DynIP Documentation

Release 0.1e

Kris Hardy

Sep 19, 2017

Contents

CHAPTER 1

About DynIP

DynIP is an embarrasingly-simple client and server purely for getting systematic updates of a client's IP address.

How it works

The client is typically configured to run on a schedule basis using crontab or Windows Scheduled Tasks. When launched, the client sends a UDP datagram to the server, with it's hostname in the data packet.

The server, when it receives the UDP datagram, opens a JSON file at the path specified in the `CLIENT_LOG_PATH`. The server looks to see if the client name (the data portion of the packet) is already in the list of known addresses. If it is, it updates the record with the new IP address from the packet header and the current date and time. If it is new, it adds it to the dict of known hosts.

Installation

Either download the source tarball from PyPi and install with *python setup.py install*, or use easy_install or pip:

```
$ easy_install DynIP
-or-
$ pip install DynIP
```

CHAPTER 2

Usage

Configuration

First, build a configuration file to start from:

```
$ dynip-init [path for configuration file]
```

DynIP will create a configuration file at the path you specify.

To configure the server, copy or edit the .conf file and set the `server_ip`, `server_port` and `client_log_path` parameters in the `DynIP:Server` section. Descriptions of each are in the .conf file.

Launching the Server

Start the server by typing:

```
$ dynipd [path to .conf file]
```

The server will endlessly listen for packets until it is killed (by pressing CTRL-C if you launched it in the foreground, or killing it by the PID if you launched it in the background).

Launching the Client

To launch the client and have it fire off a UDP packet, edit your .conf file, and modify the `server_hostname` and `server_port` lines in the `DynIP:Client` section. Descriptions of each are in the code.

Then launch the client by typing:

```
$ dynipc [path to .conf file]
```

The client will fire a single UDP packet to the server. The server will then save the hostname, ip address and date/time in the file specified in `client_log_path` in the .conf file.

Options

To enable verbose (INFO-level) logging, add *-v* to the command line when launching the server or client.

```
$ dynipd -v [path to .conf file]
$ dynipc -v [path to .conf file]
```

To enable debug (DEBUG-level) logging, add *–debug* to the command line when launching the server or client.

```
$ dynipd --debug [path to .conf file]
$ dynipc --debug [path to .conf file]
```

Both the server and client return usage information if you add *–help* or *-h* to the command line when invoking the client or server.

CHAPTER 3

Why I Built It

I am about to go on an extended trip away from home, yet need to know the dynamic IP address that is assigned to my home cable router so that I can easily help my family remotely in case they need help with their computers. In order to do this, I need to know the IP address that my cable provider as assigned to my cable modem.

By installing the client on each of my computers at home and triggering the client to run every 5 minutes, the server will always keep a log of the most recent public IP address of the boundary device.

CHAPTER 4

License

DynIP is licensed under the BSD License. See the LICENSE file.

CHAPTER 5

Modules

dynip

dynip.server

DynIP Server

A embarrisingly-simple server which listens for UDP packets and logs the data, the source IP address and time.

dynip.server.**listen_loop**(*sock*, *client_data*, *client_log_path*)

A blocking loop that listens for UDP packets, logs them, and then waits for the next one. Exits when a KeyboardInterrupt is caught.

Parameters

- **sock** (*socket.socketobject*) – The bound socket.socketobject
- **client_data** (*dict*) – The in-memory client data dict that is written out to the `client_log_path` on receipt of each packet.
- **client_log_path** (*str*) – The filepath to the JSON-encoded client log file

dynip.server.**main**()

Listen for UDP packets and save the remote IP address and data to the file specified in `client_log_path` in the configuration file

Notes: This reads the entire JSON file in on each packet, so this is not suitable for any significant load or anything but a trivial number of clients.

dynip.server.**usage**()

Print usage information

dynip.client

DynIP Client

A embarrisingly-simple client with sends UDP packets to the DynIP server.

dynip.client.**main**()
Send a single UDP datagram to the server

dynip.client.**send_packet**(*destination_ip*, *destination_port*)
Send a single UDP packet to the target server.

Parameters

- **destination_ip** – IP address of the server
- **destination_port** (*int*) – Port number of the server

dynip.client.**usage**()
Print usage information

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

d

Index