

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/338825326>

osint + python: extracting information from tor network and darkweb

Presentation · August 2019

DOI: 10.13140/RG.2.2.31123.43045

CITATIONS

0

READS

1,000

1 author:



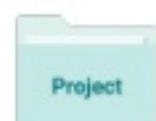
José Manuel Ortega

University of Alicante

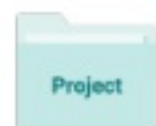
30 PUBLICATIONS 0 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Desarrollo seguro [View project](#)

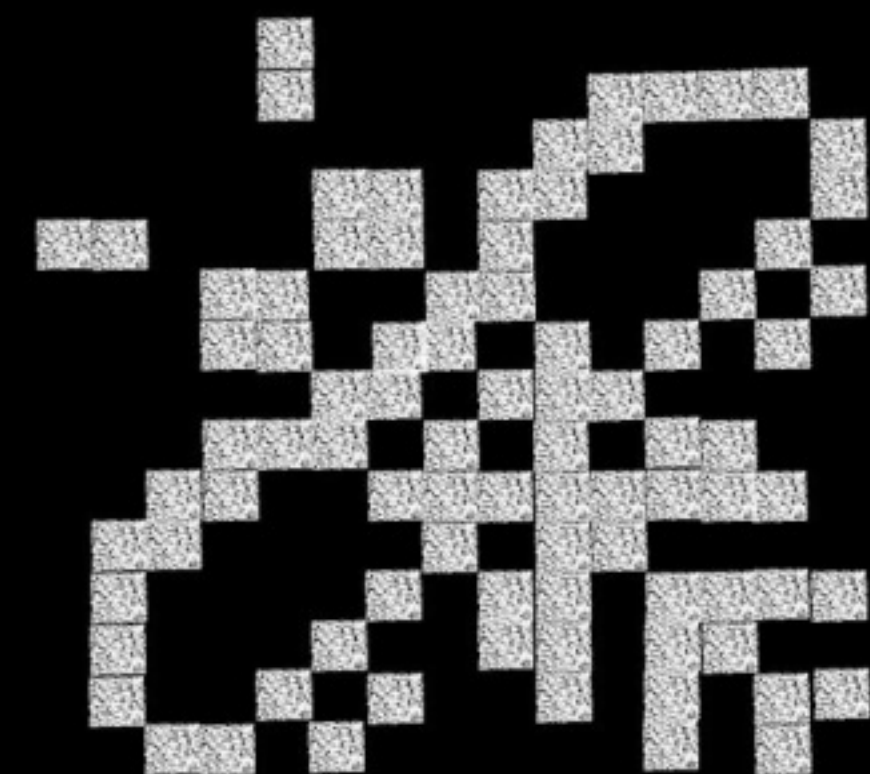


Machine Learning para proyectos de seguridad [View project](#)

BSIDES

29:08:2019

2019



MANCH35T3R

OSINT + PYTHON: Extracting information from TOR network and Darkweb

@jmortegac

BSIDES MANCHESTER, 2019

About me



<http://jmortega.github.io/>



PYTHON & JAVA & DOCKER SECURITY CONFERENCES

MUSIC BOX

PREVIOUS SITE

TALKS

CONFERENCES



Testing python security

[YouTube]

[PYCONIE]



Common Vulnerabilities & Exposures (CVE)
In Docker Containers

[YouTube]
[Sonatype]

[ALLDAYDEVOPS]

Python & OSINT para
proyectos de seguridad

Jose Manuel Ortega @jmortegac

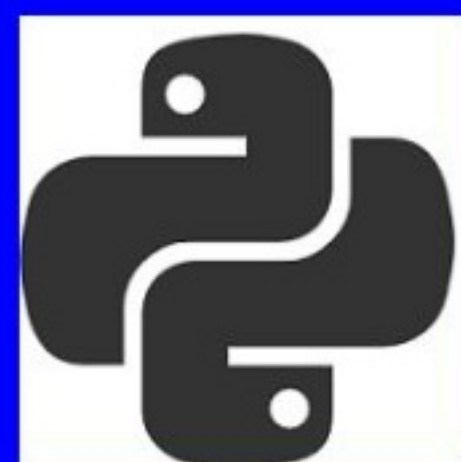
Python & OSINT para proyectos de
seguridad

[YouTube]

[BITUP]



TESTING DOCKER IMAGES
SECURITY



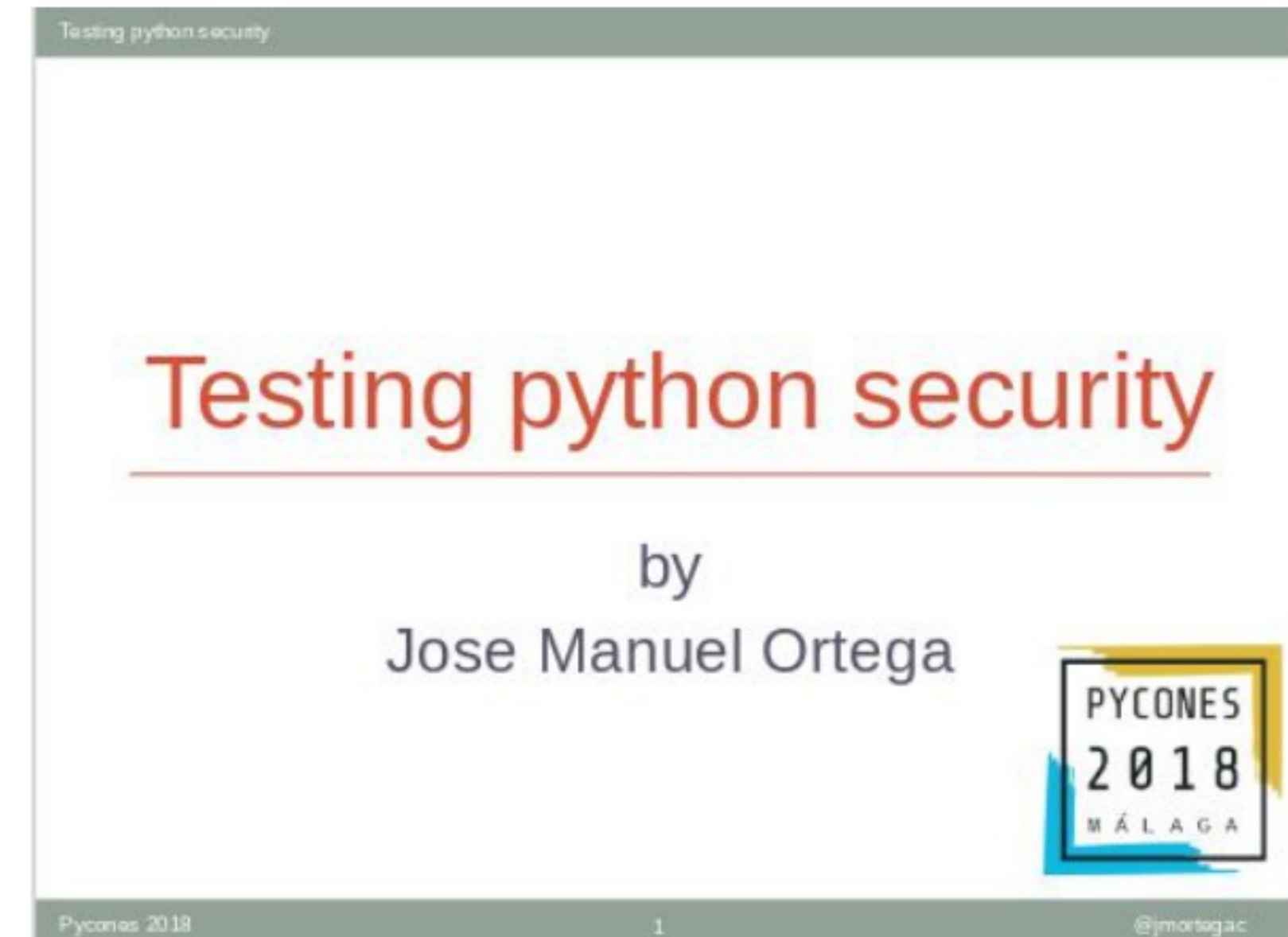
DISCOVERING PYTHON SEARCH
ENGINE



TESTING DOCKER IMAGES
SECURITY



OSINT TOOLS FOR SECURITY
AUDITING



About me



autentia

Testing python security [Agenda](#)

1. Secure coding
2. Dangerous functions
3. Common attack vectors
4. Static analysis tools
5. Other security issues



Jose Manuel Ortega
@jmortegac

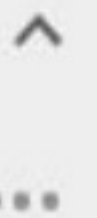


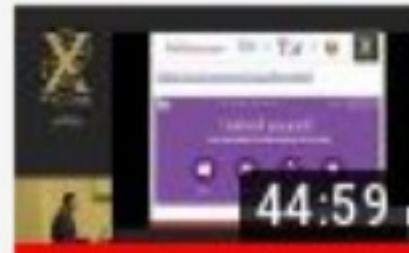
Pycones 2018 2 @mortegac

0:00 / 25:31 Testing python security

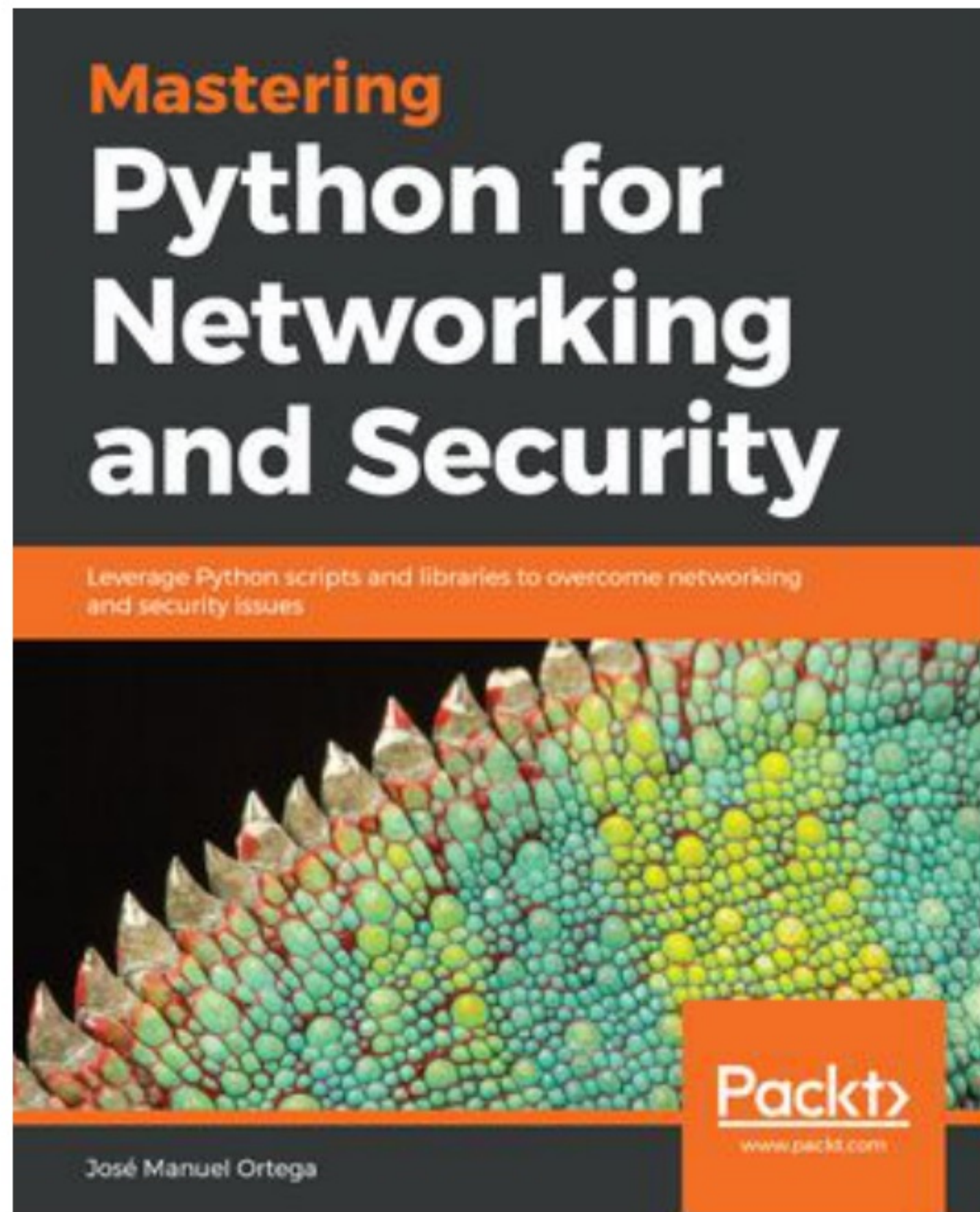
Python and docker security conferences

J.M. Ortega - 7/59



- 1  44:59 Jose Manuel Ortega - Darkweb + Python: discover, analyze and...
PyCon Italia
- 2  38:29 Python & OSINT para proyectos de seguridad
Palabra de hacker
- 3  24:45 Hacking NodeJS applications for fun and profit
J.M. Ortega
- 4  47:04 Jose Manuel Ortega | Testing NodeJS Security | Codemotion...
Codemotion
- 5  TESTING PYTHON SECURITY
Jose Manuel Ortega PYCONES 2018

