The Network Documentation Tool - Netdot

User's Manual

Contents

1 Copyright

Version 1.0

1.1 Purpose

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

2 Introduction

Netdot 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 group of the University of Oregon.

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,
  - Sysmon,
  - RANCID,
  - Cacti.
  - ISC BIND and ISC DHCPD
  - Smokeping
- Netdot implements role-based access control, allowing tasks such as IP address management, documentation of switch/router ports and updating of contact information to be delegated to specific groups with limited access to the web interface.

2.1 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 available. 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.

3 Installation

3.1 Obtaining and unpacking the packaged distribution file

Download the latest Netdot package from the netdot website

`https://osl.uoregon.edu/redmine/projects/netdot/wiki/DownLoad`

Unpack the file in a directory other than where you want to install Netdot, i.e.

```
~# tar xzvf netdot.tar.gz -C /usr/local/src/
```

3.2 Requirements

- Perl 5.6.1 or later
- Apache2 with `mod_perl2`
- MySQL or PostgreSQL
- Authentication Server (optional). Netdot supports local authentication, as well as RADIUS, LDAP and Kerberos.

- The RRDtool package, including its Perl modules, available at: http://oss.oetiker.ch/rrdtool/
- The GraphViz package, available at: http://www.graphviz.org/
- The latest Netdisco MIBs. http://sourceforge.net/projects/netdisco/files/netdisco-mibs/
- Various Perl modules.
- The 'make' utility.

3.2.1 Installing dependencies

There are two ways to install dependencies: The first and the recommended way is through package managers of your distribution (this will also install other necessary packages, not just Perl modules).

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

  ```
  ~# apt-get install build-essential
  ```

  ```
  ~# make apt-install
  ```

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

  ```
  ~# yum install make
  ```

  ```
  ~# make rpm-install
  ```

  Tip If you are still missing Perl modules after running this step, you can complete the process in the next step.

- If your package manager is not supported, or if you are missing dependencies, you can install those by hand. However, you can at least take advantage of the CPAN to install Perl modules automatically.

  To test for missing modules in your system, run:

  ```
  ~% make testdeps
  ```

  Then, use this to install the missing modules:

  ```
  ~# make installdeps
  ```

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

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

3.3 Configuration

Netdot comes with a configuration file that you need to customize to your needs. You need to create a copy of `Default.conf` with the name `Site.conf`

```
~% 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.

3.4 Upgrading

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

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

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

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.

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

(make a backup of your database first!)

```
~# make upgrade
```

Finally, install the new Netdot code and restart Apache:

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

3.5 Installing the package for the first time

- Prepare your database administrator (DBA) account

  MySQL users: The DBA account for MySQL is usually created when installing the package. Make sure to set a password during the installation.

  Pg users: PostgreSQL normally comes with a default DBA account named 'postgres'. After installing, you may need to set the password for this account as follows:

  ```
  ~% sudo -u postgres psql postgres
  ```

  Set a password for the "postgres" database role using the command:

  ```
  \password postgres
  ```

  and give your password when prompted. Type Control+D to exit the prompt.

- Adjust your database configuration if necessary

  MySQL users: If you intend to use the IPAM functionalities in Netdot, you might need to increase the maximum packet buffer size in my.conf to something like:

  ```
  max_allowed_packet = 16M
  ```

- Make sure you have created the file etc/Site.conf with your configurations (See above).

- 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
  ```

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

3.6 Apache Configuration

Edit the supplied Apache config template for either Local, RADIUS, Kerberos 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|krb5>.conf
```

Alternatively, some Apache environments provide a directory from which files are included automatically when Apache starts. In that case, you can create a link to the file in said directory.

For example, in Debian or Ubuntu, it's a two-step process:

```
~# cd /etc/apache2
~# ln -s /usr/local/netdot/etc/netdot_apache2_local.conf sites-available/netdot
~# ln -s ../sites-available/netdot sites-enabled/netdot
```

Or, in other distributions with just one directory:

```
~# ln -s /usr/local/netdot/etc/netdot_apache2_local.conf /etc/apache2/conf.d/netdot
```

> **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 one of the external authentication options, you should have Netdot(radius|ldap|krb5)FailToLocal set to "yes" in your `netdot_apache2_x.conf` file.

> **Warning** Please remember to change the "admin" password! Go to `Contacts -> People`, search for 'Admin', click on [edit] and type in a new password. Then click on the Update button.

3.7 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
- With time, old data like forwarding and ARP table entries, audit records, etc. 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. See details later in this document.
- 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
  ```

4 Operation

4.1 Device Management

In Netdot, devices are network infrastructure equipment: switches, routers, firewalls, access points, servers, etc. End nodes such as desktop computers, laptops and printers are not devices.

