nrlnhdp User's Guide

Titleabbrev *nrlnhdp*User's Guide

Abstract The NRL Neighborhood Discovery Protocol (*nrlnhdp*) project is an open source implimentation of the NHDP protocol being standardized within the IETF. This software was developed by the Naval Research Laboratory (NRL) PROTocol Engineering Advanced Networking (PROTEAN) Research Group. The goal of this effort is to provide an implementation of neighborhood discovery with support for multiple heterogeneous interfaces in dynamic, wireless networks such as Mobile Ad-hoc Networks (MANETs). This effort will support other NRL MANET related projects such as OLSRv2 and SMF projects.

1 Overview

The *nrlnhdp* software is NRL's implementation of the Neighborhood Discovery Protocol (NHDP), which provides two-hop neighborhood discovery for IP based networks. This implementation was developed by the Naval Research Laboratory (NRL) PROTocol Engineering Advanced Networking (PROTEAN) Research Group. The goal of this effort is to provide an implementation of the Internet Engineering Task Force (IETF) standards track protocol NHDP, developed within the Mobile Ad hoc Networking (MANET) working group. The *nrlnhdp* application supports multiple interface operation with independent settings per interface. It maintains a single graph two hop graph structure which contains the neighborhood information. Both IPv4 and IPv6 operation are supported. Versions of *nrlnhdp* can be built for the following operating systems: Linux, MacOS; with BSD, Win32, and others pending.

2 Theory of Operation

The *nrlnhdp* program uses hello protocol messaging to build and maintain a local two-hop neighborhood state. These hello messages conform to the packet format defined in RFC 5444. All NHDP nodes periodically send hello messages on each of their NHDP enabled interfaces to advertise there presence (all of their IP addresses) to other NHDP protocol enabled nodes/interfaces. Each *nrlnhdp* enabled machine maintains a single neighborhood state which contains all discovered IP addresses of one and two hop neighbors, link state between those neighbors (HEARD, SYMMETRIC, LOST) and IP node associations where appropriate. Each interface running *nrlnhdp* maintains its own link state to one hop connected interfaces. Upon receiving a hello a message from a neighboring node, future hello packets on that receiving interface will advertise the addresses listed as HEARD. Upon receiving a hello message, in which the local interface is listed as HEARD (your neighbors hello message lists you as HEARD), the local link state transitions to SYMMETRIC. State times out after a certain period of time without reception of an update. Links are then advertised as LOST for a amount of time. By sending hello messages with all known (HEARD, SYMMETRIC) links and all recently known (LOST) links receivers of those hellos build and maintain two hop neighborhood databases.

3 Usage

The *nrlnhdp* program provides a command-line syntax with which it can be launched. Many of the same commands (see Commands) available via the command-line may be applied during run-time using the "remote control" interprocess control interface (see Remote Control Interface). To launch *nrlnhdp*, use the following command-line syntax:

```
nrlnhdp [-d <debugLevel>][-l <debugLogFile>][-h][-v]
        [-i <interfacename> <Interface::Options>]
```

```
<Interface::Options>
        [-b <broadAddress> <maskLength>][-port <portNumber>
        [-hi <HelloInterval>][hmi HelloMinInterval][-hpj HelloPeriodicJitter>][-hht <He
        [-ipv6][-ipv4]
        [-qos <qosvalue>]
```

3.1 Usage Examples

To get *nrlnhdp* version information:

```
"nrlnhdp -v"
```

To get basic *nrlnhdp* usage help:

```
"nrlnhdp -h"
```

To run *nrlnhdp* on a single interface "eth0" with default settings :

```
"nrlnhdp -i eth0"
```

To run *nrlnhdp* on a two interface "eth0" and "192.168.0.101" with default settings :

```
"nrlnhdp -i eth0 -i 192.168.0.101"
```

An example running *nrlndhp* on two interfaces with the "eth0" using non-default settings:

```
"nrlndhp -i eth0 -hi 60.0 -hj 30 -qos 7 -i eth1"
```

In the previous example hello messages will be sent out on eth0 once every sixty seconds with a jitter of 30 seconds. Hello messages will be sent out at the default rate on eth1.

To get debug information in a file /tmp/nhdp.log

```
"nrlnhdp -d 8 -l /tmp/nhdp.log -i eth0"
```

Note that -d and -l are global options and MUST come before any -i option. Any command line option following a -i option will be assigned to that interface.

By default *nrlnhdp* uses the default multicast address and default port as defined in RFC 5498. 224.0.0.109 for IPv4 and FF02::6D for IPv6 with port 269.

An example of running *nrlnhdp* using a different broadcast/multicast address/port.

```
"nrlnhdp -i 192.168.0.101 -b 192.168.0.0 16 -port 5001"
```

4 Commands

The *nrlnhdp* program supports a variety of commands that can be invoked via command-line at startup or at run-time using the "remote control" interprocess communication interfaces that is provided. There a three categories of commands:

1. "Global Commands" to control or enable general *nrlnhdp* capabilities and features.
2. "Interface Commands" to enable and control NHDP operation on network interfaces

It should be noted that all of the commands listed here can be invoked via the run-time remote control interface (see Remote Control Interface).

