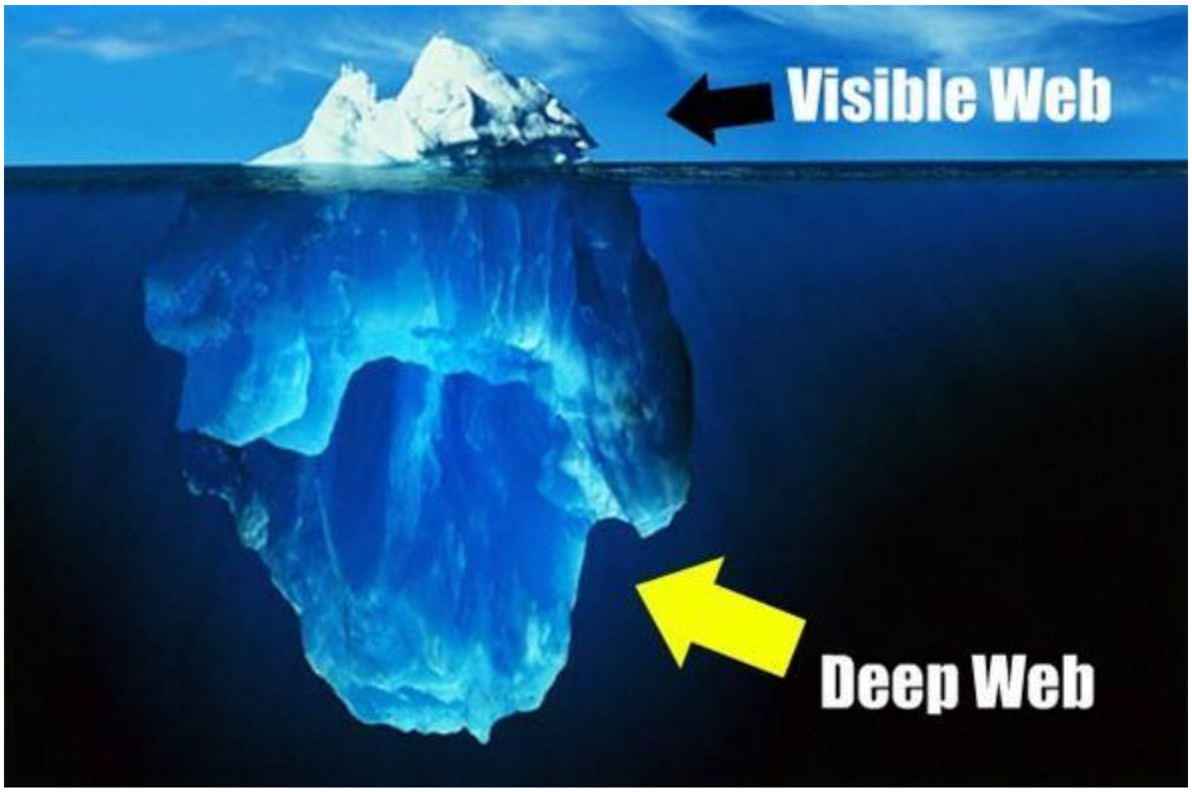
Anonymity, Darknets and Staying Out of Federal Custody



Author & Credits:

Allen Freeman

https://creator.wonderhowto.com/allenfreeman/

Part One: Deep Web

You've probably seen those deep-web images floating around on the Internet. Usually, it goes something like this: There is a towering iceberg and the deeper the underwater portion extends, the more "hidden" and "exotic" the content is described to be. Sometimes these images are accurate to a point, but most are just making things up.

So what exactly is 'deep web' then? Are there really hidden secrets and treasure buried under some cloak and dagger type conspiracy? Well, in short, the answer depends on your idea of treasure and conspiracy.

In this series of articles, I am going to break down the idea of a deep web, what is it, how it got there, and most importantly, how we can use it for our security—maybe even for lulz.

Anonymity and Darknets

As it stands today, practical knowledge of how darknets (non-indexed portions of the Internet) function will allow you to make more informed decisions regarding risks when rooting a box, hiding files, or communicating securely. Now, keep in mind that nothing short of unplugging your computer will make you 100% anonymous. You can have fifty proxies and a handful of VPNs, but never consider yourself to be completely masked. Look at anonymity as a tradeoff between function and speed.

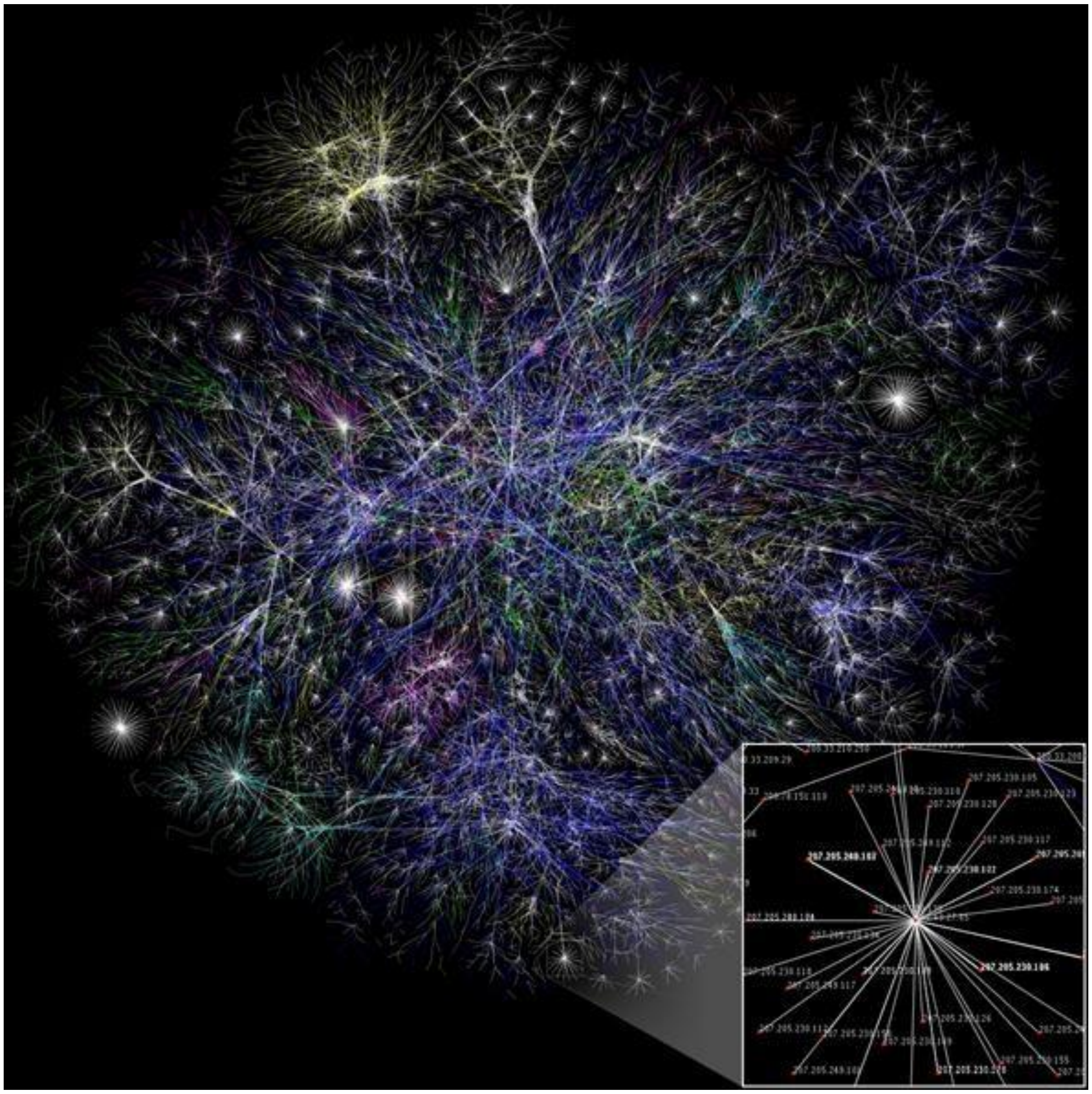
The idea is to have enough masking, while maintaining a level of usefulness. While it is not impossible to track your actions, ideally you want to make it logistically too complex to be attempted in a realistic way.

