The Netdot Manual

Network Documentation Tool

Version 0.9

Table of Contents

This manual documents the installation, administration and operation of the Netdot application.

The Network Documentation Tool

Netdot [http://netdot.uoregon.edu] is an open source tool designed to help network administrators collect, organize and maintain network documentation.

Netdot is actively developed by the Network and Telecommunication Services [http://ns.uoregon.edu] group of the University of Oregon [http://www.uoregon.edu].

Netdot features include:

- Device discovery via SNMP
- Layer 2 topology discovery and graphing, using multiple sources of information: CDP+LLDP, Spanning Tree Protocol, switch forwarding tables, router point-to-point subnets.
- IPv4 and IPv6 address space management (also referred to as *IPAM*), including hierarchical organization, address block visualization and IP and MAC address location and tracking.
- Cable plant information including: sites, rooms, jacks, closets, inter and intra-building wiring, circuits, etc.
- Contact information for related entities: departments, providers, vendors, etc.
- Netdot can generate configuration files for various other tools, including: Nagios [http://www.nagios.org], Sysmon [http://www.sysmon.org], RANCID [http://www.shrubbery.net/rancid], Cacti [http://www.cacti.net]. Starting with version 0.9, Netdot also generates configurations for ISC BIND and ISC DHCPD.

Structure

Netdot consists of several components:

1. The database

   Our goal has been to make Netdot database-agnostic as much as possible. In principle, it should be able to use any database supported by Perl DBI. There are, however, some limitations to this, for example, schema migration scripts are db-specific and may not always be availab.e. Currently MySQL is fully supported. There is currently partial support for PostgreSQL.

2. The libraries

   The back-end code is a hierarchy of object-oriented Perl classes. It can function as an API as well. One advantage of this model is that presentation, collection and database can be separated among different physical machines.

3. User Interface (UI)

   The web user interface is built on a templating system called HTML::Mason.

4. Command Line scripts (CLI)

   Certain tasks, like device discovery, can be executed from the command line. Therefore, these tasks can be automated by running them periodically via CRON.

Installation

Requirements

- Perl 5.6.1 or later
- Apache2 with mod_perl2
- MySQL (5.x preferred) or PostgreSQL (limited support)

- Authentication Server (optional). Netdot supports local authentication, as well as Radius and LDAP.
  - For Radius, we recommend FreeRadius, available at:

    http://www.freeradius.org
  - For LDAP, you can try OpenLdap, available at:

    http://www.openldap.org
- The RRDtool package, including its Perl modules, available at:

  http://oss.oetiker.ch/rrdtool/
- The GraphViz package, available at:

  http://www.graphviz.org
- Various Perl modules. To test for missing modules in your system, run:

  ```
  #make testdeps
  ```
- There are two ways to install the missing modules: The first (and the recommended) way is through package managers of your distribution.

  For systems with APT (e.g. Debian-based systems), run:

  ```
  #make installdeps-apt-get
  ```

  For systems with RPM (e.g. Red Hat), run:

  ```
  #make installdeps-rpm
  ```

  ### Tip

  If you are still missing modules after running this step, you can complete the process by running the next step.
- Alternatively, you can install any missing modules automatically using the CPAN, by running:

  ```
  #make installdeps
  ```

  If you want to install modules individually, you can do this instead:

  ```
  #cpan
  >install Module::Blah
  ```

Configuration

Netdot comes with a configuration file that you need to customize to your needs. You need to:

```
#cp etc/Default.conf etc/Site.conf
```

Then, modify Site.conf to reflect your specific options. The original file contains descriptions of each configuration item.

Netdot will first read Default.conf and then Site.conf

The reason for keeping two files is that when an upgrade is performed, the Default.conf file can be re-written (to add new variables, etc.), without overwriting your site-specific configuration.

Tip

Notice that, each time you modify Site.conf, you must restart Apache for the changes to take effect in the web interface.