About me



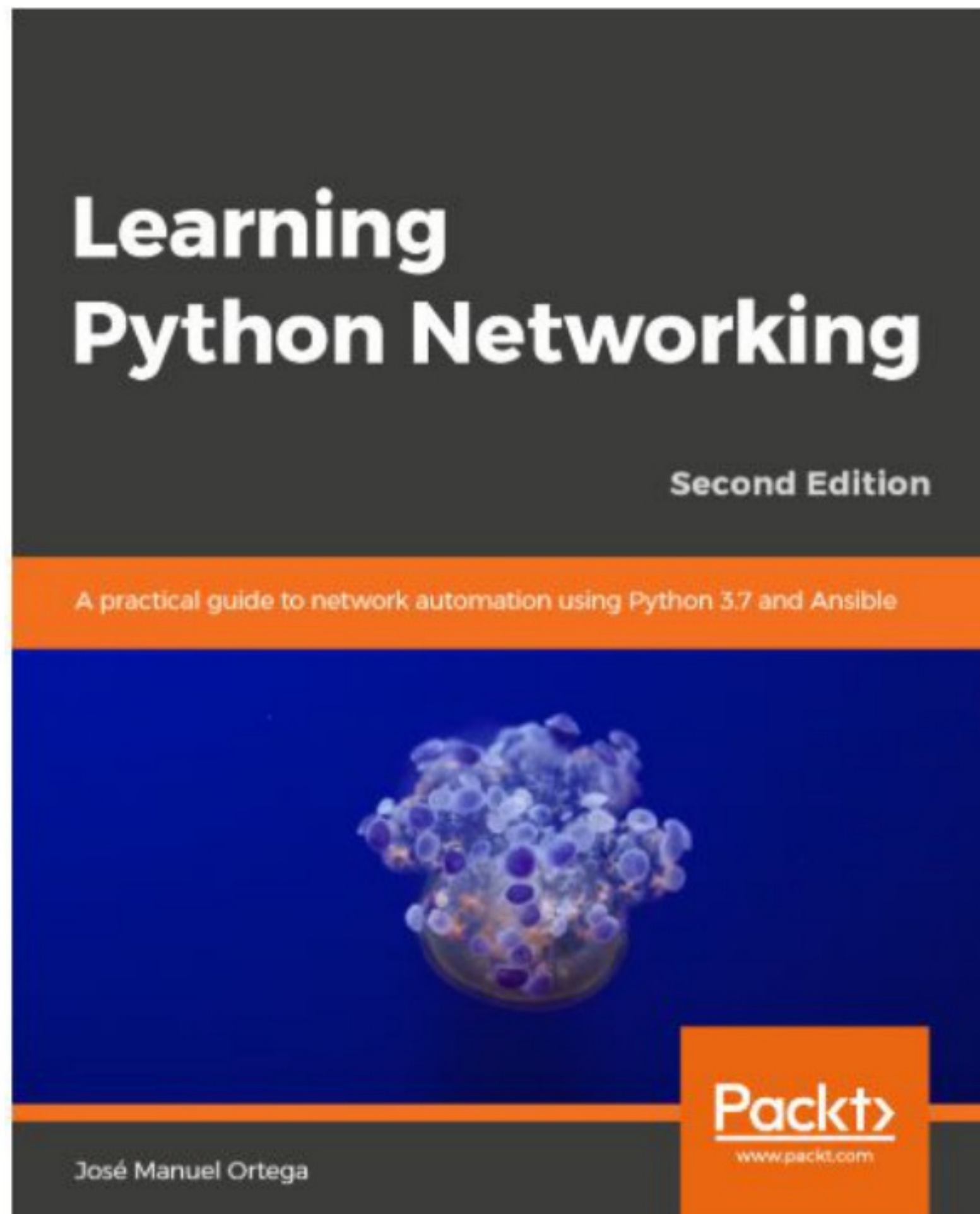
SHODAN



FTP



About me



django



Scan me

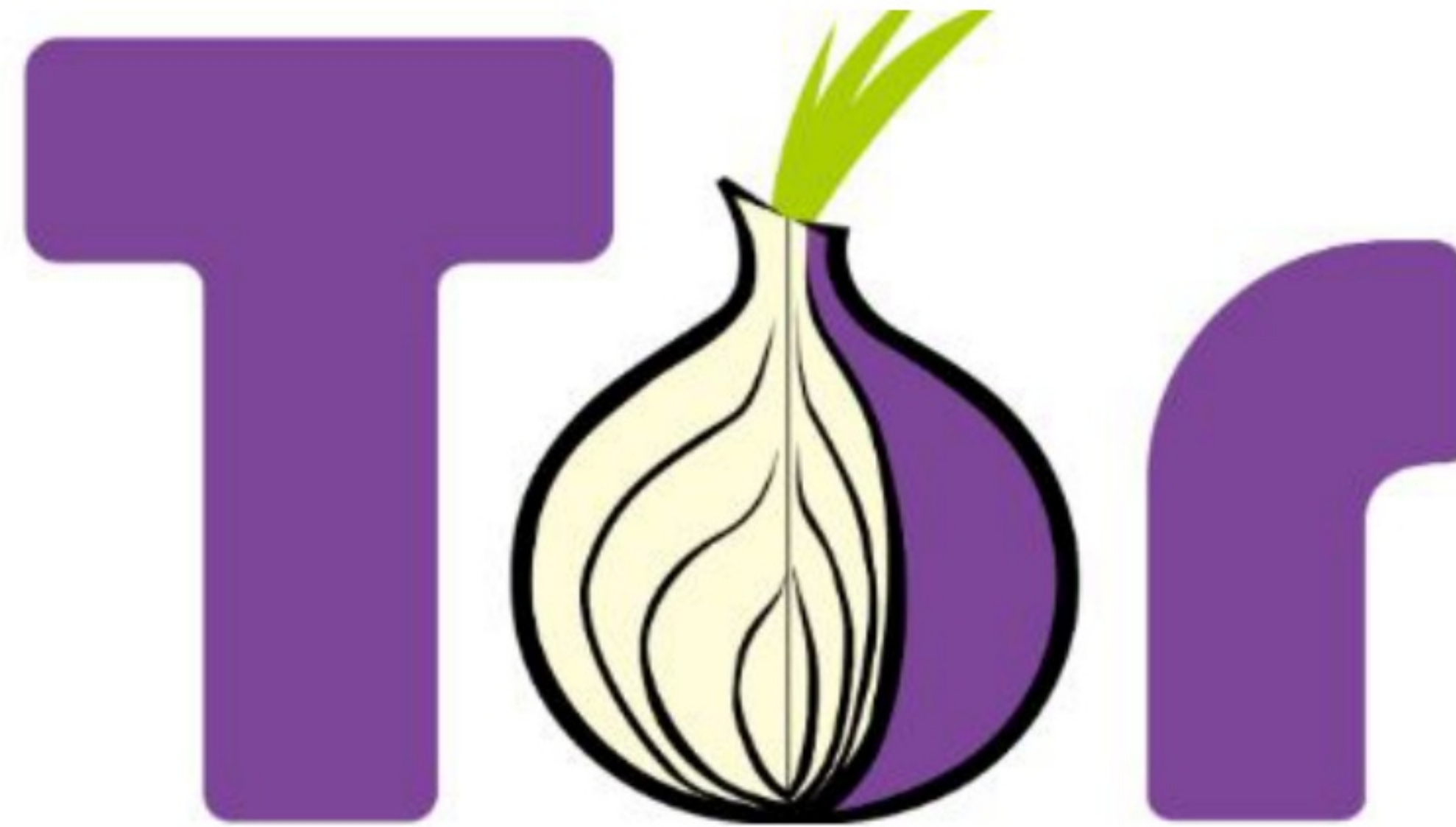
- **Introduction to Tor project and discover hidden services**
- **Modules and packages we can use in python for connecting with Tor network**
- **Tools that allow search hidden services and automate the crawling process in Tor network**
- **OSINT TOOLS for discovering hidden services**

Surface vs Deep vs Dark Web



What is Tor?

- Tor is a free tool that allows people to use the internet anonymously.
- Tor anonymizes the origin of your traffic



The Onion Router

What is Tor?



WikiLeaks

Leaks News About Partners

Search  Shop [Donate](#) [Submit](#)

WikiLeaks

- Main Page
- Country index
- Follow us

WikiLeaks:Tor

The following method requires some technical ability. If you are used to installing new software and configuring proxy servers you should have the required skills, otherwise you may wish to use one of our [other submission methods](#). Don't let the technology defeat you!

Tor, or The Onion Router, is a cryptographic technique first implemented by US Navy research to permit intelligence agents to use the internet without being traced, by encrypting and routing communications through many different internet servers. Subsequently, Tor has been developed by the US university [MIT](#) and by the California internet rights watchdog the [Electronic Frontier Foundation](#) and subsequently incorporated into [Wikileaks](#).

Using our anonymous access package ([see below](#)) you can prevent internet spies knowing that your computer has connected to [Wikileaks](#).

Most Wikileakers do not need this extra security, and there are simpler and possibly safer alternatives for once-off high-risk leaks (see [Submissions](#)). But for those who are at risk and want to access Wikileaks from the comfort of their homes or offices or need to bypass Internet censorship, Tor ([Onion Routing](#)) is an excellent solution.

When you have installed our Tor access package (see below), you may then connect to [Wikileaks](#) via our anonymous address (the ".onion" is short for "Onion Routing", but you do not need to be concerned with this detail). *NB: the original .onion link for browsing [Wikileaks](#) is currently unavailable; however, if you have installed Tor and are redirecting all of your browsing through the Tor network, you can still browse the normal WikiLeaks site with a high degree of anonymity (but not end-to-end encryption). The secure .onion address provided here and on the [submissions page](#) should still work, in any case.*

To upload a document anonymously using Tor:

<http://suw74isz7wqzpmgu.onion/>
(this link will only work once you have installed and configured Tor.)

What is Tor?

The Advantages of TOR:

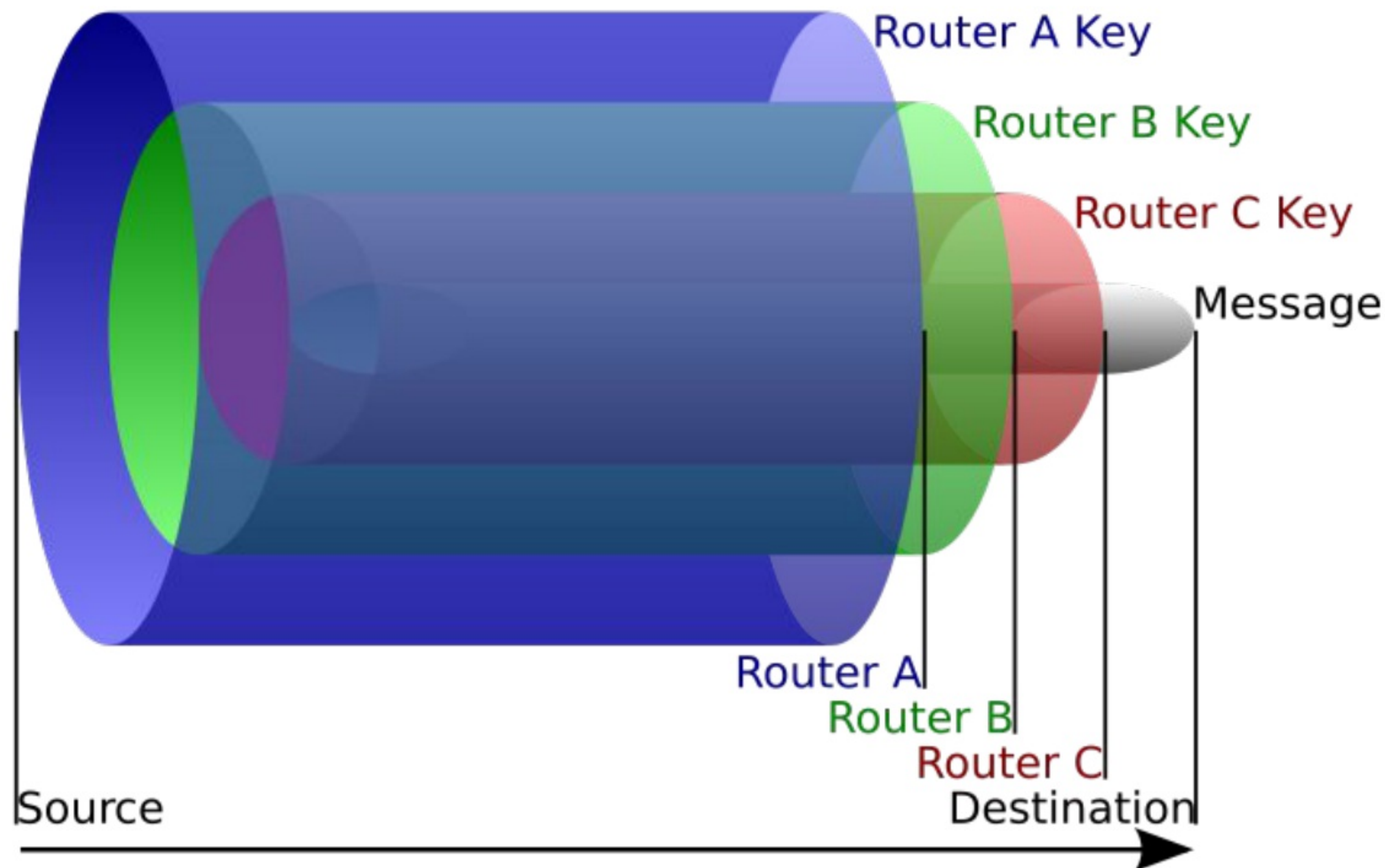
1. Tor is considered a strong security tool against surveillance
2. It's the low latency anonymity network
3. Unlike Mix-Networks, the message doesn't have to wait for more messages

The Disadvantages of TOR:

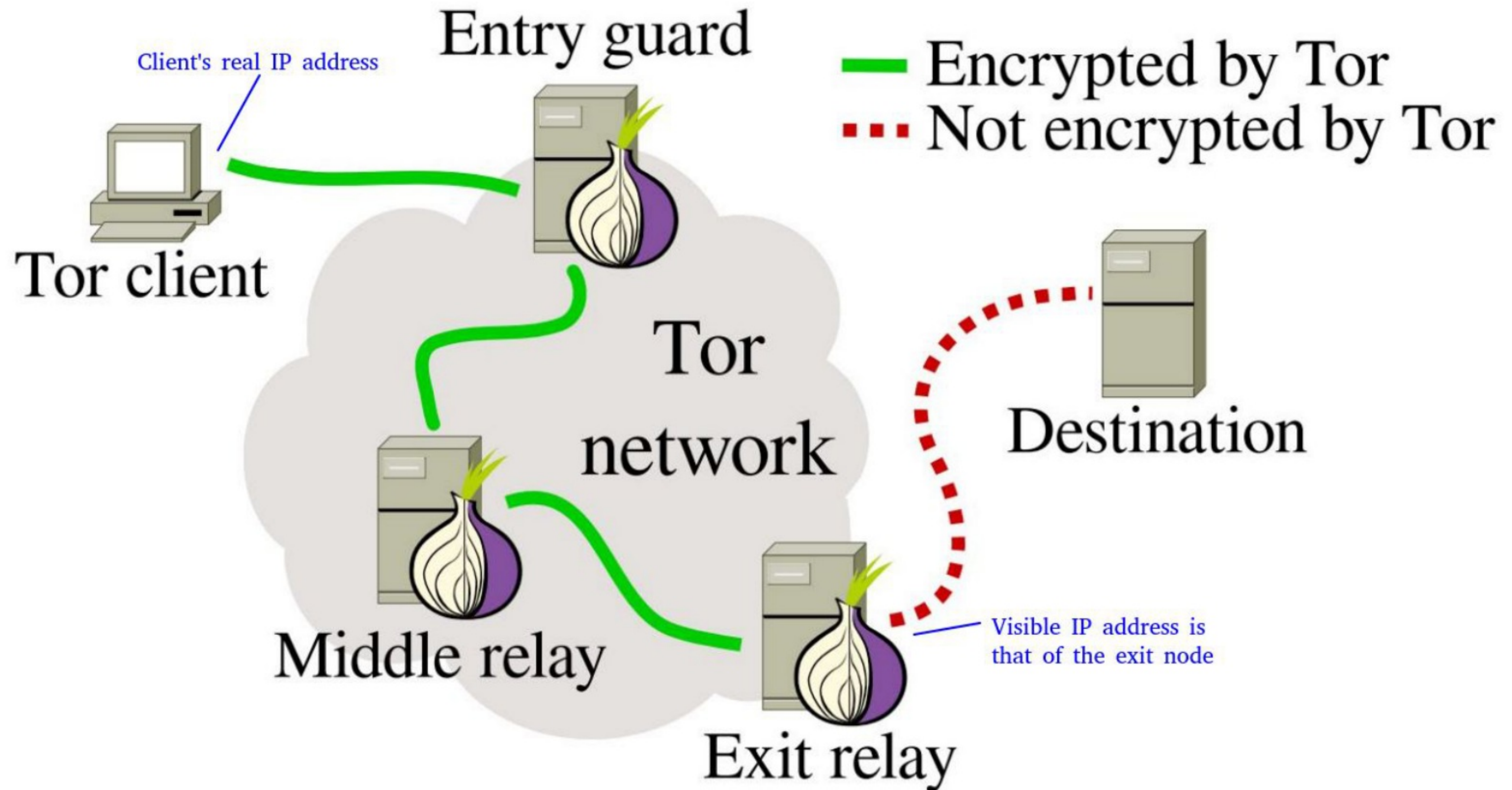
1. It isn't safe against end-to-end correlation (attacks at the boundaries of Tor network) and manual traffic analysis.
2. It's slow as the message is routed many times
3. Some ISP tries to search and block Tor relays