If you need to escape your school's firewall, a simple HTTP proxy may work for this. If you are rooting an AT&T server, you will want several layers between yourself and the target. Cybercrime laws are changing rapidly around the world, from Cairo to Chicago, and learning how to blend in with the masses is a valuable skill to have. You are harder to track when you are nobody at all.

First though, we need to talk about Google. Yes Google.

Why Google?

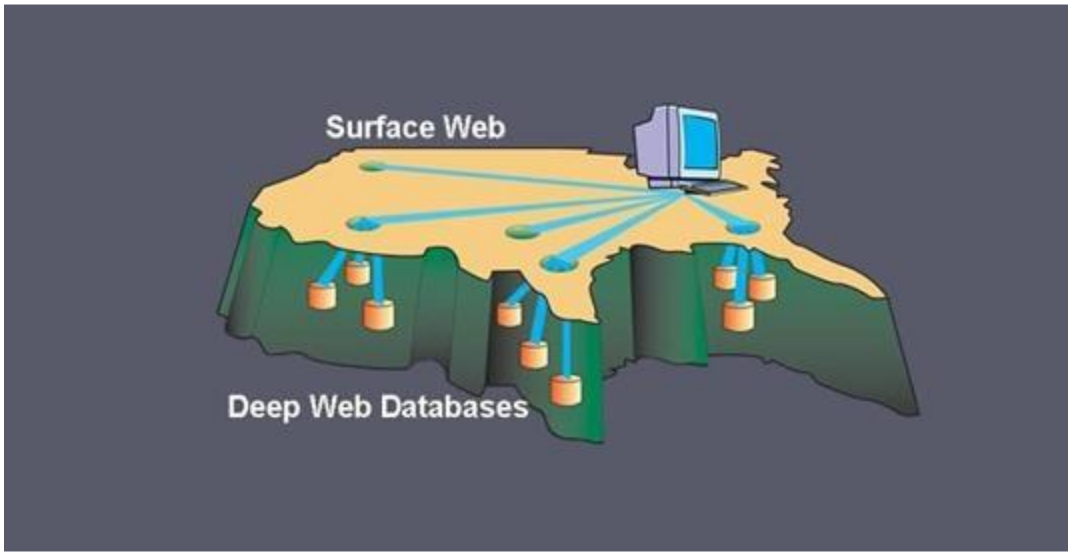
When you use Google to search, it does not take your query and search the entire Internet for results; there is simply too much data for that. It runs your search on databases of sites that have already been located by Google's web crawlers. These crawlers are bots, coded to search for, find, and index content on the web. Primarily, this is achieved by 'seeding' the crawler with a few initial links to start with. It scans for more hyperlinks on those websites, connecting to and repeating this process over and over while creating a 'map' of the results.



It is this 'map' of collected links that your search request is actually looking at. While this is innovative, it contains a few inherent flaws.

The Internet is large. In fact, the Internet is *very* large, and estimates on just how much of it is actually indexed and publicly searchable range from 40 to 70 percent. Problems arise in the fact that most search engines do not crawl through non-HTTP protocols like Gopher or FTP. And if they choose to, developers can take steps to minimize indexing of their sites altogether (controls like the *robots.txt* standard and spider traps are commonly used). It is worth noting that network resources requiring authorization are not crawled, under normal circumstances.

The surface web makes up over 90 percent of what you use and do online. The remaining network services require you to directly connect to them, log in to them, or otherwise know they are there beforehand.



So, what good is all of this for you?

Right now, not as much as you might think. Though the content might not be searched, and is sometimes exciting and risque, you are still held to the basic laws of the Internet, TCP/IP.

Every packet you send that zips back and forth has your IP address inside. It has to have your IP address, or the remote server would not know where to route the requests back to. This means that even if you are snooping around where you shouldn't be, even if it's not on an indexed site, those server logs can still give you up, even when a normal HTTP/SOCKS proxy is used. When your door gets kicked open and the Feds storm your living room, you will have wished you took the time to truly hide yourself.

Picture the Internet like a city, with each building as a resource with an address. Envision the non-indexed parts as alleyways, still connected to the main streets, but lacking public addresses of their own.

Let's take it one step farther... what if these alleyways had gates? What if you could create a path through the city using just the alley and your own private keys? You would be much harder to locate and follow. You could always pop out into the city, go into a house as you needed; people would only see you come and go from the alley, would not know where you started nor where you intend to go.

This is the basic idea behind low-latency anonymous networks like Tor and i2p. We will go over both of these in more detail, including installation and configuration, in the upcoming articles, so stay tuned.

