Event Handlers

Introduction

Event handlers are optional system commands (scripts or executables) that are run whenever a host or check state change occurs.

An obvious use for event handlers is the ability for Shinken to proactively fix problems before anyone is notified. Some other uses for event handlers include:

- Restarting a failed check
- Entering a trouble ticket into a helpdesk system
- Logging event information to a database
- Cycling power on a host
- etc.

Note that cycling power on a host that is experiencing problems with an automated script should not be implemented lightly. Consider the consequences of this carefully before implementing automatic reboots.

When Are Event Handlers Executed?

Event handlers are executed when a check or host:

- Is in a SOFT problem state
- Initially goes into a HARD problem state
- Initially recovers from a SOFT or HARD problem state

SOFT and HARD states are described in detail here.

Event Handler Types

There are different types of optional event handlers that you can define to handle host and state changes:

- Host-specific event handlers
- check-specific event handlers

Event handlers offer functionality similar to notifications (launch some command) but are called each state change, soft or hard. This allows to call handler function and react to problems before Shinken raises a hard state and starts sending out notifications.

Individual hosts and checks can have their own event handler command that should be run to handle state changes. You can specify an event handler that should be run by using the "event_handler" directive in your host and check definitions.

Enabling Event Handlers

Event handlers can be enabled or disabled on a program-wide basis by using the enable_event_handlers in your main configuration file.

Host- and check-specific event handlers can be enabled or disabled by using the "event_handler_enabled" directive in your host and check definitions. Host- and check-specific event handlers will not be executed if the global enable_event_handlers option is disabled.

Event Handler Execution Order

As already mentioned, global host and check event handlers are executed immediately before host- or check-specific event handlers.

Event handlers are executed for HARD problem and recovery states immediately after notifications are sent out.

Writing Event Handler Commands

Event handler commands will likely be shell or perl scripts, but they can be any type of executable that can run from a command prompt. At a minimum, the scripts should take the following data as arguments:

For checks:

- $SERVICESTATE$
- $SERVICESTATETYPE$
- $SERVICEATTEMPT$

For Hosts:

- $HOSTSTATE$
- $HOSTSTATETYPE$
- $HOSTATTEMPT$

The scripts should examine the values of the arguments passed to it and take any necessary action based upon those values.

Permissions For Event Handler Commands

Event handler commands will normally execute with the same permissions as the user under which Shinken is running on your machine. This can present a problem if you want to write an event handler that restarts system services, as root privileges are generally required to do these sorts of tasks.

Ideally you should evaluate the types of event handlers you will be implementing and grant just enough permissions to the shinken user for executing the necessary system commands. You might want to try using sudo to accomplish this.