

# **Freenet: Setup Experiments on the Testbed**

**University of Hawaii at Manoa**

## **Project Members**

Dr. Yingyei Dong

Todd Baumeister

Dwayne Yuen

## **Author**

Todd Baumeister

## **Date**

5/19/2011

## Table of Contents

1. Purpose	Page 3
2. Definitions	Page 4
3. Prerequisites	Page 5
4. Pre-exeriment Configuration	Page 6
5. Experiment Setup	Page 7
6. Tools	Page 10
7. First Use Experiment Example	Page 12
8. None First Use Experiment Example	Page 19
9. References	Page 21

## 1. Purpose

The purpose of this document is to outline the execution of experiments on the Freenet testbed. The Freenet testbed is defined in the document *Freenet Test Bed Design*[3]. This document will cover the prerequisites that must be completed before this document should be used. It will also walk through the pre-experiment setup steps and how to setup and experiment on the testbed. This document also provides an example experiment walk through. The walk through will guide users through setting up the testbed and then starting and stopping all of the Freenet nodes in it.

## 2. Definitions

Virtual Machine (VM) - Software used to emulate physical hardware. In this document we use VMWare to manage our VMs; however, any equivalent product could be used.

VM Server - Physical machine used to host multiple VMs.

Testbed Machine - This is a VM running in our testbed. It is used to emulate a physical machine that a typical Freenet user would have. The testbed machine is used to run the Freenet software.

Freenet Node - This is an instance of the Freenet software running on a testbed machine. Each of the Freenet nodes connected together form a Freenet network.

Seed Node - This is a special kind of Freenet Node that is used to seed new general nodes into the Freenet network.

General Node - This is the kind of node that a user typically is running to browse the Freenet network.

### 3. Prerequisites

This document assumes that the Freenet testbed is already partially configured. The setup instructions in this documents assume that a VM server is already configured. Also that an initial template is created for the Freenet testbed machine VM. The VM template does not need to be finished, but it should at least be created. The directions for the initial Freenet testbed setup can be found in the document *Freenet Testbed Setup*[4].

## 4. Pre-experiment Configuration

This section will cover configuration of the testbed VM template and cloning the template to produce testbed machines. We suggest creating a template VM because it is the easiest way to create a lot of VMs. Also if the template is properly configured before it is cloned, it will drastically cut down on the testbed setup time. The alternative to using the template method is to create scripts to perform setup actions in the testbed such as software installation.

Freenet should be installed on the template VM. The easiest way to do this is install the official Freenet from their website. This way only a few files need to be replaced to get the testbed up and running. Just check that you are able to start Freenet up at this point. It doesn't even need to be able to connect to the network. In addition to installing Freenet, any other software that may be needed during experiments should be installed. This includes software like SSH, network bandwidth limiting software, and others.

Once the template is complete you will clone it to produce all of the testbed machines in your testbed. We have a simple script that automates the process of copying and renaming the VMs. The script can be found at <https://github.com/tbaumeist/FreeNet-Analysis/blob/master/Scripts/server/CloneVMs.sh>. This script will need to be customized to meet your needs before it is run. Additional information on this script can be found in the document *Freenet Testbed Setup*[4]. After you have created all of the testbed machines that you will need, then boot up the VMs and configure them. This will include things such as assigning a static IP, renaming the computer, and any other unique settings that need to be changed.

Now that the testbed machines are ready to be used, you will probably need to setup a computer to build the custom Freenet software. The source for our version of Freenet can be found at [https://github.com/tbaumeist/FreeNet-Analysis/tree/master/eclipse\\_workspace/FreeNet-Source](https://github.com/tbaumeist/FreeNet-Analysis/tree/master/eclipse_workspace/FreeNet-Source). This version of Freenet has several modifications to facilitate the testbed. It doesn't check file hashes, which allows us to modify configuration files easily. It also has several minor bug fixes. Instructions for setting up a build environment can be found in the document *Freenet: Setup Eclipse and Debug Environment*[2]. Build this Freenet project will create a freenet.jar file that can be distributed to all of the testbed machines. The steps for distributing this file and others will be described later in section 5.

## 5. Experiment Setup

Section 7 should be used in conjunction with this section. Section 7 will provide you with a walk through of performing the steps in this section. This section will explain why the steps are performed in the order that they are. Section 7 will focus more on the commands that you need to run. Section 8 describes the procedures for running a testbed once you setup the testbed and run it a few times.

It is assumed that all of the tasks from section 4 have been completed before doing this section. All of the VMs for the Freenet testbed should be created, configured and running at this point. This section will cover how to run the Freenet maintenance scripts to setup and start the Freenet nodes.

The first task is to choose a machine to run the Freenet maintenance scripts from. This machine must be able to access all of the testbed machines. If the logging scripts are used, the machine will also need to have access to a large data storage. Once a machine is chosen you will need to check out the Freenet Analysis project from GitHub. The command to checkout the project is “git clone git://github.com/tbaumeist/FreeNet-Analysis.git”. This will create a local copy of the repository on your machine. You can then find the Freenet testbed maintenance script in the folder /FreeNet-Analysis/Scripts/. Figure 7.1 gives an example of how to run this command. There are also additional options when checking out the project, and they can be found in the document *Freenet: Setup Eclipse and Debug Environment*[2].