Part Two: Onions and Daggers

In the first part of this series, we learned about darknets, as well as how they came about. But these patches of forgotten Internet are not the oasis of free information you might think. Despite being hidden—or just harder to come across—these networks are no safer than anywhere else on the 'clear' Internet. The nature of networking and routing means your location is always known in server logs. It only takes one phone call to your ISP with your IP address to obtain both your physical address and a search warrant along with it. Therefore, a method must be used to cloak your action from those ever-growing prying eyes.

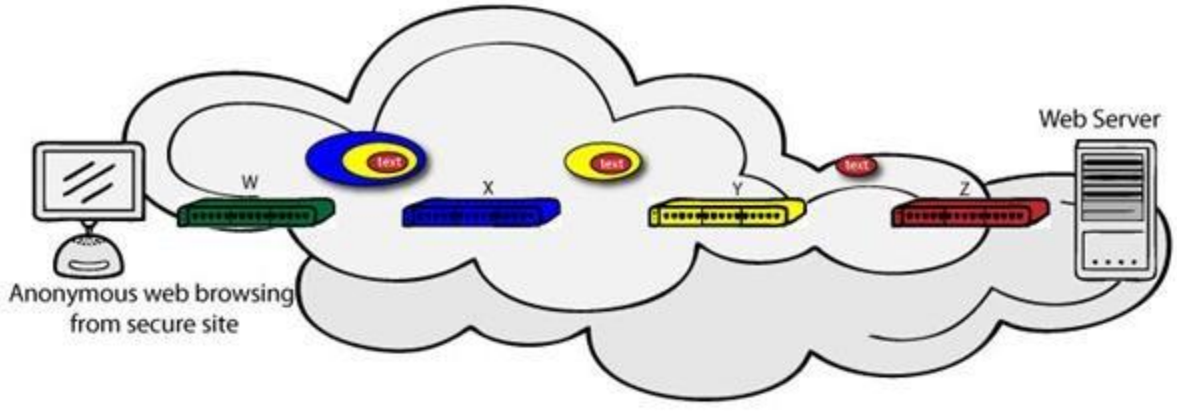
Enter Tor and Onion Routing

Simple traffic analysis and deep packet inspection can give up what you are sending, where you are sending it, where it is coming from and who it is going to. This is performed by your ISP to enforce data limits or anti-spam and piracy measures, and it can be performed by anyone with access to your network. This could be the FBI outside in a parked van or even your neighbor after he cracked your wireless password.

Tor works on the idea of onion routing. Let's take a more in-depth look at the idea behind this, as it's critical to how the Tor (and i2p) network operates. Each machine that's running the Tor service is running a *Tor router* on their network. Each of those routers (which we'll refer to as "nodes" from now on) work by forwarding traffic from other random nodes.

When you wish to send some data on the network, your node calculates a path through all the running nodes, and wraps your data in that many layers of encryption. When each of these random nodes receives your packet, they can only decrypt the layer assigned to them (only they have the key). As such, each node can only see the previous node a packet came from and the next node the packet should be sent to. At no point can the nodes dig deeper into the packet. They can only *peel* away the layer assigned to them. In this way, each layer is peeled away, one at a time, giving us the term *onion routing*.

The figure below illustrates how each layer of the "test" packet is peeled away one at a time on its way to the web server.



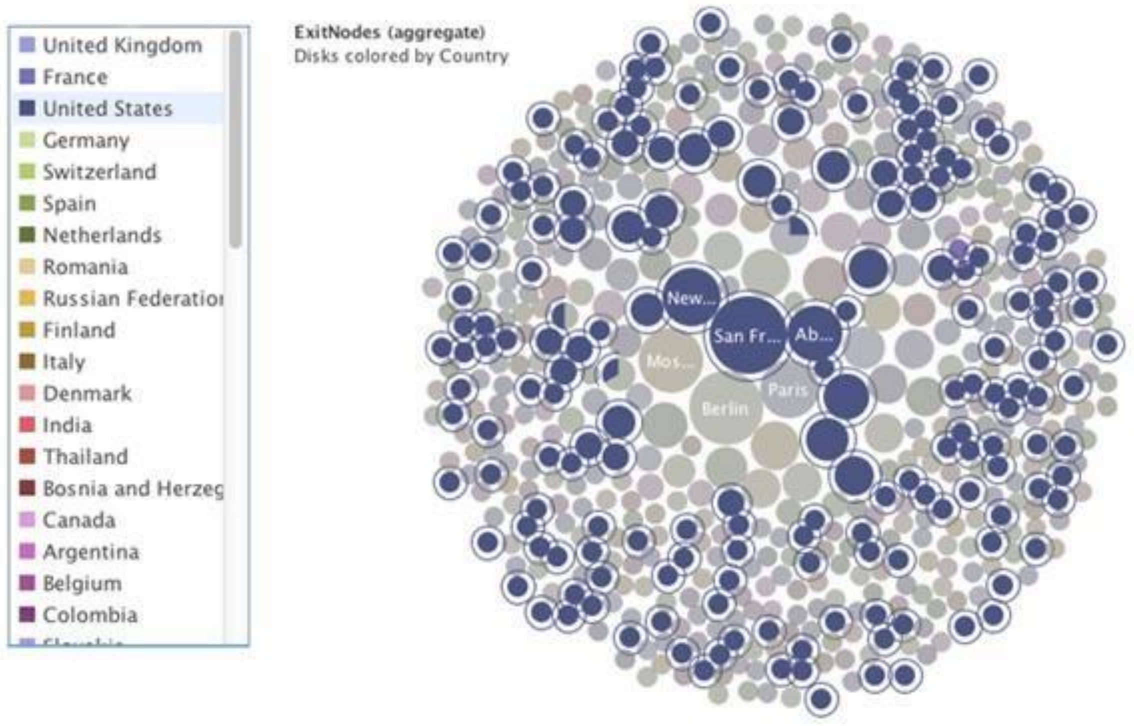
Exit Nodes and Escaping Content Filtering Firewalls

Inside the Tor network, there is a wide range of hidden content, a lot of which is not for the weak of heart. This is because Tor allows you to create your own internal website, known as a Tor Hidden Service. We'll dive deeper into those services in the next part of this series. For now, let's go over how Tor allows you to browse the web anonymously.

Normally you send your request to a web server using the normal HTTP port of 80. The server then reads what you need, and sends it over to you. By the very nature of this system, the server needs to know your router's IP address to communicate back to you, and this is contained in the header of the each packet sent. Essentially, it's connecting what you are doing on the Internet to where you live.

The problem occurs when a company or ISP peeks into those packets and blocks access based on their contents. You have seen examples of this at your high school computer lab and maybe even in your office at work. Websites and content that are not wanted can be filtered out before it even reaches you. Sometimes this is undertaken on a national level, like in China and Iran, by filtering directly at the ISP.

