pHostInfo: Detecting and Sharing Host Info

June 2018

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139

1 Overview

The `pHostInfo` application is a tool with a simple objective - determine the IP address of the machine on which it is running and post it to the MOOSDB. Although this information is available in a number of ways to a user at the keyboard, it may not be readily available for reasoning about within a MOOS community. Often, from an application's perspective, the host name is simply known and configured as *localhost*. This is fine for most purposes, but in situations where a user is on a machine where the IP address changes frequently, and the user is launching MOOS processes that talk to other machines, it may be very convenient to auto-determine the prevailing IP address and publish it to the MOOSDB. The typical usage scenario for `pHostInfo` is shown in Figure 1.

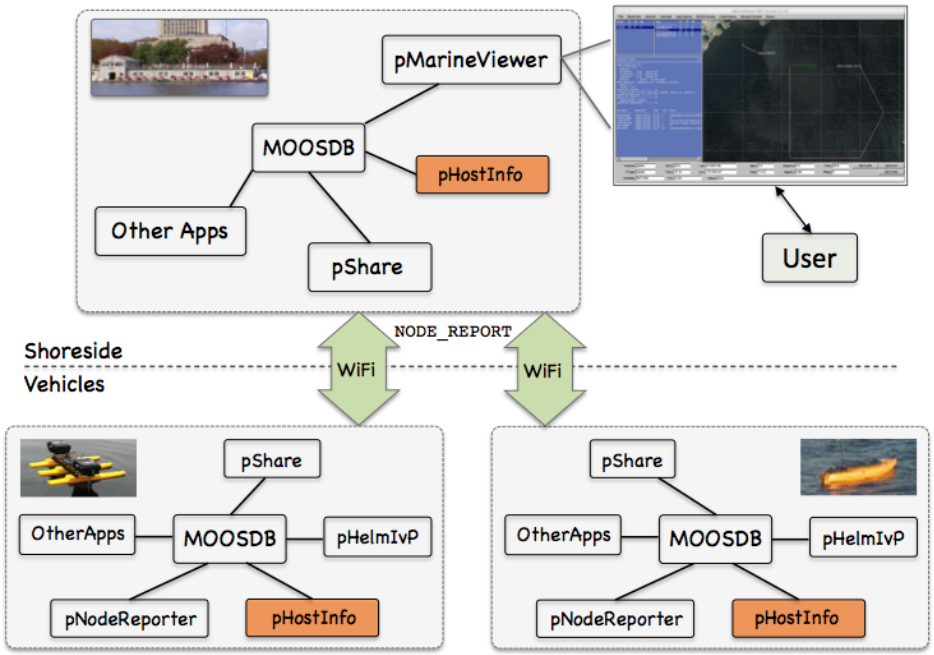

Figure 1: **Typical pHostInfo Topology:** A shoreside or topside community is receiving information from several deployed vehicles, in the form of node reports. The node reports contain time-stamped updated vehicle positions, from which the speed and distance measurements are derived and posted to the shoreside MOOSDB.

There are two scenarios where this is currently envisioned to be useful. The first is when the fielded vehicles are on the network with their IP addresses set via DHCP. For example if on the network via a cellular phone connection. The second is if the "vehicle" is a simulated vehicle running on a user's laptop also with its IP address set via DHCP. In both cases there may be another MOOS community with a known IP address, e.g., a shoreside community, to which the local vehicle wishes to inform it of its current IP address. This simple process however does not get involved in any activity regarding the communication to other MOOS communities, but simply tries to determine and post, the IP address, e.g., `PHI_HOST_IP ="192.168.0.1"` for other applications to do as they see fit.

2 Configuration Parameters for pHostInfo

The `pHostInfo` application may be configured with a configuration block within a MOOS mission file, typically with a `.moos` file suffix. The following parameters are defined for `pHostInfo`.

Listing 2.1: Configuration Parameters for pHostInto.

`temp_file_dir`:	Directory where temporary files are written. Default is `"~/"`.
`default_hostip`:	IP address used if no IP address can otherwise be determined.
`default_hostip_force`:	This IP address will override any IP address from an auto-discovered network interface. Useful for debugging.

prefer_inerface_: If multiple interfaces have a valid IP address then a preference can be specified. Valid options are `wlan0`, `wifi`, `eth0`, `eth1`, `usb0`, `usb1`,`usb2`.

An Example MOOS Configuration Block

An example MOOS configuration block may be obtained from the command line with the following:

```
$ pHostInfo --example or -e
```

Listing 2.2: Example configuration of the pHostInfo application.

```
1  ============================================================
2  pHostInfo Example MOOS Configuration
3  ============================================================
4
5  ProcessConfig = pHostInfo
6  {
7    AppTick   = 4
8    CommsTick = 4
9
10   temp_file_dir  = ./
11   default_hostip = 192.168.0.55    // default is "localhost"
12
13   default_hostip_force = 192.168.0.99
14 }
```

3 Publications and Subscriptions for pHostInfo

The interface for pHostInfo, in terms of publications and subscriptions, is described below. This same information may also be obtained from the terminal with:

```
$ pHostInfo --interface or -i
```

3.1 Variables Published by pHostInfo

The primary output of pHostInfo to the MOOSDB are the following five PHI_* variables. Once these variables are published, pHostInfo does not publish them again unless requested by receiving mail HOST_INFO_REQUEST. Thus pHostInfo is mostly idle once the below five variables are posted.