Onion Routing

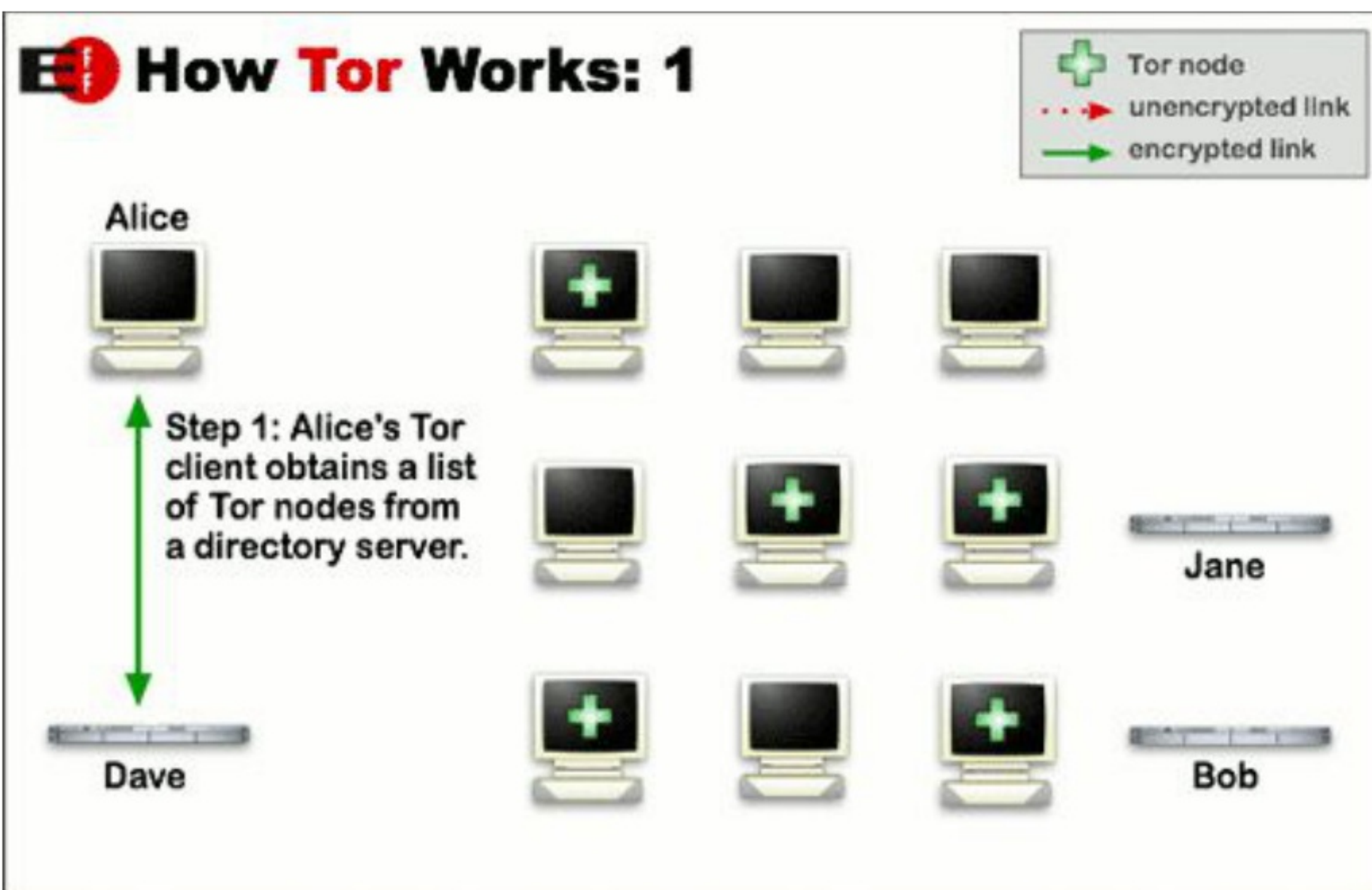
Tor is based on Onion Routing, a technique for anonymous communication over a computer network.



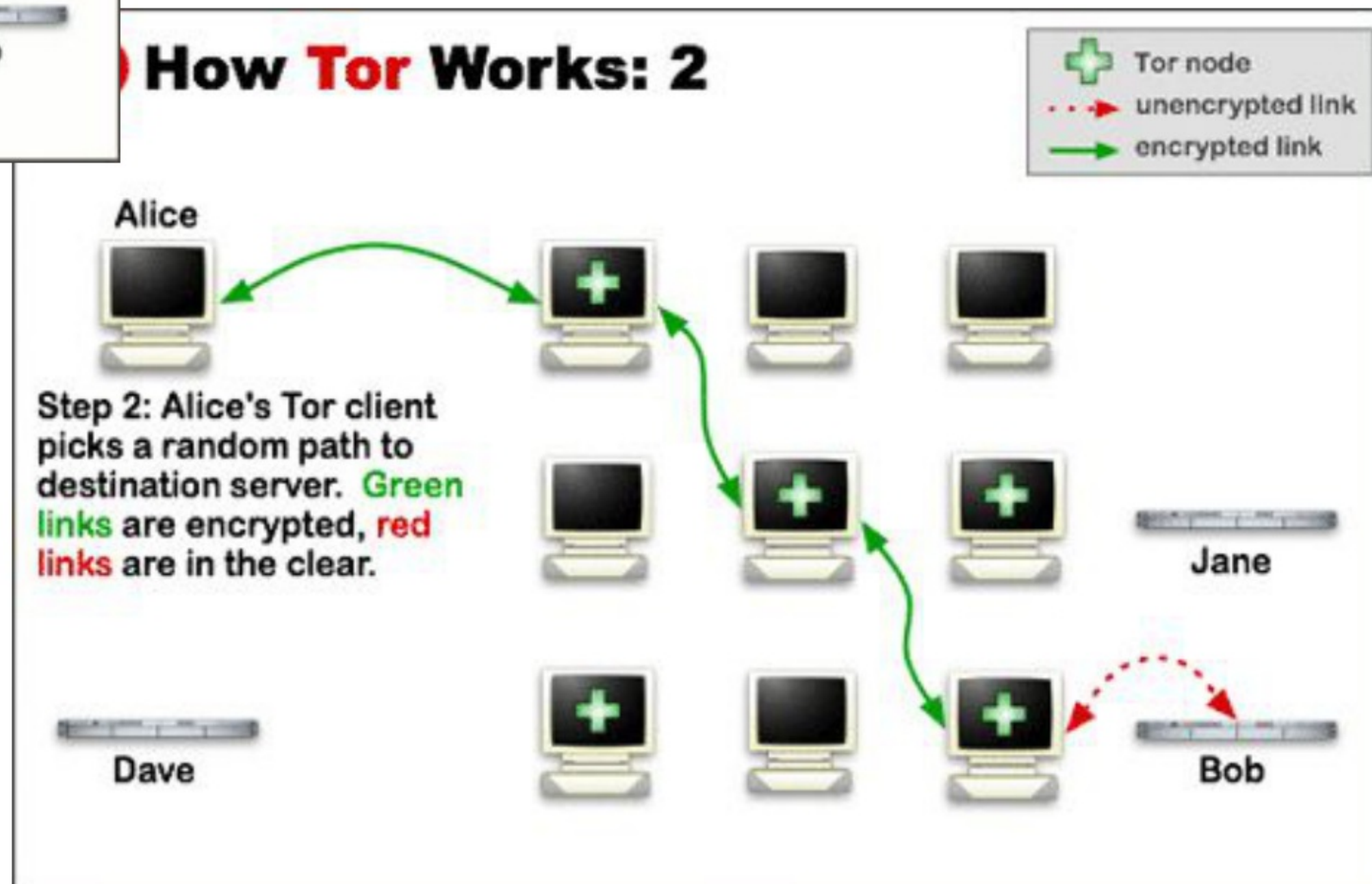
Onion Routing



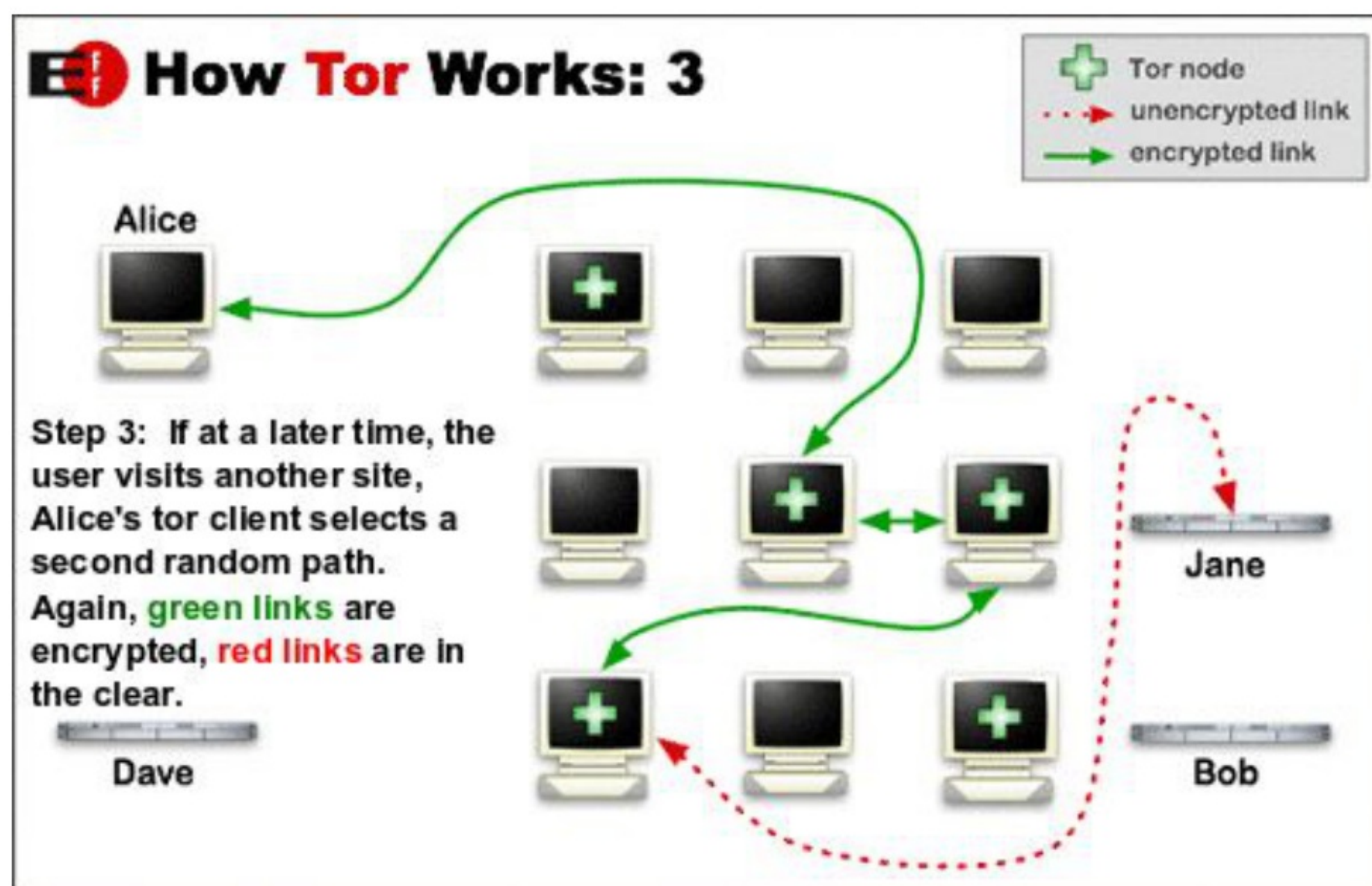
Establish TOR circuit



User's software or client incrementally builds a circuit of encrypted connections through relays on the network.