Upgrading

You should check if the version you are installing has any new requirements that need to be satisfied:

```
#make testdeps
#make installdeps (or installdeps-rpm, installdeps-apt-get)
```

Netdot's database schema **usually** only changes between major versions. For example, if upgrading from 0.8.x to 0.9.x, you will need to run an upgrade script to adapt your current database to the new schema.

Look for a file called doc/UPGRADE for upgrade notes in a particular distribution.

If you are supposed to upgrade, this can be accomplished by running this command:

```
#make upgrade
```

Finally, install the Netdot code and restart Apache:

```
#make install
#/etc/init.d/httpd restart
```

Installing the package for the first time

- Make sure you have created the file *etc/Site.conf* with your configurations (See above).
- From the top directory in the package, do:

  ```
  #make install [parameters]
  ```

  Possible parameters include:

  ```
  PREFIX=YOUR-PREFIX (default: /usr/local/netdot)
  APACHEUSER=USER-YOUR-APACHE-RUNS-AS (default: apache)
  APACHEGROUP=GROUP-YOUR-APACHE-RUNS-AS (default: apache)
  ```

- ### Tip

  Ubuntu users: will probably need to set the APACHEUSER and APACHEGROUP variables to "www-data", which is the user that Apache runs as.

- You will then be ready to initialize the database.

  ```
  #make installdb [parameters]
  ```

  Remember you need to set *DB_DBA* and *DB_DBA_PASSWORD* to your database's admin username/password in *etc/Site.conf* before running this command. Or if you prefer, you can specify the *DB_DBA* and *DB_DBA_PASSWORD* values as parameters (however, these are used by many functions in Netdot, they will need to be set to the correct value in *etc/Site.conf* eventually)

```
DB_DBA=DATABASE-ADMIN-ACCOUNT
DB_DBA_PASSWORD=DATABASE-ADMIN-PASSWORD
```

Apache Configuration

Edit the supplied Apache config template for either Local, Radius or LDAP authentication, copy it to your Apache config directory and include it somewhere in your Apache configuration file (httpd.conf) (e.g.):

```
Include conf/netdot_apache2_<local|radius|ldap>.conf
```

Alternatively, some Apache environments provide a directory from which files are included automatically when Apache starts. In that case, it's sufficient to copy the file into that directory, for example:

```
#cp /usr/local/netdot/etc/netdot_apache2_radius.conf /etc/apache2/conf.d/
```

Tip

Make sure you use the version of the file that gets copied into your install directory by *make install*, not from the source directory. This file contains relevant path substitutions based on your chosen install prefix.

Once this is done, you can restart Apache2. If you used the default settings, point your browser to:

```
http://servername.mydomain/netdot/
```

You should be able to log in with:

```
username: "admin"
password: "admin"
```

Tip

If you are using the Radius or LDAP authentication options, you should have NetdotRadiusFailToLocal or NetdotRadiusFailToLocal set to "yes" in your netdot_apache2_x.conf file.

Warning

Please remember to change the "admin" password!

CRON jobs

Netdot comes with a few scripts that should be run periodically as cron jobs.

- Retrieval of forwarding tables and ARP caches for IP/MAC address tracking
- Devices should be re-discovered via SNMP frequently to maintain an accurate list of ports, ip addresses, etc.
- Rediscovery of network topology
- Netdot keeps history records for some objects every time they are updated. With time, old history should be deleted from the database to save disk space.
- Netdot can generate text documentation that is easy to find using simple grepping commands, for example, information about people, locations, device port assignments, etc. This documentation should be kept up to date by exporting it frequently.

- Configurations for external programs can be generated using Netdot data. Current supported programs include: Nagios, Sysmon and Rancid.
- The netdot.cron file included in the package is a sample crontab containing recommended periodic jobs. You should customize it to your liking and copy it to your cron directory, for example:

```
#cp etc/netdot.cron /etc/cron.d/netdot
```

Operation

Device Management

Netdot can discover and maintain an extensive amount of information about network devices. The easiest way to gather and store this information is by querying the devices using the Simple Network Management Protocol (SNMP). Devices can be discovered individually, by subnet, or by providing a text file with a list of devices.