- APPCAST: Contains an appcast report identical to the terminal output. Appcasts are posted only after an appcast request is received from an appcast viewing utility.
- PHI_HOST_IP: The single best guess of the Host's IP address.
- PHI_HOST_IP_ALL: A comma-separated list of IP addresses if multiple addresses detected.
- PHI_HOST_IP_VERBOSE: A comma-separated list of IP addresses, with source information, if multiple addresses detected.
- PHI_HOST_PORT_DB: The port number of the MOOSDB for this community.

- PHI_HOST_PORT_INFO: A comma-separated list of parameter-value pairs describing all relevant aspects of the host.

3.2 Variables Subscribed for by pHostInfo

The pHostInfo application subscribes to the following MOOS variable:

- APPCAST_REQ: A request to generate and post a new apppcast report, with reporting criteria, and expiration. Section **??**.
- HOST_INFO_REQUEST: A request to re-determine and re-post the platform's host information.
- PSHARE_INPUT_SUMMARY: The input routes used by pShare for listening for incoming messages from other MOOSDBs.

3.3 Command Line Usage of pHostInfo

The pHostInfo application is typically launched with pAntler, along with a group of other modules. However, it may be launched separately from the command line. The command line options may be shown by typing:

```
$ pHostInfo --help or -h
```

Listing 3.3: Command line usage for the pHostInfo tool.

```
1  Usage: pHostInfo file.moos [OPTIONS]
2
3  Options:
4    --alias=<ProcessName>
5        Launch pHostInfo with the given process
6        name rather than pHostInfo.
7    --example, -e
8        Display example MOOS configuration block
9    --help, -h
10       Display this help message.
11   --HOSTIP=<HostIP>
12       Force the use of the given IP address as the reported IP
13       address ignoring any other auto-discovered IP address.
14   --interface, -i
15       Display MOOS publications and subscriptions.
16   --version,-v
17       Display the release version of pHostInfo.
18
19 Note: If argv[2] is not of one of the above formats
20       this will be interpreted as a run alias. This
21       is to support pAntler launching conventions.
```

4 Usage Scenarios for the pHostInfo Utility

4.1 Handling Multiple IP Addresses

It is possible that a machine has more than one valid IP address at any given time, e.g., if its ethernet cable is plugged in, and it has a wireless connection. In this case, pHostInfo will make a guess that

the ethernet connection takes precedent, and it will report this in the variable PHI_HOST_IP. The full set of IP addresses can be found in the other postings. For example it may not be uncommon to see something like the following three postings at one time:

```
PHI_HOST_IP         = 118.10.24.23
PHI_HOST_IP_ALL     = 118.10.24.23,169.224.126.40
PHI_HOST_IP_VERBOSE = OSX_ETHERNET2=118.10.24.23,OSX_AIRPORT=169.224.126.40
```

5 A Peek Under the Hood

The pHostInfo application currently only works for GNU/Linux and Apple OS X. It determines the IP information by making a system call within C++. A system call when generated will act as if the argument were typed on the command line. In this case the system call is generated and the output is redirected to a file. In a second step, pHostInfo then tries to read the IP address information from those files.

In GNU/Linux, the system call is based on the `ifconfig` command. In OS X, the system call is based on the `networksetup` command. Rather than determining in pHostInfo whether the user is running in a GNU/Linux or OS X environment, the system calls for both are invoked. Presumably a system call on a command not found in the user's shell path will not generate something that is confusable with a valid IP address.

5.1 Temporary Files

The temporary files are written to the user's home directory by default. This may be changed with the temp_file_dir cofiguration parameter, for example, temp_file_dir=/tmp. The set of temporary files are put into a folder named `.phostinfo/`. The set of temporary files may look like:

```
  $ cd ~/.phostinfo
  $ ls -a .ipinfo*
.ipinfo_linux_ethernet.txt    .ipinfo_osx_airport.txt     .ipinfo_osx_ethernet2.txt
.ipinfo_osx_wifi.txt          .ipinfo_linux_wifi.txt      .ipinfo_osx_ethernet1.txt
.ipinfo_osx_ethernet.txt
```

Some of these files may be empty, or some may contain error output if one of the system commands was not found, or was given an improper argument. The pHostInfo app will try to parse all of them to find a valid IP address. If more than one IP address is found, then this handled in the manner described previously in Section 4.1.

5.2 Possible Gotchas

The system calls invoked by pHostInfo need to be in the users shell path. A typical user default environment would have these in their shell path anyway, but it may be worth checking if things aren't working properly. Below is a list of commands that are run under the hood, and their probable locations on your system.

```
For Linux:
  /sbin/ifconfig
  /bin/grep
  /usr/bin/cut
  /usr/bin/awk
  /usr/bin/print
For OS X:
  /usr/sbin/networksetup
```