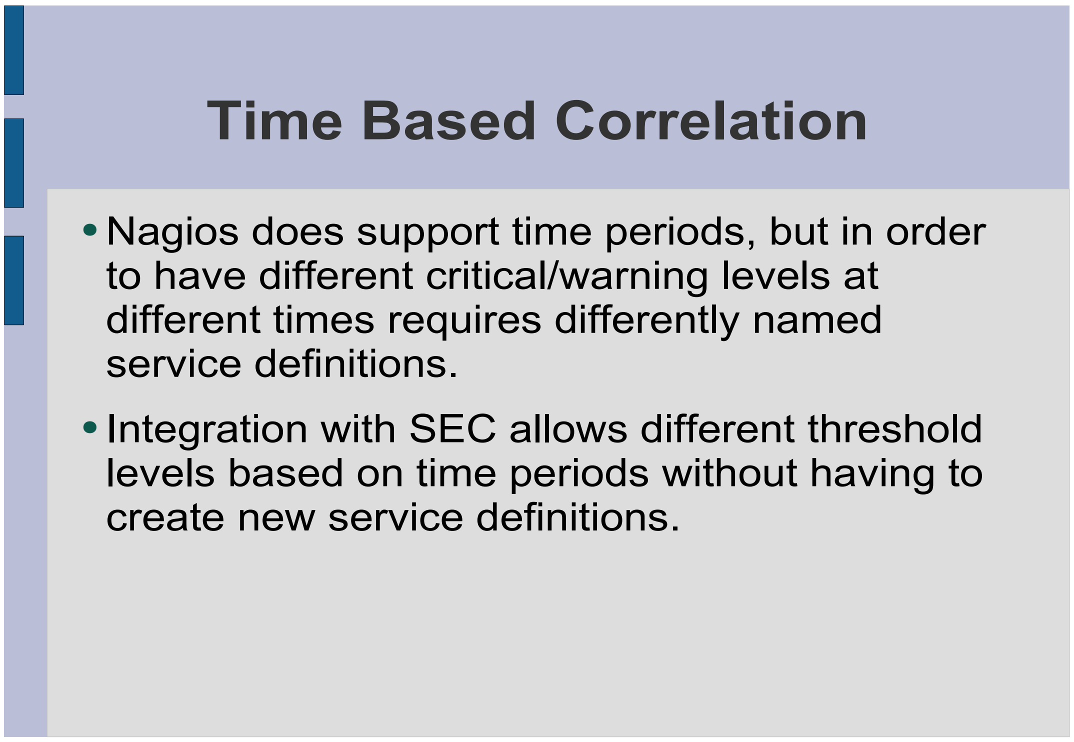
Counting consecutive ok states is equivalent to HPOV's NNM threshold rearm parameter. Can also count ok states within a window (as nagios does it).



More SEC use cases

- Automatically disable notifications
- Implement acknowledgments that expire after some period of time.
- Max check attempts can change depending on the error.
- Automatic load redistribution to even load on polled clients
- Automatically schedule downtime

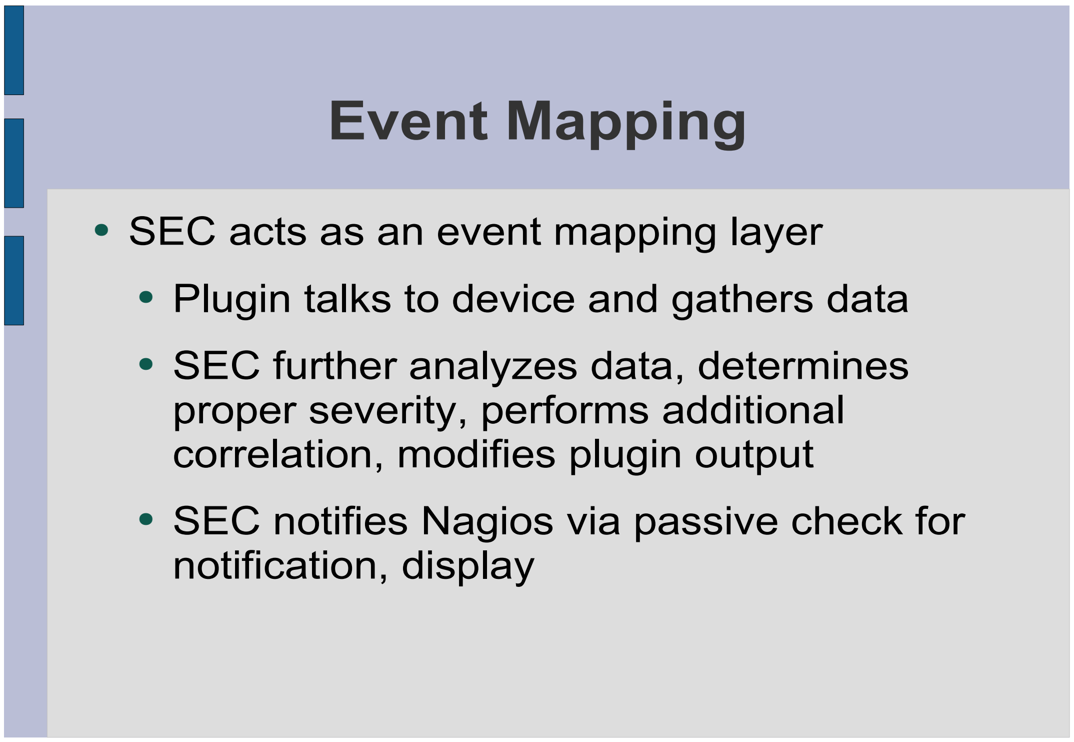
Not used in presentation.