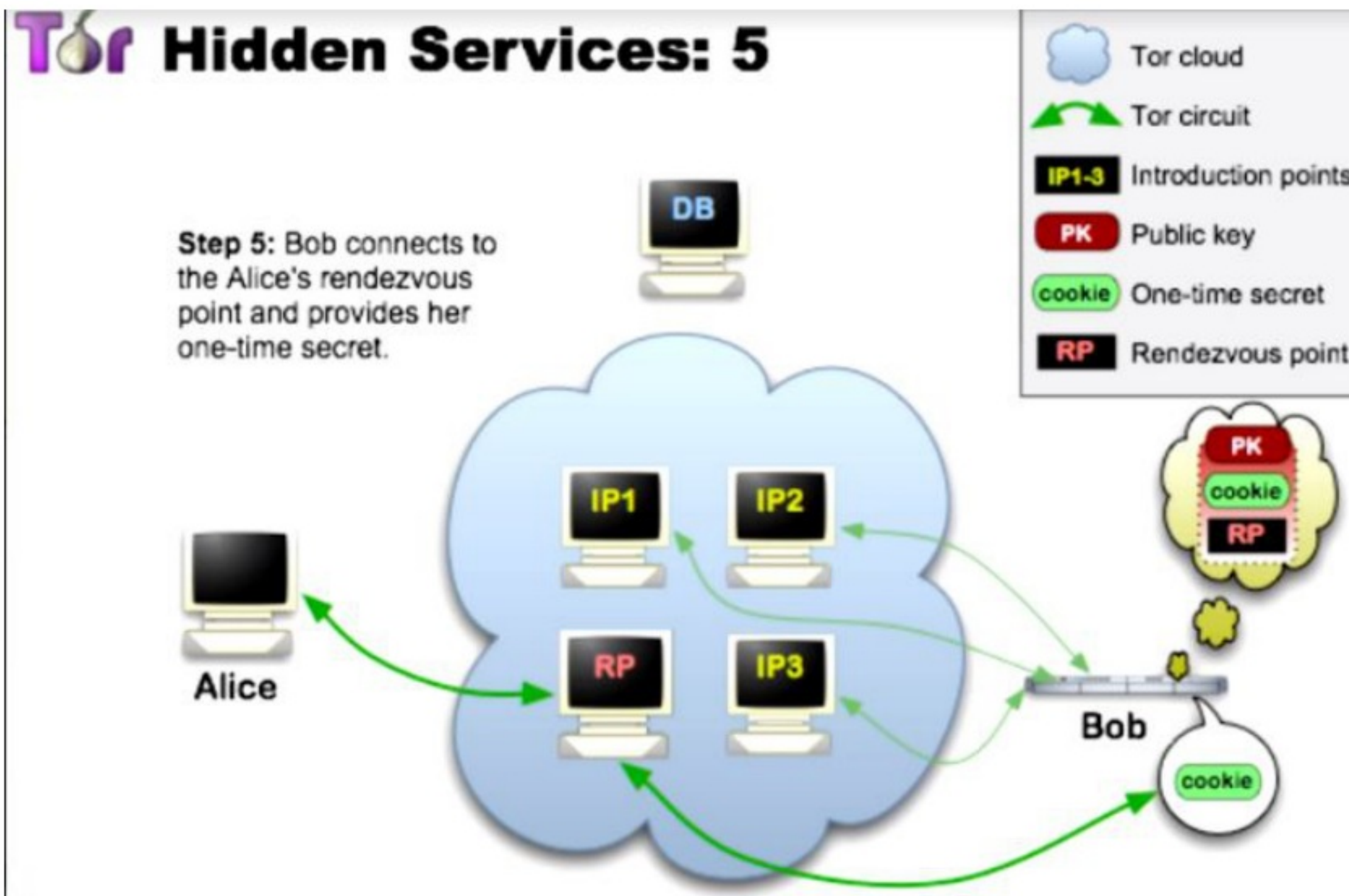


Establish TOR circuit

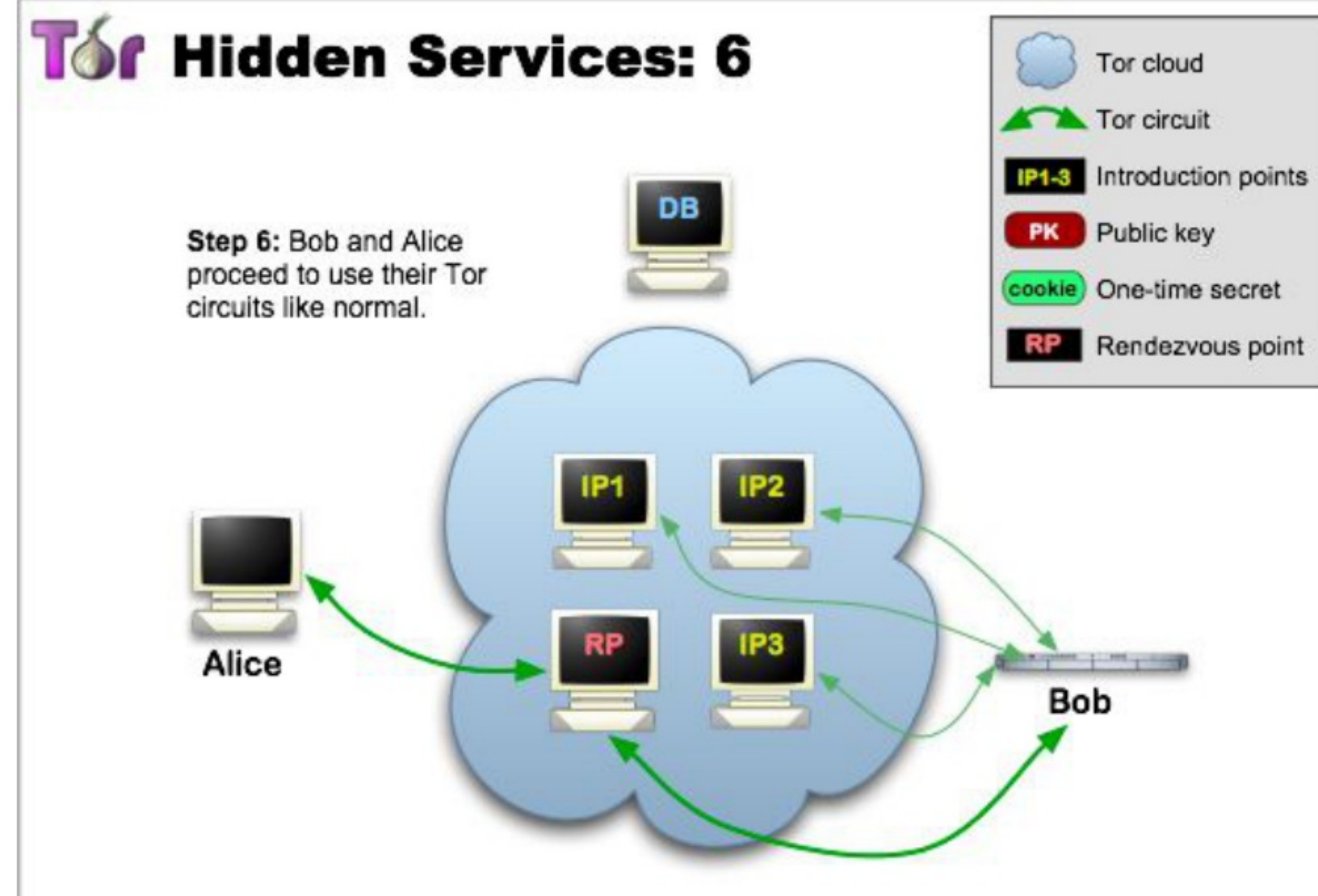


When we connect to the TOR network, we do it through a circuit formed by 3 repeaters, where the encrypted packet sent from the client is passing. Each time the packet goes through a repeater, an encryption layer is added.

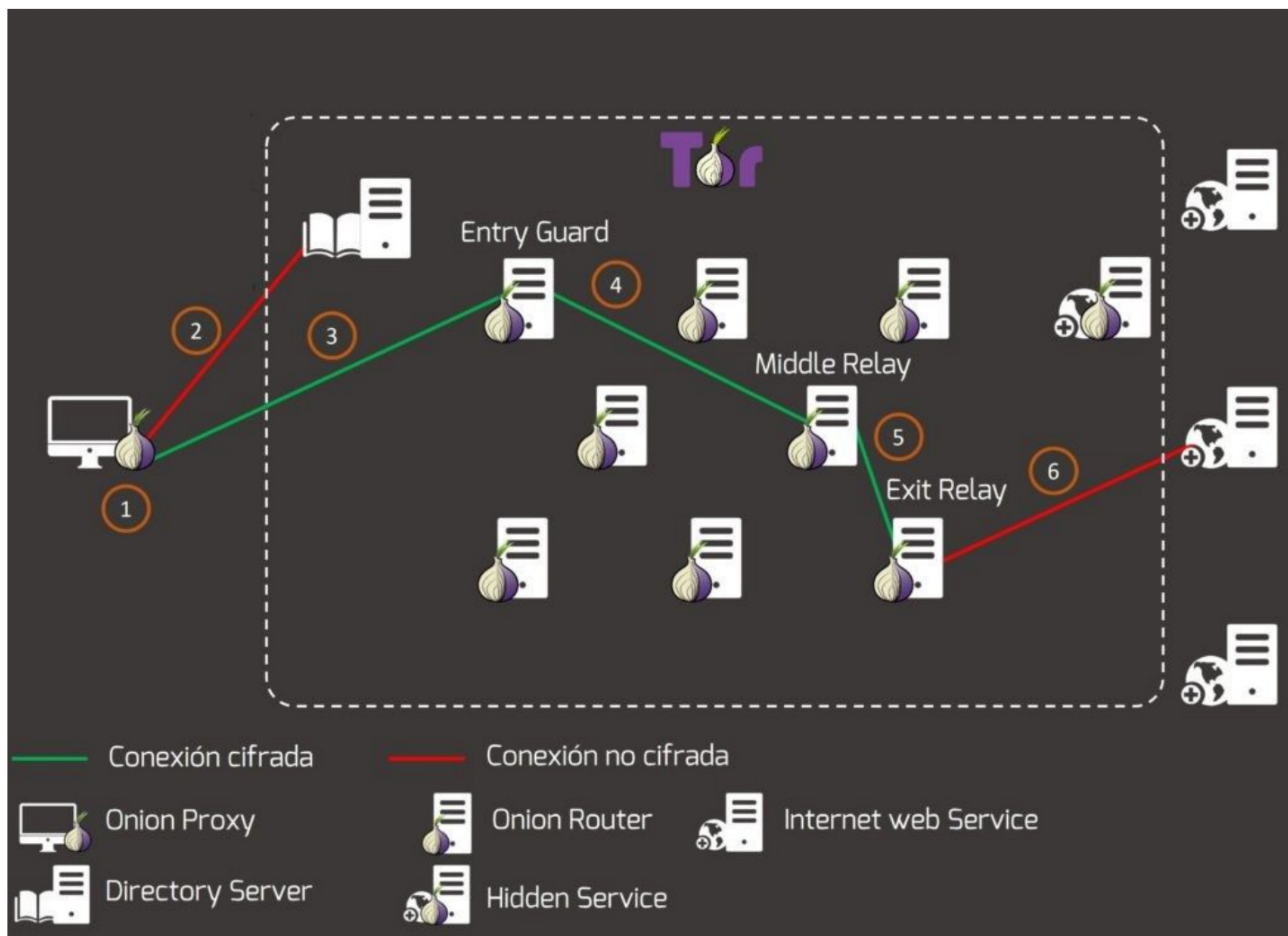
Hidden services



User's software or client incrementally builds a circuit of encrypted connections through relays on the network.



Directory server

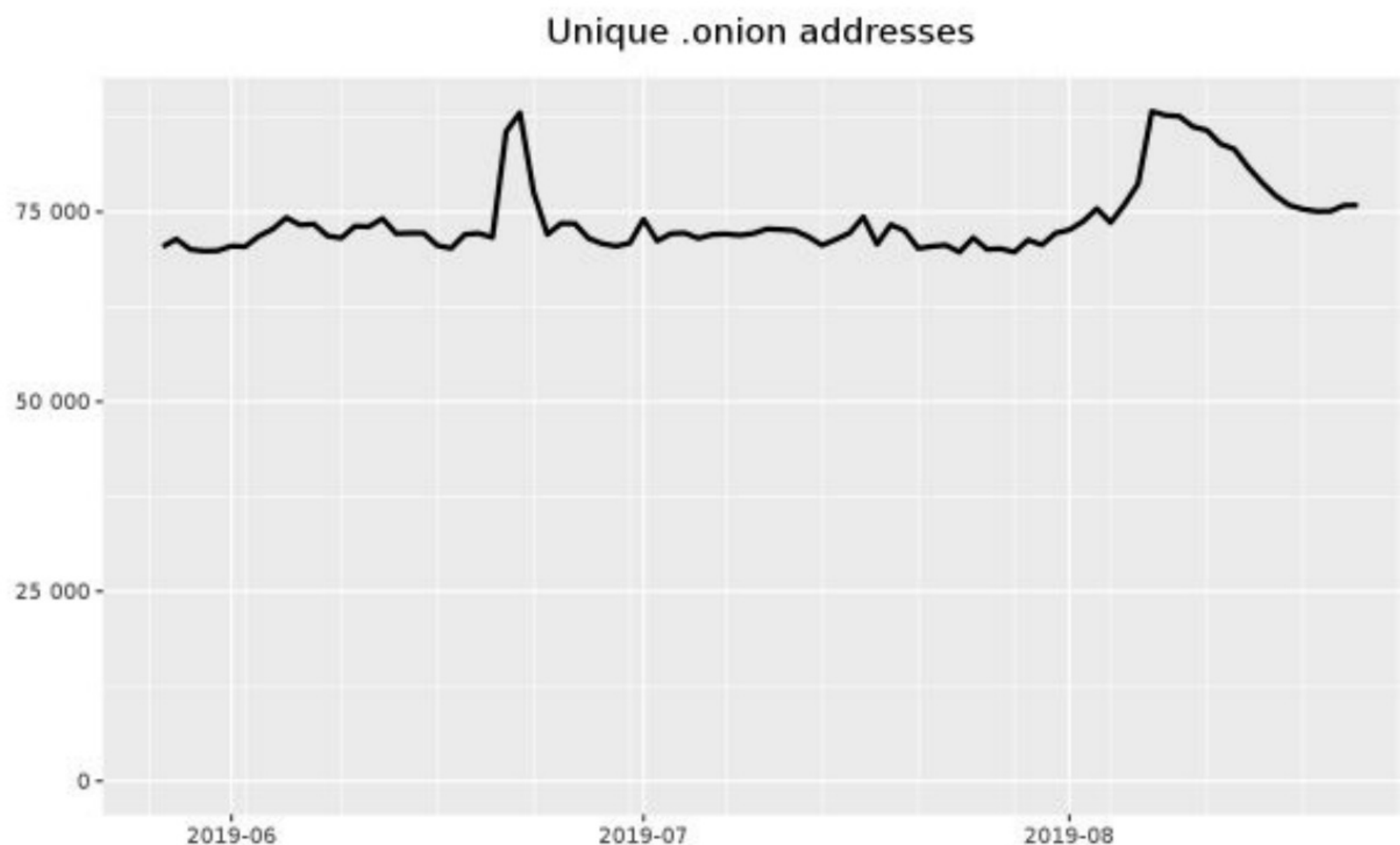


Hidden services

<https://metrics.torproject.org/hidserv-dir-onions-seen.html>

Unique .onion addresses (version 2 only)

Onion-service traffic (versions 2 and 3)



The Tor Project - <https://metrics.torproject.org/>

Tor NODE List

Aggregate Network Statistic Summary Graphs / Details		
Total Bandwidth of displayed Routers [MByte/s]	20734	
Total Number of Routers	6641	100%
Routers in Current Query Result Set	6640	99.98%
Total Number of 'Authority' Routers	10	0.2%
Total Number of 'Bad Directory' Routers	0	0%
Total Number of 'Bad Exit' Routers	1	0%
Total Number of 'Exit' Routers	921	13.9%
Total Number of 'Fast' Routers	5783	87.1%
Total Number of 'Guard' Routers	2922	44%
Total Number of 'Hibernating' Routers	1	0%
Total Number of 'Named' Routers	0	0%
Total Number of 'Stable' Routers	5592	84.2%
Total Number of 'Running' Routers	6641	100%
Total Number of 'Valid' Routers	6641	100%
Total Number of 'V2Dir' Routers	6012	90.5%
Total Number of 'Directory Mirror' Routers	4457	67.1%

Tor NODE List



<https://www.dan.me.uk/tornodes>

<http://torstatus.blutmagie.de>

Router Name	Bandwidth (KB/s)	Uptime	Hostname	ORPort	DirPort	Bad Exit	FirstSeen	ASName
IPredator	95512	9 d	exit1.ipredator.se [197.231.221.211]	443	9030	X	2014-04-19	CYBERDYNE, L
BizNasty	73940	80 d	46.166.187.77 [46.166.187.77]	9001	9030	X	2019-02-03	NFORCE, NL
Hansa	72654	15 d	ns3130724.ip-51-75-145.eu [51.75.145.219]	443	80	X	2019-04-03	OVH, FR
exit4	59340	19 d	exit4.tor-network.net [31.220.40.54]	443	80	X	2019-03-13	AMARUTU-TEC
flo	58785	62 d	delta.flo-du.com [144.76.196.92]	9001	9090	X	2018-10-19	HETZNER-AS, I
drjohn	51026	125 d	ns3129544.ip-51-75-144.eu [51.75.144.67]	443	8080	X	2018-12-21	OVH, FR
AIKO	50623	112 d	78.129.218.56 [78.129.218.56]	443	80	X	2019-01-03	IOMART-AS, GE
PIAzhexit	46313	42 d	zrh-exit.privateinternetaccess.com [195.206.105.217]	443	80	X	2018-12-21	M247, GB
exit3	45161	42 d	exit3.tor-network.net [31.220.0.225]	443	80	X	2018-12-15	AMARUTU-TEC
PrivacyRepublic0001	44514	157 d	tor-exit-node.1.privacyrepublic.org [178.32.181.96]	443	80	X	2014-11-21	OVH, FR
ExitNinja	44412	59 d	46.165.245.154 [46.165.245.154]	443	80	X	2014-11-06	LEASEWEB-DE
haedus	44146	13 d	static.232.44.9.176.clients.your-server.de [176.9.44.232]	9001	9030	X	2018-08-03	HETZNER-AS, I
torrelay01	43837	128 d	78.129.150.54 [78.129.150.54]	9001	9030	X	2018-12-18	IOMART-AS, GE
HORUS1	43349	4 d	ra.horus-it.com [94.130.34.199]	9001	9030	X	2016-08-04	HETZNER-AS, I
hyacinthinus	41936	21 d	94.23.150.81 [94.23.150.81]	443	80	X	2017-05-01	OVH, FR
theWOPR	41336	125 d	customer.worldstream.nl [212.8.243.229]	9001	9030	X	2018-12-21	WORLDSTREA
martasy	41094	50 d	ip81-173-112-100.pbiaas.com [81.173.112.100]	9030	9040	X	2018-12-11	PROFITBRICKS
sellerie	40211	55 d	tor-exit-anonymizer-09.appliedprivacy.net [109.70.100.10]	443	80	X	2018-05-01	NEXTLAYER-AS
idideditheconfig	40084	58 d	185-171-130-240.skpnet.nl [185.171.130.240]	9001	9030	X	2018-02-11	UNET Unet Netv
poiuty	39976	44 d	static.166.105.4.46.clients.your-server.de [46.4.105.166]	443	80	X	2016-01-26	HETZNER-AS, I
gurke	39557	60 d	tor-exit-anonymizer-07.appliedprivacy.net [109.70.100.8]	443	80	X	2018-04-16	NEXTLAYER-AS
rucola	38926	60 d	tor-exit-anonymizer-08.appliedprivacy.net [109.70.100.9]	443	80	X	2018-04-18	NEXTLAYER-AS
Unnamed	38806	128 d	78.129.150.83 [78.129.150.83]	9001	9030	X	2018-12-18	IOMART-AS, GE
erbse	38362	60 d	tor-exit-anonymizer-02.appliedprivacy.net [109.70.100.3]	443	80	X	2018-04-05	NEXTLAYER-AS
PIAnyexit	38119	42 d	nyc-exit.privateinternetaccess.com [209.95.51.11]	443	80	X	2019-01-22	HOSTINGSERV
radieschen	37486	60 d	tor-exit-anonymizer-01.appliedprivacy.net [109.70.100.2]	443	80	X	2018-08-16	NEXTLAYER-AS
sofia	37327	12 h	chomsky.torservers.net [77.247.181.162]	443	80	X	2017-10-03	NFORCE, NL

Tor NODE List



<https://onionite.now.sh>

Top nodes by consensus weight

#	Nickname	Bandwidth	Uptime	Country	Flags	Type
1	drakeforce1	76.2 MB/s	254d 19h	United Kingdom	      	Relay
2	Unnamed	87.9 MB/s	257d 1h	United Kingdom	      	Relay
3	privacyguardian	77.3 MB/s	251d 2h	United Kingdom	      	Relay
4	LittleFoxRahja	81.1 MB/s	172d 3h	Germany	      	Relay
5	Unnamed	85.4 MB/s	257d 1h	Germany	      	Relay
6	drjohn	78 MB/s	247d 1h	Germany	      	Relay
7	martinsrelay	77.7 MB/s	251d 7m	United Kingdom	      	Relay
8	cryptocrax0r	72.4 MB/s	247d 1h	Germany	      	Relay
9	PicklePower	72.9 MB/s	4d 7h	United Kingdom	      	Relay
10	torrelay01	69.6 MB/s	250d 7h	United Kingdom	      	Relay

Exonera TOR



<https://metrics.torproject.org/exonerator.html>

[Home](#) » [Services](#) » ExoneraTor

ExoneraTor

Enter an IP address and date to find out whether that address was used as a Tor relay:

IP address

86.59.21.38

Date

dd/mm/yyyy

Search

About Tor

Tor is an international software project to anonymize Internet traffic by [encrypting packets and sending them through a series of hops before they reach their destination](#). Therefore, if you see traffic from a Tor relay, this traffic usually originates from someone using Tor, rather than from the relay operator. The Tor Project and Tor relay operators have no records of the traffic that passes over the network and therefore cannot provide any information about its origin. Be sure to [learn more about Tor](#), and don't hesitate to [contact The Tor Project, Inc.](#) for more information.

