



Switching between DHCP and Static IP address Acquisition in NET+OS

1 Document History

Date	Version	Change Description	
2/20/2014	V1	Initial Entry	
2/26/2014	V1.1	Continued entry	
2/27/2014	V1.2	Continued entry	
3/18/2014	V1.3	Continued entry	
3/18/2014	V1.4	Fill in glossary	
3/19/2014	V1.5	Grammar edits	

2 Table of Contents

1	Document History.....	2
2	Table of Contents.....	3
3	Introduction.....	4
3.1	Problem Solved.....	4
3.2	Audience.....	4
3.3	Assumptions.....	4
3.4	Scope.....	4
3.5	Theory of Operation.....	5
4	Basics.....	5
4.1	Why did my DHCP-acquired IP address become unavailable?.....	5
4.1.1	I acquired an IP address via DHCP but lost it.....	5
4.1.2	I never had an IP address in the first place.....	6
4.2	How do I go about switching from DHCP to static address acquisition?.....	6
4.3	How do I go about switching from Static to DHCP address acquisition?.....	6
4.4	Why would I save to NVRAM?.....	7
5	Example Application Explanation.....	7
5.1	fromStaticToDHCP.....	8
5.1.1	Don't update NVRAM.....	8
5.1.2	Update NVRAM.....	8
5.2	fromDHCPToStatic.....	9
5.2.1	Don't update NVRAM.....	9
5.2.2	Update NVRAM.....	9
6	Conclusion.....	9
7	Appendix.....	10
7.1	Glossary of terms.....	10

3 Introduction

This paper describes a method for programmatically switching between DHCP and static IP address acquisition methods in the NET+OS development environment.

3.1 Problem Solved

Many embedded devices, developed using the NET+OS development environment set their primary network interfaces to acquire their IP address via DHCP. A problem can arise when the DHCP server becomes unavailable. So the question is how can the application deal with this issue? One solution is to use the device's autoip address. Since autoip does not route, this is non-optimal solution. Another option that I have heard from customers is to switch over to a static IP address until the DHCP server becomes available again. This paper describes how to accomplish this.

The method used to describe implementing this IP acquisition method switching will be through a sample application. The application switches back and forth between acquiring its IP address via DHCP method and static method. A sample application that performs this is not very useful except in demonstrating the switching. You can set up two DOS shells, one pinging the DHCP-acquired IP address and the other pinging the DHCP-acquired IP address. You'll be able to see the switching back and forth as one will succeed and one will fail, alternately.

3.2 Audience

This paper is intended for a reader with knowledge of IP addressing, IP address acquisition methods and developing applications in the NET+OS development environment.

3.3 Assumptions

This paper assumes that your device has the need to switch between acquiring its IP address statically and from a DHCP server. This paper describes a method for doing so.

3.4 Scope

This paper has a very narrow focus. It looks and explains setting up a module to switch between DHCP and static IP address acquisition methods. This paper does not discuss any of the following topics:

- General NET+OS development
- AWS
- SNMP
- Sntp
- Email
- TCP/IP
- Socket programming
- HTML

- javascript

3.5 Theory of Operation

The application uses IAM calls to either release the DHCP-acquired IP address and disable DHCP IP address acquisition while enabling static IP address acquisition or release the statically acquired IP address and disable static IP address acquisition while enabling DHCP IP address acquisition. Additionally, when running the application, you have the option of continually updating NVRAM or updating “live” interface information only.

4 Basics

The basic assumption of this application and this paper is that your device has stored a statically acquired IP address into NVRAM using IAM APIs. Also I assume that the static method of acquisition on this interface is disabled, by default. Further I will assume that you initially want your device to boot up and acquire its IP address via DHCP. If the DHCP-acquired address becomes unavailable, you’d like to switch to the statically-acquired IP address.

So, the initial state of your main IP interface is as follows: DHCP is enabled, static is disabled and a static IP address is associated with the static state. This static IP address was pulled from NVRAM.

4.1 Why did my DHCP-acquired IP address become unavailable?

There are two reasons that come to my mind as to why my DHCP-acquired IP address became unavailable. Both reasons involve the DHCP server being unavailable. So let’s quickly look at these two cases.

4.1.1 I acquired an IP address via DHCP but lost it.

Generally, when your device boots up it will issue a DHCP discover broadcast packet. Hopefully a DHCP server will receive it and responds with a DHCP offer. The offer includes a proposed IP address that the DHCP server will lease to you. If your device successfully received the offer, your device will respond with a DHCP request for that IP address. Finally the DHCP server replies with either an ACK (OK) or a NAK (not OK). If your device received an ACK, then your device has an IP address.

In the previous paragraph, I mentioned that the DHCP server leases the IP address to your device. This means that your device holds that IP address for a limited period of time and then has to request it again. What happens if a DHCP server is not available when your lease expires? Your device would be left with no IP address. This would leave it incommunicado. What now?

If your device is left with no IP address due to the lack of a DHCP server following a DHCP lease expiration, you might still want your device to be available to PCs locally. For this reason you might want to be able to provide your device with a temporary statically-acquired IP address, until such time as a DHCP server becomes available.