You will now need to set up the testbed configuration file. This file is used by all of the testbed maintenance scripts. The configuration file contains several pieces of information for finding and locating resources on each of the testbed machines. Before setting up the configuration file you will need to decide which Freenet nodes will act as seed nodes. You will need at least one seed node, but we recommend several. Seed nodes will have slightly different information in their configuration file entry. Information about the configuration file can be found in the *script readme* file. An example of a couple entries in the file would look like this:

```
192.168.0.102,SEED,freenetuser,/home/freenetuser/Freenet/,../NodeImages/  
SeedNode/  
192.168.0.103,NODE,freenetuser,/home/freenetuser/Freenet/,../NodeImages/General/
```

The file is delimited by commas. The first column is the IP address of a testbed machine. The second column specifies what type of Freenet node the testbed machine will be running. Currently the options are either SEED which is a seed node or NODE which is a general node. The third column specifies the user name on the testbed machine to log in as. The fourth column is where Freenet is installed on the testbed machine. The last column specifies the *master folder* that the testbed machine is using. A *master folder* is a centralized folder that is used to distribute files to a bunch of testbed machines. When the update script is run it will copy all of the files in the *master folder* to the Freenet install folder on a testbed machine. You will need to create an entry in the configuration file for each of the testbed machines. The default configuration file used by all of our scripts is <https://github.com/tbaumeist/FreeNet-Analysis/blob/>

[master/Scripts/config/remoteMachines.dat](#), and will be located at /FreeNet-Analysis/Scripts/config/remoteMachines.dat on the machine running the maintenance scripts. It is also important that the Freenet install folder entry in the configuration file matches the base folder paths used in the freenet.ini. If they don't match some of the maintenance scripts would work properly.

The next step will be to create seed nodes reference file, but before you do that you need to run the seed nodes. To run the seed nodes first run the update script. Figure 7.2 shows how to run this script. The script will copy all of the files that are in the *master folders* to the testbed machines. You need to run this script so it can replace the freenet.ini with the one that we have configured to run on the testbed. After running the update script you can start the seed nodes. Figure 7.3 shows one way of starting these nodes. After the nodes have completed their start up process they should have a opennet-[port #] file in their install directory (Figure 7.4). After that file is generated you can shut down the seed nodes.

You now have everything that you need to generate a new seed nodes reference file. First create an empty file named seednodes.fref. Then copy/paste the contents of one of the seed node opennet-[port #] in to the file(Figure 7.5). Then go through the pasted content and remove all of the private information entries: clientNonce, location, dsaPrivKey, ark.privURI. The resulting contents should look like figure 7.6. You will need to repeat this process for each of the seed nodes. Each time adding to the end of the seednodes.fref file. The final file will look like figure 7.7. Once the file is complete you will need to put it in the *master folder* for the general Freenet nodes. In the example configuration file entries above, we use ./../NodeImages/General/ as the location for the general node *master folder*. You will want to leave the seednodes.fref in the seed nodes *master folder* empty, so that seed nodes will not try to connect to the other seed nodes.

You are now going to need to run the update script again(Figure 7.2). This will distribute the updated seednodes.fref file to all of the testbed machines. There are several other files that can be found in the *master folders* that are distributed by default. We will briefly explain the changes in these files here. The freenet.ini in both of the *master folders* has several settings purposely removed. These settings are related to identifiable information, that is filled in by the Freenet when the software is started up and they are missing from the ini. You can also find the freenet.jar file in the *master folders*. This is the custom build of the Freenet project. You will be placing your own jar file in here each time that you make source code changes. Typically the default freenet.jar found in the master folders that is from GitHub will be built from the source code currently checked in GitHub. In addition to the files mentioned here, other files that you want to be distributed to the testbed machines can be added.

All of the testbed machines should now be properly configured, and you are ready to start up the Freenet nodes. Figure 7.8 show how to run the script to start up the nodes. This script will quickly cycle through all of the testbed machines and issue the start command. You will need to give the testbed a few minutes to start up though. It takes a few minutes for the Freenet nodes to gather entropy, find neighbors, and connect to neighbors. You can check if a Freenet nodes is connected to other nodes by using a nodes web interface. You can access the



web interface at `http://[IP]:8888` in any web browser. If you on the testbed machine you get to the web interface at `http://localhost:8888`. You can find your peers on the page at “Status” - > “Connections to strangers”.

Now that the Freenet nodes are up and running, you can perform any experiments that you need to. The start up script also starts a logging process. The subject of logging is not covered in this document.

To stop the Freenet nodes you use the same script that you started them with. Figure 7.9 shows you how to run the script.

Now that you are done with the experiment and have stopped the Freenet nodes, you can do any post experiment analysis needed. Section 6 describes some of the additional tools that can be used to analyze the testbed.

You will need to occasionally reset your testbed before an experiment is run. This will remove any data stored on the Freenet nodes. It will put the nodes in a state where they have not connected to any peers yet. Running either the update or clean script will cause the testbed machines to reset their Freenet nodes. Figure 7.10 shows how to run the clean script.

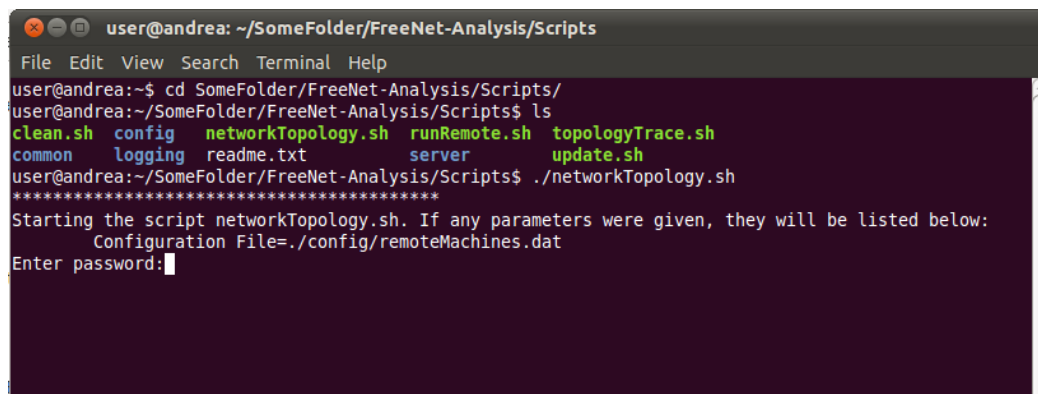
## 6. Tools

### 6.1 Network Topology

One of the tools included with the Freenet testbed is a network topology mapping tool. This is a script that will poll all of the Freenet nodes in the testbed for their neighbor information. The script will then combine all of the information into a topology graph. Additional information about the network topology script can be found in the *script readme*[1].