About ExoneraTor

The ExoneraTor service maintains a database of IP addresses that have been part of the Tor network. It answers the question whether there was a Tor relay running on a given IP address on a given date. ExoneraTor may store more than one IP address per relay if relays use a different IP address for exiting to the Internet than for registering in the Tor network, and it stores whether a relay permitted transit of Tor traffic to the open Internet at that time.

Relay search



<https://metrics.torproject.org/rs.html#simple>

Relay Search

Simple Search

Aggregated Search

Advanced Search

The relay search tool displays data about single relays and bridges in the Tor network. It provides useful information on how relays are configured along with graphs about their past.

1.161.127.163

Search

Top Relays

You can search for Tor relays and bridges by using keywords. In particular, this tool enables you to search for (partial) nicknames (e.g., "moria"), IP addresses (e.g., "128.31."), and fingerprints (e.g., "9695DFC3"). It is also possible to combine searches, e.g., "moria 128.31.". Finally, you can use qualifiers to search for relays in specific countries (e.g., "moria country:us"), with specific contact information (e.g., "contact:arma"), or with specific flags (e.g., "flag:Authority").

If you are searching for a bridge, you will need to search by the hashed fingerprint. This prevents leaking the fingerprint of the bridge when searching. You can find this in the `hashed-fingerprint` file in the Tor data directory. On Debian systems, this is in `/var/lib/tor` but may be in another location on your system. The location is specified as `DataDirectory` in your `torrc`.

Relay search

<https://metrics.torproject.org/rs.html#simple>

Relay Search

country:it

country:it

Show 10 entries

Nickname†	Advertised Bandwidth	Uptime	Country	IPv4	IPv6	Flags	Add. Flags	ORPort	DirPort	Type
● pingu (4)	20.4 MiB/s	23d 22h	🇮🇹	176.126.83.211	-	⚡🛡️🔒🔄🔍🔑🔒		9001	9030	Relay
● Dazzle (1)	18.86 MiB/s	97d 6h	🇮🇹	54.37.207.82	-	⚡🛡️🔒🔄🔍🔑🔒	⚠️	443	0	Relay
● bauruine55 (8)	18.1 MiB/s	51d 7h	🇮🇹	158.58.170.183	2a02:29e0:2:5::7	⚡🛡️🔒🔄🔍🔑🔒	🔑	443	80	Relay
● mailaddressislegit (1)	17.6 MiB/s	28d 9h	🇮🇹	83.136.106.130	2a02:29e0:2:6:1:1:123d:4d1c	⚡🛡️🔒🔄🔍🔑🔒	🔑	54200	54400	Relay
● CanopoIT (6)	16.88 MiB/s	47d 8h	🇮🇹	37.9.231.195	2001:4b78:2006:ffc3::1	⚡🛡️🔒🔄🔍🔑🔒	🔑 v6	443	80	Relay
● bauruine66 (8)	15.15 MiB/s	51d 7h	🇮🇹	94.198.98.21	2a02:29e0:2:5::1c	⚡🛡️🔒🔄🔍🔑🔒	🔑	3443	8888	Relay
● bauruine56 (8)	12.95 MiB/s	51d 7h	🇮🇹	94.198.98.21	2a02:29e0:2:5::1c	⚡🛡️🔒🔄🔍🔑🔒	🔑	443	80	Relay
● bauruine65 (8)	12.71 MiB/s	51d 7h	🇮🇹	158.58.170.183	2a02:29e0:2:5::7	⚡🛡️🔒🔄🔍🔑🔒	🔑	4443	5580	Relay
● Unnamed (1)	11.49 MiB/s	203d 17m	🇮🇹	91.134.147.134	-	⚡🛡️🔒🔄🔍🔑🔒	⚠️	9001	9030	Relay
● blackmamba (1)	11.49 MiB/s	45d 8h	🇮🇹	195.135.194.134	2001:678:7dc:134::dead:beef	⚡🛡️🔒🔄🔍🔑🔒	⚠️🔑 v6	443	80	Relay
Total	378.56 MiB/s									

Relay search



<https://metrics.torproject.org/rs.html#simple>

Relay Search

1.161.127.163

Details for: Unnamed ●

Configuration

Nickname

Unnamed

OR Addresses

1.161.127.163:80

Contact

kcwu A csie org

Dir Address

none

Exit Addresses

1.161.127.163

Advertised Bandwidth

1.58 MiB/s

IPv4 Exit Policy Summary

accept
53
80
443
1194
...

Properties

Fingerprint

0A50AA36FD41F4BED31810DED7CDB9B3D32E1686

Uptime

1 day 1 hour 23 minute and 45 seconds

Flags

Exit Fast Running V2Dir Valid

Additional Flags

none

Host Name

1-161-127-163.dynamic-ip.hinet.net

Country

Taiwan (🇹🇼)

AS Number

AS3462

AS Name

Data Communication Business Group

Discover hidden services



HiddenWiki: <http://wikitjerrta4qgz4.onion/>

Dark Links: <http://wiki5kauuihowqi5.onion>

Tor Links: <http://torlinkbgs6aabns.onion>

Dark Web Links:

<http://jdpskjmgy6kk4urv.onion/links.html>

HDWiki: <http://hdwikicorldcisiy.onion>

OnionDir: <http://dirnxxdraygbifgc.onion>

DeepLink: <http://deeplinkdeatbml7.onion>

Ahmia: <http://msydqstlz2kzerdg.onion>

Tor onion services



Browser address bar: wikitjerrta4qgz4.onion

The Hidden Wiki

Hidden Wiki - Tor Wiki - Onion Links Directory [\[edit\]](#)

Welcome to the new Hidden Wiki, your Deep Web url list. Partly moderated and without spam links, now located at easy to remember url: **wikitjerrta4qgz4.onion**

Editor's picks [\[edit\]](#)

Bored? Pick a random page from the article index and replace one of the five slots with it.

1. [TORLINKS](#) Directory for .onion sites, moderated.
2. [OnionWallet](#) - Anonymous Bitcoin Wallet and Bitcoin Laundry.
3. [EasyCoin](#) - Bitcoin Wallet with free Bitcoin Mixer.
4. [Bitcoin mixing guide](#) - Mixing/Cleaning bitcoins before using them on Silkroad.

Volunteer TODO [\[edit\]](#)

Bored? Here are five random things to help out with

1. Plunder other hidden service lists for links and place them here
2. File the [SnapBBSIndex](#) links wherever they go.
3. Set external links to HTTPS where available, good certificate, and same content.
4. Care to start recording onionland's history? Check out [Onionland's Museum](#).
5. Perform Dead Services Duties.

Introduction Points [\[edit\]](#)

OnionLand link indexes and search engines

[li](#) - Clearnet search engine for Tor Hidden Services (allows you to add new sites to

Contents [\[hide\]](#)

- 1 Policy Announcement
- 2 [Editor's picks](#)
- 3 Volunteer TODO
- 4 Introduction Points
- 5 Marketplace
 - 5.1 Financial Services
 - 5.2 Commercial Services
- 6 Hosting / Web / File / Image
- 7 Blogs / Essays
- 8 Forums / Boards / Chans
- 9 Email / Messaging
- 10 Political Advocacy
- 11 Whistleblowing
 - 11.1 WikiLeaks
 - 11.2 Operation AntiSec
 - 11.3 Other
- 12 H/P/A/W/V/C
- 13 Audio - Music / Streams
- 14 Video - Movies / TV
- 15 Books

Tor onion services

https://en.wikipedia.org/wiki/List_of_Tor_onion_services

https://en.wikipedia.org/wiki/The_Hidden_Wiki

V · T · E		Tor onion services	[hide]
		List · Category	
Web directories	The Hidden Wiki		
Search engines	Ahmia · DuckDuckGo · Grams · MetaGer · Searx		
File storage and peer-to-peer file sharing	BTDigg · Freedom Hosting · Free Haven Project · KickassTorrents · The Pirate Bay		
Email and instant messaging	Bitmessage.ch · Riseup · Tor Mail · TorChat · SIGAINT · ProtonMail		
Social media and forums	8chan · Dark0de · Facebook · HackBB · Russian Anonymous Marketplace · The Hub · Tor Carding Forum		
Cryptocurrency tumblers	Bitcoin Fog · Blockchain.info · Helix		
Commerce	Agora · AlphaBay · Assassination market · Atlantis · Black Market Reloaded · Dream Market · Evolution · The Farmer's Market · Hansa · Sheep Marketplace · Silk Road · TheRealDeal · Utopia		
News, whistleblowing, and document archives	Archive.is · BuggedPlanet · DeepDotWeb · Doxbin · Filtrala · GlobalLeaks · Independent Media Center · Ljust · NawaatLeaks · ProPublica · SecureDrop · Sci-Hub · The Daily Stormer · The Intercept · WildLeaks · WikiLeaks		
Nonprofit organizations	Courage Foundation · Freedom of the Press Foundation · La Quadrature du Net · Telecomix		
Pornography	Lolita City · Playpen · Childs Play		
		Tor · .onion domain · Tor2web	



<https://www.onion.to>



onion.to Tor Hidden Services Gateway

This gateway to Tor hidden services provides convenient access to Tor hidden services. It is a pure proxy that forwards requests to the respective hidden service. We do not store any data and are not liable for the content.

No anonymity!

Onion.to as a gateway **cannot offer any anonymity for the visitor**. For example, both onion.to and the hidden service itself can see the visitor's IP address, and use [browser fingerprinting](#) to track users across different sessions. [In all cases, it is better to download the Tor Browser Bundle](#) and access the hidden service using Tor (don't forget to remove the .to from the URL!).

Enter onion address here:

open via onion.to proxy

TOR browser



<https://www.torproject.org/download/>

A screenshot of the Tor Project website. The header features the Tor logo, navigation links (About, Documentation, Support, Blog, Donate), a language selector (English (En)), and a 'Download Tor Browser' button. The main content area has a large heading 'Defend yourself.' followed by the text 'Protect yourself against tracking, surveillance, and censorship.' Below this are four circular icons representing different operating systems: Windows, Apple, Linux (Tux penguin), and Android. Each icon has a 'Sig' button underneath it.

Tor

About Documentation Support Blog Donate

English (En) Download Tor Browser

Defend yourself.

Protect yourself against tracking, surveillance, and censorship.

Windows Apple Linux Android

Sig Sig Sig

Onion Routing

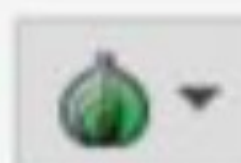
How do circuits work?

Circuits are made up of randomly assigned relays, which are computers around the world configured to forward Tor traffic. Circuits allow you to browse privately and to connect to onion services.

1 of 3

Next

About Tor



Tor Browser

Search or enter add

New Identity Ctrl+Shift+U

New Tor Circuit for this Site Ctrl+Shift+L

Privacy and Security Settings...

Tor Network Settings...

Check for Tor Browser Update...



Duck Duck Go, Inc.
Secure Connection



Tor Circuit

- This browser
- Germany 217.182.196.70 **Guard**
- Germany 144.76.196.92
- Germany 80.240.31.211
- Relay
- Relay
- Relay
- 3g2upl4pq6kufc4m.onion

New Circuit for this Site

Your **Guard** node may not change. [Learn more](#)

Circuit Display

This diagram shows the relays that make up the circuit for this website. To prevent linking of activity across different sites, each website gets a different circuit.

2 of 3

Next

Installing TOR

```
sudo apt-get update  
sudo apt-get install tor  
sudo /etc/init.d/tor restart
```



TORrc



```
## Configuration file for a typical Tor user
## Last updated 22 September 2015 for Tor 0.2.7.3-alpha.
## (may or may not work for much older or much newer versions of Tor.)
##
## Lines that begin with "## " try to explain what's going on. Lines
## that begin with just "#" are disabled commands: you can enable them
## by removing the "#" symbol.
##
## See 'man tor', or https://www.torproject.org/docs/tor-manual.html,
## for more options you can use in this file.
##
## Tor will look for this file in various places based on your platform:
## https://www.torproject.org/docs/faq#torrc
##
## Tor opens a SOCKS proxy on port 9050 by default -- even if you don't
## configure one below. Set "SOCKSPort 0" if you plan to run Tor only
## as a relay, and not make any local application connections yourself.
#SOCKSPort 9050 # Default: Bind to localhost:9050 for local connections.
#SOCKSPort 192.168.0.1:9100 # Bind to this address:port too.
```


Running TOR

```
$ tor --SocksPort 9050 --ControlPort 9051
```

```
Tor 0.2.9.14 running on Linux with Libevent 2.0.21-stable, (c) 2014-2017 The Tor Project, Inc.  
Tor can't help you if you use it wrong! Learn how to be safe  
Read configuration file "/etc/tor/torrc".  
ControlPort is open, but no authentication method has been configured.  
Upgrade your Tor controller as soon as possible.  
Opening Socks listener on 127.0.0.1:9050  
Opening Control listener on 127.0.0.1:9051  
Bootstrapped 0%: Starting  
Bootstrapped 80%: Connecting to the Tor network  
Bootstrapped 85%: Finishing handshake with first hop  
Bootstrapped 90%: Establishing a Tor circuit  
Tor has successfully opened a circuit. Looks like client functionality is working.  
Bootstrapped 100%: Done
```


Running TOR

```
Bootstrapped 0%: Starting
Bootstrapped 5%: Connecting to directory server
Bootstrapped 10%: Finishing handshake with directory server
Bootstrapped 15%: Establishing an encrypted directory connection
Bootstrapped 20%: Asking for networkstatus consensus
Bootstrapped 25%: Loading networkstatus consensus
I learned some more directory information, but not enough to build a circuit:
Bootstrapped 40%: Loading authority key certs
Bootstrapped 45%: Asking for relay descriptors
I learned some more directory information, but not enough to build a circuit:
have 0% of guards bw, 0% of midpoint bw, and 0% of exit bw = 0% of path bw.)
Bootstrapped 50%: Loading relay descriptors
Bootstrapped 55%: Loading relay descriptors
Bootstrapped 61%: Loading relay descriptors
Bootstrapped 66%: Loading relay descriptors
Bootstrapped 72%: Loading relay descriptors
Bootstrapped 80%: Connecting to the Tor network
Bootstrapped 90%: Establishing a Tor circuit
Tor has successfully opened a circuit. Looks like client functionality is work.
Bootstrapped 100%: Done
```


Tor service

service tor start/restart
service tor status

```
● tor.service - Anonymizing overlay network for TCP (multi-instance-master)
   Loaded: loaded (/lib/systemd/system/tor.service; enabled; vendor preset: enabled)
   Active: active (exited) since lun 2019-04-22 14:51:32 CEST; 6s ago
 Process: 6371 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
 Main PID: 6371 (code=exited, status=0/SUCCESS)
    Tasks: 0
   Memory: 0B
      CPU: 0
   CGroup: /system.slice/tor.service
```


Connecting with TOR

Stem

<https://stem.torproject.org/>

TorRequest

<https://github.com/erdiaker/torrequest>

Requests + socks5

pip install stem

```
.. data:: Signal (enum)
    # Nombre
    # Description
    # Signals that the tor process will accept.
    # Descendants (0)
    .. versionchanged:: 1.3.0
        Added the HEARTBEAT signal.
    # Completes (0)
    # =====
    # Signal          Description
    # =====
    # **RELOAD** or **HUP**      reloads our torrc
    # **SHUTDOWN** or **INT**    shut down, waiting ShutdownWaitLength first if we're a relay
    # **DUMP** or **USR1**       dumps information about open connections and circuits to our log
    # **DEBUG** or **USR2**      switch our logging to the DEBUG runlevel
    # **HALT** or **TERM**       exit tor immediately
    # **NEWNYM**                 switch to new circuits, so new application requests don't share any circuits
    # **CLEARDNSCACHE**          clears cached DNS results
    # **HEARTBEAT**              trigger a heartbeat log message
    # =====
    # Shutdown (0)
```


TOR descriptors



Server descriptor: Complete information about a repeater

ExtraInfo descriptor: Extra information about the repeater

Micro descriptor: Contains only the information necessary for TOR clients to communicate with the repeater

Consensus (Network status): File issued by the authoritative entities of the network and made up of multiple entries of information on repeaters (router status entry)

Router status entry: Information about a repeater in the network, each of these elements is included in the consensus file generated by the authoritative entities.