Time Based Correlation

- Nagios does support time periods, but in order to have different critical/warning levels at different times requires differently named service definitions.
- Integration with SEC allows different threshold levels based on time periods without having to create new service definitions.

Not used in presentation. Currently people are told to schedule downtime for a service if there is an interval in which it is known to exceed the normal monitoring thresholds. The problem is that I may have 2 cron processes while backups are running, but if I have 10 during that interval I will never be notified because the service is in downtime.



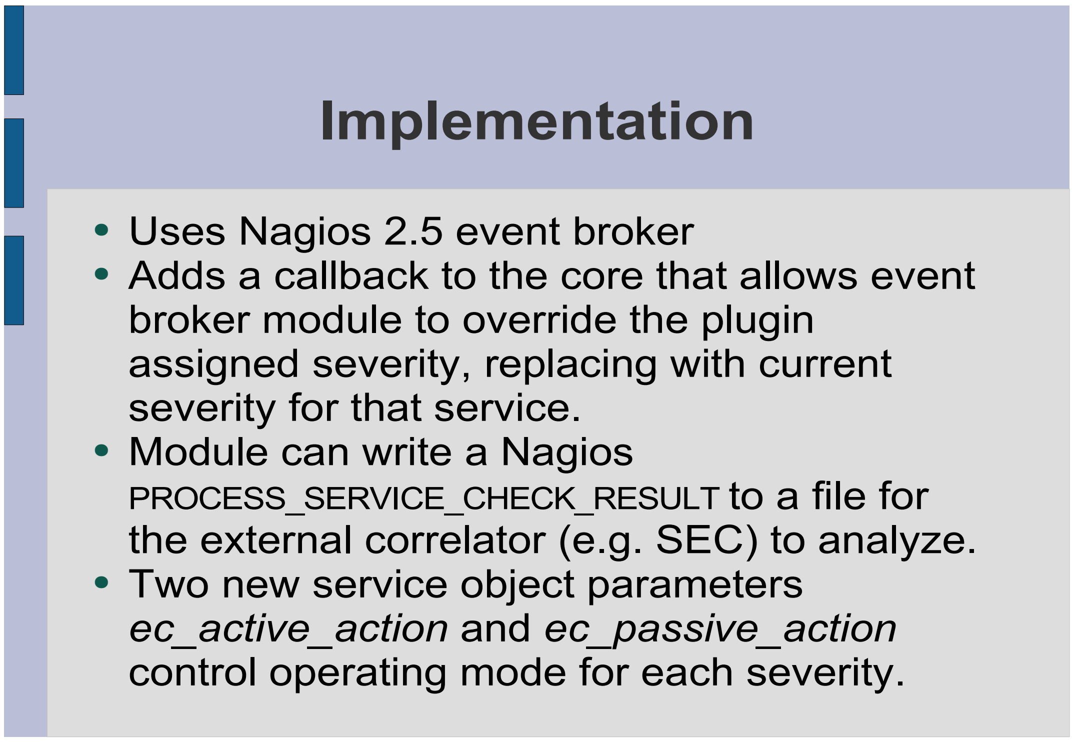
Event Mapping

- SEC acts as an event mapping layer
  - Plugin talks to device and gathers data
  - SEC further analyzes data, determines proper severity, performs additional correlation, modifies plugin output
  - SEC notifies Nagios via passive check for notification, display

This heads towards my personal multi-step setup for monitoring:

1 data gathering via plugin
2 analysis of data
3 correlation between devices
4 display/notification

currently plugins provide both 1 and 2 while nagios supplies 3/4. SEC works at the 2/3 level. IMO these are seperate levels tnad should be able to be handled independently.

