

# Censorship.no! User Manual

This manual is aimed at users of the **CENO Browser** and related technologies, as created for the [Censorship.no!](#) project by [eQualitie](#).

If you are looking for technical documentation, please refer to the [CENO documentation repository](#), where you will find pointers for further reading and details on implementation, as well as the [protocol specifications](#).

# Introduction

The Internet and the World Wide Web have become more and more important for people around the world as a source of all kinds of information, and a way to exercise fundamental rights. At the same time, recent years have seen an increase in all kinds of network censorship and other types of network interference (see the reports from [OONI](#), [Magma](#), [Censored Planet](#)), both from private and state actors.

The Web relies on your devices being able to reach special computers called *web servers* (run by content creators, publishers or Internet services providers) that hold the content you want to retrieve, and to do so in a synchronized way - more like having a live chat on the telephone than exchanging some letters. Unfortunately, this requires that the web server you need is connected to the network and has enough available resources to talk to your device at that moment.

The advent of *content delivery networks* (or CDNs, like the commercial Akamai and Cloudflare, or the civil society-oriented Deflect) has taken some of the load off these web servers by distributing copies of the content to data centers all over the world, so that it can be closer to your devices and thus faster to reach, while keeping the origin servers protected from direct access. However, currently CDN servers (and thus the organizations that run them) need to be trusted by both the origin server and your devices, and they also need to be reachable at all times.

Unfortunately again, there are situations in which general connectivity is scarce (developing countries or underserved, impoverished or rural regions), expensive (with some countries charging more for international traffic) or has been actively blocked by an state player (explicitly or as a result of a general shutdown). In these cases, reaching origin web servers or even CDN servers is difficult or impossible, and your device will not be able to get that content - even if I was somehow able to access it a few hours ago and we're living some streets apart.

This is where the CENO Browser and Ouinet come into play. This chapter will introduce you to them.

# What is the CENO Browser?

CENO (short for [Censorship.no!](#)) is a Web browser for mobile Android devices (such as smartphones and tablets) that uses a novel approach to circumvent Internet censorship and share retrieved content among all users in a [peer-to-peer](#) (P2P) fashion. This reduces overall reliance on international network exchanges, and allows frequently requested Web content to persist in the network even during heavy filtering and throttling events.

What sets CENO apart from most other circumvention initiatives is that users can continue to share Web content even when no (or extremely limited) connectivity exists across national borders. CENO is thus built in anticipation of aggressive Internet filtering and the establishment of [national intranets](#) to fence off nations from the Web.

The CENO Browser is built on an adapted version of [Firefox for Android](#), a popular, modern, feature-rich and secure Free/Libre/Open Source browser. CENO extends Firefox with **Ouinet**, the underlying technology allowing it to share content between devices (described in later sections).

## Who develops CENO?

The *Censorship.no!* project is run by [eQualitie](#) in support of Articles 18, 19 and 20 of the [Universal Declaration of Human Rights](#). CENO and related technologies are developed as Free/Libre/Open Source software ([project source](#)), allowing anyone to use, study, share and enhance it. Please contact [cenors@equalit.ie](mailto:cenors@equalit.ie) in case of doubt or for further information.

## Who is it for?

CENO is meant for everyone but is particularly convenient for people interested in Web content that is censored on their network, and those those living in countries where connectivity to the global Internet is spotty, unreliable or expensive. It encourages and enables the sharing of web content among users, creating a decentralized network of peers helping each other.

You do not need to be an advanced computer user or even understand how peer-to-peer networks operate to use CENO. The user experience is akin to operating a standard Web browser (CENO-specific functionality is largely hidden under the hood).

CENO can, however, generate more Internet traffic than your usual Web browser – since it

needs to let other users know what Web content it is sharing to the network, and possibly deliver this content to those who request it. CENO thus relies on moderately good in-country connectivity. We recommend using CENO when connected to Wi-Fi not just to avoid exhausting mobile data limits (should you have them), but also to increase the chances of being able to deliver Web content to other users.

---

**Warning:** *CENO is not an anonymity tool.* In fact, using CENO may allow others to know whether you have accessed or are sharing certain Web content. Please take careful consideration of which risks you can assume by using this tool. See the sections on [public vs. private browsing](#) and [risks](#) for further information.

---

# What is Ouinet?

[Ouinet](#) is a core technology that allows the CENO browser to share Web content with other devices. Ouinet comes in the form of reusable computer code (a library) that an application like CENO can use to become a participant in a network of cooperating nodes that communicate directly ([peer-to-peer](#) or P2P) to help access and store new Web content, and to request and deliver previously accessed content to others.

Ouinet is based on a clever combination and use of existing technologies to accomplish each of its functionalities: locating other participants is done with techniques coming from the world of file-sharing (BitTorrent's distributed hash table), communicating with them uses common Web and file-sharing protocols (HTTP proxy requests and BitTorrent's  $\mu$ TP), and modern industry standards ensure the security of communications and the authenticity of exchanged content (TLS encryption and Ed25519 signatures). Ouinet allows replacing particular technologies with others if needed (for instance, some uses of  $\mu$ TP can be replaced by Tor's Pluggable Transports).

On mobile devices, Ouinet can be embedded into end-user applications (as an Android library). In computers, it can be used by normal Web clients like browsers (as a local HTTP proxy).

Same as the CENO Browser, Ouinet is developed by [eQualitie](#) as Free/Libre/Open-Source software.

## Who is it for?

Ouinet is mostly useful for software developers, content creators and publishers who want to enable users of their applications to share retrieved content with other users. This reduces the overall demand on the application server and improves content accessibility for users living in countries that block access to that server.

Please note that Ouinet is an evolving experimental project: some features may not work reliably enough in certain scenarios, bugs may exist and crashes may occur. We encourage you to reach out to us at [cenoers@equalit.ie](mailto:cenoers@equalit.ie), test it and report back - your feedback is very welcome!

---

**Warning:** *Ouinet is not an anonymity tool.* If you are unsure about its adequacy for a certain task, do not hesitate to contact us.

---

# Quick start guide

The CENO Browser allows you to access any website on the Internet, even if it is censored in your country. CENO uses a peer-to-peer infrastructure to route your requests, as well as to store and share retrieved content with others. [Read more about CENO](#).

## How to get started

You will need an Android device:

1. Install the CENO Browser from [Google Play](#), [GitHub](#) or [Paskoocheh](#). *No special permissions are needed.*
2. Run it.
3. Browse pages normally to help other users access them; if concerned about privacy for some page or if it is not loading as expected, use a private tab (see [public vs. private browsing](#)).
4. Tap on the CENO notification to stop it completely.

Detailed installation instructions are [here](#).

## Configuration

The CENO Browser should work out-of-the-box. You can find some [diagnostics and settings](#) under the *CENO* menu entry.

If you want to make sure that your app is also helping others access blocked content, please [read this section](#).

## More questions?

- Please see the [FAQ](#).
- Refer to the [troubleshooting guide](#).
- Contact us by writing to [cenoers@equalit.ie](mailto:cenoers@equalit.ie).

# Frequently Asked Questions

## Usage

### **Can the CENO Browser replace my current browser (Chrome/Firefox/Safari)?**

Short answer: yes, at least for some non-critical browsing.

Since CENO is based on Mozilla Firefox for Android, it provides all the features that are expected from a modern browser. However, its reliance on Ouinet technology for retrieving web content may affect its operation in subtle ways (some of them potentially involving your privacy).

Also, since CENO and Ouinet are under continuous development, you may experience some instability. That may also mean that backwards-incompatible changes are introduced which require you to uninstall the application or remove stored data (including bookmarks and site settings) before upgrading.

For important work on Web sites not subject to censorship, we recommend that you use your habitual Web browser instead of CENO. Please read the section on [risks](#) for more information.

### **Can I use the CENO Browser to access Twitter, Facebook and Gmail?**

Short answer: yes, by using private tabs.

Although CENO strives to provide a user experience as similar to normal Web browsing as possible, some of the techniques used to avoid network interference do not play well with such dynamic sites. This is the case with CENO's default mode of operation (i.e. public browsing), since it strips down all private data (like passwords and cookies) from Web traffic to ensure that it does not leak to other untrusted CENO or Ouinet users.

To avoid this and enable the use of such dynamic sites in CENO, you can use private tabs (i.e. private browsing), which leave private data intact and keep connections to the sites

encrypted and thus protected from everyone else. However, this requires that network traffic from your device can reach those websites in some way.

For further details, please refer to the section on the differences between [public and private browsing](#).

## Privacy and security

### Will my device store content that I did not request?

Short answer: no.

CENO only shares content that you requested (using [public browsing](#)).

Please note that a malicious website may still try to trick your browser into retrieving content from other sites without your knowledge so as to force your device to store and share it with other users. While Firefox code already does a pretty good job at detecting and blocking such attempts, you should still avoid visiting suspicious sites.

Read more about how your CENO Browser retrieves and shares Web content with others [here](#).

### Can anyone see if I am using the CENO Browser to access censored Web sites?

Short answer: yes, with some technical knowledge and resources.

CENO is not an anonymity tool. An attacker able to spy on your network traffic can see content being requested from or served to another user from your device. The attacker can also assess whether you are sharing a particular website, although they cannot list all the content that you are sharing.

However, content being retrieved into the network for the first time or using [private browsing](#) will travel over encrypted connections. See [how content retrieval works](#) and the associated [risks](#) for more information.