Device Discovery using the web UI

Go to *Management -> Devices*. In the Tasks section, click on [new] and type the hostname or IP address of the device in question, along with the SNMP community and click [discover]. Netdot will then query the device using SNMP and present a window where you can assign an owner entity (for example, your organization), the entity that uses the device (for example, your customer), the location and a contact list.

If you are discovering a layer 3 device with IP forwarding turned on (such as a router or firewall), Netdot will ask you if you would like to automatically create subnets, based on the IP configuration of the device interfaces. This is a convenient way to add all your subnets into Netdot.

Another option is to specify whether Netdot should assign any newly created subnets the same owner and user entities assigned to the device.

Once you click the on [update] button, Netdot will show the discovery information and a link to the device page at the button.

You can always re-discover a device manually by using the [snmp-update] button on the top right corner of the device page. For example, if you have added a new port adapter, new interface cards, or if the device has been replaced with a different unit.

Device discovery using the command line interface (CLI)

For brevity, let's assume you are located at the Netdot installation prefix, for example, */usr/local/netdot*.

You can discover a single device by executing:

```
#bin/updatedevices.pl -H <device-name> -I -c <community>
```

You can also try discovering a whole subnet like this:

```
#bin/updatedevices.pl -B 192.168.1.0/24 -I -c <community>
```

In addition, you can give Netdot a specific list of devices you would like to discover:

```
#bin/updatedevices.pl -F <text-file> -I -c <community>
```

The file should contain a list of devices and their communities, separated by spaces, one device/community per line, for example:

```
device1 community1
device2 community1
device3 community2
...
```

Netdot can retrieve ARP caches and bridge forwarding tables. You will probably want to fetch ARP caches from your layer 3 devices (i.e. routers and firewalls), and forwarding tables from your layer 2 devices (switches). Examples:

```
#bin/updatedevices.pl -H <router> -A -c <community>
```

```
#bin/updatedevices.pl -H <switch> -F -c <community>
```

Netdot can also try to discover the network topology. For that, you need to run:

```
#bin/updatedevices.pl -T
```

Ideally, once you have discovered all your devices, you should combine all this functionality and have it run periodically (e.g. every hour) via CRON. A sample crontab entry would be:

```
0 * * * * root /usr/local/netdot/bin/updatedevices.pl -I -A -F -T
```

You will find some examples of cron jobs in the file named *netdot.cron*

Device Documentation

Once you have created a device, you can go ahead and add more information about it.

Going to *Management -> Devices* you can search for a device by name, IP or MAC address.

From the device page, you can navigate to the different sub-sections depending on the information you want to edit.

Interfaces: Here you can edit interface descriptions, assign network jacks, etc. by clicking on the [edit] button. You can also edit a specific interface by clicking on its number or on its name. If you are running topology discovery, you will probably see neighbor information. If for some reason the topology discovery process cannot dectect a neighbor, you can add it manually by clicking on the [add] button in the neighbor column.

Manually adding a neighbor sets the "Neighbor Fixed" flag in the Interface object. This flag prevents the topology discovery process from removing the neighbor relationship.

Tip

Neighbor relationships tend to change frequently as hardware is replaced and connections are moved. Therefore, fixed neighbor settings can become out of date pretty soon. It is preferable to let the topology discovery process maintain neighbor relationships.

IP Address Space Management

Netdot is a useful tool for IP address space management (referred to as *IPAM* these days). Some key features are:

- Address space is hierarchically organized through the use of a fast binary tree algorithm, which is the same technique used by routers when doing prefix lookups.
- IPv4 and IPv6 support

- New subnets can be automatically created based on the interface information retrieved from routers and firewalls.
- Easy visualization of used vs. available address space for easier allocations

IP blocks

IP objects are called IP blocks. These objects can represent individual end-node addresses, as well as groups of addresses. The distinguishing characteristic is the prefix attribute. For example, an IPv4 block with a 32 bit prefix is an end-node address, while a block with a 24 prefix represents a group of 254 end-node addresses.