The network topology script requires the *Graphviz* framework, so it can generate the topology graph. This framework can be found in the *KGraphEditor* package on the Ubuntu OS. The network topology script also requires that the Freenet nodes be running for it to work.

The figure 6.1.a and 6.1.b show an example execution of the network topology script. Figure a uses the default testbed configuration file, and figure b uses a user specified testbed configuration file. The command to run the script is “./networkTopology.sh”. The script can be found in the /FreeNet-Analysis/Scripts/ folder, which is also located at <https://github.com/tbaumeist/FreeNet-Analysis/tree/master/Scripts> on GitHub. When the script is done running it will produce a graphical representation of the network topology. The location that the script saves the output image to can be passed in as a parameter; otherwise it will use the default location specified in the *script readme*[1]. Figure 6.1.c is an example of a possible network topology image that is generated by this script.

A terminal window titled 'user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts'. The window shows the following commands and output:

```
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh  config  networkTopology.sh  runRemote.sh  topologyTrace.sh
common    logging  readme.txt          server         update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./networkTopology.sh
*****
Starting the script networkTopology.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

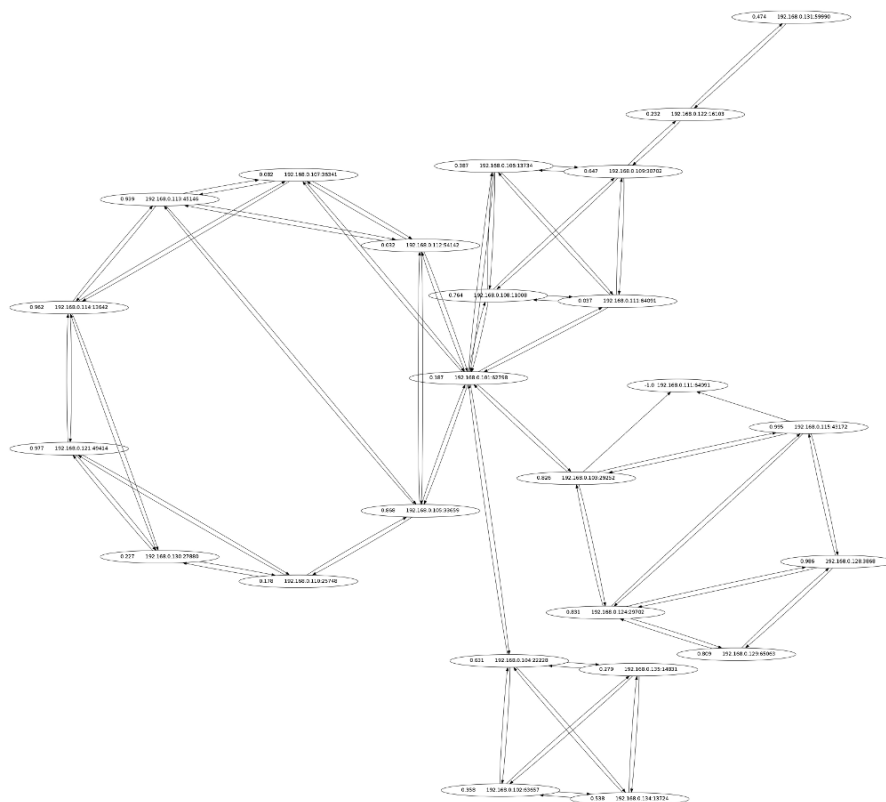
[Figure 6.1.a Network Topology Script]

```

user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh  config  networkTopology.sh  runRemote.sh  topologyTrace.sh
common    logging  readme.txt          server         update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./networkTopology.sh ./config/remoteMachines.dat
*****
Starting the script networkTopology.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:

```

[Figure 6.1.b Network Topology Script with Configuration File Parameter]



[ Figure 6.1.c Example Network Topology]

## 7. First Use Experiment Example

This section should be used with section 5. Section 5 will explain in more detail why you will be running the commands that you are.

We will walk through executing an example experiment in this section. It is assumed that an experiment has never been run on the Freenet testbed yet. If you have already run at least one experiment on the testbed then skip to the example experiment in section 8. Additional information on all of the scripts run in this section can be found in the *script readme*[1].

Note that all of the example commands run in section will have a two very similar figures. We will show executing all of the commands once using the default parameters in the scripts and once specifying custom parameter values. Figures in this section that end in “a” will typically use the default parameter values, and figures that end in “b” will use a custom parameter value.

There are several prerequisites before you can run this example experiment. It is assumed that you have already created several testbed VMs from a single template VM. All of the testbed VMs need to have an executable version of Freenet installed (doesn't need to be configured, just need to be able start the program). Also all of the testbed VMs need to be up and running, and all individual VM settings have already been taken care of. It is also assumed that you have already configured the testbed configuration file. This file is used by all of the scripts in this section. If you have not completed any of these requirements please see section 5 for setup instructions.

The first task is to pick a computer to run the Freenet maintenance scripts from. The computer will need to be able to connect to all of the testbed machines. Typically it is easiest to create an additional VM from the testbed template VM, and use that VM to run the scripts. If the machine used to run the maintenance scripts also runs the logging scripts, it will need to have a large data storage capacity.

Once a computer is chosen to run the scripts, then you will need to check out the scripts from source control. The command to check out the code is “git clone git://github.com/tbaumeist/FreeNet-Analysis.git”. This will check out a read only version. If you want to be able to check changes into the project on GitHub, you will need to contact the project's administrator. Additional instructions about checking out the source can be found in the *Freenet: Setup Eclipse and Debug Environment*[2] document. Figure 7.1 shows how to run the project checkout command.