## Resource usage



## **Does the CENO Browser use a lot of data?**

Short answer: more than your usual browser.

Whenever your CENO Browser serves content to another user or forwards their traffic, it is consuming extra data depending on factors like how popular or large the content is, and how well-connected your device is. Sharing more content also implies more overhead.

Although CENO is much lighter in resources than other data-sharing applications, it can still result in increased data usage and fees. We recommend keeping an eye on the app's data usage (under Android settings) and running CENO while connected to Wi-Fi instead of using mobile data.

## **Does the CENO Browser use a lot of battery on my device?**

Short answer: more than your usual browser.

CENO and Ouinet use various techniques for different users to cooperate in avoiding network interference and outages. Serving content to and forwarding traffic for other users consumes extra power. Also, even when your device is not actively helping other users, it still needs to stay reachable over the network, which prevents the use of some energy-saving features.

The net result is that CENO may keep on draining your battery even when not in use. Our tests have not shown power consumption hiking too much, but your mileage may vary. When in need of battery, we recommend stopping CENO completely (it offers a convenient shortcut for this, see [here](#)).

## **Do I need to be connected to Wi-Fi to use the CENO Browser?**

Short answer: no, but we strongly recommend it.

Although CENO should work fine on a mobile connection, there are two reasons why we recommend using a Wi-Fi connection instead:

1. CENO consumes an extra amount of data which may result in a higher phone bill (see above).
2. Mobile connections often make reaching your device more difficult from the outside than Wi-Fi ones, thus decreasing the chances that you can help other users get content.

# Main concepts

To fulfill its objective of circumventing several kinds of network interference and shortages, CENO makes use of assorted techniques from the fields of the World Wide Web, file-sharing systems and advanced cryptography. These techniques are cleverly combined by Ouinet so as to make the experience of using CENO as close as possible to normal Web browsing.

Nonetheless, to get the most out of CENO it is helpful to understand how Ouinet operates, the different ways it can be used depending on the kind of content we are trying to access, and the advantages and risks that they entail. This chapter will cover these topics.

# How does it work?

This section will explain CENO and Ouinet operation by going over a series of scenarios. Terminology and concepts important to Ouinet will be introduced (highlighted in **bold letters**) and used afterwards for efficiency and to avoid confusion.

## Accessing content directly

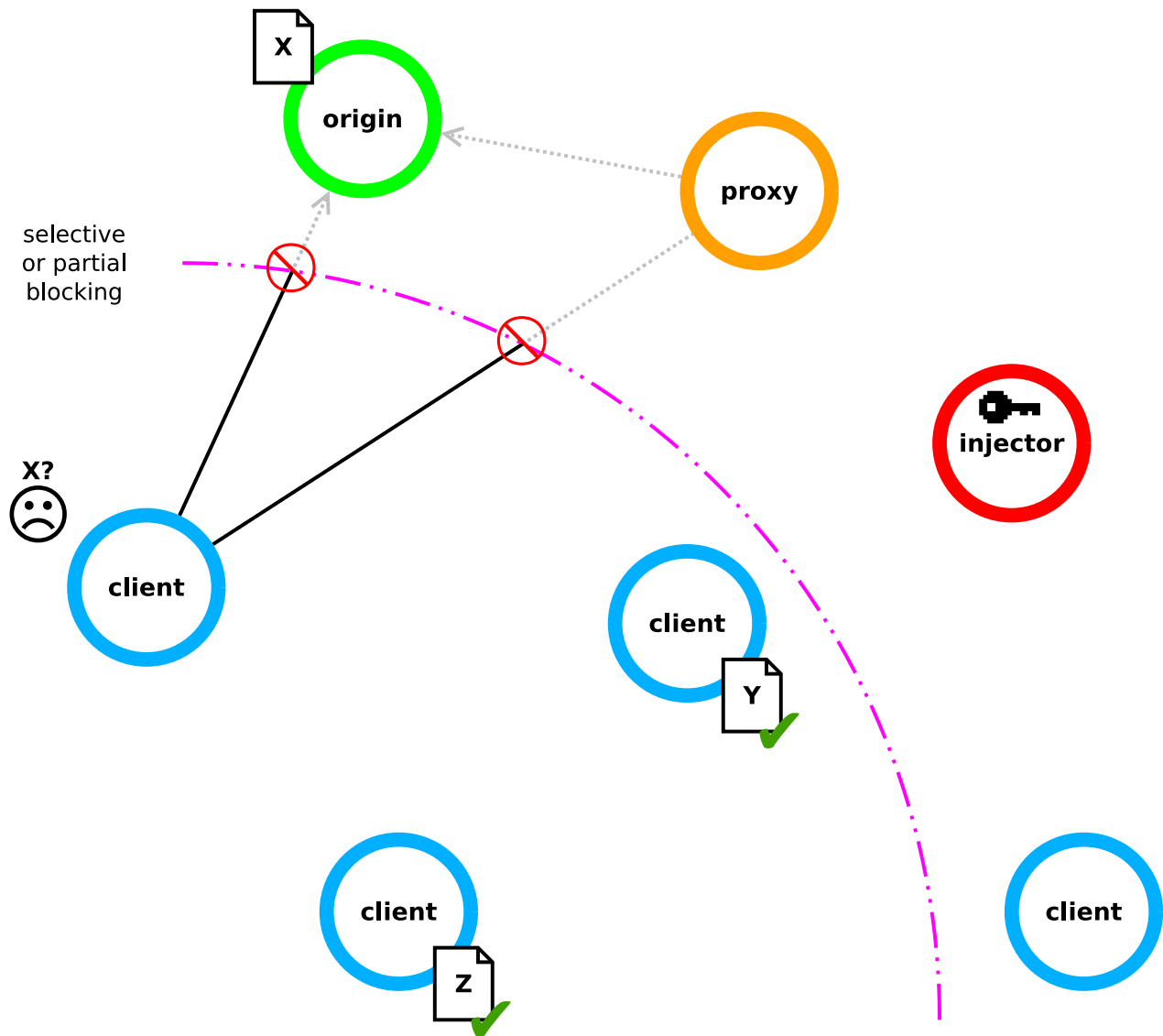
The CENO Browser is an example of an application that uses Ouinet technology to retrieve and share Web content. We call such an application a Ouinet **client**. When you use your client (i.e. CENO) to try to access some content *X*, hosted on a Web server (which we will call *X*'s **origin** server), your client tries to contact the origin server over the Internet either directly or via some other machine configured to contact Web servers on behalf of others (a so-called **proxy** server) and then requests the desired content. This is no different from the way in which any normal Web browser works.

---

**Technical note:** There is in fact one small gotcha. Since the client acts as an HTTP proxy running on your device, for the client to be able to decrypt and act upon HTTPS content requests, the application using the client (i.e. the Web browser part – like Firefox in CENO) needs to accept a special certificate issued by the client itself (and only used on your device). The CENO Browser already takes care of setting this certificate up for its private use so that you do not need to worry.

---

However, these direct paths may not be available. For instance, your Internet service provider (ISP) may be blocking access to *X*'s origin server or the proxy because of a state order (even if other traffic is still allowed). As the user of the top left client depicted below, both attempts to reach content *X* (the little document close to its origin server) would fail for you. You may also note the "injector" node on the diagram. We will explain that in a moment.



With a normal browser you would be out of luck. However, with Ouinet you can ask other clients for their copies of content *X*, should they already have a copy. Let us see how Ouinet performs this request.

## Searching for shared content

The set of all content stored by Ouinet clients is called the **distributed cache**, i.e. a store which sits in no single place. But how can your client find which other clients forming the cache have the desired content?

In any Web browser, to access content *X* it needs to know its [Uniform Resource Locator](#) (URL), that is the address in the browser's location bar, e.g. `https://example.com/foo/x`. From that URL, a normal browser would infer that it has to contact the Web server called

example.com using the HTTP protocol (the language used to exchange Web resources) over SSL/TLS (a security layer over TCP, the Internet's rules for programs to talk to each other) and request the resource /foo/x .

Quinet looks for the content in a different way. It uses an index not unlike that of a book: in Quinet's **distributed cache index** you look up the whole URL of the content and get a list of clients holding a copy of it. The index itself is distributed, with clients in charge of announcing what content they have to others. Actually, only a *hint* on each URL is announced, so that someone spying your device's traffic cannot tell which content you have, but someone looking for a particular content can follow the hints towards your client.