Each address or block has a corresponding status. Let's see those in detail.

IP block Status

IP objects are assigned a status to better represent their nature. Depending on whether the IP is an end address or a block, different status values can be assigned.

The status of an end-node address can be one of:

- Static: These are addresses that have been statically assigned to hosts or device interfaces.
- Dynamic: Addresses that belong to a DHCP pool
- Discovered: Addresses that have not been assigned as static or dynamic, but have been seen on the network (as part of ARP entries, for example).
- Reserved: Addresses that should not be assigned

On the other hand, the status of an IP block can be one of:

- Container: This kind of block is meant to group or contain other blocks, such as Subnet blocks or other Container blocks. For example, let's say your whole IPv4 address space is 192.168.0.0/16. You also have partitioned this space into two /17 blocks, and from these blocks, you allocate subnets that you configure in your routers. In this case, you would have:

```
192.168.0.0/16 -> Container
    192.168.0.0/17  -> Container
        192.168.0.1/24 -> Subnet
        192.168.0.2/24 -> Subnet
    192.168.128.0/17 -> Container
        192.168.128.10/24 -> Subnet
        192.168.128.20/24 -> Subnet
```

- Subnet: This kind of block is meant to represent actual subnets that are configured on the interfaces of your layer 3 devices such as routers or firewalls. Subnets usually contain the end-node addresses that you assign to your users.
- Reserved: Similarly to reserved addresses, reserved blocks are not supposed to be allocated for whatever reason.

DNS

Netdot can maintain DNS zone data. Zones can be exported as text files to be used by DNS server software. Currently, only ISC BIND zone file exporting is supported.

Tip

The mechanisms by which zone files are transferred to and loaded by authoritative name servers are left to the administrator. A simple way to do this is by running a name server locally in the machine that runs Netdot, and saving those zone files in the location where the software can load them periodically. A more complex setup could involve saving these files into revision control systems (CVS, SVN, etc), which could then be used by system configuration engines like Puppet or CfEngine to run syntax checks and load them into the appropriate name servers.

Netdot supports the following DNS records: A, AAAA, CNAME, DS, HINFO, LOC, MX, NAPTR, SRV, and TXT.

You can import your existing BIND zones into netdot with the help of the tool *import_bind_zones.pl* from the *import* subdirectory

```
usage: import/import_bind_zones.pl
 [ -n|domain <name>, -f|file <path> ] (for single zone)       [ -c|config <path>,

 -c  --config <path>   Bind config file containing zone definitions
 -d, --dir <path>      Directory where zone files are found
 -n, --domain <name>   Domain or Zone name
 -f, --zonefile <path> Zone file
 -w, --wipe            Wipe out existing zone data           -g, --debug
 -h, --help            Print help
```

To add a new zone manually, go to *Management -> DNS Zones* and provide a name for the zone. Optionally, select an existing zone which you would like to use as a template. This will tell Netdot to basically clone this template zone and all its records, but saving it with the name you provide. This is useful in cases when multiple zones share the same information, such as NS records, MX records, etc. Click on [add]. You will see a new zone created using the values from the template zone, or with default values extracted from the configuration file.

Once a zone is created, it should be linked to an IP block (Subnet or Container). You can do this by clicking on the [add] button of the IP blocks section in the zone page.

The most convenient way to create reverse zones (in-addr.arpa or ip6.arpa) is to go to the corresponding IP block page, *DNS Zones* section, and click on [add]. If the corresponding reverse zone does not exist, Netdot will present the user with the appropriate zone name and an option to create it. This is especially useful with IPv6 blocks, which tend to require very long reverse zone names.

At this point, you can add new records by clicking on the [add] button on the Records section. Records can also be added from other parts of the user interface, for example, from the IP address page, or the *DNS Records* page.