```
user@andrea: ~/SomeFolder
File Edit View Search Terminal Help
user@andrea:~$ mkdir SomeFolder
user@andrea:~$ ls
Classes      Downloads      missfont.log  Public        texmf
Desktop      examples.desktop  Music         SomeFolder    Videos
Documents    FreeNet-Analysis  Pictures      Templates     workspace
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ git clone git://github.com/tbaumeist/FreeNet-Analysis.git
Initialized empty Git repository in /home/user/SomeFolder/FreeNet-Analysis/.git/
remote: Counting objects: 156224, done.
remote: Compressing objects: 100% (41112/41112), done.
remote: Total 156224 (delta 111153), reused 155868 (delta 110904)
Receiving objects: 100% (156224/156224), 42.20 MiB | 92 KiB/s, done.
Resolving deltas: 100% (111153/111153), done.
user@andrea:~/SomeFolder$ ls
FreeNet-Analysis
user@andrea:~/SomeFolder$
```

[ Figure 7.1 Checkout Freenet Analysis Project]

Now that you have a copy of the scripts on a computer we will set up some seed nodes. You should have already chosen which testbed machines will be seed nodes. We will need to create entries in the seednodes.fref for each of the seed nodes. However, before we can do that the seed nodes need to be run so that they can generate a public private key pair. We are going to run the update script before we start the seed nodes. The update script will replace the original freenet.ini file with the one we have configured for the testbed. Run the “./update.sh” script. Figure 7.2.a and 7.2.b show the update script being run.

```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ ls
FreeNet-Analysis
user@andrea:~/SomeFolder$ cd FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./update.sh
*****
Starting the script update.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

[ Figure 7.2.a Update Script ]

```

user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ ls
FreeNet-Analysis
user@andrea:~/SomeFolder$ cd FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse_workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./update.sh ./config/remoteMachines.dat
*****
Starting the script update.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:

```

[ Figure 7.2.b Update Script with Configuration File Parameter ]

Now that the seed nodes have our freenet.ini, they need to be started. The easiest way to do this is to manually start the Freenet nodes on the the seed node testbed machines. You can either manually run the following command on the seed node testbed machines or SSH into them. To start the nodes run the “./run.sh” command. Figure 7.3 shows this command being run. You can either start the node with the “console” parameter which prints status information out to the console or with the “start” parameter which starts Freenet silently. Once the seed nodes have generated a opennet-[some port #] file, you can shut down the seed nodes.

```

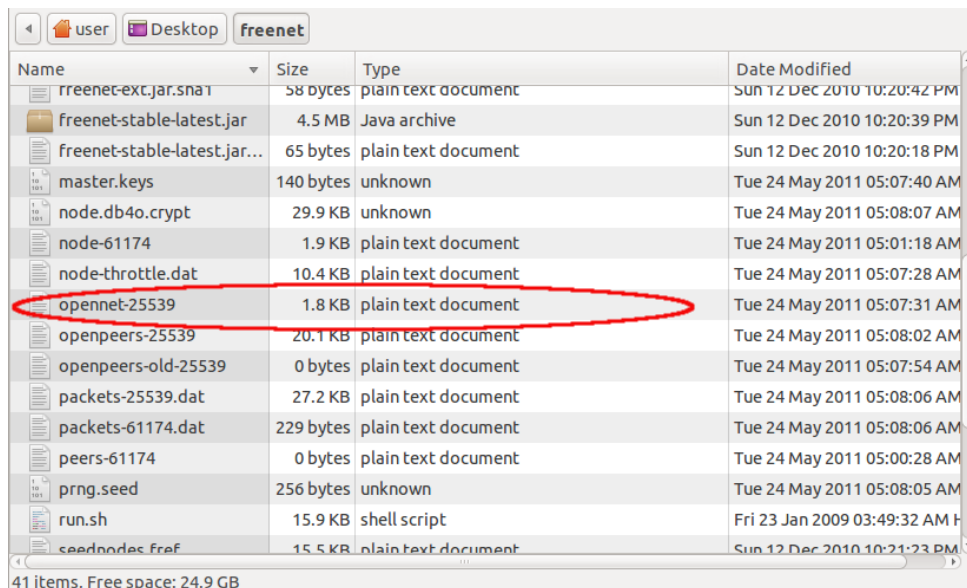
user@andrea: ~/Desktop/freenet
File Edit View Search Terminal Help
user@andrea:~$ cd Desktop/freenet/
user@andrea:~/Desktop/freenet$ ls
bin                license                prng.seed
bookmarks.dat      logs                  run.sh
bootID             master.keys           seednodes.fref
client-throttle.dat node-61174            seednodes.fref.sha1
datastore          node.db4o.crypt       startssl.pem
downloads          node-throttle.dat     temp-61174
extra-peer-data-61174 opennet-25539         update.sh
freenet-ext.jar    openpeers-25539       update.sh.sha1
freenet-ext.jar.sha1 openpeers-old-25539  uptime.dat
freenet.ini        packets-25539.dat     wrapper.conf
freenet.jar        packets-61174.dat    wrapper_Linux.zip
freenet-stable-latest.jar peers-61174          wrapper_Linux.zip.sha1
freenet-stable-latest.jar.sha1 persistent-temp-61174 wrapper.log
lib                plugins
user@andrea:~/Desktop/freenet$ ./run.sh console