To reach regular websites on the 'clear' Internet, some Tor nodes are configured as *exit nodes*, which route traffic in and out of the Tor network. This hides who you are from the police, the web server, your ISP and anyone else listening in. As far as any of them know, you are the exit node! Also, as Tor uses its own port and encrypts the traffic, your content restrictions can be bypassed with ease.



Even as these exit nodes are often in other countries, there are still methods of tracking you down. None the less, putting a few layers between yourself and the world can never hurt. At the end of this article, I will explain the most efficient way to obtain the software you need and provide you a few links to start your adventure with.

Why Aren't We All Using Tor Then!?

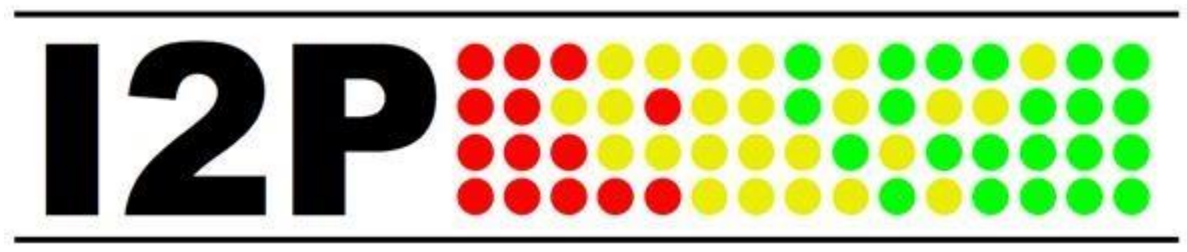
Tor has a few serious issues with it that do need to be addressed. The most critical is a lack of end-to-end encryption. This means anyone who can sniff the traffic from an exit node can see EVERYTHING. They still have to trace packets back and forth to determine a location, but all the encryption is gone. Any personal info you transmitted will be wide open for the stealing. You can negate this by wrapping your data in a presentation-layer protocol like SSH, but that's not always an option.

Tor also does not provide protection against end-to-end timing attacks. If your attacker can watch the traffic coming out of your computer, and also the traffic arriving at your chosen destination, he can use statistical analysis to discover that they are part of the same circuit and locate you. This problem is tied into the one above in a way, as well.

Why Should I Use It?

Tor is still very effective at outsmarting firewalls and port scanning, and though more advanced, tunneling SQL injections. Tor is secure enough to communicate with and even better if you wrap that data in application-layer encryption beforehand. It is not 100 percent, but it's better than just rolling dice and hoping no one is watching.

Over the years, the model has been improved upon and added to. If you are looking for the next step up from Tor, you need to enter the world of i2p. In my next article, I will explain how they're similar and how they differ, sometimes drastically.



tl;dr

Tor is great at browsing the web anonymously and accessing the network's hidden services. The best way is to download the browser bundle directly from the website. There is no installation required, and you can even put it on a USB drive for mobile use as en emergency proxy!

Simply download and extract the archive, navigate to its directory and:

```
$ ./start-tor-browser
```

That's it! The Tor/Vidalia bundle includes the script that starts and connects to the Tor proxy and a modified version of Firefox that is set up to be secure and work with Tor out of the box. This includes disabling Javascript, Flash and any other add-ons that can be tricked into giving up your real IP.

Below is a list of links for a few Tor hidden services to start you off.

- dppmfxaacucguzpc.onion - The TorDIR, a listing of various hidden services, some on and offline.
- kpvz7k12v5agwt35.onion - The Hidden Wiki, a listing of more links and services. Be careful here.
- eqt5g4fuenphqinx.onion - Core.Onion, a great intro for new users.

Part Three: Hidden Services

For a moment, picture a situation where you want to host some files or images, but you do not want it traced back to you. Perhaps you're working on a project with others and need secure data storage. Anonymity is the new shield of the 21st century—and you best protect yourself. As always here at Null Byte, we are trying to make that happen. Before someone can learn how to root a box, they need to learn how *not* to be found. I cannot stress that point enough.

In the last article, we spent some time going over the concepts of onion routing and how it can be useful to you. Before that, we covered the concept of deep web. In this article I will talk about Tor hidden services, what they are, and what they can be used for.

Tor Hidden Services

A Tor hidden service is simply a web server that runs locally and only takes requests from inside the Tor network. It is bound to localhost, so it is not visible or accessible to the outside Internet. Hidden services do not require registration or processing because they are anonymous, so there is no domain name to purchase and use. You may simply set up a web server, and point it to your loopback address. The network will then generate a Tor host name to be located by. Knowing this also explains the sometimes varied uptime of hidden services in general.

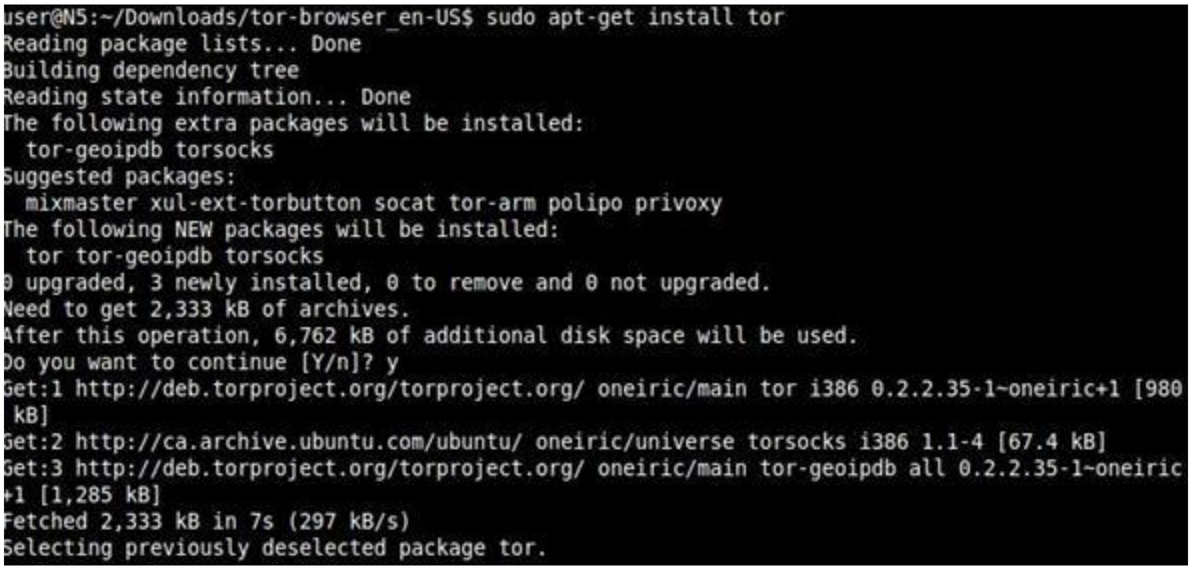
Due to the nature of hidden services being anonymous to a degree, the content can be extreme. Like anything in life, please exercise caution, but enjoy yourself. You will find hackers, crackers, porn and political discontent. You will find places to have your hashes cracked and places that sell drugs. It is the wild west of the modern day Internet. Viewer discretion is advised.