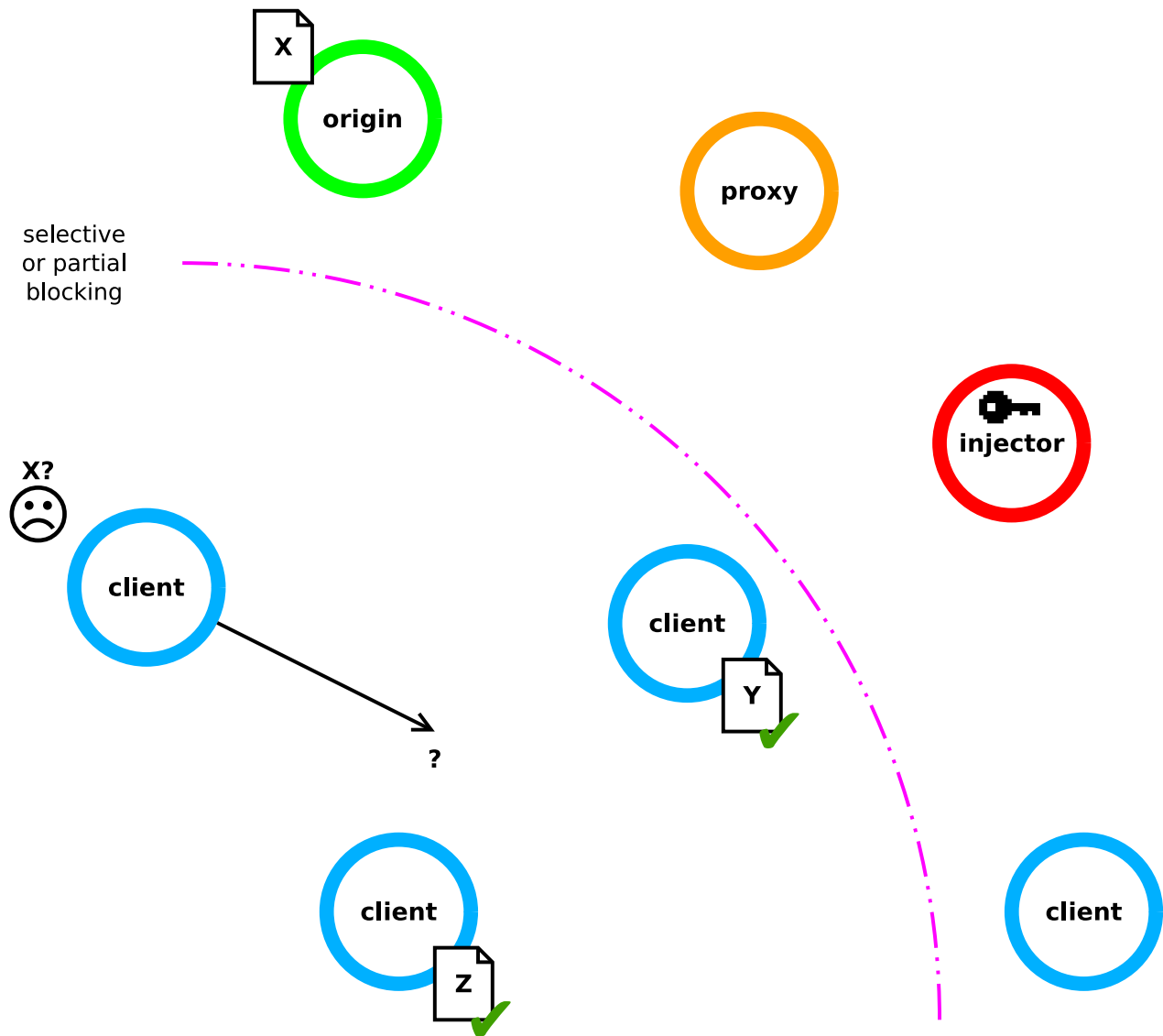
---

**Technical note:** One way the index is implemented is using [BitTorrent's Distributed Hash Table](#) (DHT) to get the addresses (IP and port) of the clients with the content. The DHT uses a [Cryptographic hash function](#) to compute the table key from the content's URL and some other parameters as the injector key (see below), so that several indexes can coexist.

Also, the CENO Browser does not announce the URL of every single resource it holds: with any modern page having tens or hundreds of components (images, style sheets, scripts...), that would create a lot of traffic. Instead, resources are grouped under the URL of the page pulling them, and only that URL is announced. This is done with the help of an *ad hoc* browser extension (described later on).

---

Clients offering some particular content over the distributed cache are said to be **seeding** it or to be their *seeders* (these terms come from the P2P file-sharing world). Going back to our example scenario, there are two clients seeding some content. Unfortunately, one is seeding content Y and the other one content Z, so your client would find no entries for content X in the distributed cache index, as depicted below:



Fortunately, Ouinet offers a way to retrieve such content and furthermore make it available to other clients in the distributed cache. Please read on to learn how.

## Sharing new content

### Proxies on steroids

In Ouinet, there are special kinds of proxy servers called **injectors** which sit in the (hopefully) free side of the Internet and try very hard to stay reachable despite blocking measures:

- First of all, connections between clients and injectors are encrypted (using standard

SSL/TLS like in HTTPS) to avoid attackers from identifying injectors by eavesdropping on web traffic.

By the way, injector certificates are shipped in the CENO Browser, allowing it to detect attackers trying to impersonate injectors.

- If encryption was not sufficient, connections to injectors can use special obfuscation techniques (like I2P and Tor's Pluggable Transports) to make identification even more difficult.
- Even if an injector was identified and access to it was blocked by your ISP, there are several of them and it does not matter which one your client contacts over the Internet.
- Some or all injectors may be blocked, but then the set of injectors can vary over time (with new ones being added).

Your client need not know their Internet addresses in advance; instead, it performs a lookup in the **injector swarm** (another term from P2P file sharing), a single-entry distributed index similar to the distributed cache index which yields the addresses of currently available injectors.

- Finally, even if your client may not be able to reach any injector, some other clients may. When a client is able to reach an injector and believes itself to be reachable by other clients, it becomes a **bridge** client and adds its own Internet address to the **bridge swarm**, another single-entry distributed index.

So your client can look for such an address, connect to the bridge behind it and tell it to establish another connection to an injector on its behalf, creating a **tunnel** between your client and the injector. Then a connection can be established between them inside of the tunnel.

Please note that since client-to-injector connections are encrypted, bridges are not able to see the information flowing between them.

An injector can behave like a normal (though extra available) proxy server, and this is indeed what Ouinet clients (including the CENO Browser) do currently when trying to access content over a proxy. In this case, the injector will neither see the actual information flowing between your client and the origin server (unless it is a plain, unencrypted HTTP connection itself).

But there exist other tools allowing you to reach proxies in stringent network interference conditions so, what is so special about Ouinet injectors?

## Trusting shared content

Well, the point is that an injector does not just retrieve content on behalf of your client, it also allows you to *share that content with others at a later time, even when there is no longer access to the injector or most of the Internet*.

You could of course download a page from your browser and copy the resulting files to other people, which should be fine if you know each other. But what if you received such files from an unknown person? How could you be sure that the content actually came from the website it claims to, that it was retrieved at a certain date or that the information in it was not manipulated?

We want CENO and Ouinet usage to scale and provide as much content to as many people as possible, so we do want you to be able to receive content from people you do not know. To enable you to accept such content, Ouinet uses **content signing**: your client is configured to trust content that is signed using a special key owned by injectors. Whenever a client tells an injector to retrieve some Web content for sharing, the injector gets it from the origin server, uses the key to sign it, and returns the signed content to the client.

---

**Technical note:** In fact, the injector signs individual blocks of data as they come, so even if the connection is cut in the middle while retrieving a big file, the downloaded data can still be shared by the client that received it.

---

Different injectors may have different keys, so you can choose which injectors to trust. Picture it like this: you may trust a document signed by a *notary public* from your country, no matter who gave it to you (national or foreigner), while you would not be required to accept a document signed by a notary from another country. The CENO Browser is already configured to trust a set of injectors run by eQualitie.

---

**Technical note:** Injectors use a public/private key pair to create Ed25519 signatures; public keys are small enough to allow them to be sent along signatures, and encoded as 64 hexadecimal characters or 52 Base32 characters. They may even be exchanged on the phone or written down on a piece of paper.