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 device addresses.

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

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.

4.1.2 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 -E <text-file> -I
```

The file should contain a list of device names or IP addresses, one per line, for example:

```
device1
device2
device3
...
```

Optionally, each device line can be accompanied by its SNMP community:

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

Netdot can retrieve ARP 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
```

If the configuration option `ADD_UNKNOWN_DP_DEVS` is set to 1 (true), then Netdot will attempt to discover any devices seen (via CDP/LLDP) on existing device interfaces. With the previous command, Netdot will only try to discover directly connected devices. In order to attempt to discover all unknown neighbors, and the neighbors of those neighbors, use the following parameter:

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

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 -DIFAT
```

If you want to only update a subset of the devices in your database, you may use the "–matching" parameter to specify a regular expression, which will be applied against devices' fully qualified names. For example if all your routers have the suffix "-gw", you could do something like:

```
0 * * * * root /usr/local/netdot/bin/updatedevices.pl -DIFAT --matching "-gw"
```

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

4.1.3 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. Notice that clicking on any field name will open a browser window with a description of that field.

Basics Tab: In this section you can view and edit general information about the device, including its location, contact information, and management details.

Interfaces Tab: 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 detect 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.

Modules Tab: If the device provides module information via SNMP, Netdot will show it in this tab. Modules are shown hierarchically based on how they are contained within each other.

IP Info Tab: This section lists all the IP addresses found in the device, together with the subnets they belong in, the device interface where they were found, and optionally, their DNS names.

At the bottom of this section, you will find an option to set the "Auto DNS" flag on all interfaces with IP addresses. The purpose of this flag is to tell Netdot whether it should generate DNS names for each IP address based on the interface name and the device name.

The logic of this operation is handed off to a plugin module, which means that you can write your own plugin to generate DNS names based on your own naming scheme (see configuration file for more details). The included plugin generates names such as "ge-0-1.router1.mydomain.com", assuming that the device name is router1 and that the interface is GigabitEthernet0/1. This is very useful for when you are using the traceroute utility.

For this to work you need the following:

- The device has to have its "Auto DNS" flag set (Basics section of the device page).
- Each interface with an IP address on the device has to have the "Auto DNS flag on"
- The IP address must exist within an IP block which has been assigned a DNS zone (Management -> Address Space).
- For PTR records to be generated as well, the IP block must have a reverse zone (in-addr.arpa or ip6.arpa) associated with it.

BGP Peers Tab: If the device is a router with BGP peering sessions, and those are seen via SNMP, Netdot will show that information in this tab. Information includes the remote IP address, the BGP ID and the AS. The BGPPeering record also includes fields to document things such as the maximum number of allowed IPv4 and IPv6 prefixes, whether the peering should be monitored (e.g. with Nagios), etc.

For each AS discovered, Netdot tries to look up its information using WHOIS. If the information is found, an entity record is created with the AS number, AS name, etc. You can expand this record to include contact information, comments, etc.

Topology Tab: Netdot can use the neighbor relationships from the device interface to draw a graph of this device and its neighbors. By default, Netdot only shows directly connected neighbors. However, you can expand the graph to include neighbors of neighbors by specifying a larger "Search Depth" value.

4.2 VLANs

Netdot creates VLANs when these are found in devices. You can add additional information to the VLAN record, such as a description, or comments.

When viewing a VLAN, you can see which interfaces in which devices are currently members (or trunks) of that VLAN. Also, in the device page you can see which VLANs are configured on each interface.

Currently Netdot assumes that VLANs are unique. If your VLAN IDs are reused around your network for different physical segments, Netdot information could be confusing. We intend to address this limitation in a future release.

4.2.1 Finding VLANs

You can search for specific VLANs by going to "Management" -> "VLANs". Netdot will match the search string against VLAN IDs (numbers) or names.

4.2.2 VLAN Groups