```

[ Figure 7.3 Manually Start Freenet Node ]

We now have all of the information that we need to generate a seednodes.fref file. The seednodes.fref file is used by general Freenet nodes to seed themselves into the Freenet network. First create an empty document named seednodes.fref. Next copy the contents of the opennet-[port #] a seed node into the seednodes.fref file. Figure 7.4 and 7.5 show an example opennet-[port #] file and its contents. You will now need to remove the private information from the data you pasted into the seednodes.fref file. Remove the entries “clientNonce, location,

dsaPrivKey, ark.privURI” from the file. Also edit the “physical.udp” entry so that it only contains a single reachable IP address. After cleaning up the entries the file should look similar to Figure 7.6. These steps will need to be repeated for each of the seed nodes. Each new seed node entry gets appended to the end of the seednodes.fref file. Figure 7.7 show a seednodes.fref file with multiple entries. When you have completed the seednodes.fref file place it in the *master folder* for the seed nodes. In our example the seed node *master folder* is located at /FreeNet-Analysis/NodeImages/SeedNode/. Further information on *master folders* can be found in section 5.



Name	Size	Type	Date Modified
freenet-etc.jar.snaf	58 bytes	plain text document	Sun 12 Dec 2010 10:20:42 PM
freenet-stable-latest.jar	4.5 MB	Java archive	Sun 12 Dec 2010 10:20:39 PM
freenet-stable-latest.jar...	65 bytes	plain text document	Sun 12 Dec 2010 10:20:18 PM
master.keys	140 bytes	unknown	Tue 24 May 2011 05:07:40 AM
node.db4o.crypt	29.9 KB	unknown	Tue 24 May 2011 05:08:07 AM
node-61174	1.9 KB	plain text document	Tue 24 May 2011 05:01:18 AM
node-throttle.dat	10.4 KB	plain text document	Tue 24 May 2011 05:07:28 AM
opennet-25539	1.8 KB	plain text document	Tue 24 May 2011 05:07:31 AM
openpeers-25539	20.1 KB	plain text document	Tue 24 May 2011 05:08:02 AM
openpeers-old-25539	0 bytes	plain text document	Tue 24 May 2011 05:07:54 AM
packets-25539.dat	27.2 KB	plain text document	Tue 24 May 2011 05:08:06 AM
packets-61174.dat	229 bytes	plain text document	Tue 24 May 2011 05:08:06 AM
peers-61174	0 bytes	plain text document	Tue 24 May 2011 05:00:28 AM
prng.seed	256 bytes	unknown	Tue 24 May 2011 05:08:05 AM
run.sh	15.9 KB	shell script	Fri 23 Jan 2009 03:49:32 AM
seednodes.fref	15.5 KB	plain text document	Sun 12 Dec 2010 10:21:23 PM

41 items, Free space: 24.9 GB

[ Figure 7.4 Example Opennet File ]

```

opennet=true
identity=VTrzgKnCqLXnpKH0~qq3y06tdE7StzospluRc68~x8Y
clientNonce=3YImgUgSSmhiLsC46BD1iN~GFZFk41lFu163VvjzP1M
location=0.9932114427620465
lastGoodVersion=Fred,0.7,1.0,1310
sig=6a022b684ee263ef5083e299f4e3c53f8d8b25a11df26aec55e5ec3fb59f748a,175b349cd5d5c656255ccd5e93dc401dfcddf64099444cd
version=Fred,0.7,1.0,1310
dsaPubKey.y=VC~e02SGkIcd29k6vVvtkPYUUSl~z6n1Rhm502kybfyRabx89wjsHsNuzdlCARJYd0Jno~tpwQeeGVQJuKkEnZlgkKzSgsJGk6niP0t6
physical.udp=184.158.48.223:25539
dsaGroup.g=UaRatnDByf0QvTLaaAXTMzn1Z15LDTXe~J~g0qXCv0zp83CVngSkb~-bVRuZ9R650Fg~ATKcuw8VJJwn1~A9p5jRt2NPj2EM7bu72085
dsaGroup.q=ALFDNoq81R9Y1kQNVBc5kzmK0VvVwosXY5t9E9S1tN5
dsaGroup.p=AIYIre9VNmM38qPjirGGT~PJjWZBHY0q~JxSYyDFQfZQe0hrx4SUpdc~SppnWD~UHymT7WyX28eV3YjwkVyc~-H5Tc83hPjx8qQc7kQb
dsaPrivKey.x=Z~CjGf0I7dMFVtp~NvtfaBH54bVyg3GF~0wzdNTWfiE
ark.privURI=SSK@AKSzzjwEGgOmLgVCS2VFe0pRK09SoCvN6ft3TfLYG~7~L,bDLE0VgpQuvLY4kb~Pnj7pCQT5C6ZrJ3wugN5mjG~-8,AQECAAE/ark
ark.pubURI=SSK@~xQzkNWixhZDP43Q0puTMJaEJrKRs5oPf9G05p6VwvfQ,bDLE0VgpQuvLY4kb~Pnj7pCQT5C6ZrJ3wugN5mjG~-8,AQACAAE/ark
ark.number=0
auth.negTypes=2;4
End