Records can also be imported in bulk into the zone by going to the Zone page, clicking on the [import] button of the *Records* section and pasting the text from a BIND zone file into the text box.

Each time the zone or its contents are modified, the transaction is added to a list of "pending changes". This list is used to determine when a zone needs to be exported. Zones can be exported manually via the UI by going to the *Export* menu, or via cron jobs. When a zone is exported, its serial number is increased and its list of pending changes is flushed.

Netdot can auto-generate and maintain DNS records for IP addresses belonging to device interfaces, for example, routers. The logic of how these names are generated is handed off to a plugin. In order to have Netdot generate DNS names for device IPs, set the "Auto DNS" option to "yes" in the management section

of the Device page. From here, go to the IP Info tab for the same device and for each IP address, make sure that the interface it belongs to has the "Auto DNS" option set to "yes".

The '@' record

In Netdot, as in BIND, the '@' record symbolizes the domain (a.k.a "zone apex"). In order to add records that apply to the domain itself, such as NS records, MX records, A records, etc. this record must exist. At zone creation time, Netdot automatically adds this record, together with two NS records for the zone, with the names (ns1.zone.name and ns2.zone.name).

DHCP

Netdot can maintain DHCP information and generate configurations for ISC DHCPD.

DHCP information is organized hierarchically around the DHCP Scope object. Netdot supports scopes of the following types: global, subnet, shared-subnet, group, and host. Each of these scopes can be assigned one or more attributes.

Global Scopes

A global scope will represent a DHCP server (or a pair of failover servers). Attributes in this scope are the default attributes inherited by all other scopes. Attributes in more specific scopes override the global scope attributes.

To create a new global scope, go to *Management->DHCP*. Click on the [new] button. Assign the scope a name (for example, the host name of your DHCP server) and select type "global". Global scopes are not contained by any other scope, so leave the Container field unselected.

Once a scope is created, you can add attributes to it. For example, click on the [attributes] button and then [add]. You will see a new page where you can create a new attribute. Let's say, for example, that you want to add a list of name servers. Type "name-servers" in the Name search box and click on "List". Select the "domain-name-servers" attribute name from the list and add a list of values. Then click Insert.

Subnet Scopes

Subnet scopes contain attributes that apply to all hosts within a subnet. Subnet scopes are contained by a global scope.

The easiest way to enable DHCP for a particular subnet is from within the Subnet page. First, make sure that the subnet exists (you can create it manually or by discovering the router that serves that subnet). You can view the subnet by going to *Management->Address Space* and navigating to where the subnet is, or by simply searching for its address.

Once in the subnet page, look for the *Dhcp Scope* section and click on [enable]. This will bring an input section where you can select the global scope and the routers option. By default, Netdot shows the first address of the subnet as the routers option value. You can change this value if your router interface has a different address. Click Save. You will now see the subnet scope listed in the Subnet page. You can click on the scope name and that will take you to the DHCP Scope page, from which you can add any other necessary attributes.

Host Scopes

Host scopes allow you to assign attributes that apply to particular hosts. Host scopes also link a host's Ethernet address with its IP address.

You can create a new host scope from the host page. First of all, a Static IP address object needs to exist. You can create new static IP objects by selecting the desired address from the Subnet page. Once the Static IP address is created, you need to give it name. Look for the *DNS A records* section and click on [add]. Once you provide a name for the A record, you will be redirected to the host page. Here, find the *DHCP for <IP address>* section and click on [add]. Type the Ethernet address and save your changes. When you click on the Ethernet address, you'll go to the MAC address page, which has a "DHCP Scopes" section. Clicking on the IP address will take you to the DHCP scope page. Here, you can add any specific attributes for that specific host.

Template Scopes

A template scope is not a real scope, but only a collection of attributes that you want to apply to things as a group. For example, the DHCP host scope for an IP phone may have one or more attributes that define where it should get its configuration from and other things. You can create a template containing these attributes and then use that template each time you create a host scope for IP phones.

Contact Information

Netdot uses the concept of "Contact Lists" to show contact information for different objects, for example devices, sites, entities (departments, providers, etc.).

