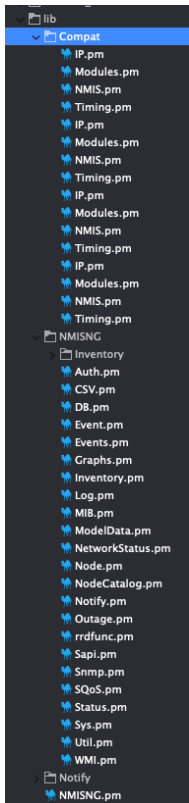


# NMIS 8 - Migrate plugins to NMIS 9

- [Internal libraries](#)
- [Database Access](#)
  - [Get the node](#)
  - [Get the node configuration](#)
  - [Get the node model](#)
  - [Get the node catchall](#)
  - [Get the node interfaces](#)
  - [Get the node inventory](#)
- [Example](#)
- [Related Documentation](#)

## Internal libraries

The internal libraries had been refactored.



The basic structure:

- lib/Compat: Legacy files to keep compatibility with some system functions.
- lib/NMISNG: All the domain entities, like Nodes, Events or inventory objects.
- lib/Notify: All the notification methods.
- lib/NMISNG.pm: The main module that have access to all the other objects and instantiates the DB access. Instantiate this object carefully: If we create an object without reusing it we could have connection leak issues.

Some replacements in NMIS 8:

- use NMIS; Replaced by use NMISNG;
- use func; Replaced by use NMISNG::Util;

Please, check the internal structure for other libraries.

## Database Access

As the new data backend has been moved from NMIS files to the mongo database, so the node information access is different.

Here you can find a guide about the main structures and how to replace them:

**Sys::ndinfo** has been replaced by **Sys::nmisng\_node** and **Node::nmisng\_node** can be accessed directly rather than accessing **Sys::nmisng\_node**

- **Sys::ndinfo->{system}** becomes hash returned by **Node::configuration()**
  - **Node::Configuration()->{roleType}** replaces **NMIS::loadLocalNodeTable()->{<a\_node\_name>} {role}**
  - Be warned that **Sys::ndinfo->{system}** key values often carry the strings "true" and "false" for boolean whereas **Node::configuration()** returns hash key values will carry 0 and 1 for boolean.
    - Therefore one should always use Common::getBool() to evaluate key values.
  - Furthermore, **Node::configuration()** returns a clone copy each call, therefore it is advisable to set a variable if re-used:
- **Sys::ndinfo->{status}** becomes **Node::get\_status\_model\_data()**
  - NMIS8 **ref(Sys::ndinfo->{status})** versus NMIS9 **ref(Node::get\_status\_model\_data)**
    - NMIS8 **Sys::ndinfo->{status}** was a hash of hashes, whereas
    - NMIS9 **Node::get\_status\_model\_data()** is an array of hashes.
  - NMIS8 **Sys::ndinfo->{status}->{source}** is not a hashkey in **Node::get\_status\_model\_data()**
    - NMIS8 **Sys::ndinfo->{status}** has entries either by 'source' or by 'index', whereas
    - NMIS9 **Node::get\_status\_model\_data()** only has entries by 'index'
  - NMIS8 **<object>->{<a\_node>->{active}}** should be replaced by
    - **Node::configuration{active}**
- **Sys::ndinfo->{sysUpTime}** is replaced by **Node::node\_summary->{sysUpTimeSec}**

## Get the node

Now it is possible to get the node object using the main object nmisng.

```
my $S = NMISNG::Sys->new(nmisng => $nmisng);
my $nodeobj = $nmisng->node(name => $node);
```

## Get the node configuration

The node configuration:

The screenshot shows a web-based configuration interface titled "Edit Node". The node name is "asgard". The interface is organized into several sections:

- Table Nodes:**
  - Name: asgard
  - New Name: (empty text box)
  - UUID: ed8e5402-aa5f-40f8-aa20-ab99b88c9493
  - Host Name/IP Address: asgard.opmantek.net
  - Fallback Host Name/IP Address: (empty text box)
  - IP Protocol: IPv4 (dropdown menu)
  - Group: 123 (dropdown menu)
  - SNMP Community: OMKread
- WMI Options:**
  - WMI Domain: (empty text box)
  - WMI Username: (empty text box)
  - WMI Password: (empty text box)
- Service Management Options:**
  - Customer: Opmantek (dropdown menu)
  - Business Service: Core Network, Web Page, blaclabla, eCommerce, eMail (list box)
  - Service Status: Development (dropdown menu)
- Name and URL for additional node information:**
  - Node Context Name: (empty text box)
  - Node Context URL: (empty text box)