```

[ Figure 7.5 Example Opennet File Contents ]

```

opennet=true
identity=VTrzgKnCqLXnpKH0~qq3y06tdE7Stzosp1uRc68-x8Y
lastGoodVersion=Fred,0.7,1.0,1310
sig=6a022b684ee263ef5083e299f4e3c53f8d8b25a11df26aec55e3fb59f748a,175b349cd5d5c656255ccd5e93dc401dfcdf64099444cd
version=Fred,0.7,1.0,1310
dsaPubKey.y=VC~e025GkICD29k6vVwtkPYUUSl-z6n1Rhm502kybfyRabxB9wjsHsNuzdlCARJYd0Jno-tpwQeeGVQJuKkEnZigKzSgsJGk6niP0t6
physical.udp=184.158.48.223:25539
dsaGroup.g=UaRatnDByf0QvTlaaAXTMzn1Z15LDTXe-J~g0qXCv0zpz83CVngSkb--bVRuZ9R650Fg-ATKcuw8VJjwn1-A9p5jRt2NPj2EM7bu72085
dsaGroup.q=ALFDNoq81R9Y1kQNVBc5kzmK0VvVcWosXY5t9E9S1tN5
dsaGroup.p=AIYIrE9VNHm38qPjirGGT-PJjWZBHY0q-JxSYyDFQfZQe0hrx4SUdpdc-SppnWD-UHymT7WyX28eV3YjwkVyc--H5Tc83hPjx8qQc7kQb
ark.pubURI=SSK@~xQzkNWixhZDP43Q0puTMJaEJRKR5oPf9G05p6VwfQ,bDLE0VgpQuvLY4kb-Pnj7pCQT5C6ZrJ3wugN5mjG--8,AQACAAE/ark
ark.number=0
auth.negTypes=2;4
End

```

[ Figure 7.6 Example Seednodes.fref File ]

```

opennet=true
identity=VTrzgKnCqLXnpKH0~qq3y06tdE7Stzosp1uRc68-x8Y
lastGoodVersion=Fred,0.7,1.0,1310
sig=6a022b684ee263ef5083e299f4e3c53f8d8b25a11df26aec55e3fb59f748a,175b349cd5d5c656255ccd5e93dc401dfcdf64099444cd
version=Fred,0.7,1.0,1310
dsaPubKey.y=VC~e025GkICD29k6vVwtkPYUUSl-z6n1Rhm502kybfyRabxB9wjsHsNuzdlCARJYd0Jno-tpwQeeGVQJuKkEnZigKzSgsJGk6niP0t6
physical.udp=184.158.48.223:25539
dsaGroup.g=UaRatnDByf0QvTlaaAXTMzn1Z15LDTXe-J~g0qXCv0zpz83CVngSkb--bVRuZ9R650Fg-ATKcuw8VJjwn1-A9p5jRt2NPj2EM7bu72085
dsaGroup.q=ALFDNoq81R9Y1kQNVBc5kzmK0VvVcWosXY5t9E9S1tN5
dsaGroup.p=AIYIrE9VNHm38qPjirGGT-PJjWZBHY0q-JxSYyDFQfZQe0hrx4SUdpdc-SppnWD-UHymT7WyX28eV3YjwkVyc--H5Tc83hPjx8qQc7kQb
ark.pubURI=SSK@~xQzkNWixhZDP43Q0puTMJaEJRKR5oPf9G05p6VwfQ,bDLE0VgpQuvLY4kb-Pnj7pCQT5C6ZrJ3wugN5mjG--8,AQACAAE/ark
ark.number=0
auth.negTypes=2;4
End
opennet=true
identity=VTrzgKnCqLXnpKH0~qq3y06tdE7Stzosp1uRc68-x8Y
lastGoodVersion=Fred,0.7,1.0,1310
sig=6a022b684ee263ef5083e299f4e3c53f8d8b25a11df26aec55e3fb59f748a,175b349cd5d5c656255ccd5e93dc401dfcdf64099444cd
version=Fred,0.7,1.0,1310
dsaPubKey.y=VC~e025GkICD29k6vVwtkPYUUSl-z6n1Rhm502kybfyRabxB9wjsHsNuzdlCARJYd0Jno-tpwQeeGVQJuKkEnZigKzSgsJGk6niP0t6
physical.udp=184.158.48.223:25539
dsaGroup.g=UaRatnDByf0QvTlaaAXTMzn1Z15LDTXe-J~g0qXCv0zpz83CVngSkb--bVRuZ9R650Fg-ATKcuw8VJjwn1-A9p5jRt2NPj2EM7bu72085
dsaGroup.q=ALFDNoq81R9Y1kQNVBc5kzmK0VvVcWosXY5t9E9S1tN5
dsaGroup.p=AIYIrE9VNHm38qPjirGGT-PJjWZBHY0q-JxSYyDFQfZQe0hrx4SUdpdc-SppnWD-UHymT7WyX28eV3YjwkVyc--H5Tc83hPjx8qQc7kQb
ark.pubURI=SSK@~xQzkNWixhZDP43Q0puTMJaEJRKR5oPf9G05p6VwfQ,bDLE0VgpQuvLY4kb-Pnj7pCQT5C6ZrJ3wugN5mjG--8,AQACAAE/ark
ark.number=0
auth.negTypes=2;4
End

```

[ Figure 7.7 Example Seednodes.fref with Multiple Entries ]

Now that we have a seednodes.fref that will work in your testbed, we need to distribute it to all of the general Freenet nodes. The seednodes.fref file should be placed in the general Freenet nodes *master folder*. Once it is there you can run the update script again to distribute the new file. Figure 7.2.a and 7.2.b show how to run the update script.