A Person object in Netdot contains a person's information, including location, e-mail address, phone numbers, pager numbers, etc.

Since a given person often times is the point of contact for different things, a person can have many "roles", which link that person with a particular Contact List.

You can create new Person, Entity, Site and Contact List objects by going to the Contacts section.

Cable Plant

Netdot allows you to document inter-building and intra-building fiber and copper wiring, closets, jacks, etc.

Advanced DB operations

The Advanced section of the top menu shows basic *Browse*, *Search* and *Add* operations on particular tables of the database. This often requires certain familiarity with the database schema.

In this section you can also write your own SQL queries, which can be saved for future use. SQL query output can also be saved in comma-separated (CSV) format.

Reports

The Reports section provides a number of useful types of reports.

Device Reports

By Type/Model

Lists devices grouped by type (switches, routers, servers, etc), then by model, and gives a total count per type and model.

By Model/OS

Lists devices by manufacturer, then model, showing each model's recommended OS version (which you would have had to previously specify) and all the other existing versions of that OS in your network, with counts.

Device in Downtime

Since Netdot can be used to export configurations for monitoring tools (e.g. Nagios), particular devices can be assigned a downtime period, which will exclude them from the monitoring tool during the time frame specified. This report shows you all the devices that are within a downtime period.

Duplex Mismatches

This report shows a list of neighboring device interfaces whose duplex settings are mismatched.

OS mismatches

This report lists devices whose operating system version differs from the recommended version. The list is grouped by manufacturer, then model, then device name and it shows the current OS version.

IP Reports

Unused Subnets

Here you will see a list of subnets that have no IP addresses. You can select only IPv4 subnets or IPv6 subnets.

Maxed out Subnets

This report lists subnets that are used beyond a given threshold. This threshold is configurable by modifying the SUBNET_USAGE_MINPERCENT item in the etc/Site.conf file

Unused Static Addresses

This report shows static addresses that have not been seen in the network for a given time. This makes it easy to free up subnet address space.

MAC Addresses

This report shows a list of MAC address OUIs, sorted by number of addresses. You have the option to include all addresses, only MAC addresses belonging to infrastructure devices or only MAC addresses found in ARP caches and forwarding tables.

Exporting Configurations for External Programs

You can use the exporter tool to generate text files that can be used as configurations for third-party tools and programs.

The exporter tool is available in the web UI, under the *Export* tab. Simply select one or more programs and click on the [submit] button. Netdot will show some output from the exporter tool, including the paths to the new files.

Additionally, the exporter can be called from the command line. For example, to generate Nagios configurations:

```
#bin/exporter.pl -t Nagios
```

Or you can export several in one call:

```
#bin/exporter.pl -t Nagios,Sysmon,Rancid,BIND,DHCPD
```

There are specific export parameters for each of these which you can customize by editing your Site.conf file.

Cacti Integration

Cacti integration is done a little differently (it's more of an "import"than an "export". You will find a script called "netdot_to_cacti.php"under export/cacti in the Netdot package. This script should be placed(together with its configuration file) in your Cacti's cli directory(it doesn't need to be the same machine running Netdot, but you need to make sure that the script can connect to Mysql on the Netdot machine).You will then need to run it periodically via CRON, say, once a day.

Authorization

Starting with version 0.9, netdot supports multi-level user access.

There are three types of users that correspond with levels of access in Netdot:

- Admin: Full access to the UI and operations on objects.
- Operator: Full access to the UI, but read-only access to objects.
- User: Limited UI, with view, edit, and delete access to particular objects.

Assigning permissions to users

Permissions can be assigned to individuals or to groups. Individuals are grouped in contact lists. A user who is a member of a contact list inherits the permissions from the list. However, the individual can have more specific permissions (or no permissions) if necessary.