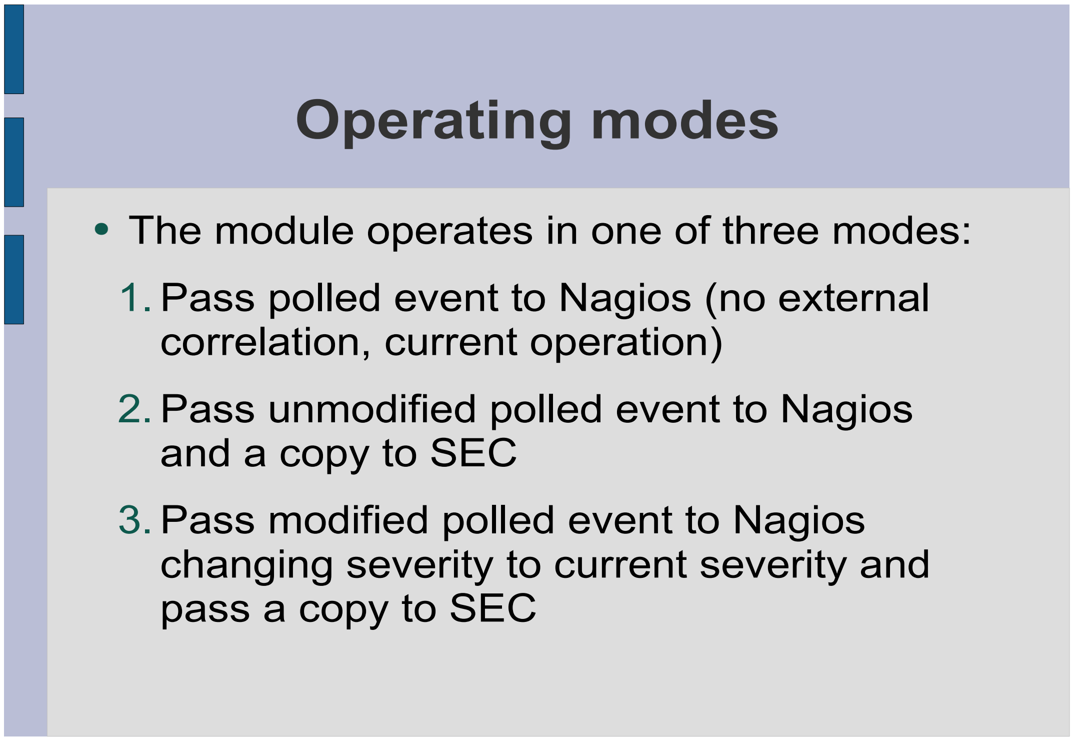


Implementation

- Uses Nagios 2.5 event broker
- Adds a callback to the core that allows event broker module to override the plugin assigned severity, replacing with current severity for that service.
- Module can write a Nagios PROCESS_SERVICE_CHECK_RESULT to a file for the external correlator (e.g. SEC) to analyze.
- Two new service object parameters *ec_active_action* and *ec_passive_action* control operating mode for each severity.

Note that the PROCESS_SERVICE_CHECK_RESULT may be annotated with additional information such as:

prior severity info
process mode for the service



Operating modes

- The module operates in one of three modes:
  1. Pass polled event to Nagios (no external correlation, current operation)
  2. Pass unmodified polled event to Nagios and a copy to SEC
  3. Pass modified polled event to Nagios changing severity to current severity and pass a copy to SEC

I want to add a fourth mode where Nagios forgets about the current active polled data and updates only when SEC sends in a passive check result.

Current Testing

- Running with 1460 services. SEC uses 0.4% of cpu for processing 9 events/second. 11% of the events are sent back into Nagios. RTT (Nagios -> SEC -> Nagios) 2-10 seconds.
- SEC configuration has 8 rules that split apart every event line. Has 2 event mapping rules to provide more useful output messages and 1 rescheduling rule to distribute polls in time
- SEC also monitors Nagios log file for errors in addition to handling active event stream.

The Future

- **Need Beta Testers -** Beta release in early January 2007
- Stop losing events when writing to a FIFO.
- Command line parser to allow user to set items such as output mode at module load.
- Finalize annotation format for events
- Have module receive other data from Nagios (e.g. acknowledgments, host events)
- More stress testing
- Add a 4th operational mode to stop Nagios processing of active check result.

Questions

- Left as an exercise for the viewer.
- Presented by:
  John Rouillard (Renesys Corp)
  - rouilj@renesys.com
  - rouilj+ns@cs.umb.edu