---

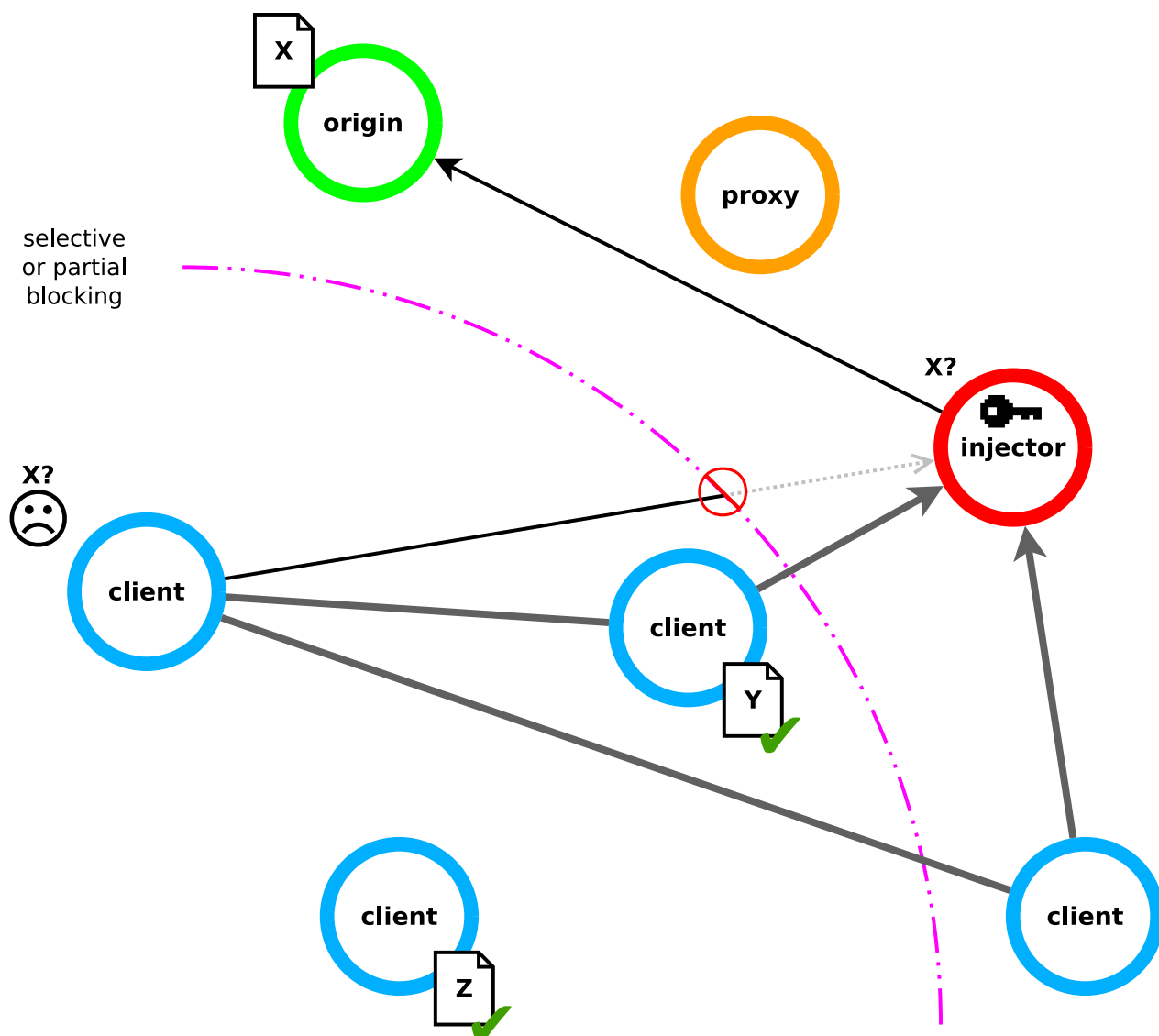
## Content injection

Remember that in our example scenario your client had already tried to retrieve content *X* directly from the origin server and from other clients to no avail. The client plays its last



Quinet card and tries to contact a trusted injector to get a signed copy of the content that it can share with other clients.

In the figure below you can see a possible outcome of that operation: the client first tries to contact the injector directly (e.g. using an Internet address that it got from the injector swarm), but sadly it is already blocked by your ISP; fortunately, the bridge swarm shows the Internet addresses for two other clients which are still able to reach an injector. Your client opens a tunnel to the injector through one of these clients, so the injector gets the request for content *X* from your client, and asks its origin server for it.

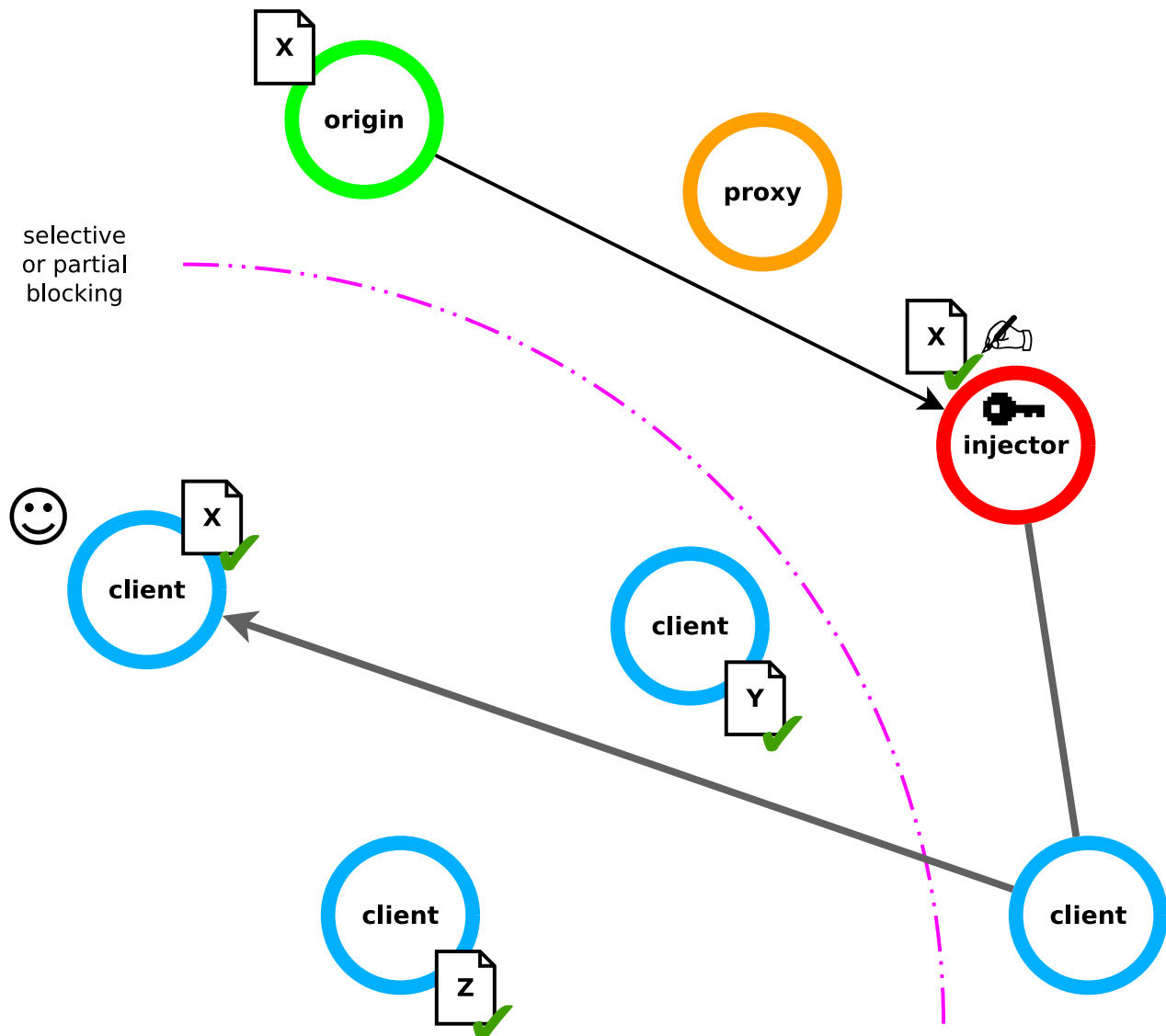


As content *X* is received by the injector, it signs it with its key, adds the signature to the content and sends it back to your client via the tunnel it arrived from (say, through the client sitting beyond the blocking). Once the content reaches your client, it does three things:

1. It delivers it to you (in the case of CENO, it shows the content on the browser).

2. It saves the content on your device for further seeding to other clients. It will stay there for a configurable amount of time, or until you decide to clear all stored content.
3. It announces in the distributed cache index that it is in possession of a copy of that content, so that other clients can find it.

The whole combined operation of retrieval, signing, storage and announcement is what we call **content injection**, as shown in the figure below.



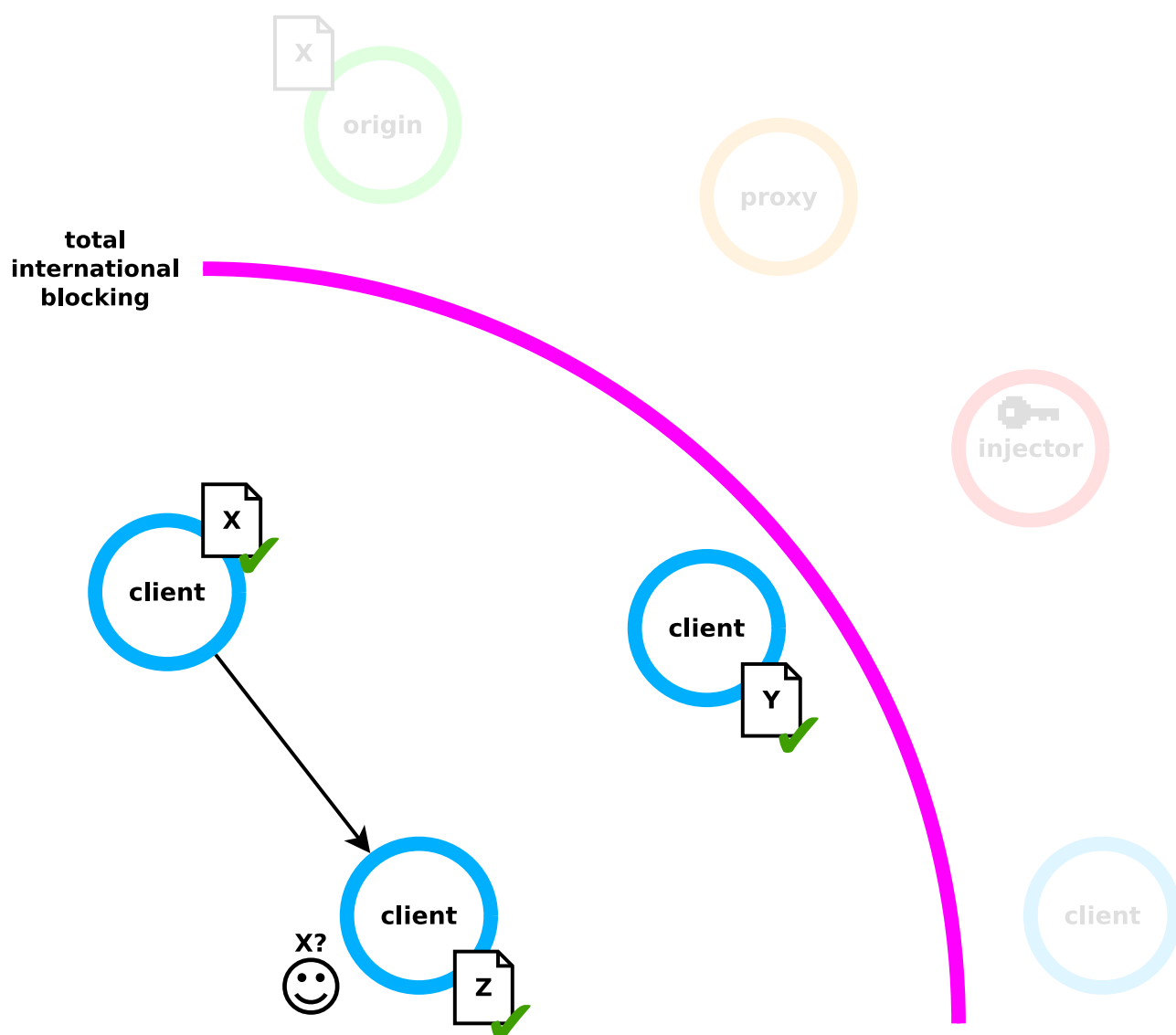
## Browsing under complete blocking

Please note that the mechanism described above still requires that *some path exists* across blocking and towards the rest of the Internet. But sometimes that path will also be missing: think about complete international disconnections, natural disasters, or simply excessive

congestion of the few existing paths (due to everybody trying to go across them). This is where the real power of the distributed cache comes into play.