VLAN Groups are basically VLAN ID ranges that can help organize your VLAN assignments. For example, you might want to assign all your VOIP VLANs from the range 2000-2500.

You can create a VLAN group by going to "Management" -> "VLANs" and clicking on [new]. Provide a name for the group and a range of IDs.

4.3 Assets

An asset in Netdot is a record which contains information about device hardware. For example, serial number, inventory number, MAC address, product name, etc.

The difference between an Asset and a Device in Netdot is that a Device is an Asset which has been deployed and discovered.

Asset records can be used to document equipment that is not yet deployed. Once the asset is discovered in the network, it is referenced by the new device or device module record.

4.3.1 Importing Assets

Go to Management -> Assets -> [import] This form allows you to import multiple hardware assets. For example, you can use a bar code scanner to scan the information from vendor boxes as you receive your equipment.

Create a text file composed of part number, serial number, and optionally other fields. The part number must match the value from an existing product in Netdot. The order of fields in each line must match the list of fields in the "Fields for import" select menu.

Once imported, you can view a report of your assets in the Reports section.

4.4 IP Address Space Management

Netdot can be helpful in managing IPv4 and IPv6 address spaces. Some of its 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.
- New subnets can be automatically created based on the interface configuration retrieved from routers and firewalls.
- Visualization of used vs. available address space for easier block and address allocations
- DNS and DHCP configuration management

4.4.1 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
- *Available*: Addresses that were previously used, but have been freed.

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.

Associating IP blocks with other objects IP blocks can be linked to sites in a many-to-many relationship. A site can use one or more IP blocks and one IP block can be in use at one or more sites.

Similarly, IP blocks can be linked to DNS zones. This helps Netdot determine which domain a new DNS A, AAAA or PTR record should belong to.

4.5 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, PTR, 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>, -d|dir <path>  ] (for multiple zones)
  [ -g|--debug ] [-h|--help]

  -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           Print debugging output
  -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 kept in a database table called "hostaudit" and 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 the changes' "pending" status is cleared.

4.5.1 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).

4.6 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.

4.6.1 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.

4.6.2 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.

4.6.3 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. If you don't see a `DHCP for <IP address>` section, the IP is not within a subnet that has DHCP enabled.
- 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.

4.6.4 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.

4.6.5 Active and Inactive Scopes

The 'active' flag in the scope object determines whether this scope will be used while exporting DHCP configurations. For example, if you wish to document the assignment of IP addresses to MAC addresses in a given subnet, but you do not want to run DHCP on that subnet, you can create a Subnet scope and make it inactive.

Similarly to DNS records, DHCP changes are recorded in the "hostaudit" table, which Netdot uses to determine whether the DHCP configuration needs to be exported. Once exported, all changes' "pending" status is cleared.

4.7 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.

4.8 Cable Plant

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

4.8.1 Sites

Sites are usually buildings with one or more floors, closets and rooms. Sites can be associated with other things, such as people, departments, subnets, etc.

To create a new Site, go to Cable Plant -> Sites and click on [new]. You will need to enter a name. The "Site ID" is a value that can represent the (shorter) unique identification of that building within the organization.

You can also insert pictures of Sites in the database.

4.8.2 Closets

Communications closets house network equipment and cable terminations. A Closet is located in a Room, which is located in a Floor, which is located in a Site.

To create a new Closet, go to Cable Plant -> Closets and click on [new]. It is also possible to include pictures of closets in the database. This is useful for technicians that might want to review the physical characteristics of the closet space without visiting it in person.

4.8.3 Backbone Cables

Backbone cables exist between two closets.

- If a physical cable traverses closets in various sites, for the purpose of documentation,those sections of cable should be represented as different backbone cables.
- Backbone cables can interconnect closets within the same site (risers).