And the node overrides:

Node Configuration Fri 10:46

Select node: **asgard** Optional Node and Interface Configuration

		Original value	Replaced by (active after update of node)
<b>Node</b>			<input type="button" value="Store"/> <input type="button" value="Store and Update Node"/>
	Contact	default	<input type="text"/>
	SNMP Location	Varsity Lakes, Gold Coast	<input type="text"/>
	Node Type	router	<from model> <input type="text"/>
<b>Interfaces</b>			
<b>Dialer1194</b>	Description	Demonstration of change detection	<input type="text"/>
	Display Name		<input type="text"/>
	Speed In	60000	<input type="text" value="60000"/>
	Speed Out	56000	<input type="text"/>
	Speed Limit	strict	<input type="radio"/> normal <input checked="" type="radio"/> strict <input type="radio"/> off
	Collect	false	<input checked="" type="radio"/> unchanged <input type="radio"/> true <input type="radio"/> false
<b>FastEthernet0/0</b>	Description	Opmantek LAN	<input type="text"/>
	Display Name	the loony lan	<input type="text" value="the loony lan"/>
	Speed In	100000000	<input type="text"/>
	Speed Out	100000000	<input type="text"/>
	Speed Limit	off	<input type="radio"/> normal <input type="radio"/> strict <input checked="" type="radio"/> off
	Collect	true	<input checked="" type="radio"/> unchanged <input type="radio"/> true <input type="radio"/> false
	Events	true	<input checked="" type="radio"/> unchanged <input type="radio"/> true <input type="radio"/> false
	Thresholds	true	<input checked="" type="radio"/> unchanged <input type="radio"/> true <input type="radio"/> false
<b>FastEthernet0/1</b>	Description	WAN	<input type="text"/>
	Display Name	GCHUB Connection	<input type="text" value="GCHUB Connection"/>
	Speed In	100000000	<input type="text" value="100000000"/>
	Speed Out	100000000	<input type="text" value="100000000"/>

```
my $conf = $nodeobj->configuration;
my $overrides = $nodeobj->overrides;
```

## Get the node model

```
$S->init(node => $nodeobj, snmp => 0);
my $mdl = $S->mdl;
```

## Get the node catchall

Catchall Is a part of the node inventory and this is where all the main information collected from the node is saved.

The node model, the last poll date, or the health data collected like the sysDescr or the sysLocation:

```
my $catchall_data = $S->inventory( concept => 'catchall' )->{_data};
```

## Get the node interfaces

```

my $sectionIds = $S->nmisng_node->get_inventory_ids(
    concept => "interface",
    filter => { historic => 0 });

if (@$sectionIds)
{
    for my $sectionId (@$sectionIds)
    {
        my ($section, $error) = $S->nmisng_node->inventory(_id => $sectionId);
        if ($error)
        {
            print "Failed to get inventory $sectionId: $error \n";
            next;
        }
        my $interface = $section->data();
    }
}

```

## Get the node inventory

```

my $sectionIds = $S->nmisng_node->get_inventory_ids(
    concept => $concept,
    filter => { historic => 0 });

if (@$sectionIds)
{
    for my $sectionId (@$sectionIds)
    {
        my ($section, $error) = $S->nmisng_node->inventory(_id => $sectionId);
        if ($error)
        {
            print "Failed to get inventory $sectionId: $error \n";
            next;
        }
        my $data = $section->data();
    }
}

```

## Example



dev\_tests.pl

- You can download and run the complete example :

- Place it `/usr/local/nmis9/admin`
- Run with:

```
/usr/local/nmis9/admin/dev_tests.pl act=show_node node=asgard what=ALL concept=addressTable
```

## Related Documentation

- You can see a complete example of a collect plugin in the Host\_Resources plugin (Released in 9.1.2).
- [NMIS 9 collect and update plugins](#)