Setting Up Your Own Hidden Service

If you have gotten this far in my series, you are probably interested in setting up your own service. This can be for your friends or even to share things with random people across the world. You could even encrypt files and place them on your hidden service to be picked up by others, adding extra security.

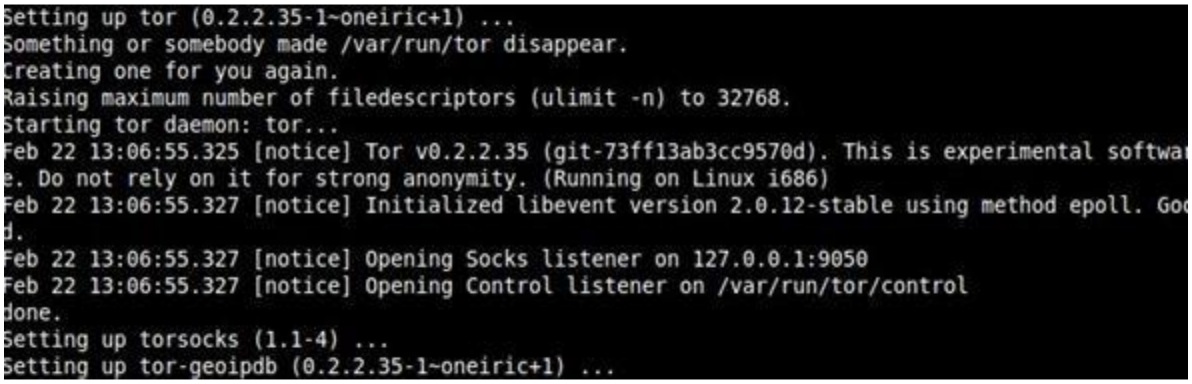
Step 1 Get the Tor Client

In the previous article, we downloaded and used the Tor browser bundle. This works great for simple web browsing, but if we want to operate a hidden service, we need to download and install the actual Tor client. The installation is a simple matter in most cases, but might require a few extra steps for Ubuntu/Debian users.



```
user@N5:~/Downloads/tor-browser_en-US$ sudo apt-get install tor
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  tor-geoipdb torsocks
Suggested packages:
  mixmaster xul-ext-torbutton socat tor-arm polipo privoxy
The following NEW packages will be installed:
  tor tor-geoipdb torsocks
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 2,333 kB of archives.
After this operation, 6,762 kB of additional disk space will be used.
Do you want to continue [Y/n]? y
Get:1 http://deb.torproject.org/torproject.org/ oneiric/main tor i386 0.2.2.35-1~oneiric+1 [980
 kB]
Get:2 http://ca.archive.ubuntu.com/ubuntu/ oneiric/universe torsocks i386 1.1-4 [67.4 kB]
Get:3 http://deb.torproject.org/torproject.org/ oneiric/main tor-geoipdb all 0.2.2.35-1~oneiric
+1 [1,285 kB]
Fetched 2,333 kB in 7s (297 kB/s)
Selecting previously deselected package tor.
```

After we're finished, we have the Tor client running on a default port of 9050 and are ready to go!



```
Setting up tor (0.2.2.35-1~oneiric+1) ...
Something or somebody made /var/run/tor disappear.
Creating one for you again.
Raising maximum number of filedescriptors (ulimit -n) to 32768.
Starting tor daemon: tor...
Feb 22 13:06:55.325 [notice] Tor v0.2.2.35 (git-73ff13ab3cc9570d). This is experimental softwa
e. Do not rely on it for strong anonymity. (Running on Linux i686)
Feb 22 13:06:55.327 [notice] Initialized libevent version 2.0.12-stable using method epoll. Goo
d.
Feb 22 13:06:55.327 [notice] Opening Socks listener on 127.0.0.1:9050
Feb 22 13:06:55.327 [notice] Opening Control listener on /var/run/tor/control
done.
Setting up torsocks (1.1-4) ...
Setting up tor-geoipdb (0.2.2.35-1~oneiric+1) ...
```

Step 2 Set up Your Web Server

If you are using Linux, your absolute best choice for a web server is ThttpD. You may use more mainstream servers like Apache. Windows users will find a good home there, but in this tutorial, we will focus on ThttpD.

```
$ wget http://www.acme.com/software/thttpd/thttpd-2.25b.tar.gz
$ tar zxvf thttpd-2.25b.tar.gz
```