4.1.2 I never had an IP address in the first place

In this case, your device boots up and sends out a DHCP discover but your device received no answer from any DHCP server. Again your device has no voice. What are you to do? As described above, you can give your device a statically acquired IP address, until such time as a DHCP server becomes available.

4.2 How do I go about switching from DHCP to static address acquisition?

There is a four step process for transitioning from DHCP to static IP address acquisition. This assumes that you do not want to update NVRAM. If you do not update NVRAM, then when your device reboots, it will start up the same way it did so previously. You may want to update NVRAM while switching and I will touch on this when we look at the application.

The four steps are as follows:

- Release the DHCP acquired address (naIamRelease)
- Disable the DHCP acquisition process (naIamDisable)
- Enable the static acquisition process (naIamEnable)
- A small delay

That is your device should now have the statically acquired IP address that you had previously stored in NVRAM (I will also look at this when we look at the application).

BTW, if you have a terminal emulator (such as hyperterm) connected to the serial port of your device that is assigned to STDOUT, you will be able to see the transition.

You may choose to update NVRAM and thereby change the way your device boots up. This is done by acquiring the current configuration structure from NVRAM, updating its fields as needed and writing that configuration structure back to NVRAM. I will explain how this is done when I explain the application.

4.3 How do I go about switching from Static to DHCP address acquisition?

There is a four step process for transitioning from static to DHCP address acquisition. You do not have to have a valid static-acquired IP address to transition from static to DHCP address acquisition.

The four steps are as follows:

Switching between DHCP and Static IP Address Acquisition Methods

- Release the statically-acquired address (naIamRelease)
- Disable the static acquisition process (naIamDisable)
- Enable the DHCP acquisition process (naIamEnable)
- A small delay

Your device will now begin the four step DHCP IP address acquisition process (described above). Assuming a DHCP server is available; your device should very soon have a DHCP-acquired IP address. Again if you have a terminal emulator attached to the STDOUT serial terminal of your device, you should see the transition.

You may choose to update NVRAM and thereby change the way your device boots up. This is done by acquiring the current configuration structure from NVRAM, updating its fields as needed and writing that configuration structure back to NVRAM. I will explain how this is done when I explain the application.

4.4 Why would I save to NVRAM?

The answer to this is application specific. One way to look at this is to answer the following question: when I reboot my device next time, do I want its characteristics to be the same as when I rebooted it previously? Let's assume your device acquired its IP address using DHCP but due to current events the next time it reboots you'd like it to acquire its IP address statically, then you might want to update NVRAM to reflect this. Conversely, due to local events you may have switched to a statically-acquired IP address, but on its next reboot, you'd like the device to attempt to acquire its IP address using DHCP methods, then you probably do not want to update NVRAM. Again, this is application specific. It is a decision that you'll want to think about.

5 Example Application Explanation

The example can be found at this [link](#).

The application that accompanies this paper was written to further illustrate the concepts involved in using IAM calls to switch between using DHCP-associated and static-associated IP address acquisition methods. Further, this application demonstrates how your application might store/update NVRAM information related to IP address acquisition, as described above. To switch between updating NVRAM and not updating NVRAM modes I have included the macro WANT_TO_UPDATE_NVRAM. When this macro is defined, the application updates NVRAM as it switches from one mode to another. When this macro is undefined, NVRAM is not so updated. You will notice that at the beginning of function changeIPAddressThread, that there is a call to function setUpStaticAddressing. The purpose to initially set up a static IP address in NVRAM so that when the application switches from DHCP to static IP address acquisition modes, there is a static IP address to use. This is not required but it is used for demonstration purposes.

At the beginning of the program is a table of the `IamInfo_t` entries. This is used to control the flow of the main loop and to give the static code an IP address to use. This is probably not required in your application and should be taken as an instrument of demonstration only.

To start out the function `applicationStart` starts a thread that performs the bulk of the work of this sample application. The thread runs function `changeIPAddressThread`.

At about line 248 of file `root.c`, function enters a loop that represents the main loop of the application. The first thing the loop does is acquire the IP address acquisition method associated with the interface I am concerned with. That is the current IP address acquisition method. In my case I am concerned with `eth0`. I use function `anIamGetCurrentMethod`. The function returns the method into variable `theAcqMethod`. I compare the currently used method against enum `NA_IAM_NETHOD_STATIC`. If our current method is static, I want to call function `fromStaticToDHCP`. If the comparison is false then I check the currently used method against `NA_IAM_NETHOD_DHCP`. If that comparison is true, then I call function `fromDHCPToStatic`. If neither is true, then I was returned a method that I am not prepared to deal with. At the end of each loop iteration I call `dumpIamBuffer` which displays information to show that the switch actually took place.

Let's look at function `fromStaticToDHCP`.