We are ready to start all of the Freenet nodes. To do this we will be running the run script. Figure 7.8.a and 7.8.b show how to execute the run script. This script will start Freenet on each of the testbed machines. Note that when you execute the run script in start mode, it will also start the logging process. To check if the Freenet nodes are properly connecting load the web interface for a Freenet node and look at the “Status” -> “Connections to strangers” page.



```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ cd FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse_workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:s
```

[ Figure 7.8.a Start Freenet Nodes Script ]

```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse_workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh ./config/remoteMachines.dat
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:s
```

[ Figure 7.8.b Start Freenet Nodes Scripts with Configuration File Parameter ]

You will also execute the run script to stop all of the Freenet nodes. However, you will tell the script to stop the nodes instead of start them. Figure 7.9.a and 7.9.b show how to stop the Freenet nodes.

```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:x
```

[ Figure 7.9.a Stop Freenet Nodes Script ]

```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh  config  networkTopology.sh  runRemote.sh  topologyTrace.sh
common    logging  readme.txt          server         update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh ./config/remoteMachines.dat
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:x
```

[ Figure 7.9.b Stop Freenet Nodes Script with Configuration File Parameter ]

Now that we are able to start and stop the Freenet nodes, we need to be able to reset the Freenet nodes. The clean script will remove stored data and return a node to a point where it has not been connected to a Freenet network yet. Figure 7.10.a and 7.10.b show how to clean the Freenet nodes on the testbed. The clean script should only be run when the Freenet nodes are off.

```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh  config  networkTopology.sh  runRemote.sh  topologyTrace.sh
common    logging  readme.txt          server         update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./clean.sh
*****
Starting the script clean.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

[ Figure 7.10.a Clean Script ]

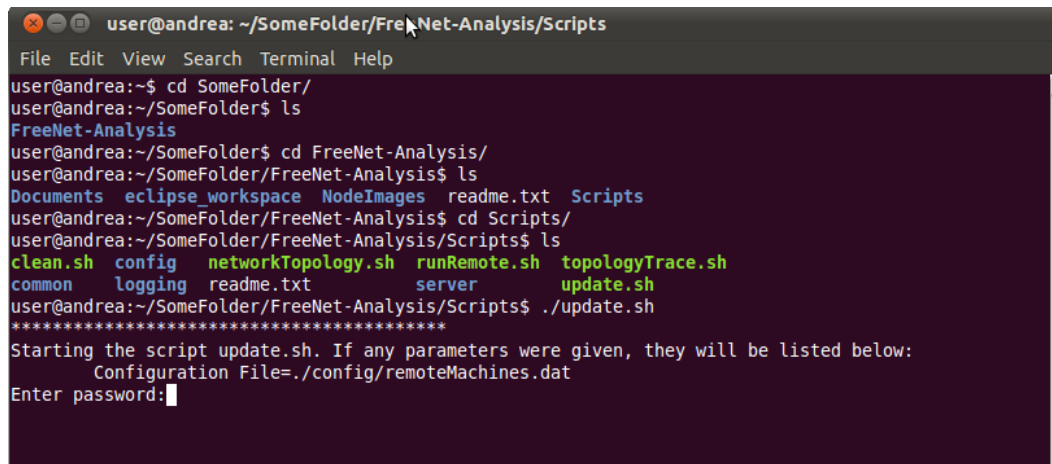
```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh  config  networkTopology.sh  runRemote.sh  topologyTrace.sh
common    logging  readme.txt          server         update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./clean.sh ./config/remoteMachines.dat
*****
Starting the script clean.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

[ Figure 7.10.b Clean Script with Configuration File Parameter ]

## 8. Non First Use Experiment Example

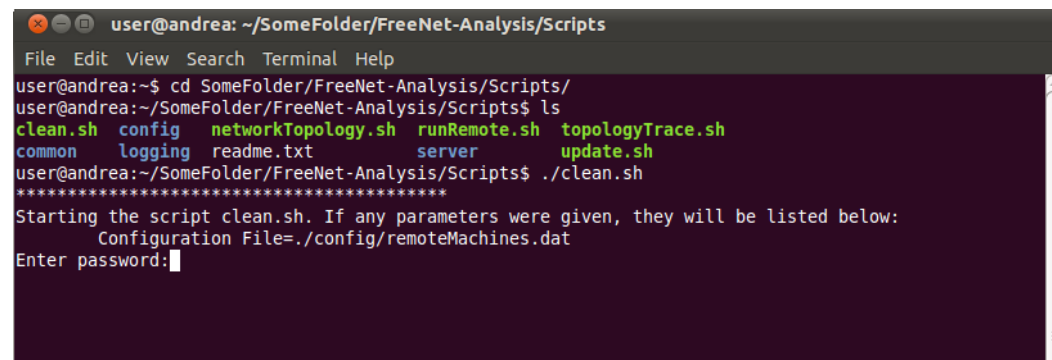
This example experiment assumes that you have already run the first use example experiment in section 7. This example experiment shows you the steps that you will typically be repeating as you are using the testbed to run experiments. This example experiment will only show running the commands using the default parameters. If you want to see an example of running the commands with parameters please refer to section 7. Additional information on all of the scripts run in this section can be found in the *script readme*[1].