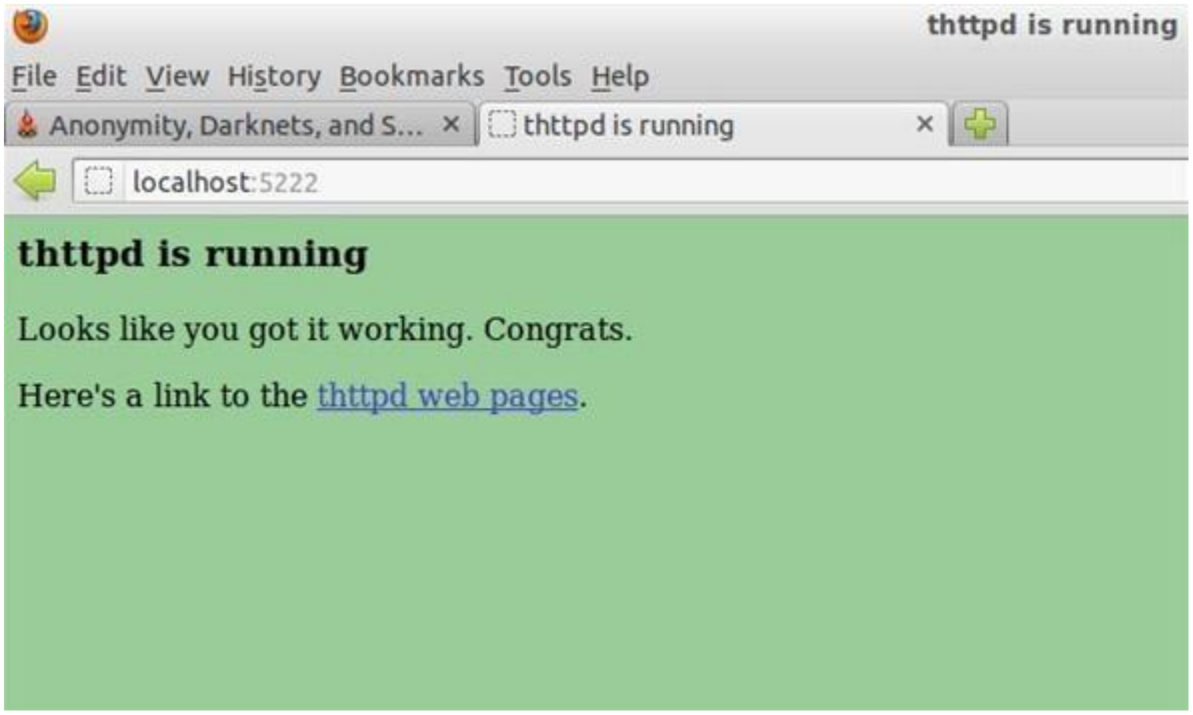
Move into the new directory to configure and compile from source:

```
$ sudo ./configure && make
$ mkdir hidserv; cd hidserv
$ cd ..
$ ./thttpd -p 5222 -h localhost
```

Tip

- Adding a semi colon to a command allows us to follow it up with another command!

The last command started the web server and told it to use port 5222 and localhost as its host. You can click this link to see if it worked. If you see the page below, you are on the right track.


thttpd is running
File Edit View History Bookmarks Tools Help
Anonymity, Darknets, and S...
thttpd is running
localhost:5222

thttpd is running

Looks like you got it working. Congrats.

Here's a link to the thttpd web pages.

That's it! We are now ready to finish up and connect with Tor!

Step 3 Turning Your Web Server into a Hidden Service

Navigate to your /etc/tor directory and open the file named *torrc* with your favorite editor (I use vim). This is a configuration file for letting Tor know you are running a hidden service. Scroll down about halfway until you see the title:

### This section is just for location-hidden services ###

And add the following lines under that:

```
HiddenServiceDir /home/username/hidden_service/
HiddenServicePort 80 127.0.0.1:5222
```

Username is the name you log into on your Linux box with and *hidden_service* is whatever name you wish to give it.

Now restart Tor. When it starts back up, it reads the file *torrc*, follows the instructions, and bootstraps your service into the Tor network. It will then automatically create the 'hidden_service' directory that you specified (if necessary), and it will create two files there, *private_key* and *hostname*.

The private key is identical to the idea we discussed in our GPG article. Keep this key secret! Anyone who obtains it can impersonate your hidden service!

The hostname is a URL created by the network used to identify you from other services, it should look something like *uy4htf7ssvv3gfh8g.onion*.

Conclusion

This was just a taste of what's out there. From here you can branch out to configure your service however you need it. I implore you to spend some time on the Tor network and see what is all there. However, for the more savvy and security minded, Tor is not the best choice. In my next and final article in this series, I will go over the Invisible Internet Project known as i2p. The development of i2p started where Tor left off, improving upon its base and adding more features for the security minded.

Part Four: The Invisible Internet

In the last article, we left off with the Tor network and its hidden services. As I mentioned, Tor is not the only option in the game, and I want to offer a general introduction to I2P.

For a web browsing proxy and hidden services host, Tor gets the job done well. But if you are looking to expand your options for secure communication, then enter I2P! This article will show you how to obtain and set up I2P, and in the process, we will also configure an IRC client to access the IRC2P network and get you up and running.

Invisible Internet and Onion Routing

I2P initially began in 2003 as a proposed modification to Freenet. To deal with a wide range of attacks, I2P is fully distributed with no centralized resources— hence there are no directory servers keeping statistics regarding the performance and reliability of routers within the network. I2P is not 100% secure, as nothing is 100% secure, but using it will provide you with meaningful security nonetheless.

Content sent over I2P is encrypted through three-layer garlic encryption, used to verify the delivery of the message to the recipient. All messages passing through a tunnel are encrypted by the tunnel gateway to the tunnel endpoint, and undergo inter-router transport layer encryption along the way. You also have the ability to tunnel TCP/IP based applications (IRC, Jabber, steaming music, etc.) through the network. In fact, you can even tunnel your torrent downloads!

Step 1 Download I2P

Let's start with a visit to the download page to grab the Windows binary or source code, if needed. If you are using Debian/Ubuntu or Mint, we can add the repositories by simply typing:

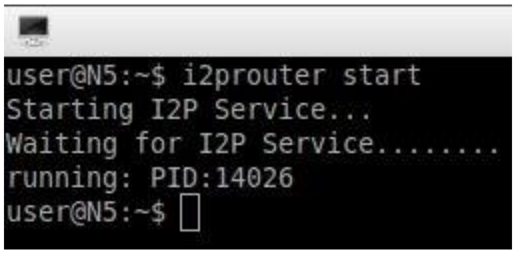
```
$sudo apt-add-repository ppa:i2p-maintainers/i2p
$ sudo apt-get update
$ sudo apt-get install i2p
```

This downloads and installs the I2P router on your computer.