TOR spec



index : torspec

Tor's protocol specifications

summary refs log tree commit diff

log msg ▼

Branch	Commit message	Author
master	Merge remote-tracking branch 'tor-github/pr/73'	Nick Mathewson

Age	Commit message	Author
2019-04-04	Merge remote-tracking branch 'tor-github/pr/73' HEAD master	
2019-03-30	Merge remote-tracking branch 'tor-github/pr/74'	
2019-03-30	bandwidth-file: Add time to report half network header KeyValues	
2019-03-30	bandwidth-file: Add relay line diagnostic KeyValues	
2019-03-30	bandwidth-file: Add summaries for format version 1.3.0 and 1.4	
2019-03-30	bandwidth-file: Add new RelayLine monitoring KeyValues	
2019-03-30	dir-spec: Update the Tor version for bandwidth-file-digest	
2019-03-29	bandwidth-file: Remove SP from the end of KeyValues	
2019-03-29	bandwidth-file: Fix missing cardinalities, format versions, and h	
2019-03-29	bandwidth-file: Add version 1.4.0 examples	
[...]		

Recognized keys and their values include:

"version" -- The version of the server's software, which MAY include the name of the software, such as "Tor 0.0.9.4". The name of the software, if absent, is assumed to be "Tor".

"config-file" -- The location of Tor's configuration file ("torrc").

"config-defaults-file" -- The location of Tor's configuration defaults file ("torrc.defaults"). This file gets parsed before torrc, and is typically used to replace Tor's default configuration values. [First implemented in 0.2.3.9-alpha.]

"config-text" -- The contents that Tor would write if you send it a SAVECONF command, so the controller can write the file to disk itself. [First implemented in 0.2.2.7-alpha.]

"exit-policy/default" -- The default exit policy lines that Tor will *append* to the ExitPolicy config option.

"exit-policy/reject-private/default" -- The default exit policy lines that Tor will *prepend* to the ExitPolicy config option when ExitPolicyRejectPrivate is 1.


```
from stem import Signal
from stem.control import Controller

with Controller.from_port(port = 9051) as
controller:
    controller.authenticate(password='your
password set for tor controller port in torrc')
    print("Success!")
    controller.signal(Signal.NEWNYM)
    print("New Tor connection processed")
```


Periodic Tor IP Rotation



```
import time
from stem import Signal
from stem.control import Controller
def main():
    while True:
        time.sleep(20)
        print ("Rotating IP")
        with Controller.from_port(port = 9051) as controller:
            controller.authenticate()
            controller.signal(Signal.NEWNYM) #gets new identity

if __name__ == '__main__':
    main()
```


Stem.Circuit status



```
from stem.control import Controller  
  
controller = Controller.from_port(port=9051)  
controller.authenticate()  
  
print(controller.get_info('circuit-status'))
```


Stem.Network status



```
from stem.control import Controller

controller = Controller.from_port(port=9051)
controller.authenticate(password)

entries = controller.get_network_statuses()
for routerEntry in entries:
    print(routerEntry)
```


Stem.circuits

```
for circ in sorted(controller.get_circuits()):  
    if circ.status != CircStatus.BUILT:  
        continue  
  
    print("")  
    print("Circuit %s (%s)" % (circ.id, circ.purpose))  
  
    for i, entry in enumerate(circ.path):  
        div = '+' if (i == len(circ.path) - 1) else '|'  
        fingerprint, nickname = entry  
  
        desc = controller.get_network_status(fingerprint, None)  
        address = desc.address if desc else 'unknown'  
  
    print(" %s- %s (%s, %s)" % (div, fingerprint, nickname, address))
```


Circuit 10 (GENERAL)

```
| - CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIK0, 78.129.218.56)
| - 12CF6DB4DAE106206D6C6B09988E865C0509843B (ATZv5, 159.69.114.110)
+ - E19D4503D2FD584C8099A954270A9BC819596E74 (Unnamed, 51.68.206.35)
```

Circuit 11 (GENERAL)

```
| - CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIK0, 78.129.218.56)
| - 44DF1007B545B4D8057F279025EBB33CF99BE227 (Kroell, 80.241.214.102)
+ - 9612664500871798CFB52E8A71A956F316AA0503 (Polaris, 130.230.113.235)
```

Circuit 12 (GENERAL)

```
| - CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIK0, 78.129.218.56)
| - 9E1E4F5B5F94812D02C4D18CB4086CE71CA5C614 (torpidsDEhetzner1, 78.46.217.214)
+ - 615ABEA2DE76EB3760BC51E7306BAA59F15CD8F2 (Cloud, 5.135.158.101)
```

Circuit 13 (GENERAL)

```
| - CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIK0, 78.129.218.56)
| - 91B14EB2893544F0EC8F16086261A10B8E46B5C5 (okthx, 163.172.210.167)
+ - 03EE7DDD931D92BB57B81B3038AE7C40A08AB237 (Shockrealm, 123.30.128.138)
```

Circuit 14 (GENERAL)

```
| - CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIK0, 78.129.218.56)
```


Server descriptors

```
import stem.descriptor.remote

server_descriptors = stem.descriptor.remote.get_server_descriptors().run()

with open('/tmp/descriptor_dump', 'wb') as descriptor_file:
    descriptor_file.write(''.join(map(str, server_descriptors)))
```

```
class stem.descriptor.server_descriptor.RelayDescriptor(raw_contents, validate=False, annotations=None, skip_crypto_validation=False)
```

Bases: `stem.descriptor.server_descriptor.ServerDescriptor`

Server descriptor (**descriptor specification**)

- Variables:**
- **certificate** (`stem.certificate.Ed25519Certificate`) -- ed25519 certificate
 - **ed25519_certificate** (`str`) -- base64 encoded ed25519 certificate
 - **ed25519_master_key** (`str`) -- base64 encoded master key for our ed25519 certificate
 - **ed25519_signature** (`str`) -- signature of this document using ed25519
 - **onion_key** (`str`) -- * key used to encrypt EXTEND cells
 - **onion_key_crosscert** (`str`) -- signature generated using the onion_key
 - **ntor_onion_key_crosscert** (`str`) -- signature generated using the ntor-onion-key
 - **ntor_onion_key_crosscert_sign** (`str`) -- sign of the corresponding ed25519 public key
 - **signing_key** (`str`) -- * relay's long-term identity key
 - **signature** (`str`) -- * signature for this descriptor

* attribute is required when we're parsed with validation

Changed in version 1.5.0: Added the `ed25519_certificate`, `ed25519_master_key`, `ed25519_signature`, `onion_key_crosscert`, and `ntor_onion_key_crosscert_sign` attributes.

Changed in version 1.6.0: Moved from the deprecated `pycrypto` module to `cryptography` for validating signatures.

Changed in version 1.6.0: Added the certificate attribute.

Deprecated since version 1.6.0: Our `ed25519_certificate` is deprecated in favor of our new `certificate` attribute. The base64 certificate is available via the certificate's `encoded` attribute.

Introduction points

```
from stem.control import Controller

with Controller.from_port(port = 9051) as controller:
    controller.authenticate()
    desc = controller.get_hidden_service_descriptor('3g2upl4pq6kufc4m')

    print("DuckDuckGo's introduction points are...\n")

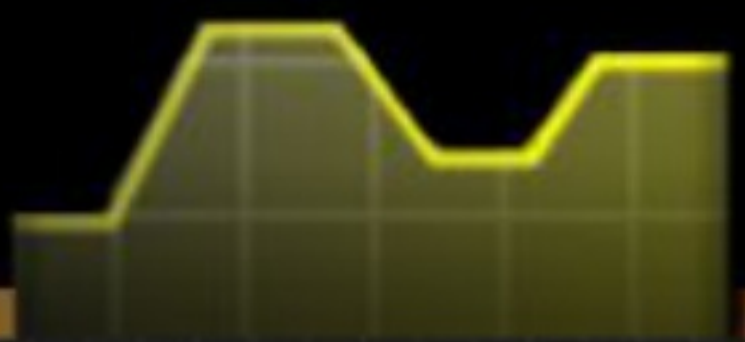
    for introduction_point in desc.introduction_points():
        print('  %s:%s => %s' % (introduction_point.address, introduction_point.port, introduction_point.introduction_point_id))
```

DuckDuckGo's introduction points are...

```
93.72.79.64:9001 => c2otnj2rbdm5o62cfse67j4dz66gmk7y
82.64.78.170:9001 => 2fszgcjszlkiiw66lzfsnedgu6rylbd3
95.216.136.46:9001 => atr3les6wx3gy3wiccr5o4bobsr7tvph
65.49.20.10:9001 => 2nlifk3ofhmrviqb4a5jx22coagznipp
194.187.249.116:443 => zzmm6ztpecmfoubpz2xrznjpmofp7pld
144.76.236.14:443 => lgnhbb2wasezhiictmtxl3xyzhsaxnsa
144.217.94.84:443 => ujqniwgkiagv6t6265uqeqbqlmvkkdeu
212.89.225.242:443 => eeyq6sagnjgkvxvsx7jsj4ftr6hbisgw
88.198.70.137:9090 => fdcszw5xrwsgcjg2mlwb2z4wncphdeqq
198.16.70.10:9001 => r3ri5lyiosxzwa7lxv43l22qlhrfi42v
```


Tor nyx

<https://nyx.torproject.org/>



Tor Nyx

Download

Nyx is available Mac OSX, Linux, and BSD but **not Windows**. Find your platform below to get started. For what's changed see our **change log**.

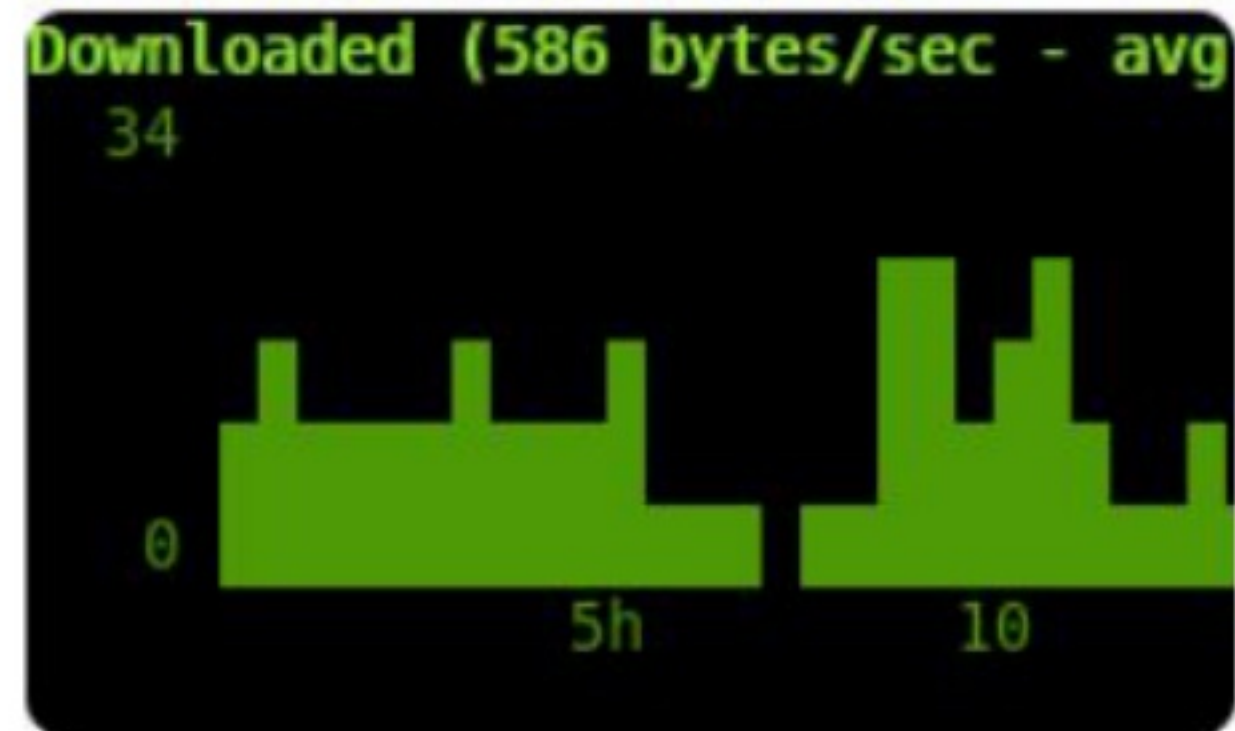


Python Package Index

Signed releases and instructions for both Python 2.x and 3.x. You can easily install from its **tarball (sig)**, or with **pip**...

```
% sudo easy_install pip
% sudo pip install nyx
```


What does Nyx provide?



<< Bandwidth Graph

Bandwidth used by Tor. You can press 'i' to pick the graphing **interval**, or 's' to show other usage **statistics**.

```
18:49:57 [INFO] router_pick_pub
18:49:57 [INFO] resolve_my_addr
public IP addresses.
18:49:57 [INFO] resolve_my_addr
18:49:57 [INFO] resolve_my_addr
18:49:57 [BW] READ: 586, WRITTE
18:49:56 [BW] READ: 0, WRITTEN:
18:49:55 [BW] READ: 0, WRITTEN:
18:49:54 [BW] READ: 1172, WRITT
```

<< Event Log

Tor logs a wealth of information about itself. We present it, colored and deduplicated. Press 'e' to select what **events** are logged and 'f' to **filter** to just what you want.

```
tor.globenet.org:58010
tor.maimed.org:48250
tor.noreply.org:41249
tor2.digineo.de:52112
torserver.uvt.nl:45561
unpatented.com:57261
vecna.uncg.edu:33392
wiredwings.org:41099
zitaraku.ath.cx:62753
```

<< Connections

Connection data similar to netstat or lsof, but correlated with Tor relay information to make it much richer. Press 'enter' for more **details**, 's' to **sort**, and 'd' to see raw **descriptor** data.