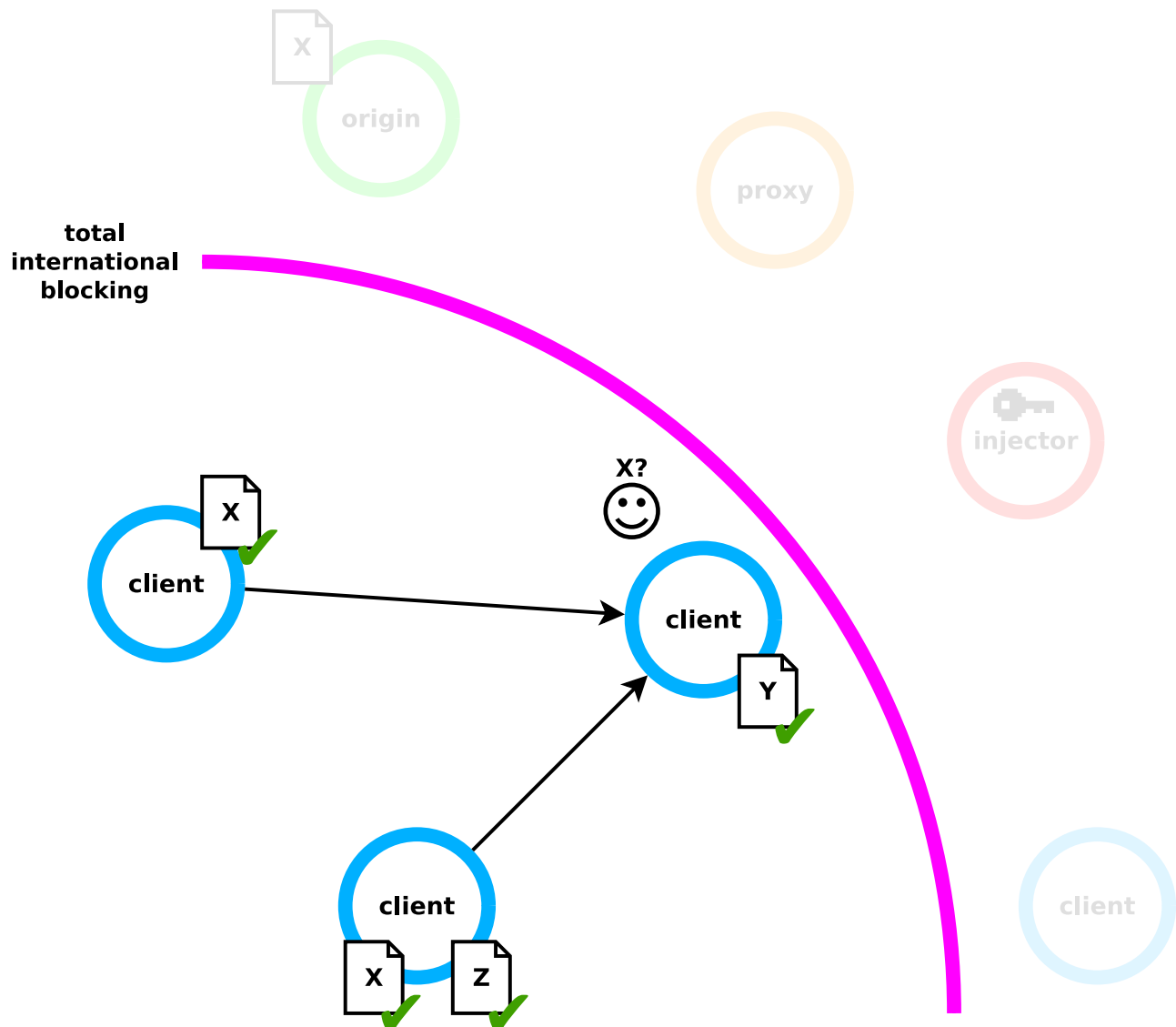
Let us imagine that after you retrieved content X from the injector, a disaster leaves your region isolated from the world. It turns out that content X becomes especially relevant since it describes some ways in which you can help your community in such a situation.

At that moment a second person using the CENO Browser also tries to get that content. Access to the origin server or to anything beyond your region is impossible, so CENO checks the distributed cache index for that content and it finds that your device is seeding it. CENO gets your Internet address from the index, connects to it and requests the content as shown below.



Now that second device also holds a copy of content X, so it announces this in the distributed cache index, thus becoming a seeder. If a third person interested in that content

uses the CENO Browser to retrieve it, CENO will now see *two* addresses in the index for the content: your device's and that of the second user. If the content is heavy (e.g. a video), this third device may try to get half of it from each of the other devices (as shown below), thus speeding up the download and reducing the traffic they use.



Finally, the situation may get even worse, and all commercial and state network infrastructure may be shut down. In this case, Ouinet and the CENO Browser also have some support for *device-to-device* sharing of content between two clients sitting on the same local network (e.g. connected to the same Wi-Fi access point), even if the network has no access to others.

# Public vs. private browsing

Because of the many techniques used to overcome connectivity issues, CENO may become a convenient way for you to get all kinds of Web content. And, as you may have already read in previous sections, whenever you retrieve and seed a page using the CENO Browser, it becomes available to others. There may be some content, however, that you do not wish to share (or you do not want to let others know that you are trying to or did retrieve), and fortunately CENO can help you in this instance as well.

The default mode when you launch the application is **public browsing**. In it, CENO accesses Web content as described previously:

1. Direct access is attempted.
2. Failing that, the distributed cache is searched.
3. Failing that, the content is requested via an injector (maybe via another client).

CENO also has a **private browsing** mode. In it, the distributed cache is never searched, and injection is never attempted:

1. Direct access is attempted.
2. Failing that, an injector is contacted (maybe via another client) and used *as a normal proxy server*. Note that in this case, neither the injector nor your client updates the distributed cache with your page.

The different behavior results in different characteristics. Thus, in public mode:

1. You have better chances to get Web content, and help others get that content (from you).
2. Pages with dynamic content (e.g. updated in real time) may break in obvious or subtle ways.
3. Pages requiring authentication do not work (as passwords and cookies are removed by the client).
4. Some browsing activity may be leaked to other users (see [risks](#)).
5. Some browsing activity may be leaked to injectors (see [risks](#)).
6. You need to trust injectors to retrieve and sign Web content.

While in private mode:

1. You may not be able to access blocked Web content if international connectivity is too scarce; even if you could, other CENO users would not get that content from you.
2. Pages with dynamic content will probably work.
3. Pages requiring authentication may work (when your connection is protected by HTTPS, the injector does not see your passwords).

4. Browsing activity is not leaked to other users.
5. Limited browsing activity is leaked to injectors (with HTTPS, only the origin server name or address).
6. You need not trust injectors (with HTTPS, usual certificate-based security still works).

In conclusion: if you are using CENO to read the news, watch videos, browse Wikipedia and other static resources that are otherwise censored in your network, consider using the default *public browsing* mode. And if you want to login to Twitter or edit your WordPress website, use *private browsing* mode.

Please read the section on [risks](#) for a more detailed explanation. Also note that your client can continue to operate as a bridge and a seeder regardless of public or private browsing. We explain this in greater detail in the [Helping others](#) section of the manual.

# Advantages of using CENO/Quinet

The main advantages of using CENO and Quinet over other circumvention technologies stem from the cooperation of clients and injectors to forward traffic for one another, sign content for later verification, and store signed content for seeding to others. Some advantages worth mentioning are:

- **Familiar usage:** Accessing Web content using the CENO Browser feels quite like surfing the Web on your habitual browser, even during complete blocking. There is no need for new custom links to popular content, or special actions from the user (like transferring files between applications).
- **Increased availability of content:** Quinet is able to provide content in an efficient and trusted manner in situations of extreme network interference. The more popular some particular content gets, the more copies of it get seeded by CENO/Quinet clients, and the more available it becomes.

Signed content may be brought into particular clients in a disconnected area over offline means (e.g. a USB drive) and thus made available to other clients.

