



## Advanced Topics

### 5.1 Smart Notification

Helix's Smart Notification was designed to eliminate sending multiple notifications when a device goes down or is unavailable. Often, many configured tests on a device have action profiles assigned to them to notify various recipients when test status reaches Warning, Critical, Unknown, or all three.

Smart notification relies on the inherent dependency between the ping packet loss test results and the availability of the device. If the ping packet loss test returns 100%, then communication with the device has somehow been lost. This could result in all tests sending notifications every test cycle, especially if they are configured to notify on Unknown, which is what the test results will likely be in this situation.

**To configure Smart Notification during new device configuration:**

1. Click on the **MANAGE** tab. You will be taken to the Manage Devices page.
2. Click on the **Create A Device** link. For a detailed description of device creation, see Section 9.1, "Managing Devices" on page 83.
3. Fill out the required information in the Create Device page.
4. Check the box labeled **Smart Notification**.

5. Proceed with test discovery by selecting the **Create Device** button.

**NOTE** You *MUST* configure a *Packet Loss* test for the device. If you don't, this feature will not work correctly.

If you are receiving notifications when the device is down you should check the following items:

- Confirm that you have in fact configured Smart Notification on the device by navigating to the Manage Devices page and selecting the **Update** link for the device; the **Smart Notification** box should be checked.
- Confirm that you have a Packet Loss test configured on the device by navigating to the Manage Devices page and selecting the **Tests** link for the device; the list of test should include the Packet Loss test.
- If both of the above are configured properly, the notifications that you have received are likely a result of having been queued up prior the Packet Loss returning 100%. That is, if tests are scheduled in the queue ahead of the Packet Loss test, they will be executed prior to the trigger that suppresses all further notifications when Packet Loss = 100%.

To avoid notifications in this case it is advisable that you change your action profiles to either 1) not notify on UNKNOWN; or 2) only notify after 2 or 3 test cycles have passed.

## 5.2 Device Dependency

In networked environments, switches, routers, etc. are often the physical gateways that provide access to other network devices. If critical “parent devices” are unavailable, monitoring may be impeded for devices that are accessed via the parents. To distinguish between devices that are genuinely in a **CRITICAL** state and those that are **UNREACHABLE** because of a problem with one or more parent devices, you can create device dependencies.

A device dependency is a parent-child relationship between monitored devices. A single parent can have multiple children, and a single child can have multiple parents. Device dependencies are cascading. If A is a child of B, and B is a child of

C, it is only necessary to configure A as a child of B and B as a child of C. Helix automatically recognizes the dependency between A and C.

If a device is tested and the result is CRITICAL (for all thresholds), UNKNOWN, or FAILED, some additional processing is used to determine if the device is reachable.

1. A current packet loss test is examined for the device. If such a test exists and packet loss is not 100%, the device is considered reachable.
2. If no packet loss test exists, all immediate parent devices are examined. If the device has no parents, it is considered reachable and the result of the test is the measured value. If all parents have a current packet loss test which was measured at 100%, the device is considered unreachable.
3. If no packet loss test exists for the parent, or no recent test result is found for an existing packet loss test, the child device is considered reachable and the result of the test is the measured value.

### **Dependency Restrictions**

Device dependencies must conform to these rules:

1. Circular dependency is not allowed. For example, if you set up the following dependencies:  
 Device A **depends on** Device B **depends on** Device C  
 You cannot configure Device C to depend on Device A.
2. Parent and child devices must belong to the same DGE Location.

#### **□ To configure device dependency:**

1. Create the parent device as described in Section 9.1, “Managing Devices” on page 83.
2. Create the child device as described in Section 9.1, “Managing Devices” on page 83.

3. Click `MANAGE | devices`.
4. On the Manage Devices page, find the device that will be the *child* device in the dependency and click **Update**.
5. On the Update Device page, click **Update Device Dependency**.
6. On the Update Device Dependency page, select the device or devices on which this child depends from the **Does Not Depend On** list, and then click **Done**. (If you return to the Device Dependency page you will see that the parent device(s) appear in the **Depends On** list).

**NOTE:** Device dependencies are cascading. If A is a child of B, and B is a child of C, it is only necessary to configure A as a child of B and B as a child of C. Helix automatically recognizes the dependency between A and C.

The next time the parent device has a **CRITICAL** ping/pl test result, the child device will have **UNREACHABLE** status.

## 5.3 External Help

External help provides Helix operators the ability to write support documentation specific to a Department, device or test and tie it directly to that same object via a **HELP** link in the Web UI. This way, less experienced system administrators can be provided with a first line of troubleshooting in the absence of live support. You can also enable actions (e.g., server restart) via the **HELP** links. This is a powerful option, as any number of files can be configured to work in this fashion, enabling a large number of background processes via the web app.

The perl script `displayTestHelp.pl` in the `utils` directory scans through `HELIX_HOME/plugin/help` for help text specific to a Department, device or test. This script expects one argument in the form:

```
<department_name> | <device_name> | <device_addr> |  
<test_type> | <test_subtype> | <test_name>
```

where `device_addr` can be fqdn or ip address. This has to match what was used for device creation. The field `test_name` should match the descriptive name that was displayed during test creation (or in test details page).

**NOTE** *The perl script converts everything (i.e. `acct_name` & `device_name`) to lowercase to avoid any case related problems when searching for the file. Therefore, the directories and subdirectories must be named in lowercase and formatted the same (i.e. spaces or special chars included) as the Department and device names.*

The script searches `HELIX_HOME/plugin/help` according to following algorithm:

1. Search for directory - `acct_name` ELSE `_default_user`, if found, cd into it.
2. Search for subdirectory - `device_name` ELSE `device_addr` ELSE `_default_device`, if found, cd into it.
3. Search for the files in the current directory in the following order: `<test_type>_<test_subtype>_<test_name>.{html,txt}` ELSE `<test_type>_<test_subtype>.{html,txt}` ELSE `<test_type>.{html,txt}` ELSE `default.{html,txt}`
4. Display the entire file on stdout (if text, then put HTML tags around the text).
5. If not found, display `NO FILE FOUND` on stdout in HTML format. The script prints out errors on stdout. The location of the script is specified in `web.xml` and it can basically be any script (or program). It is up to the target script to take the arguments and send back help text in the required format.

For example, to create a help file for device 'mail\_server' and a more specific one for the 'disk\_space', in Department 'local\_department':

```
cd HELIX_HOME/plugin/help
mkdir -p local_department/mail_server
mkdir -p local_department/_default_device
cd local_department/
```

```
vi _default_device/default_html  
vi mail_server/snmp_disk.txt  
vi mail_server/default.html
```

It is possible to use your own script, that (for example) connects to a database and retrieves escalation information based on specified criteria.