Tor nyx

```
3 User tor
4 DataDirectory /etc/tor
5 SocksPort 0
6 SocksListenAddress 127.0.0.1
7 PidFile /var/run/tor/tor.pid
8 Log notice file /var/log/tor
9 RunAsDaemon 1
10 ControlPort 9051
11 HashedControlPassword 16:22B
```

<< Torrc

Provides your torrc with line numbers and syntax highlighting. Comments can be **stripped** by pressing 's'.

```
os: Linux x86_64
version: 0.2.1.30
flags: Fast, Guard, Named, Run
exit policy: reject 1-65535
contact: www.atagar.com/contact

>>> GETINFO version
0.2.2.23-alpha (git-b85eb949b5)
```

<< Interpreter

Integration with **Stem's interpreter**, providing **raw controller access**, irc-style commands like **/help**, tab completion, history scrollback, and a python prompt.

Actions	View	Graph	Log
Close Menu	7.173.186:9001, D		
New Identity	1.0% arm	mem:	
Stop Tor	6A29789D9BC8A8879		
Reset Tor			
Setup Wizard			
Pause	to close the men		
Exit	it: 40 Mb/s, burs		

Download (0.0 b/sec);

<< ... and more!

That's just the tip of the iceberg. All pages within Nyx provide **help** information when you press 'h' and **menu** in response to 'm'. So go explore!

Tor nyx

page 3 / 5 - m: menu, p: pause, h: page help, q: quit

Tor Configuration (press 'a' to show all options):

CookieAuthentication (General Option)

Value: True (default, Boolean, usage: 0|1)

Description: If this option is set to 1, allow connections on the control port when the connecting process knows the contents of a file named "control_auth_cookie", which Tor will create in its data directory. This authentication method should only be used on systems with good filesystem security. (Default: 0)

BandwidthRate	1 GB	Average bandwidth usage limit
BandwidthBurst	1 GB	Maximum bandwidth usage limit
RelayBandwidthRate	0 B	Average bandwidth usage limit for relaying
RelayBandwidthBurst	0 B	Maximum bandwidth usage limit for relaying
ControlPort	<none>	Port providing access to tor controllers (nyx, vidalia, etc)
HashedControlPassword	<none>	Hash of the password for authenticating to the control port
CookieAuthentication	True	If set, authenticates controllers via a cookie
DataDirectory	/home/jmoc/Esitorio/tor-browser_en-US/Browser/TorBrowse...	Location for storing runtime data (state, keys, etc)
Log	notice stdout	Runlevels and location for tor logging
RunAsDaemon	False	Toggles if tor runs as a daemon process
User	<none>	UID for the process when started
Bridge	<none>	Available bridges
ExcludeNodes	<none>	Relays or locales never to be used in circuits
MaxCircuitDirtiness	10 minutes	Duration for reusing constructed circuits
SocksPort	<none>	Port for using tor as a Socks proxy
UseBridges	False	Make use of configured bridges
BridgeRelay	False	Act as a bridge
ContactInfo	<none>	Contact information for this relay
ExitPolicy	<none>	Traffic destinations that can exit from this relay
MyFamily	<none>	Other relays this operator administers
Nickname	<none>	Identifier for this relay
ORPort	<none>	Port used to accept relay traffic
AccountingMax	0 B	Amount of traffic before hibernating
AccountingStart	<none>	Duration of an accounting period
DirPortFrontPage	<none>	Publish this html file on the DirPort
DirPort	<none>	Port for directory connections
HiddenServiceDir	<none>	Directory contents for the hidden service
HiddenServicePort	<none>	Port the hidden service is provided on

VIDEO

```
Archivo Editor Ver Buscar Terminal Pestanas Ayuda
jnoc@jnoc-HP-Compaq-60... x root@jnoc-HP-Compaq-600... x jnoc@jnoc-HP-Compaq-600... x jnoc@jnoc-HP-Com
192.168.100.9:33520 => 217.182.196.70:443
127.0.0.1:9151 => 127.0.0.1:36266
127.0.0.1:9151 => 127.0.0.1:36254
jnoc-HP-Compaq-6005-Pre-SFF-PC tor_examples # clear
jnoc-HP-Compaq-6005-Pre-SFF-PC tor_examples # vi list_circuits.py
jnoc-HP-Compaq-6005-Pre-SFF-PC tor_examples # python list_circuits.py

Circuit 595 (GENERAL)
|- CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIKO, 78.129.218.56)
|- 3CCA722A00B4E676B96109061C4E9DCF2F0009EB (1EpHny, 158.69.198.177)
+- 315F0BF355D96D678267D7B708127ACBC87BA7E9 (nodvrelay10, 45.62.245.142)

Circuit 597 (GENERAL)
|- CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIKO, 78.129.218.56)
|- DE847D94E78B2E560ABB7D272DC90192D3144F17 (kohlrahi, 109.70.100.15)
+- 1C90D3AEADFF3BCD079810632C8B85637924A58E (Multivac, 163.172.53.84)

Circuit 598 (GENERAL)
|- CE3FE883C6C9EF475EA097DC3E33A6F32B852DA1 (AIKO, 78.129.218.56)
|- 6B9EA8927AB6E94E216067E65372182343A5AFA2 (angelinajolie, 62.210.83.207)
+- 13557448AD1632BF080757C2FF77698B23DF1DF3 (bauruinell, 144.76.71.91)

Circuit 600 (GENERAL)
```



```
from torrequest import TorRequest
```

```
with TorRequest() as tr:
```

```
    response = tr.get('http://ipecho.net/plain')
```

```
    print(response.text) # not your IP address
```

```
tr.reset_identity()
```

```
response = tr.get('http://ipecho.net/plain')
```

```
print(response.text) # another IP address
```


Request

```
import requests

def get_tor_session():
    session = requests.session()
    # Tor uses the 9050 port as the default socks port
    session.proxies = {'http': 'socks5h://127.0.0.1:9050',
                      'https': 'socks5h://127.0.0.1:9050'}
    return session

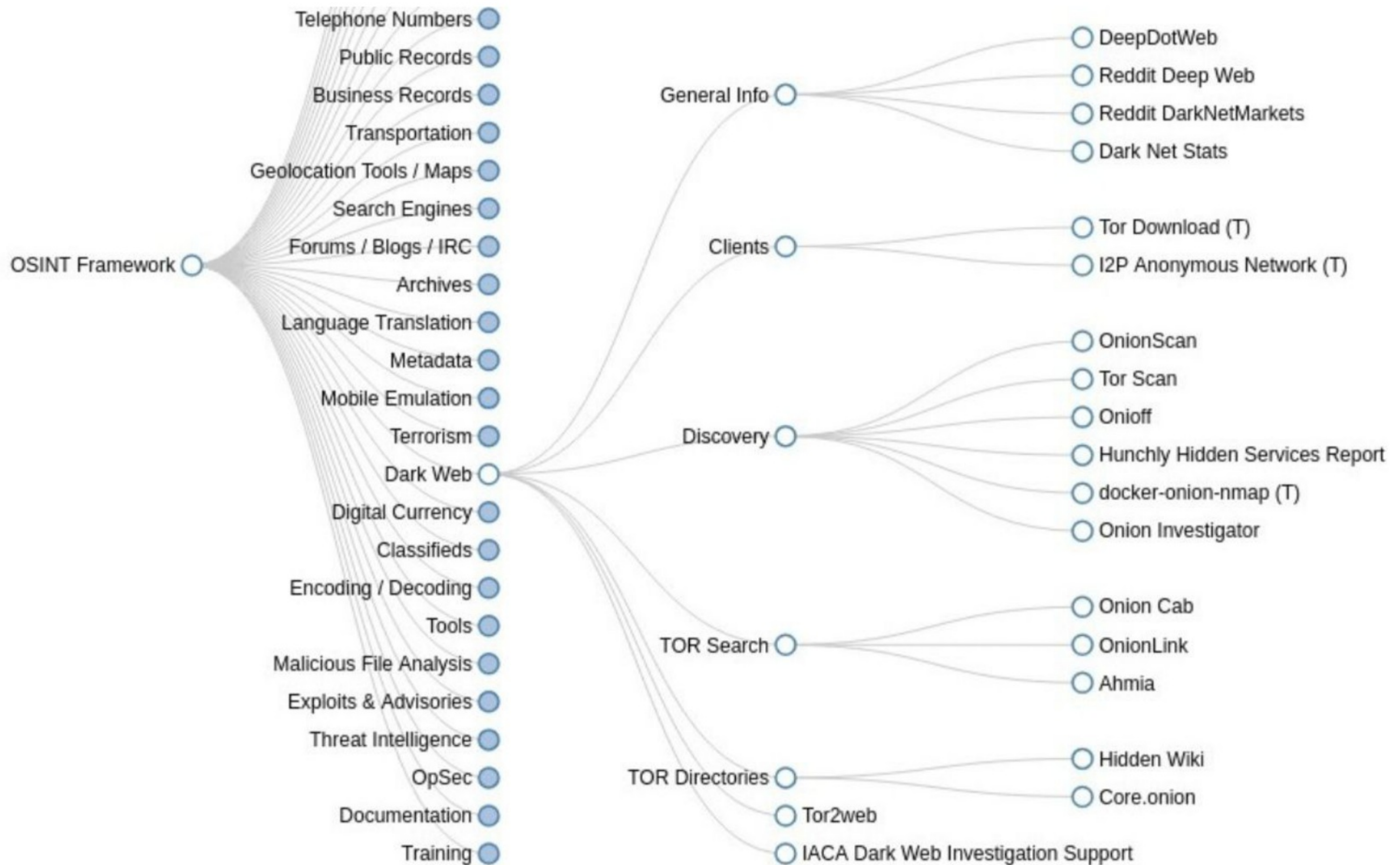
# Following prints your normal public IP
print(requests.get("http://httpbin.org/ip").text)

# Make a request through the Tor connection
# Should print an IP different than your public IP
session = get_tor_session()
print(session.get("http://httpbin.org/ip").text)
r = session.get('https://www.facebookcorewwwi.onion/')
print(r.headers)
```


Analyze hidden services



- 1) Queries to the data sources.
- 2) Filter addresses that are active.
- 3) Testing against each active address and analysis of the response.
- 4) Store URLs from websites.
- 5) Perform a crawling process against each service
- 6) Apply patterns and regular expressions to detect specific content(for example mail addresses)



Ahmia search engine

<https://ahmia.fi/>

AHMIA

Search

Ahmia searches hidden services on the Tor network. To access these hidden services, you need the [Tor browser bundle](#). Abuse material is not allowed on Ahmia. See our [service blacklist](#) and report abuse material if you find it in the index. It will be removed as soon as possible.

For more about Ahmia, see [indexing information](#) , [contribute to the source code](#).

[The Tor Project](#)

Onion service: msydstlz2kzerdg.onion

Torch search engine



<http://xmh57jrznw6insl.onion>



TORCH

Search for: [Extended](#)

Search for **bitcoin**. Search results: **bitcoin** : 15938.

Results 1-10 of 4748. Search took 4.630 seconds

Sort by: [relevancy](#) | [last modified date](#) | [title](#)

Hmm. We're having trouble finding that site.

1. [Intel chips from last 7 years can be hacked remotely | Deep Dot Web](#) [4.993%]

... Markets List Markets Chart Vpn's Chart **Bitcoin** Casinos BTC Mixer Q&A ~ Ask Here! ... guns as well, its just f... darknet **bitcoin**
dark web drug sentenced man arrested ...

- <http://deepdot35wmejd5.onion/2017/05/21...> - 94512 bytes [text/html] - Fri, 02 Nov 2018, 19:57:38 GMT
[\[Cached copy\]](#)

2. [Dav3's Android Botnet -VENDOR- - Wall Street](#) [5.058%]

... on the top we will add it for you. **Bitcoin** Price: \$150 Features: Listen to calls ...

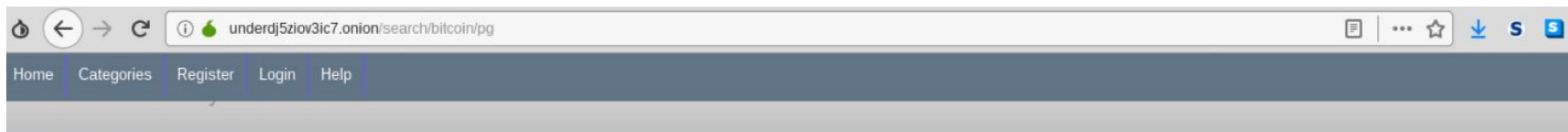
- <http://z2hjm7uhwisw5jm5.onion/viewtopic...> - 13564 bytes [text/html] - Fri, 02 Nov 2018, 19:13:57 GMT
[\[Cached copy\]](#)

3. [Login](#) [5.074%]

... это! » All that you need to know about **Bitcoin**. Всё что тебе нужно знать о Биткоин. » ...

- <http://freexd7d5vpatoe3.onion/forum/inde...> - 12422 bytes [text/html] - Wed, 31 Oct 2018, 23:26:24 GMT
[\[Cached copy\]](#)

UnderDir Search engine



Search results for bitcoin

< 1 2 3 4 5 6 7 ... 114 >

LocalBitcoins.com. Bitcoin Exchange #1 in the world

LocalBitcoins.com is a person-to-person bitcoin trading site.
LocalBitcoins.com is de-centralized, embracing the nature of bitcoin.
At LocalBitcoins.com, people from different countries can exchange their local currency to bitcoins.

★★★★★



Last check 2019-03-14 11:40 UTC Online (0 sec)

You need to login to report a site [Details](#)

Bitcoin Fog

Bitcoin anonymization taken seriously.

★★★★★



Last check 2019-03-16 10:10 UTC Online (1 sec)

You need to login to report a site [Details](#)

Bitcoin Wash

★★★★★



Last check 2019-03-21 01:20 UTC Online (1 sec)

You need to login to report a site [Details](#)

Hidden services



```
import requests

def get_tor_session():
    session = requests.session()
    # Tor uses the 9050 port as the default socks port
    session.proxies = {'http': 'socks5h://127.0.0.1:9050', 'https': 'socks5h://127.0.0.1:9050'}
    return session

def searchUnderDir(address, session):
    for page in range(1, 19):
        for searchItem in ['bitcoin']:
            addressWithCriteria = address.replace("CRITERIA_WILDCARD", searchItem)
            #http://underdj5ziov3ic7.onion/search/bitcoin/pg/1
            addressToSearch = addressWithCriteria + "/" + str(page)
            print(addressToSearch)
            response = session.get(address)
            print(response.text)

# Following prints your normal public IP
print(requests.get("http://httpbin.org/ip").text)

# Make a request through the Tor connection
# IP visible through Tor
# Should print an IP different than your public IP
session = get_tor_session()
print(session.get("http://httpbin.org/ip").text)

searchUnderDir('http://underdj5ziov3ic7.onion/search/CRITERIA_WILDCARD/pg', session)|
```


Search Hidden services

```
while len(deeplinks) <= number_results or length_of_web_links_to_crawl <= iterations:
    try:
        with timeout(10):
            crawl = session.get(deeplinks[iterations])
    except:
        error=1
    if not error:
        crawl = crawl.text
        try:
            soup = BeautifulSoup(crawl, "lxml")
        except:
            print("Error creating 'soup' object")
            os.system("sudo service tor stop")
            exit()

        for a in soup.find_all('a', href=True):
            if len(deeplinks) >= number_results:
                print(" \033[0;32m LINKS COLLECTED!\033[0m")
```

```
#process first 5 pages
for page in range(1,5):
    #http://underdj5ziov3ic7.onion/search/bitcoin/pg
    #http://www.xmh57jrznw6insl.onion/4a1f6b371c/search.cgi?s=DRP&q=bitcoin
    search_query = "http://underdj5ziov3ic7.onion/search/"+search_string+"/pg/"+str(page)
    #search_query = "http://www.xmh57jrznw6insl.onion/4a1f6b371c/search.cgi?s=DRP&q="+search_string+"&np="+str(page)

    print("Search query",search_query)
    try:
        content = session.get(search_query)
        content = content.text
```


Search Hidden services



```
# ----- MAIN PROGRAM -----  
if len(sys.argv) not in [4,5] or sys.argv[3] not in ["all", "none", "default"]:  
    print("search_dark_web.py SEARCH NUMBER_OF_RESULTS crawl_options intext")  
    print("Crawl options:")  
    print("        all) crawl each link")  
    print("        none) dont crawl")  
    print("        default) crawl if not enough links")
```


Search Hidden services



```
for a in soup.find_all('a', href=True):
    if len(deeplinks) >= number_results:
        print("\033[0;32m LINKS COLLECTED!\033[0m")
        os.system("sudo service tor stop")
        exit()

darklink = isValidOnionAdress(a['href'], session)
if darklink:
    if not darklink in deeplinks:
        if intexts in crawl or intexts == "":
            deeplinks.append(darklink)
            print(darklink)
        else:
            print("valid link, but have not '" + intexts + "' inside: \033[0;31m" + darklink + "\033[0m")
```


POOPAK - TOR Hidden Service Crawler

<https://github.com/teal33t/poopak>

Tor spider

https://github.com/absingh31/Tor_Spider

Tor router

<https://gitlab.com/edu4rdshl/tor-router>

<https://darksearch.io/>



DarkSearch

The 1st real Dark Web search engine*

SEARCH

Discover how to make advanced requests (dorks)

DarkSeach vs Ahmia



- Both offers results directly accessible on the internet thanks to Tor2Web with connecting tor network.
- **DarkSeach** provide a free API to automate searches (with some limitations to avoid the DDOS)
- **DarkSeach** indexes almost half million .onion addresses. Ahmia indexes almost 5.000 sites.
- Finally, both **search engines** not keep logs of searches done.