5.1 *fromStaticToDHCP*

5.1.1 Don't update NVRAM

First let's assume the application is not updating NVRAM. Around line 558, I call `naIamRelease`. I pass `naIamRelease` the interface (`eth0`) and the method I want to release (`NA_IAM_METHOD_STATIC`). Next I use `naIamDisable` to disable static IP address acquisition on the device. Again I pass the API the interface name and the method. Next I use `naIamEnable` to enable DHCP IP address acquisition on interface `eth0`. Last I've added a loop to wait until the device has completed the transition from static to DHCP IP address acquisition method.

5.1.2 Update NVRAM

Let's assume I want to update NVRAM. First you want to uncomment the declaration of `WANT_TO_UPDATE_NVRAM`. You'll find this at the top of the `root.c` file. Next, since I am transitioning to DHCP, I get the current NVRAM-based DHCP configuration using API `customizeIamGetDhcpConfig`. Notice at the beginning of this function I cleared out all data structures that I will be using. I highly recommend doing this. After getting the current configuration, I set `enabled` to `TRUE`. I then want to write the updated configuration to NVRAM. I do this using API `customizeIamSetDhcpConfig`. The affect of this is that the next time you boot this device, DHCP IP address acquisition method will be enabled. Next I use `customizeIamGetStaticConfig` to get the current state of static

IP address acquisition from NVRAM. I clear out all of the important fields and use API `customizeIamSetStaticConfig` to write the updated configuration controlling static IP address acquisition to be disabled. Now the next time this device is rebooted, static IP address acquisition will be disabled.

5.2 fromDHCPToStatic

5.2.1 Don't update NVRAM

First let's assume the application is not updating NVRAM. Around line 699, I call `naIamRelease`. I pass `naIamRelease` the interface (`eth0`) and the method I want to release (`NA_IAM_METHOD_DHCP`). Next I use `naIamDisable` to disable DHCP IP address acquisition on the device. Again I pass the API the interface name and the method. Next I use `naIamEnable` to enable static IP address acquisition on interface `eth0`. Last I've added a loop to wait until the device has completed the transition from DHCP to static IP address acquisition method.

5.2.2 Update NVRAM

Let's assume I want to update NVRAM. First you want to uncomment the declaration of `WANT_TO_UPDATE_NVRAM`. You'll find this at the top of the `root.c` file. Next, since I am transitioning from DHCP, I get the current NVRAM-based DHCP configuration using API `customizeIamGetDhcpConfig`. Notice at the beginning of this function I cleared out all data structures that I will be using. I highly recommend doing this. After getting the current configuration, I set `enabled` to `FALSE`. I then want to write the updated configuration to NVRAM. I do this using API `customizeIamSetDhcpConfig`. The affect of this is that the next time you boot this device, DHCP IP address acquisition method will be enabled. Next I use `customizeIamGetStaticConfig` to get the current state of static IP address acquisition from NVRAM. I set `enabled` to `TRUE` to enable static IP address acquisition. I need to set up all of the relevant static IP address fields including IP address, subnet mask and gateway. I use API `customizeIamSetStaticConfig` to write the updated configuration controlling static IP address acquisition to be disabled. Now the next time this device is rebooted, static IP address acquisition will be enabled and the device will have an IP address, a subnet mask and a gateway.

6 Conclusion

If the world was a perfect place, then our DHCP server would always be up and I'd never have to be concerned about acquiring a static IP address. Unfortunately I know that DHCP servers do, on occasion become unavailable. When they do, you may want to give your device the ability to temporarily acquire a static IP address until your DHCP server is available again. This paper described a method for transitioning between DHCP and static methods of IP address acquisition. Further I have stated that updating NVRAM is optional but have described methods for updating NVRAM with IP address acquisition methods, should your application have a requirement to do so.

We hope you find this information useful.

7 Appendix

7.1 Glossary of terms

- ACK – A positive acknowledgement. A message sent from a server signifying the positive result of some action
- AWS – Advanced Web Server. The web server included with Digi's NET+OS development environment
- DHCP – Dynamic Host Control Protocol. A protocol included within the TCP/IP suite of protocols. Used for allowing a device to acquire an IP address and other attributes over the network
- HTML – Hypertext Markup Language. The main language used to create web pages
- IAM – Internet Address Manager. A set of APIs contained within Digi's NET+OS development environment used for manipulating IP addresses.
- NAK – Negative Acknowledgement. A message sent from a server signifying the negative result of some action
- NVRAM – Non-volatile Random Access Memory. In the case of the NET+OS development environment, a portion of the FLASH device used to store system attributes and their values that will survive power cycles. Generally the last one or two sectors of the FLASH device.
- SNMP – Simple Network Management Protocol – A protocol that is part of the TCP/IP suite of protocols and generally used to manage devices connected to the network.
- SNTP – Simple Network Time Protocol – A protocol that is part of the TCP/IP suite of protocols and used to acquire and manage time within a networked device.
- Static – In the context of a device's IP address, an IP address that stays constant and does not have to be requested from a server. As opposed to a DHCP-acquired IP address which is requested from a DHCP server.
- STDOUT – Acronym for Stand Output. The port over which the output from printf commands are written.
- TCP/IP – A suite of protocols used for communications between devices over the internet.