Step 2 Start the I2P Router

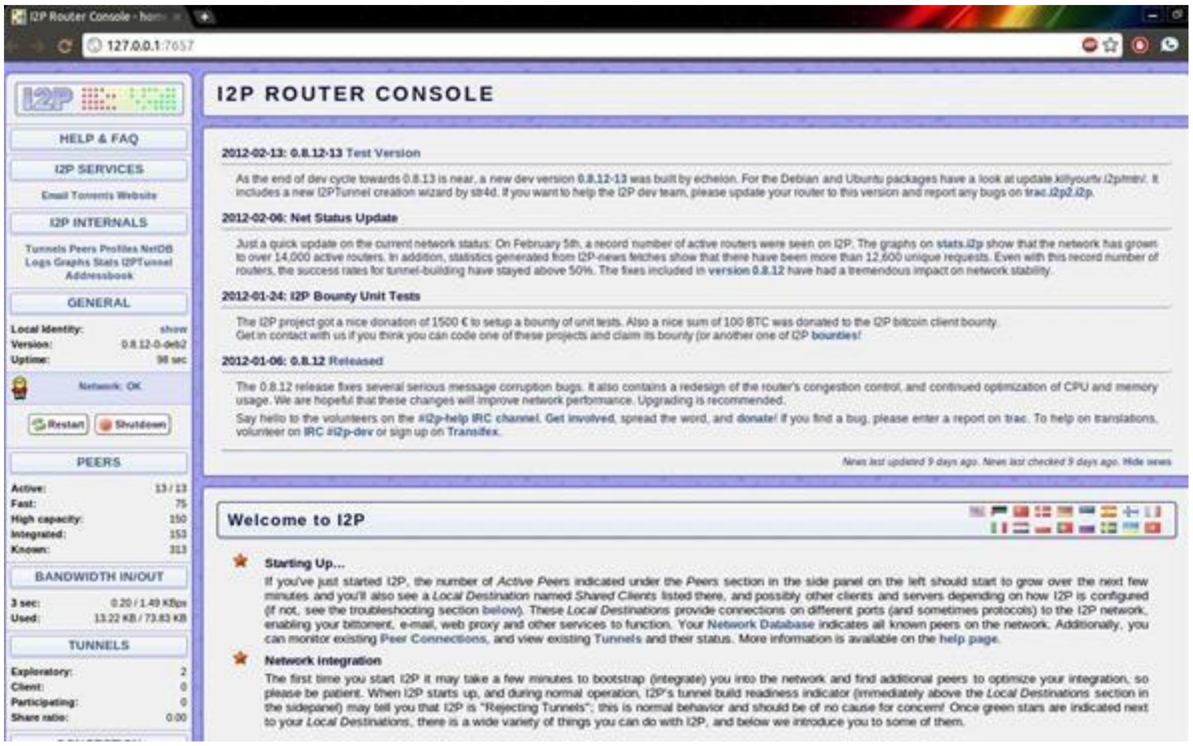
Now that we have the packages downloaded and installed, we must start the service. We do this by simply typing:

$ i2prouter start



```
user@N5:~$ i2prouter start
Starting I2P Service...
Waiting for I2P Service........
running: PID:14026
user@N5:~$ 
```

After we type that command, the I2P router will begin to bootstrap itself into the network. It is recommended to give this process some time as it seeks out and adds other nodes for you to route traffic through. After a few seconds, I2P will open a browser with the router console as seen below. You may close this.



I2P Router Console - home

127.0.0.1:7657

HELP & FAQ

I2P SERVICES

Email Torrents Website

I2P INTERNALS

Tunnels Peers Profiles NetDB Logs Graphs Stats I2PTunnel Addressbook

GENERAL

```
Local Identity:        show
Version:       0.8.12-0-deb2
Uptime:               98 sec
```

Network: OK

Restart Shutdown

PEERS

```
Active:              13 / 13
Fast:                     75
High capacity:           150
Integrated:              153
Known:                   313
```

BANDWIDTH IN/OUT

```
3 sec:       0.20 / 1.49 KBps
Used:     13.22 KB / 73.83 KB
```

TUNNELS

```
Exploratory:               2
Client:                    0
Participating:             0
Share ratio:            0.00
```

I2P ROUTER CONSOLE

2012-02-13: 0.8.12-13 Test Version

As the end of dev cycle towards 0.8.13 is near, a new dev version 0.8.12-13 was built by echelon. For the Debian and Ubuntu packages have a look at update.killyourtv.i2p/mtn/. It includes a new I2PTunnel creation wizard by str4d. If you want to help the I2P dev team, please update your router to this version and report any bugs on trac.i2p2.i2p.

2012-02-06: Net Status Update

Just a quick update on the current network status: On February 5th, a record number of active routers were seen on I2P. The graphs on stats.i2p show that the network has grown to over 14,000 active routers. In addition, statistics generated from I2P-news fetches show that there have been more than 12,600 unique requests. Even with this record number of routers, the success rates for tunnel-building have stayed above 50%. The fixes included in version 0.8.12 have had a tremendous impact on network stability.

2012-01-24: I2P Bounty Unit Tests

The I2P project got a nice donation of 1500 € to setup a bounty of unit tests. Also a nice sum of 100 BTC was donated to the I2P bitcoin client bounty.
Get in contact with us if you think you can code one of these projects and claim its bounty (or another one of I2P bounties!

2012-01-06: 0.8.12 Released

The 0.8.12 release fixes several serious message corruption bugs. It also contains a redesign of the router's congestion control, and continued optimization of CPU and memory usage. We are hopeful that these changes will improve network performance. Upgrading is recommended.

Say hello to the volunteers on the #i2p-help IRC channel. Get involved, spread the word, and donate! If you find a bug, please enter a report on trac. To help on translations, volunteer on IRC #i2p-dev or sign up on Transifex.