The first command that you will typically need to run before starting a new experiment is the update script. This script will reset all of the Freenet nodes in the testbed, and it will also copy any file changes you made in the *master folders* out to all of the testbed machines. If you do not have any files to copy out then you can just run the clean script instead. It is faster to run clean script instead of the update script. Figure 8.1 show how to run the update script, and figure 8.2 shows how to run the clean script.



```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ ls
FreeNet-Analysis
user@andrea:~/SomeFolder$ cd FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse_workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./update.sh
*****
Starting the script update.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

[ Figure 8.1 Update Script ]

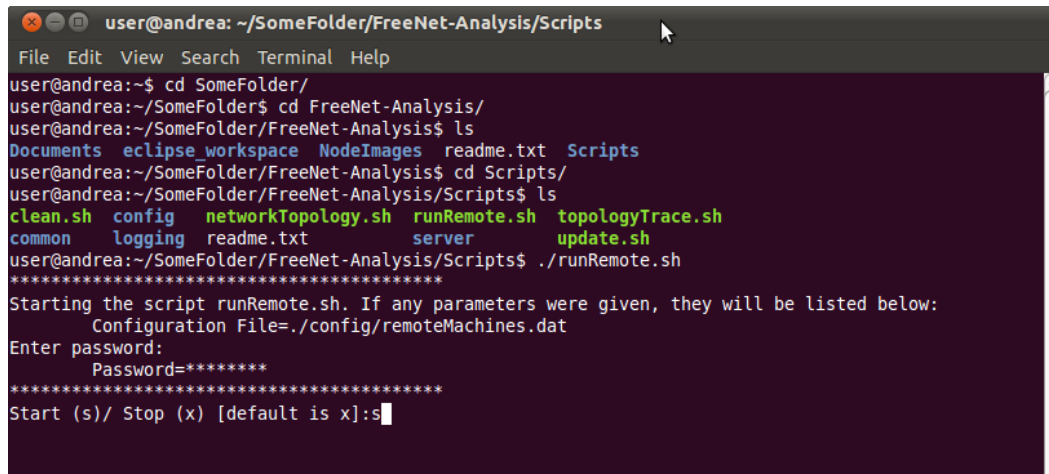


```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./clean.sh
*****
Starting the script clean.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
```

[ Figure 8.2 Clean Script ]

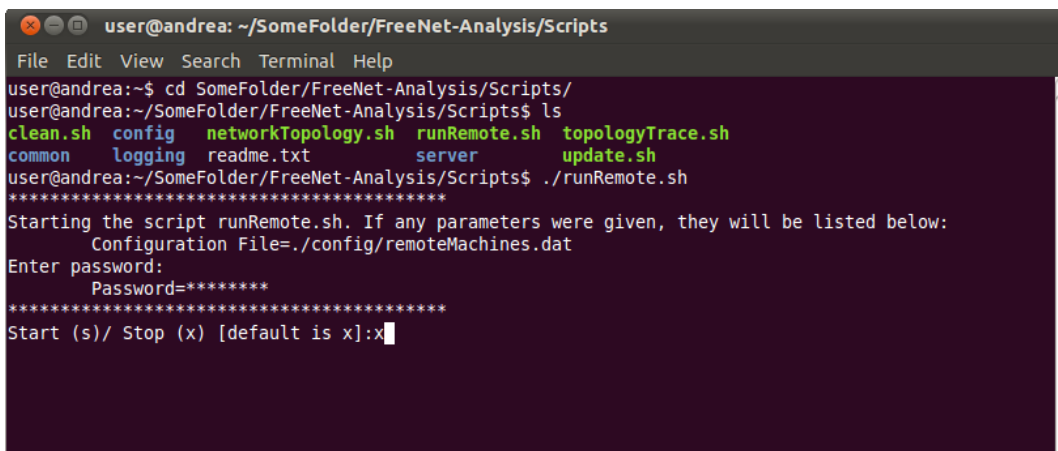
To start the Freenet nodes you can execute the run script in start mode. Once the nodes are started you can perform your experiments. Then to stop the nodes execute the run script in

stop mode. Figure 8.3 show how to start the Freenet nodes, and figure 8.4 show how to stop the Freenet nodes.



```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/
user@andrea:~/SomeFolder$ cd FreeNet-Analysis/
user@andrea:~/SomeFolder/FreeNet-Analysis$ ls
Documents eclipse_workspace NodeImages readme.txt Scripts
user@andrea:~/SomeFolder/FreeNet-Analysis$ cd Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:s
```

[ Figure 8.3 Starting Freenet Nodes Script ]



```
user@andrea: ~/SomeFolder/FreeNet-Analysis/Scripts
File Edit View Search Terminal Help
user@andrea:~$ cd SomeFolder/FreeNet-Analysis/Scripts/
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ls
clean.sh config networkTopology.sh runRemote.sh topologyTrace.sh
common logging readme.txt server update.sh
user@andrea:~/SomeFolder/FreeNet-Analysis/Scripts$ ./runRemote.sh
*****
Starting the script runRemote.sh. If any parameters were given, they will be listed below:
Configuration File=./config/remoteMachines.dat
Enter password:
Password=*****
*****
Start (s)/ Stop (x) [default is x]:x
```

[ Figure 8.4 Stopping Freenet Nodes Script ]

## 9. References

1. Script Readme. /FreeNet-Analysis/Scripts/readme.txt, or <https://github.com/tbaumeist/FreeNet-Analysis/blob/master/Scripts/readme.txt>
2. Freenet: Setup Eclipse and Debug Environment. /FreeNet-Analysis/Documents/FreeNetSetupEclipseAndDebugEnvironment.pdf, or <https://github.com/tbaumeist/FreeNet-Analysis/blob/master/Documents/FreeNetSetupEclipseAndDebugEnvironment.pdf>
3. Freenet Test Bed Design. /FreeNet-Analysis/Documents/FreeNetTestBedDesign.pdf, or <https://github.com/tbaumeist/FreeNet-Analysis/blob/master/Documents/FreeNetTestBedDesign.pdf>
4. Freenet Testbed Setup. /FreeNet-Analysis/Documents/FreenetTestbedSetup.pdf, or <https://github.com/tbaumeist/FreeNet-Analysis/blob/master/Documents/FreenetTestbedSetup.pdf>