- **Faster browsing:** Since your client can retrieve different parts of the same content from various clients at the same time, the load of delivering the content is distributed among different networks and devices, thus avoiding clogging the paths to a single client (especially when delivering a big resource like a video). This is not only useful when infrastructure connecting to other countries is limited, but also for publishers to avoid resource usage spikes at origin servers when some of their content becomes very popular (the so-called [Slashdot effect](#)).
- **Cheaper browsing:** Content popular in a particular region tends to get copied in CENO/Quinet clients in that region, even if the origin server is abroad. If you are interested in that content, your client will probably get it from some other client in your region. In some countries where international traffic is more expensive than local one (e.g. in the presence of a [national intranet](#)), this can actually save you money.

# Risks in using CENO/Ouinet

As with any sufficiently complex computing system, and especially such an innovative one, using the CENO Browser (and any Ouinet client in general) is not free from some risks. In this section we will compile and describe them to help you understand their implications according to the different roles you may play when using CENO:

- as a *user* browsing Web sites
- as a *seeder* sharing content over the distributed cache that you previously visited
- as a *bridge* allowing other users to reach an injector

## As a user

### Can bridges see my communication with the origin server?

No. The only role of a bridge is to forward raw traffic between a client and an injector. This communication is always encrypted and bridges do not have the private keys required to access the content of the communication.

### Can injectors see my communication with the origin server?

Yes and no. When the user requests content [in public browsing mode](#), all private data (like passwords and cookies) is first removed from the request by the client, and only then is the request encrypted for and forwarded to the injector, which proceeds to decrypt it.

On the other hand, when the request uses private browsing mode, it is not modified by the client, but the whole communication is encrypted for the origin server. This means that in this other case the injector cannot decrypt the content.

---

**Technical note:** Only GET HTTP requests are candidates for injection, with query parameters removed, along with all but a limited set of fundamental and privacy-preserving HTTP header fields.

---

### Can injectors see my IP address?



Yes. However, injectors cannot distinguish whether a request came from a CENO user or a bridge. Thus requests going to the injector cannot be reliably assigned an originating IP addresses.

## **Can my private data leak to the distributed cache?**

Hopefully not. As mentioned above, the CENO Browser tries hard to remove any private data (passwords, cookies...) from any request for injection. In addition, the injector does not itself do any seeding; in fact, its sole purpose is to sign content so that Ouinet clients can seed it. This means that when the content comes back to the client, it is further analyzed, and if the origin server indicated that it is of a private nature, CENO will not seed it either.

Still, there could be cases of badly designed or malicious pages which may collect some information from you (like an email address in a form or some browser fingerprints using JavaScript) and stuff it in another link URL as normal path components (e.g.

`http://example.com/subscribe/you@example.org`). If you suspect that a page may be doing that, better be on the safe side and use private browsing for it.

## **Can the origin server know whether I am using CENO?**

Most probably not. Whenever CENO contacts an origin server directly, it behaves as normal Firefox for Android does, so your particular device appears as a normal Firefox app of the same version.

However, when it uses an injector to get some content from its origin server, there are (at least) two ways for the latter to know that CENO or Ouinet is involved:

1. The source address of the connection reaching the origin server is found in the injector swarm (since the connection comes indeed from the injector);
2. The presence or absence of certain information in the request for content is characteristic of Ouinet. This happens when the injector is requesting the content because your client asked it to retrieve and sign that content, as the injector removes information unique to your particular device from the request.

Please note that these only mark the request as coming from Ouinet, but they do not link it to you or your particular device. However, if the request did for some of the reasons mentioned in the previous question still contain some personally identifiable information, it could be used to mark you as a CENO user.

In general, if a particular website (such as a governmental site) expects you to connect to it

as an identifiable individual, from a specific region (or from a [national intranet](#)), we recommend that you use a normal Web browser instead of CENO.

## **As a seeder**

### **What data is seeded from my device?**

Currently, the only content that is seeded by CENO is any non-private Web content which was requested in public browsing mode. This also means that users do not seed anything they have not accessed themselves in the recent past.

### **Can anyone find out what I seed?**

Yes and no. Anyone with enough understanding of Ouinet's operations could construct a tool to find out what IP addresses a particular content is being shared from (as with BitTorrent). However, it is not possible to target a specific IP address and get a list of all the content seeded by clients behind it.

## **As a bridge**

### **Can others find my IP address?**

Yes, every CENO Browser able to communicate with injectors will register its IP address in the bridge swarm where other Ouinet clients can find them.

### **Is it possible that I am helping someone to access content which is illegal in my country?**

Yes. However, bridges only relay encrypted communication between a Ouinet client and an injector. This means that a bridge shall never make direct requests for content to any other server on someone else's behalf.

# Using the CENO Browser

This chapter gives you some hints on the usage of CENO Browser as far as its Ouinet-related capabilities are concerned. Please always keep in mind that CENO is based on [Firefox for Android](#), so for all questions on generic browsing topics you should refer to Mozilla's [Firefox for Android Support](#) pages.

When relevant, screen captures will be included to illustrate the text. Please note that these may slightly differ from what you see on your device, especially as CENO development progresses. This documentation is up to date for CENO version 1.0.2.

If your application exhibits behaviors substantially different from those described here, do not hesitate to contact us at [cenoers@equalit.ie](mailto:cenoers@equalit.ie) and report the issue.

# Installing CENO

The CENO Browser can be installed via the following means:

- [Google Play](#) (*CENO Browser* from *eQualitie*): the recommended source for most Android users.
- [GitHub](#): for Android devices without Google Play.
- [Paskoocheh](#): for users in countries blocking access to the previous channels.

CENO requires *no special permissions* to run.

---

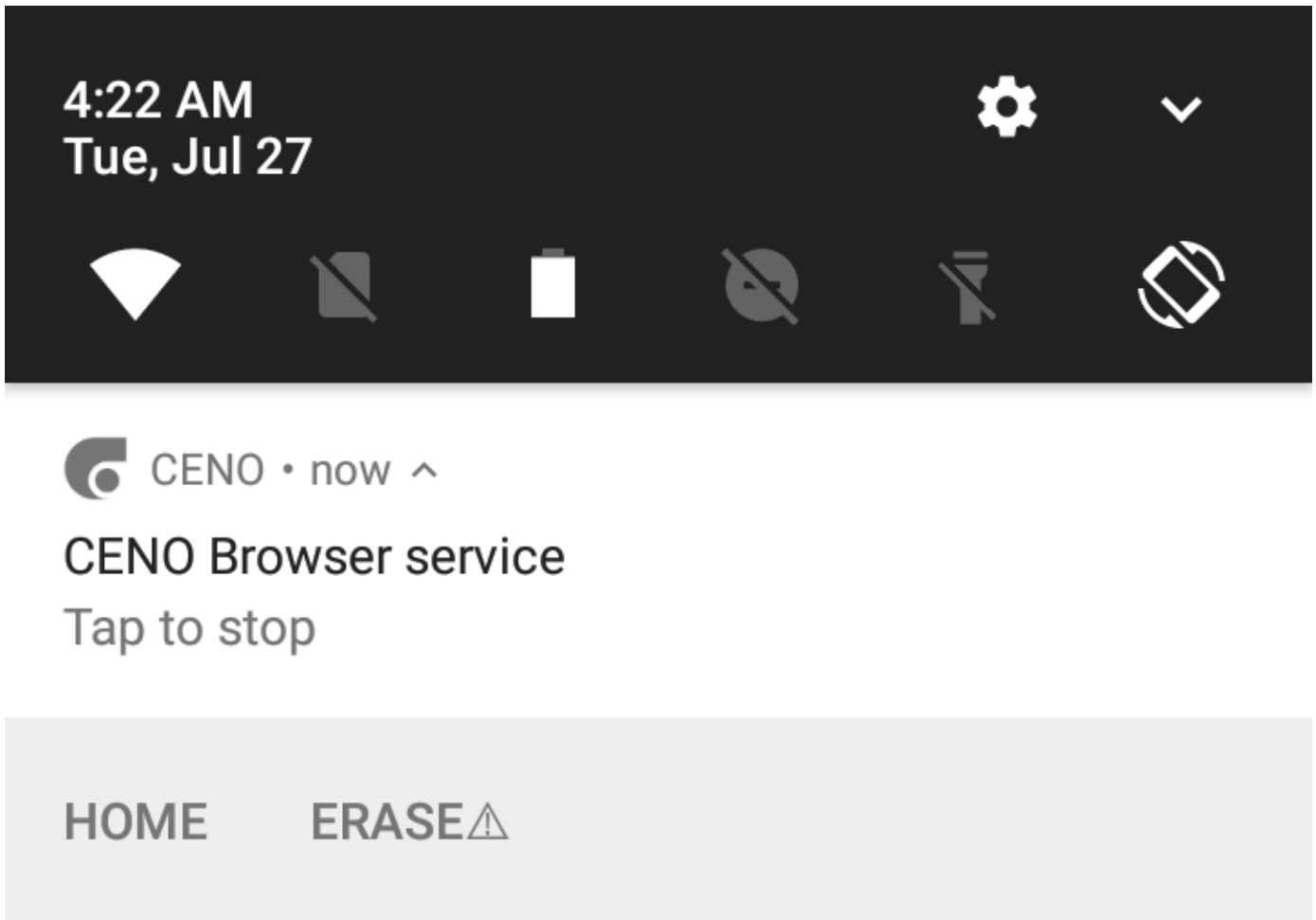
**Warning:** Please be *extremely skeptical* about installing the CENO Browser from sources other than the ones listed above. Because of the application's nature, their potential users may become a target for all kinds of fake or manipulated versions used to violate user privacy or attack other CENO and Ouinet users. If in doubt, please contact [cenoers@equalit.ie](mailto:cenoers@equalit.ie) before installing a suspicious app.