New backbone cables can be created by going to Cable Plant -> Backbone Cables and clicking on [new]. You will be asked to provide the origin and destination closets, the type of cable (Copper Bundle, Fiber, etc), and a cable ID. Netdot can suggest a cable ID value, which will be composed of the endpoint Site IDs and a sequence number, for example "123/456-1".

The field "Number of Strands" will tell Netdot to create that many strands associated with the new cable.

4.8.4 Fiber Strands

Backbone cables contain strands. These have several attributes, including:

- Sequence number
- Status - Not Terminated, Available, Damaged, In Use
- Fiber Type - Multimode or Single Mode
- Circuit - An end-to-end circuit composed of sequences of strands

You can modify ranges of strands from a backbone as a group. For example, if you have a new hybrid fiber cable with 24 strands, of which 12 are single mode and the other 12 are multi-mode, at the backbone page, after the list of strands, type Range: 1-12, then select Type: “Single Mode”, Status: “Not Terminated”. Do similarly for range 13-24.

Fiber strands from different backbone cables can be spliced together to form a sequence. To splice a range of strands, go to the bottom of the Backbone Cable page, and in the section “Manually Splice Strand Range”, type the range of strands that are spliced to another backbone, for example, “1-12”, and the corresponding strands from the next backbone, say “1-12” or “13-24”, then select the other backbone cable, and click “Go” You should now see the contiguous strands in the “Spliced With” column, and the whole sequence in “Part of Sequence”.

4.8.5 Circuits

After you have created sequences of strands from origin A to destination B, you can now create a circuit to group those strands and assign it to existing device interfaces.

To create a new circuit, go to Cable Plant -> Circuits and click on [new]. You will need to give it a unique identifier, and specify a provider. In this case, the provider might be your own organization. Circuits can also be used to document links provided by other parties. In those cases the circuit would probably not be associated with fiber strands that you own.

Circuits have these attributes, among others:

- Site Link: A record that ties two sites that are linked by this circuit. A link between two sites can use more than one circuit.
- Status: Active, Disabled, Disconnected, Pending
- Type: DS3, Ethernet, Frame Relay, etc.
- Speed: 45Mbps, 100Mbps, etc.
- Loss: Last measured loss on the circuit

Once you have created the circuit, you will have the option of associating a list of strand sequences. Simply select the origin and destination sites, then select a pair of sequences that compose this circuit (a pair, assuming that it’s a fiber circuit).

You can associate existing device interfaces to this circuit.

4.8.6 Horizontal Cables

A horizontal cable represents cabling that starts in a closet and terminates in a wall jack, usually Cat5 or similar. These are some of their attributes:

- Jack ID: The unique identifier of the jack in the organization. For example, a jack located in Site #123, terminated in closet "A" and whose sequence is 456, could be labeled uniquely with something like "123-A-456".
- Faceplate ID: Normally, faceplates contain more than one jack. This is the unique identifier of the faceplate, not the jack.
- Type: Cat5, Cat6, etc.
- Closet: The closet where the cable is terminated (one end)
- Room: The room where the cable is terminated (the other end)

Once created, you can assign this horizontal cable to a device interface by going to the Device page, selecting "Interfaces" and [edit]. You should see a list of existing cables in the "Jack(cable)" column. Notice that there are also free-form fields in the "Room" and "Jack" columns. These are available in case you don't need to document the cable, but just the interface-to-jack relationship.

4.9 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.

4.10 Reports

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

4.10.1 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.

VLAN mismatches This report shows a list of pairs of connected device interfaces whose list of VLANs differs. Interfaces can be set up as trunks, in which case they will usually carry tagged VLAN traffic for more than one VLAN, or just members of a VLAN. Unfortunately the report is not perfect because it would require knowledge about whether a VLAN is tagged or not. Currently this information is consistently available depending on the vendor and the model of the switch.

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.

4.10.2 Asset Reports

Asset reports are most useful for identifying existing device hardware, be it installed or not installed.

By Type/Model Gives a summary of device hardware by type and model, and shows quantities of each.

Detailed Shows a list of assets including their type, model serial number, inventory number, whether it has been installed or not, comments, etc.

4.10.3 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.

4.10.4 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.

5 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 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,Smokeping,BIND,DHCPD
```