There is a limited number of objects which unprivileged users can gain access to:

- DNS records: Users can create, modify and delete records from a certain zone. Permissions can be given for the entire zone or for subsets of it, based on IP blocks. For example, if a user is given view, edit and delete permissions to myzone.com, he or she can view, modify and remove any record from that zone. On the other hand, if the zone covers hosts from a supernet, i.e. 10.0.0.0/16, and the user should only have control on records within a particular subnet, i.e. 10.0.0.0/24, instead of assigning permissions on myzone.com, the administrator can assign view, edit and delete permissions on that particular subnet.
- Device interfaces: Users can view port details such as number, name, vlan, room, jack, description and neighbor. A user can only edit the room, jack and description fields. To assign permissions to a user on a list of devices, select the Device class and then select one or more devices to which the user can have access.
- Contact Lists: A user can add, modify and delete contacts from given contact lists.

To assign permissions for an individual user, perform the following tasks:

- Make sure there is a Person object for the user. You can verify if a Person object exists by going to Contacts -> People and searching for the person's name in the Search box. If the object does not exist, you can create a new one by clicking on the [new] button on the upper right corner of the same window.

- Make sure that the person object has a Username and a User Type set. If you have configured netdot to use Radius or LDAP authentication, make sure that the username corresponds with the login information in those central authentication systems. If you are using local authentication instead, make sure that you set a local password using the Password field.
- On the Person page, you can add permissions by clicking on the [access_rights] button. This will display current permissions. You can now add new ones by clicking on the [add] button on the right.
- On the UserRight window, select the Object Class, the specific object or objects, and one or more access rights (view, edit, delete). Only select the 'none' right to revoke all permissions inherited from a group. Click on 'Insert'.

RESTful Interface

The RESTful interface allows programmatic access to the Netdot database over the HTTP/HTTPS protocol. At this moment, all objects are formatted in XML using the XML::Simple Perl module. In the future, Netdot may support other formats, such as YAML.

RESTful resources

The REST interface is available using the following URL (or similar, depending on your Apache configuration):

```
https://myserver.mydomain.com/netdot/rest/
```

This should load the Netdot::REST class and return something like:

```
Netdot/0.9 REST OK.
```

RESTful resources to be acted upon represent Netdot objects and are part of the request URI. For example, in this URI:

```
http://myserver.mydomain.com/netdot/rest/device/1
```

the resource is "device/1", which for a GET request, will return the contents of Device id 1.

Using the following URI with a GET request:

```
http://myserver.mydomain.com/netdot/rest/device
```

this interface will return the contents of all Device objects in the database. You can also specify certain search filters to limit the scope of a GET request:

```
http://myserver.mydomain.com/netdot/rest/device?sysname=host1
```

This will perform a search and return all devices whose sysname field is 'host1'.

The special keyword 'meta_data' instead of an object ID will provide information about the object's class:

```
http://myserver.mydomain.com/netdot/rest/device/meta_data
```

An existing resource can be updated by using the 'POST' method with relevant parameters. For example, a POST request to the following URI:

```
http://netdot.localdomain/rest/device/1?sysname=newhostname
```

will update the 'sysname' field of the Device object with id 1 to be "newhostname". Similarly, a new object can be created with a POST request. However, in this case the object id must be left out:

```
http://netdot.localdomain/rest/person/?firstname=John&lastname=Doe
```

Lastly, specific objects can be deleted by using the 'DELETE' HTTP method.

Client module on CPAN

A convenient module is provided via CPAN for use in Perl scripts that need to access Netdot's REST interface. The module name is Netdot::Client::REST. It can be installed by doing something like this:

```
#cpan
>install Netdot::Client::REST
```

FAQ

- Q: Why can't Netdot discover my device using SNMP? I can see it responds when using snmpwalk.

  A: If your device is a Linux server or any other device using the SNMP agent from Net-SNMP, you need to add the following to /etc/snmpd.conf (or wherever it exists):

  ```
  sysservices 72
  ```

  Then, restart your snmpd process.

  The reason for this is that sysservices is essential to Netdot's device discovery. Most network devices implement this OID in their SNMP agents.