---

## Stopping CENO completely

Every time you start the app, a CENO icon will appear on your device's notification bar. This icon represents the *CENO Browser service*, which is the part of CENO that runs permanently (even when you are not browsing) and allows other clients to use your device as a bridge and retrieve content from it at any time.

Since running such service uses network and processor resources, you may want to stop it whenever you are on the move (i.e. not connected to Wi-Fi or far from a charger). Tapping on the notification attached to the icon will stop both CENO and its service at once (until you open CENO again).



## Purging all CENO data (the "panic button")

The *CENO Browser service* notification shown above includes a few accompanying actions which can be triggered by tapping on them. The *Home* action will just open CENO with a new public browsing tab showing its home page. The *Erase* action demands more explanation.

---

**Note:** If the actions below the notification are not visible, drag the notification from its center towards the bottom to unfold it.

---

Shall you ever need to quickly stop CENO and clear absolutely all data related to it (not only cached content, but also settings like favorites, passwords and all browsing history), you can tap on *Erase*. To avoid losing your data accidentally, this will not remove anything yet, but just show an additional action for a brief moment, as pictured below:

4:21 AM  
Tue, Jul 27



CENO • now ^

CENO Browser service

Tap to stop

HOME

ERASE ⚠

YES

If you tap on the *Yes* action, CENO will be stopped and all its data removed *without further questions*, effectively leaving your device as if CENO had never been used.

If you do not tap on the action, it will go away in a few seconds.

---

**Note:** The method described above requires that CENO be running on your device. To accomplish the same effect when CENO is stopped, you can use Android's general *Settings* page and, under the *Apps* entry, choose CENO and then *Clear data*.

As a harsher alternative, you may completely uninstall the app.

---

**Warning:** Android may still keep other traces of having used an app besides its data, for instance in its system log.

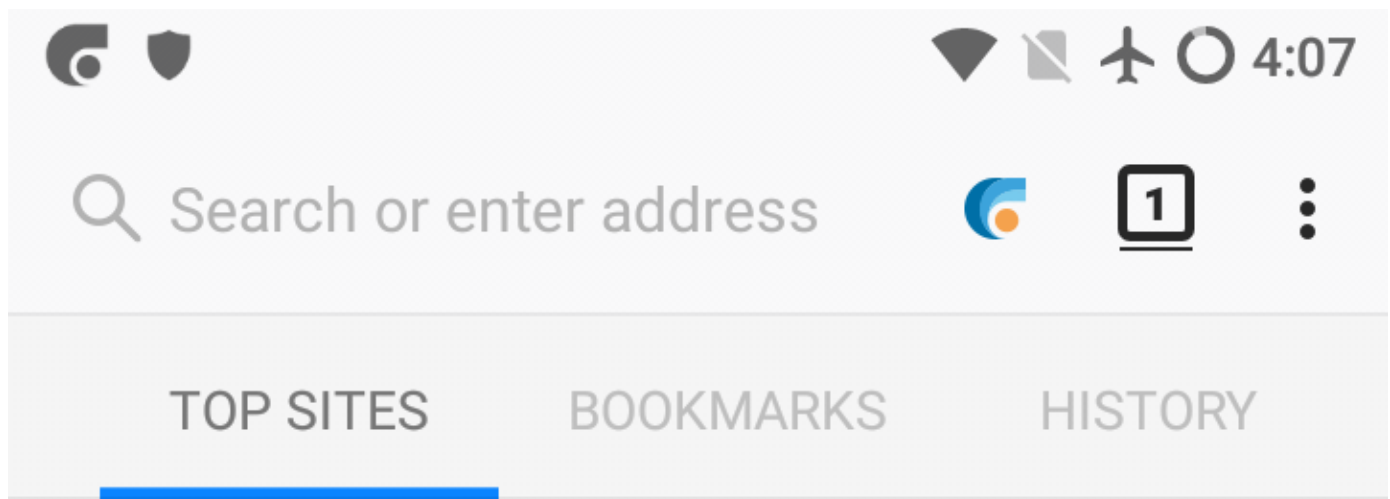
---

# Using public or private browsing

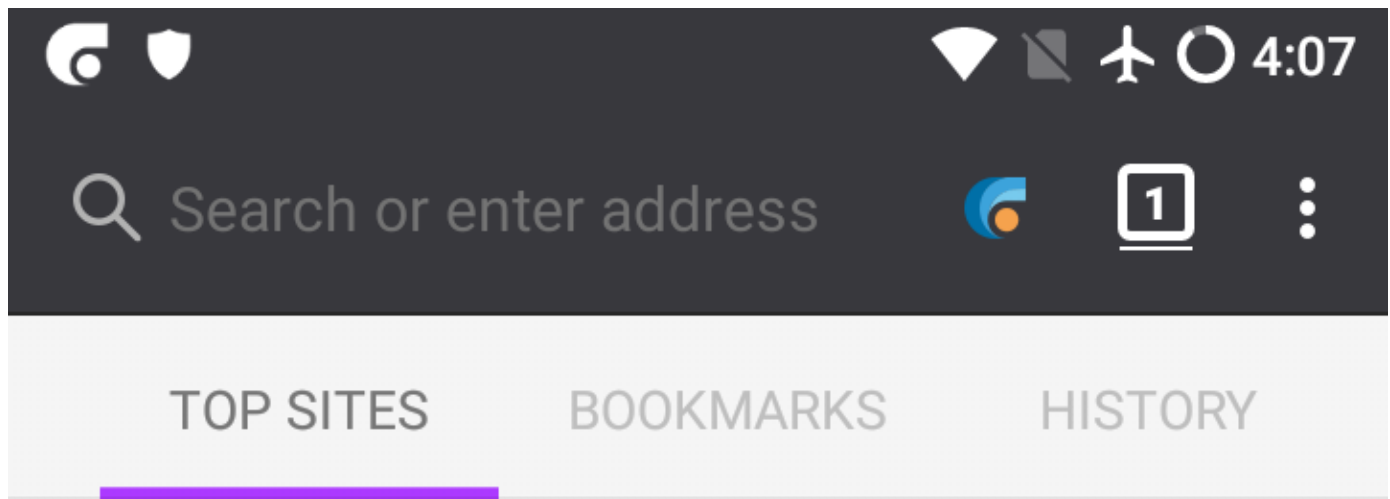
As described [in a previous section](#), CENO has two different modes of operation depending on whether you want to share the content that you browse with others (public browsing) or not (private browsing).

This setting applies to *each tab* that you open in the browser, i.e. you can have public browsing tabs and private browsing tabs. CENO's default whenever you start it or open a new tab (using *New tab* in the app's main menu) is to use public browsing. To open a new tab in private browsing mode, just choose *New private tab* in the main menu.

You can tell public tabs from private ones because public tabs have a lighter (or white) tool bar:



In contrast, private tabs have a darker tool bar:



Once you have loaded a page in a tab, the colored CENO icon in the address bar will help you know how it actually retrieved the different elements of the content. We will cover this

icon later on.



# CENO settings

CENO Browser allows you to change some Ouinet-specific settings and get information about your client in a simple manner. This should not be needed for normal operation, but it would be helpful for testing different strategies against network interference, as well as reporting issues with the app.


---

**Technical note:** These options are provided by the *CENO Extension*, a Firefox extension which comes installed out of the box with CENO and takes care of proper integration with Ouinet, like enabling content injection and cache retrieval under public browsing, hinting the user about the source of the content being visualized, and notifying about new versions of Ouinet.

---

These features are available on a page that can be accessed by choosing *CENO* in the app's main menu. Please note that the menu entry may take a few seconds to pop up on app start. The page should look like this:

# CENO Settings

 Settings on this page are only intended for testing CENO functionality in various network conditions. These are **not saved** when stopping CENO. See the [manual](#) for more information.

## Content retrieval sources:

- ☒ Direct from website
- ☒ Via the CENO network (private)
- ☒ Via the CENO network (public)
- ☒ Shared by other CENO users

---

### CENO Browser

1.0.2 Build ID 20210726080755

### CENO Extension

1.2.7

### Quinet

0.11.3 Release HEAD

808123cbd04ad80b5245b8e7731452fddeb7a7e5

192.168.1.132:28729

### UPnP status

enabled

### Reachability status

likely reachable

### Local cache size

0 B [Purge now](#)

[Content shared by you](#)

---

☐ Enable log file

## Choosing access mechanisms

The four checkboxes on the top of the page selectively enable or disable the different mechanisms or *sources* that CENO as a Ouinet client uses to retrieve content while using either [public](#) or [private browsing](#) tabs. All boxes are enabled by default.

- *Direct from website* (or **origin access**) allows CENO to try to reach the origin server directly before attempting other mechanisms enabled below.

Although this mechanism works in both private and public browsing modes, content thus retrieved cannot be shared with others.

If getting most Web content is not particularly slow or expensive, this mechanism may be more than enough for most use cases. However, such direct connections may be tracked by your ISP or government. To some extent, disabling this option may avoid such connections and trivial tracking (but not completely, see [risks](#)).