DarkSeach API



<https://darksearch.io/apidoc>

More than 467123 Tor pages currently indexed.



Completely free

Our DarkWeb search engine is completely free.



Daily indexing

Our crawlers surf the DarkWeb 24/7.



Direct access

Access the results directly, without the need to install Tor.



API available

Our API is **available for free** to automate your research (see [documentation](#))

GET

/api/search?query=hello&page=3



Description :

This API route allow you to query our darkweb search engine. This api is ruled by our Terms of service.

Params :

```
{
  "query": string,
  "page": int
}
```


The logo for BSIDES MANCH35T3R 2019 is located in the bottom right corner. It features the text "BSIDES" in a large, white, blocky font with horizontal lines through the letters. Below it, "MANCH35T3R" is written in a smaller, white, monospaced font. At the bottom, the year "2019" is displayed in a large, yellow, pixelated font. The entire logo is set against a black background.[illegible]

DarkSeach API



<https://darksearch.io/api/search?query=python>

```
1  import requests
2  import json
3
4  r = requests.get('https://darksearch.io/api/search?query=python')
5
6  data_json = r.text
7
8  json_data = json.loads(data_json)
9
10 pages = json_data['last_page']
11 print(pages)
12 for p in range(1,pages):
13     for i in range(0,20):
14         try:
15             url = 'https://darksearch.io/api/search?query=python'+'&page='+str(p)
16             r=requests.get(url)
17             data_json = r.text
18             json_data = json.loads(data_json)
19             print(json_data['data'][i]['title'])
20         except Exception as excep:
21             print(excep)
```


Onion investigator



<https://oi.ctrlbox.com/>

Statistics

Total Scanned: **16,602**

Out of **16,602** scanned .onions the system detects that there is **568** running no detectable http service on port 80 while there are **7,310** .onions run more than one detectable service and **8,724** .onions that only run a single service.

Service(Ports)

http	16034
smtp	6981
https	199
ssh	89
xmpp	41
ircd	38
imap	25

Web Technologies

Nginx	6831
jQuery	2939
Apache	2380
PHP	2174
Twitter Bootstrap	1248
Font Awesome	728
Google Font API	659

Servers

warning that there is some funky results within these.

nginx	14884
Apache	2050
Unknown	884
nginx/1.6.2	807
nginx/1.12.2	383
Apache/2.4.25 (Debian)	266

Onion investigator



<https://oi.ctrlbox.com/index.php?search=apps:Nginx>

Total Results

6,984

Top Services

http	6756
smtp	3504
https	72
http_alt2	9
ircd	12

Top Web Technologies

Nginx	6831
PHP	1273
jQuery	1185

previous 1 2 3 4 5 6 7 8 9 698 699 next

[vb.n2ohzmxofv4jrmy3.onion](#)

vb private pastebin

OS: unknown

Wed, 09 May 2018 12:35:57 +0000

[details](#)

Ports: [http](#)

Web Technologies: [Nginx](#)

[iehgay7x72zyca4z.onion](#)

Alert!

OS: unknown

Wed, 09 May 2018 07:49:02 +0000

[details](#)

Ports: [http](#)

Web Technologies: [Nginx](#)

Inspect onion address



<https://github.com/k4m4/onioff>

Usage: `python3 onioff.py {onion} [options]`

Options:

<code>--version</code>	show program's version number and exit
<code>-h, --help</code>	show this help message and exit
<code>-f FILE, --file=FILE</code>	name of onion file
<code>-o OUTPUT_FILE, --output=OUTPUT_FILE</code>	output filename
<code>-a, --active</code>	log active onions only to output file

Examples:

```
python3 onioff.py http://xmh57jrznw6insl.onion/
python3 onioff.py -f ~/onions.txt -o ~/report.txt -a
python3 onioff.py https://facebookcorewwi.onion/ -o ~/report.txt
```

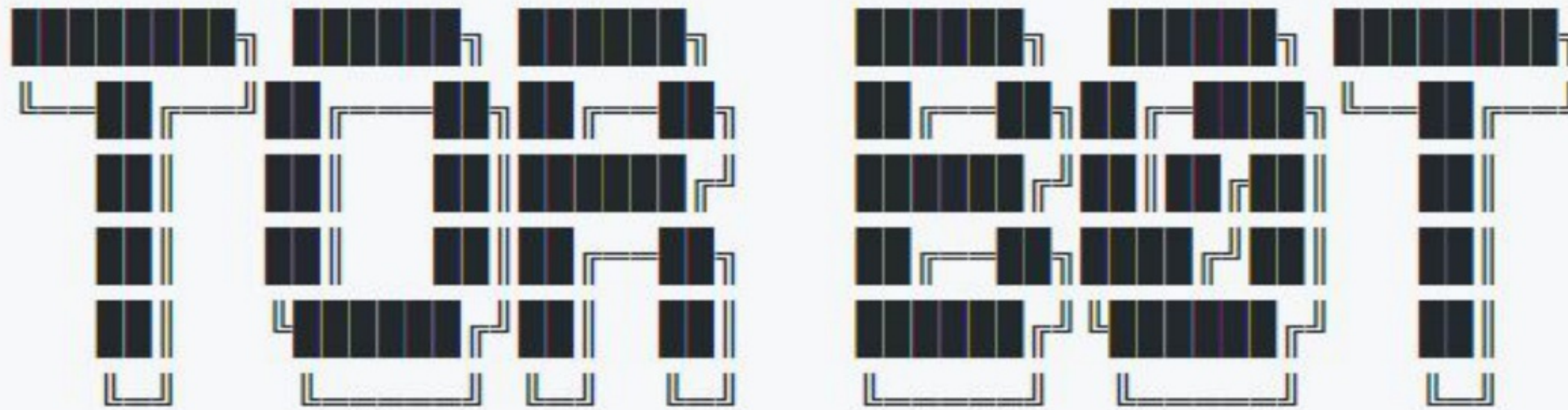

Inspect onion address

<https://github.com/k4m4/onioff>

```
[+] Commencing Onion Inspection
[+] Tor Running Normally
[!] Inspecting Onion --> http://xmh57jrznw6insl.onion/
[+] Sending Request
[+] Onion Up & Running --> ACTIVE
[+] Retrieving Onion Title
[+] Onion Title --> TORCH: Tor Search!
[!] Inspecting Onion --> http://facebookcorewwi.onion/
[+] Sending Request
[+] Onion Up & Running --> ACTIVE
[+] Retrieving Onion Title
[+] Onion Title --> Facebook – Log In or Sign Up
[!] Inspecting Onion --> http://sms4tor3vcr2geip.onion/
[+] Sending Request
[-] Onion Down --> INACTIVE
```


Crawling onion address

<https://github.com/DedSecInside/TorBot>



Open Source Intelligence Tool for the Dark Web

Crawling onion address



<https://github.com/DedSecInside/TorBot>

```
usage: torBot.py [-h] [-v] [--update] [-q] [-u URL] [-s] [-m] [-e EXTENSION]
                  [-l] [-i]
```

optional arguments:

-h, --help	Show this help message and exit
-v, --version	Show current version of TorBot.
--update	Update TorBot to the latest stable version
-q, --quiet	Prevent header from displaying
-u URL, --url URL	Specify a website link to crawl, currently returns links on that page
-s, --save	Save results to a file in json format
-m, --mail	Get e-mail addresses from the crawled sites
-e EXTENSION, --extension EXTENSION	Specify additional website extensions to the list(.com or .org etc)
-l, --live	Check if websites are live or not (slow)
-i, --info	Info displays basic info of the scanned site (very slow)

Crawling onion address

<https://github.com/MikeMeliz/TorCrawl.py>



Crawling onion address



<https://github.com/dirtyfilthy/freshonions-torscraper>

docker-onion-nmap



<https://github.com/milesrichardson/docker-onion-nmap>

```
$ docker run -e DEBUG_LEVEL=1 --rm -it milesrichardson/onion-nmap -p 80,443 facebookcorewwi.onion
[tor_wait] Wait for Tor to boot... (might take a while)
[tor_wait retry 0] Check socket is open on localhost:9050...
[tor_wait retry 0] Socket OPEN on localhost:9050
[tor_wait retry 0] Check SOCKS proxy is up on localhost:9050 (timeout 2 )...
[tor_wait retry 0] SOCKS proxy DOWN on localhost:9050, try again...
[tor_wait retry 1] Check socket is open on localhost:9050...
[tor_wait retry 1] Socket OPEN on localhost:9050
[tor_wait retry 1] Check SOCKS proxy is up on localhost:9050 (timeout 4 )...
[tor_wait retry 1] SOCKS proxy UP on localhost:9050
[tor_wait] Done. Tor booted.
[nmap onion] nmap -p 80,443 facebookcorewwi.onion
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/libproxychains4.so
[proxychains] DLL init: proxychains-ng 4.12

Starting Nmap 7.60 ( https://nmap.org ) at 2019-08-24 14:54 UTC
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... facebookcorewwi.onion:80 ... OK
[proxychains] Dynamic chain ... 127.0.0.1:9050 ... facebookcorewwi.onion:443 ... OK
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
RTTVAR has grown to over 2.3 seconds, decreasing to 2.0
```



Onion scan

<https://github.com/s-rah/onionscan>



Summary Saved Searches

Search!

Options

Save Search

Linked Tags

operation-spiky-tomato ✕

Tag Search Term

Tag!

Summary for masks3astuf5emnf.onion (We Sell Masks!)

mod_status pgp

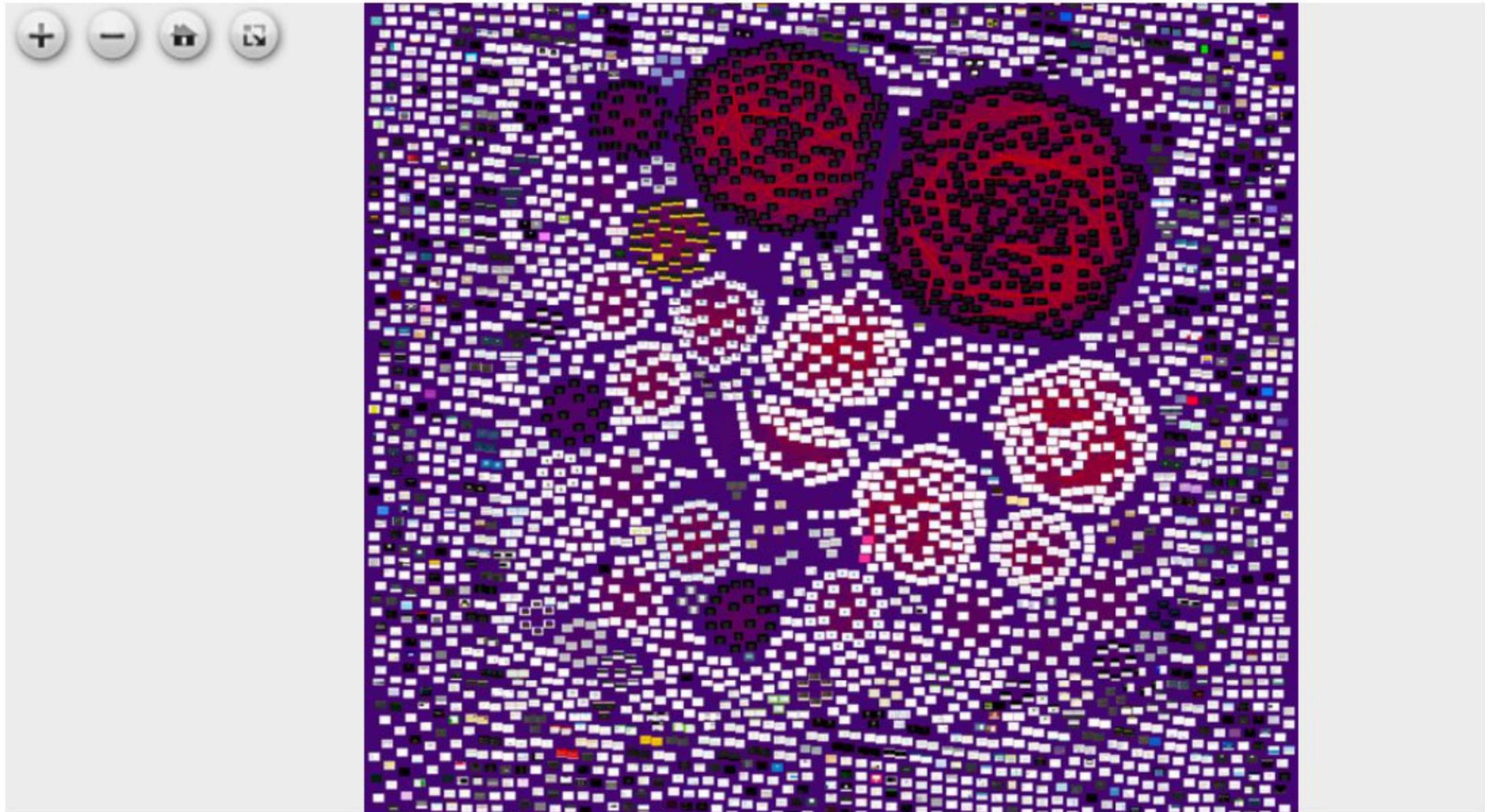
PGP Identities	1
Tag Relationships	1
Webpage Information	1
IP Addresses	13
Co-Hosted Clearnet Sites	12
HTTP Headers	4
Server Information	1

GOLANG



Dark Web map

<https://www.hyperiongray.com/dark-web-map/>



<https://github.com/serfer2/python-deepweb>

Table of contents

1. Short introduction to Tor network.
2. Tor agent installation (Ubuntu Linux) and configuration (torrc file).
3. From SOCKS to HTTP: enhance Tor with Privoxy.
4. Stem, Python's library for Tor agent managing and much more.
 - 4.1 Switching Tor circuit to get a new output IP address.
 - 4.2 Launch tor from Python program. Example: Select output IP address by country.
 - 4.3 Building hidden services in Tor network.

GitHub repositories



https://github.com/jmortega/python_dark_web

 circuit-status.py	Add files via upload	4 months ago
 current_descriptors.py	Add files via upload	4 months ago
 darkweb_python_hidden_services.pdf	Add files via upload	4 months ago
 descriptor_from_tor_data_directory.py	Add files via upload	4 months ago
 get_hidden_service_descriptor.py	Add files via upload	4 months ago
 hidden_service.py	Add files via upload	4 months ago
 info_top_relays.py	Add files via upload	4 months ago
 introduction_points.png	Add files via upload	4 months ago
 introduction_points.py	Add files via upload	4 months ago
 list_circuits.png	Add files via upload	4 months ago