Notifications

Introduction

This chapter will explain exactly when and how host and check notifications are sent out, as well as who receives them.

Notification escalations are explained in another page.

When do notifications occur?

The decision to send out notifications is made in the check and host check logic. Host and check notifications occur in the following instances...

- When a hard state change occurs. More information on state types and hard state changes can be found on the state type page.
- When a host or check remains in a hard non-OK state and the time specified by the "notification_interval" option in the host or check definition has passed since the last notification was sent out (for that specified host or check).

Who gets notified?

Each host and check definition has a "contact groups" option, and "contacts" option, that specifies which contact will receive notifications for that particular host or check. Contact groups can contain one or more individual contacts.

When Shinken Enterprise sends out a host or check notification, it will notify each contact that is a member of any contact groups specified in the "contact groups" option of the check definition. Shinken Enterprise realizes that a contact may be a member of more than one contact group, so it removes duplicate contact notifications before it does anything. Contacts specified on host or check are also notified.

What filters must be passed in order for notifications to be sent?

Just because there is a need to send out a host or check notification doesn't mean that any contacts are going to get notified. There are several filters that potential notifications must pass before they are deemed worthy enough to be sent out. Even then, specific contacts may not be notified if their notification filters do not allow for the notification to be sent to them. Let's go into the filters that have to be passed in more detail...

Program-Wide Filter:

The first filter that notifications must pass is a test of whether or not notifications are enabled on a program-wide basis. This is initially determined by the :ref:`"enable_notifications" <configuration/configmain-advanced#enable_notifications>` directive in the main config file, but may be changed during runtime from the web interface. If notifications are disabled on a program-wide basis, no host or check notifications can be sent out - period. If they are enabled on a program-wide basis, there are still other tests that must be passed...

check and Host Filters:

The first filter for host or check notifications is a check to see if the host or check is in a period of scheduled downtime. If it is in a scheduled downtime, no one gets notified. If it isn't in a period of downtime, it get passed on to the next filter. As a side note, notifications for checks are cancelled if the host they're associated with is in a period of scheduled downtime.

The second filter for host or check notification is a check to see if the host or check is flapping (if you enabled flap detection). If the check or host is currently flapping, no one gets notified. Otherwise it gets passed to the next filter.

The third host or check filter that must be passed is the host- or check-specific notification options. Each check definition contains options that determine whether or not notifications can be sent out for warning states, critical states, and recoveries. Similarly, each host definition contains options that determine whether or not notifications can be sent out when the host goes down, becomes unreachable, or recovers. If the host or check notification does not pass these options, no one gets notified. If it does pass these options, the notification gets passed to the next filter...

Notifications about host or check recoveries are only sent out if a notification was sent out for the original problem. It doesn't make sense to get a recovery notification for something you never knew as a problem.

The fourth host or check filter that must be passed is the time period test. Each host and check definition has a "notification_period" option that specifies which time period contains valid notification times for the host or check. If the time that the notification is being made does not fall within a valid time range in the specified time period, no one gets contacted. If it falls within a valid time range, the notification gets passed to the next filter...

If the time period filter is not passed, Shinken Enterprise will reschedule the next notification for the host or check (if its in a non-OK state) for the next valid time present in the time period. This helps ensure that contacts are notified of problems as soon as possible when the next valid time in time period arrives.

The last set of host or check filters is conditional upon two things: (1) a notification was already sent out about a problem with the host or check at some point in the past and (2) the host or check has remained in the same non-OK state that it was when the last notification went out. If these two criteria are met, then Shinken Enterprise will check and make sure the time that has passed since the last notification went out either meets or exceeds the value specified by the "notification_interval" option in the host or check definition. If not enough time has passed since the last notification, no one gets contacted. If either enough time has passed since the last notification or the two criteria for this filter were not met, the notification will be sent out! Whether or not it actually is sent to individual contacts is up to another set of filters...

Contact Filters:

At this point the notification has passed the program mode filter and all host or check filters and Shinken Enterprise starts to notify all the people it should. Does this mean that each contact is going to receive the notification? No! Each contact has their own set of filters that the notification must pass before they receive it.

Contact filters are specific to each contact and do not affect whether or not other contacts receive notifications.

The first filter that must be passed for each contact are the notification options. Each contact definition contains options that determine whether or not check notifications can be sent out for warning states, critical states, and recoveries. Each contact definition also contains options that determine whether or not host notifications can be sent out when the host goes down, becomes unreachable, or recovers. If the host or check notification does not pass these options, the contact will not be notified. If it does pass these options, the notification gets passed to the next filter...

Notifications about host or check recoveries are only sent out if a notification was sent out for the original problem. It doesn't make sense to get a recovery notification for something you never knew was a problem.

The last filter that must be passed for each contact is the time period test. Each contact definition has a "notification_period" option that specifies which time period contains valid notification times for the contact. If the time that the notification is being made does not fall within a valid time range in the specified time period, the contact will not be notified. If it falls within a valid time range, the contact gets notified!

Notification Methods

You can have Shinken Enterprise notify you of problems and recoveries pretty much anyway you want: pager, cellphone, email, instant message, audio alert, electric shocker, etc. How notifications are sent depends on the notification commands that are defined in your object definition files.

There are a thousand different ways to do notifications and there are already a lot of packages out there that handle the dirty work, so why re-invent the wheel and limit yourself to a bike tire? Its much easier to let an external entity (i.e. a simple script or a full-blown messaging system) do the messy stuff. Some messaging packages that can handle notifications for pagers and cellphones are listed below in the resource section.

Notification Type Macro

When crafting your notification commands, you need to take into account what type of notification is occurring. The $NOTIFICATIONTYPE$ macro contains a string that identifies exactly that. The table below lists the possible values for the macro and their respective descriptions:

Value	Decription
PROBLEM	A check or host has just entered (or is still in) a problem state. If this is a check notification, it means the check is either in a WARNING, UNKNOWN or CRITICAL state. If this is a host notification, it means the host is in a DOWN or UNREACHABLE state.
RECOVERY	A check or host recovery has occurred. If this is a check notification, it means the check has just returned to an OK state. If it is a host notification, it means the host has just returned to an UP state.
ACKNOWLEDGEMENT	This notification is an acknowledgement notification for a host or check problem. Acknowledgement notifications are initiated via the web interface by contacts for the particular host or check.
FLAPPINGSTART	The host or check has just started flapping.
FLAPPINGSTOP	The host or check has just stopped flapping.
FLAPPINGDISABLED	The host or check has just stopped flapping because flap detection was disabled
DOWNTIMESTART	The host or check has just entered a period of scheduled downtime. Future notifications will be suppressed.
DOWNTIMESTOP	The host or check has just exited from a period of scheduled downtime. Notifications about problems can now resume.
DOWNTIMECANCELLED	The period of scheduled downtime for the host or check was just cancelled. Notifications about problems can now resume.