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

5.1 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.

6 Authorization

Starting with version 0.9, netdot supports role-based authorization.

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.

6.1 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.
- When creating new DNS records, users with 'edit' rights on a subnet do not have the option to choose specific IP addresses. This helps keep ranges of IP addresses together so that Subnets can be resized more easily if necessary. If the Netdot administrator wishes to grant such rights to a user or group, there is a right called 'Choose IP' which allows that.

- 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 external authentication, make sure that the username corresponds with the login information in those external 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`.

6.2 Audit records

Once you give users permissions to update your Netdot database, you may want to know who has done what. There is a special database table called 'audit', which records every database operation made by a person (meaning that operations started by cron jobs are not recorded). Each audit record contains the following information: time stamp, username, operation type (insert, update, delete), table affected, record ID, record label, fields and values affected.

You can access these records by going to "Advanced" -> "Browse" -> "audit", or, if looking for a particular record, choose "Search" -> "audit" instead.

This table can be pruned periodically using the bin/prune_db.pl script.

7 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 or JSON.

7.1 Generic 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/1.0 REST OK.
```

- Generic 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:

```
URL: http://netdot.localdomain/rest/device/1
POST: {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:

```
URL: http://netdot.localdomain/rest/person
POST: {firstname=>'John', lastname=>'Doe'}
```

- Specific objects can be deleted by using the 'DELETE' HTTP method.

7.2 Special-purpose REST resources

7.2.1 /rest/host

The special resource '/rest/host' provides a simplified interface for manipulating DNS and DHCP records. We will illustrate its usage with the following examples:

Retrieving host data (HTTP GET)

- Retrieve all RR (DNS) objects

```
http://netdot.localdomain/netdot/rest/host
```

- Retrieve all RR objects within given zone

```
http://netdot.localdomain/netdot/rest/host?zone=localdomain
```

- Retrieve RR name "foo" and its related records

```
http://netdot.localdomain/netdot/rest/host?name=foo
```

- Retrieve RR id 1 and all related records

```
http://netdot.localdomain/netdot/rest/host?rrid=1
```

- Retrieve all Ipblock objects within given subnet

```
http://netdot.localdomain/netdot/rest/host?subnet=192.168.1.0/24
```

Creating new records (HTTP POST).

- Create new A record named 'host1' using next available address in given subnet (note: do not specify an object ID):

```
URL:  http://netdot.localdomain/netdot/rest/host
POST: {name='host1', subnet=>'192.168.1.0/24'}
```

Updating existing records (HTTP POST)

- Requires passing rrid or ipid. Rename host with RR id=2

```
URL:  http://netdot.localdomain/netdot/rest/host?rrid=2
POST: {name=>'newname'}
```

- Update DHCP scope ethernet for Ipblock with id=3

```
URL:  http://netdot.localdomain/netdot/rest/host?ipid=2
POST: {ethernet=>'DEADDEADBEEF'}
```

Deleting records (HTTP DELETE)

- Delete hostname with RR id 3 (also frees IP)

```
http://netdot.localdomain/netdot/rest/host?rrid=3
```

7.3 RESTful Interface Authorization

All user types can interact with the RESTful interface as long as they are granted permissions to do so. However only Admin users can edit or delete objects using generic REST resources. Operators and regular users can view generic resources but can only edit or delete them using specific-purpose resources such as 'rest/host'.

7.4 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:

If you are on a Debian-based system:

```
~# apt-get install libnetdot-client-rest-perl
```

or

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

8 Database Maintenance

Netdot's database will grow with time, thus it will be necessary to remove old information as it becomes stale. You will find a CLI utility called "prune_db.pl" in the bin/ directory of the distribution.

The sample CRON file "netdot.cron" included with the package contains recommended uses of this command.

> **Note**: Be especially careful when using the -I and -M options to remove old IP and MAC addresses. The criteria for deletion relies on the "last seen" timestamp on these records. That means that if Netdot is not collecting ARP and FWT tables from the routers, firewalls and switches where these addresses can be seen in the network, then Netdot will assume that they are not active anymore, thus included for deletion.