Global Commands

`-v`	Outputs version information.
`-h`	Outputs basic usage information.
`-d <debugLevel>`	Assigns the debug level for all interfaces where debugLevel is an integer from 0-10. Higher numbers will provide more detailed output. Default value is 0.
`-l <debugLogFile>`	Assigns the file location to send debug output to, where debugLogFile is the file location. If partial paths are given the location in which *nrlnhdp* was launched will be used as the current directory. Default is stderr.
`-i <interfaceName>`	This command invokes an instance of ndhp for a specific interface where <interfaceName> is the name or IP address of an active interface on the machine. All commands other than other "-i" commands which follow a "-i" command are interpreted by that interface instance, ie all "Global Commands" must proceed any "-i" command. If no "-i" command is given *nrlnhdp* will run but no messages will be sent/recieved, however the Remote Control Interface will be established.

The "Interface Commands" apply specific interfaces. These commands MUST follow a "-i <interfaceName> command to be valid. As these commands are specific to the interface which proceeded it different interfaces can and will end up with different settings. This can be useful for configuring heterogeneous interfaces with parameters which more closely match the physical properties of those interfaces.

Interface Commands

`-b <broadAddress> <maskLength>`	*nrlnhdp* will use the provided broadcast address for messaging. This broadcast address can be a single multicast address or a network broadcast address. Default is 224.0.109 32 when in IPv4 mode and FF02::6D 64 in IPv6 mode.
`-hi <HelloInterval>`	*nrlnhdp* sends "Hello" messages at periodic intervals. <HelloInterval> is a double which represents in seconds the amount of time to set this interval. Default is 2.0 seconds.
`-hmi <HelloMinInterval>`	*nrlnhdp* provides a method for sending "Hello" messages faster than the Hello Interval (both opportunistically(TBD) and) through the use of jitter. This command sets the minimum amount of time in seconds between sent "Hello" messages. <HelloMinInterval> is a double in seconds. This value should be significantly less than the HelloInterval (10%-25%) unless tight periodic timings are desired. Default is .5 seconds.
`-hpj <HelloPeriodicJitter>`	To prevent pathological collisions of packets based on periodic timing jitter is introduced to randomly spread Hello messages over time. For each sent "Hello" message the next message is scheduled for sending at <HelloInterval> time in the future. A random amount of jitter is subtracted from the scheduled time for the next "Hello" See RFC 5148 for more specific information. <HelloPeriodicJitter> is the maximum ratio of the <HelloInterval> to subtract from the next scheduled time. Valid values are from 0 up to (but not including) 1. A HelloPeriodicJitter value of 0 will provide no jitter. A value of .99 can possibly send messages with very close timing. Default is a .5 ratio.

`-hht <HelloHoldTime>`	NHDP messages must include a validity time to convey to recievers the amount of time for which the information contained in the "Hello" message is valid. This value in *nrlnhdp* is set using this option, where <HelloHoldTime> is a double with value from <HelloInterval> to MAXDOUBLE. This value is often set to 3-5 times the amount of <HelloInterval>. Much larger values of validity time may be appropriate depending on network properties. Much smaller numbers are often not appropriate. Default is 6.0 seconds.
`-ipv6`	*nrlnhdp* will operate with IPv6 addresses. This option when used, should be used among all interfaces. Default is NOT USED.
`-ipv4`	*nrlnhdp* Will operate with IPv4 addresses. This option should be used among all interfaces. Default is USED.
`-qos <qosValue>`	*nrlnhdp* can set the set the type of service bit. This by itself will not effect performance but may depending on network/interface configuration may be used to give precedence to NHDP messages. <qosValue> is a positive integer. Default N/A.
`-port <portNumber>`	Sets the port on which to listen/send NHDP messages. The default of 269 is provided by RFC 5498. For certain systems this port number is restricted and firewalls may block this by default. <portNumber> is a positive integer. Default is 269.

5 Run-Time Remote Control

The *nrlnhdp* application provides an interprocess communication "remote control" interface for receiving run-time instructions from other processes. The "remote control" interface is identified by a canonical name <*instanceName*>. The default name used by *nrlnhdp* is, strangely enough, "nrlnhdp".

On UNIX systems, the `<instanceName>` corresponds to a Unix-domain datagram socket named "`/tmp/<instanceName>`" that is opened and monitored for commands (thus the default *nrlnhdp* Unix-domain socket would be identified as "`/tmp/nrlnhdp`"). On WIN32 systems, a "mailslot" named "`\\.\mailslot\<instanceName>`" is created and used while on WinCE systems a semaphore is instantiated along with a corresponding registry entry mapping to a locally-bound UDP socket provides equivalent functionality.

6 Future Plans

There are a number of additional features and refinements planned for the *nrlnhdp* implementation. Some of these include:

1. Support SMF operation.
2. OLSRv2 implimentation based on the *nrlnhdp*.
3. Provide a method for database information extraction.
4. Option to load a "config" file for complex configurations.
5. More robust packet handling for non-*nrlnhdp* generated NHDP messages.