News last updated 9 days ago. News last checked 9 days ago. Hide news

Welcome to I2P

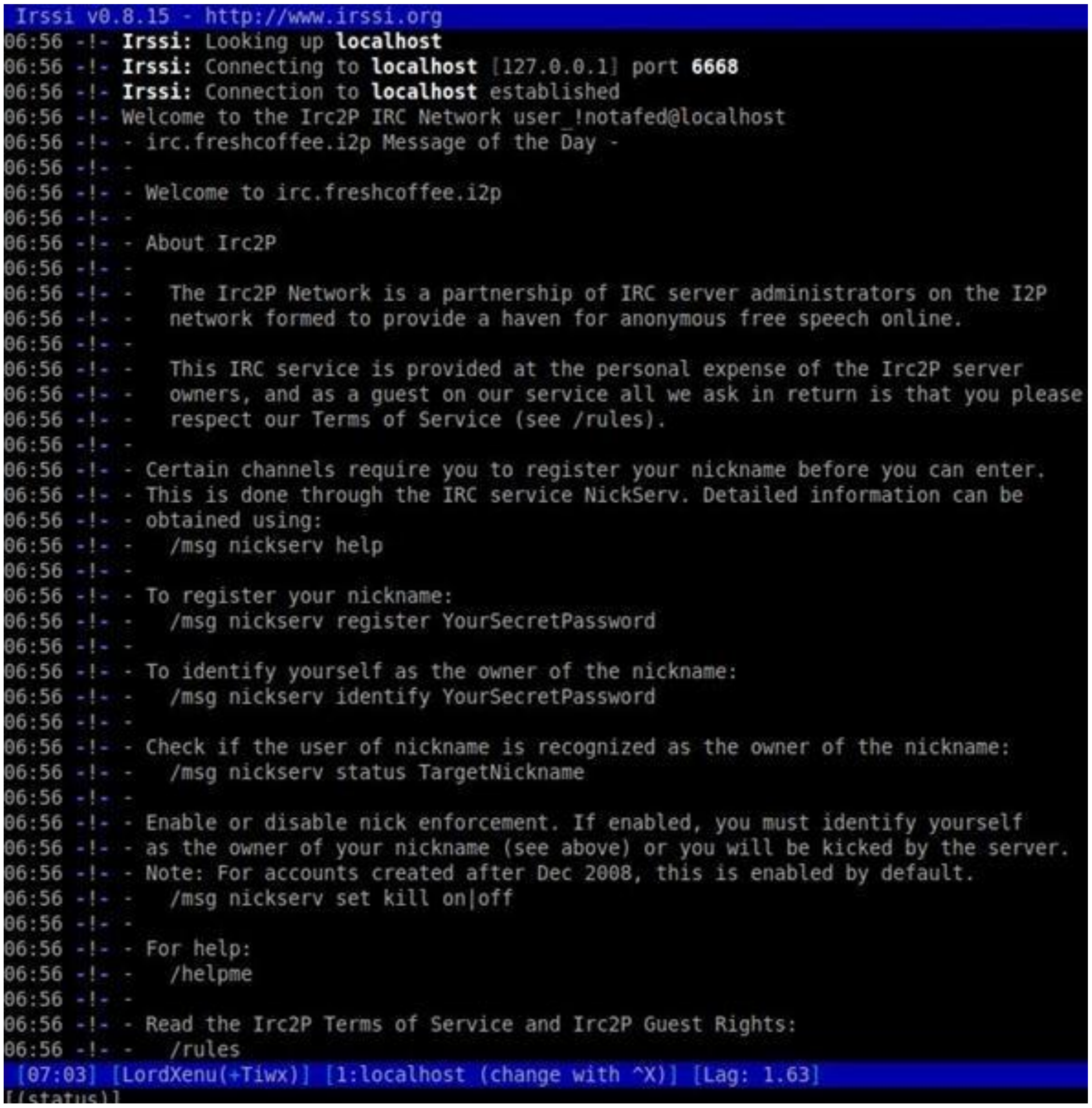
Starting Up...
If you've just started I2P, the number of *Active Peers* indicated under the *Peers* section in the side panel on the left should start to grow over the next few minutes and you'll also see a *Local Destination* named *Shared Clients* listed there, and possibly other clients and servers depending on how I2P is configured (if not, see the troubleshooting section below). These *Local Destinations* provide connections on different ports (and sometimes protocols) to the I2P network, enabling your bittorrent, e-mail, web proxy and other services to function. Your Network Database indicates all known peers on the network. Additionally, you can monitor existing Peer Connections, and view existing Tunnels and their status. More information is available on the help page.

Network integration
The first time you start I2P it may take a few minutes to bootstrap (integrate) you into the network and find additional peers to optimize your integration, so please be patient. When I2P starts up, and during normal operation, I2P's tunnel build readiness indicator (immediately above the *Local Destinations* section in the sidepanel) may tell you that I2P is "Rejecting Tunnels"; this is normal behavior and should be of no cause for concern! Once green stars are indicated next to your *Local Destinations*, there is a wide variety of things you can do with I2P, and below we introduce you to some of them.

Step 3 Invisible Internet Relay Chat

Some of you might have a preferred IRC client already. I use Irssi myself, but Xchat will work for most people. Normally, when you open the client you have the chance to connect to a server and port number. Because I2P runs as a service connected to your loopback address, we need to connect to it to access the IRC2P network by typing:

/server localhost 6668



```
 Irssi v0.8.15 - http://www.irssi.org
06:56 -!- Irssi: Looking up localhost
06:56 -!- Irssi: Connecting to localhost [127.0.0.1] port 6668
06:56 -!- Irssi: Connection to localhost established
06:56 -!- Welcome to the Irc2P IRC Network user_!notafed@localhost
06:56 -!- - irc.freshcoffee.i2p Message of the Day -
06:56 -!- -
06:56 -!- - Welcome to irc.freshcoffee.i2p
06:56 -!- -
06:56 -!- - About Irc2P
06:56 -!- -
06:56 -!- -   The Irc2P Network is a partnership of IRC server administrators on the I2P
06:56 -!- -   network formed to provide a haven for anonymous free speech online.
06:56 -!- -
06:56 -!- -   This IRC service is provided at the personal expense of the Irc2P server
06:56 -!- -   owners, and as a guest on our service all we ask in return is that you please
06:56 -!- -   respect our Terms of Service (see /rules).
06:56 -!- -
06:56 -!- - Certain channels require you to register your nickname before you can enter.
06:56 -!- - This is done through the IRC service NickServ. Detailed information can be
06:56 -!- - obtained using:
06:56 -!- -   /msg nickserv help
06:56 -!- -
06:56 -!- - To register your nickname:
06:56 -!- -   /msg nickserv register YourSecretPassword
06:56 -!- -
06:56 -!- - To identify yourself as the owner of the nickname:
06:56 -!- -   /msg nickserv identify YourSecretPassword
06:56 -!- -
06:56 -!- - Check if the user of nickname is recognized as the owner of the nickname:
06:56 -!- -   /msg nickserv status TargetNickname
06:56 -!- -
06:56 -!- - Enable or disable nick enforcement. If enabled, you must identify yourself
06:56 -!- - as the owner of your nickname (see above) or you will be kicked by the server.
06:56 -!- - Note: For accounts created after Dec 2008, this is enabled by default.
06:56 -!- -   /msg nickserv set kill on|off
06:56 -!- -
06:56 -!- - For help:
06:56 -!- -   /helpme
06:56 -!- -
06:56 -!- - Read the Irc2P Terms of Service and Irc2P Guest Rights:
06:56 -!- -   /rules
 [07:03] [LordXenu(+Tiwx)] [1:localhost (change with ^X)] [Lag: 1.63]
[(status)]
```

That's it! You are now connected to the IRC2P network. You can use normal IRC commands such as */list* to locate channels. Also please feel free to */join* #nullbyte and #i2p and say hello!

tl;dr

Just want the commands to get up and running? Look no further.

```
$ sudo apt-add-repository ppa:i2p-maintainers/i2p
$ sudo apt-get update
$ sudo apt-get install i2p
$ i2prouter start
```

Then start your IRC client and type:

```
/server localhost 6668
/join #nullbyte
```

See you in the Deep Web!