Also, when accessing a Web site over insecure HTTP (instead of the more secure

HTTPS), a censor may intercept the connection and supply the user with a bogus site, a tampering which CENO cannot detect by itself. In such cases, it may help to disable this option and thus always resort to other, safer CENO mechanisms. Please check the section on [troubleshooting](#) to learn more about this issue.

- *Via the CENO network (private)* (or **proxy access**) allows CENO to use injectors as normal HTTP proxy servers to reach origin servers.

This mechanism only works in private browsing mode.

When accessing content over HTTPS, only origin servers will be able to decrypt traffic. When using plain HTTP, the injector may also see the unencrypted traffic (but it should still not sign or share its content with others). Other participants, such as bridges, will never see the unencrypted traffic.

- *Via the CENO network (public)* (or **injector access**) enables CENO to strip any private information from requests and send them to an injector. The injector gets the content from an origin server, signs it, and sends it back to CENO - which then begins seeding it.

Other participants (such as bridges) will not see the unencrypted traffic.

This mechanism only works in public browsing mode.

- *Shared by other CENO users* allows CENO to try to retrieve content from the **distributed cache**, i.e. from other CENO and Ouinet clients seeding it.

This mechanism only works in public browsing mode.

Disabling all of the mechanisms available for either public or private browsing mode will render them useless. If you establish such a configuration, a warning will be shown as depicted below:

## Content retrieval sources:

- ☐ Direct from website
- ☐ Via the CENO network (private)
- ☒ Via the CENO network (public)
- ☒ Shared by other CENO users

 **Private browsing** will fail with current settings.

---

**Warning:** Please note that CENO does not remember these settings upon restarting the app. If you require some of the previous mechanisms to be off while using CENO, please remember to open the Settings page whenever you start the app and uncheck their boxes before browsing. We apologize for the inconvenience.

---

## About your app

This page also provides you with some information about your CENO Browser app and Ouinet client:

- *CENO Browser* indicates the exact version of CENO that you are using. Please include this information in your issue reports.
- *CENO Extension* shows the version of the extension that integrates Firefox with CENO. Also include in reports.
- *Ouinet* shows the version of Ouinet backing CENO. Also include in reports.
- *Ouinet protocol* is the version number of the protocol that CENO uses to talk to other Ouinet clients and injectors. Also include in reports.
- *Local UDP endpoints* are the Internet addresses used by CENO to seed signed content to other clients. These are shown to help test and debug the app, and should not be generally disclosed.
- *UPnP status* indicates whether CENO was able to tell your router or access point to

allow incoming connections towards it. Also include in reports.

- *Reachability status* indicates how likely it is for your device to be able to effectively seed content to other clients. Also include in reports.
- *Local cache size* shows an approximation of how much storage is taken by the content being seeded from your device's local cache.
- *Content shared by you* allows you to check the content being announced by your device.

## Purging the local cache

Next to the *Local cache size* value above, there is a button which allows you to stop seeding and drop all content shared by your device over Ouinet. This allows you to free up some storage space in your device while keeping other CENO settings like Favorites.

If you want to clear CENO's normal browsing cache (the one used by the browser but not shared with others) or other items like cookies, browsing history or favorites, you should choose *Settings* in the app's main menu, then *Clear private data*. You will be asked about which items you want to clear.

To drop everything at the same time (especially if you are in a hurry), please learn how to use the "panic button" feature described in [Installing CENO](#).

## Collecting log messages

At the bottom of the page there is an *Enable log file* check box that allows you to collect all of Ouinet's internal messages and download them to a file. This should only be used when diagnosing some problem in CENO; just follow these steps:

1. At the *CENO Settings* page, check *Enable log file*.
2. Go back to browsing and do whatever actions that trigger the troublesome behavior.
3. Return to the *CENO Settings* page and click on the *Download* link next to the *Enable log file* check box. Save the file for later use. Android may ask you at this point whether to allow CENO access to stored media: this is needed to be able to save the file.
4. Uncheck *Enable log file* to avoid the logs from growing too large.

You can now use the saved log file to document an issue report, but try to avoid making it public since it may contain sensitive information about your browsing.

# Testing the Browser

Now that you know how to [install](#) and [configure](#) CENO, let us follow some steps to test whether different Ouinet-specific features work. This will involve selectively enabling and disabling different access mechanisms. Keep in mind however that in day-to-day usage of CENO, you will seldom need to change the default settings at all.

To complete all the tests below you will need at least two devices connected to the same Wi-Fi network, and a third one on a completely different network.

All of the tests will be performed using [public browsing](#). If something does not work as expected, please be patient and check the section on [troubleshooting](#). Specifically, if the browser gets stuck for more than a couple of minutes while loading a page, you may hit stop and reload it (although this may alter results slightly).

## Accessing an injector

Let us first check whether your CENO Browser can reach an injector. This may look trivial, but your client will already be exercising several Ouinet features in the process: looking up the injector address in the injector swarm, trying to contact it directly and, if it is blocked by your access provider or country, looking up the bridge swarm and trying to contact the injector via some other Ouinet client.

In the first device perform the following steps:

1. First of all, install CENO and start it. Its home page will appear.
2. Open the app's main menu and choose *CENO* to open the *CENO Settings* page. Since we only want to test injector access, uncheck all the boxes for content sources except *Via the CENO network (public)*.
3. Go back to CENO's home page. Either select one of the recommended Web sites, or enter the URL of some other site at the address bar at the top of the window. If you know about a site which is usually blocked for you, go ahead and enter it!
4. The chosen site should eventually show up.

**Note:** If you get a fake page instead of a legitimate one, use the address bar to enter the site's URL with `https://` at the beginning (instead of `http://`) and repeat the test.

If the site does load, you can be happy that your device can reach the injector! Since you are able to query swarms and contact other clients, you are also likely to be able to retrieve content from the distributed cache.

By the way, if you push the CENO icon in the address bar, it will open a popup like the one below, showing how many elements from the site were retrieved from the different sources. Only *Via CENO network (public)* should have a non-zero value since the other sources were disabled.

## CENO Sources

### Number of page elements per source:

Direct from website	0
Via the CENO network (private)	0
Via the CENO network (public)	6
Shared by other CENO users	0
Shared by you	0

See the [manual](#) for more information on content sources.

In contrast, if you had used private browsing with default settings, you may have seen a popup like the one below, with non-zero counts in *Direct from website* or *Via CENO network (private)*.



# CENO Sources

## Number of page elements per source:

Direct from website	0
Via the CENO network (private)	6
Via the CENO network (public)	0
Shared by other CENO users	0
Shared by you	0

See the [manual](#) for more information on content sources.

## Getting content from close users

Since your first device was able to get some content from an injector, let us test if it is able to share it with another device over the distributed cache. The simplest way is to use CENO's device-to-device support to check whether getting and verifying signed content works.

After successfully completing the test above on the first device, leave CENO running on it (the CENO icon should appear in its notification bar). Then get hold of a second device (you can invite a friend over to help test) and connect it to the same Wi-Fi network. Next, follow the steps below on the second device:

1. Install CENO if needed and start it as above.
2. Open the *CENO Settings* page as above. Since we only want to test distributed cache access, uncheck all the boxes for content sources except *Shared by other CENO users*.
3. Go back to CENO's home page and visit the same site in the same manner as you did above (i.e. by selecting one of the recommended web sites or entering its URL in the address bar).

4. The chosen site should eventually show up.

If it does not work, your Wi-Fi network may be blocking direct communication between its devices. This "client isolation" may happen in public networks like those of parks, bars or hotels. Please try again on a different network.

If it works, it means that both devices are able to deliver that content to other clients. Pushing the CENO address bar icon should show a popup like the one below, where only *Shared by other CENO users* has a non-zero value.

## CENO Sources

### Number of page elements per source:

Direct from website	0
Via the CENO network (private)	0
Via the CENO network (public)	0
Shared by other CENO users	6
Shared by you	0

See the [manual](#) for more information on content sources.

Finally, you may have also noticed that there is a counter for *Shared by you*. This is not a different source *per se*: elements counted here are indeed part of the distributed cache, but they happen to already be stored in your device, so CENO does not need to retrieve them over the network.

## Getting content from remote users

We have done a small-scale test of the distributed cache. Let us now test how it works over

the Internet.

After successfully completing the test above, leave CENO running on the first device as with the previous test, and stop CENO on the second device (by showing its notifications and pushing "Tap to stop").

This time you will need a third device, but it must be connected to a different Wi-Fi network (maybe another friend can help from home). The steps to follow for that device are exactly the same as those in the previous test.

If the third device can load the site used for the test, you are all set. The first device is able to seed content to others, and it can most probably also act as a bridge.

Congratulations!

# Helping other CENO users browse the Web

A peer-to-peer network is built from every node connected to it (yes, that means you as well!). The more nodes, the stronger and more versatile the network becomes. If you are running the CENO Browser from a country that does not censor the Internet (or not as heavily as some), consider helping other CENO users by becoming a **bridge** node. You will then begin to route traffic between clients living in heavily censored countries and CENO injectors. You will not be able to see their traffic (it will be sent through an encrypted tunnel), nor will any of this traffic remain on your device.

---

**Note:** The configuration described in this section may also help your device to effectively seed content to others on the distributed cache, so please consider applying it as well when using CENO in a censoring country (but keep in mind the [risks](#) of serving such content to others).

---

## How to become a CENO bridge

This functionality is already built into the CENO Browser. Your device will need to be connected to a Wi-Fi network that has either UPnP enabled or explicit port forwarding configured for CENO. See the next sections for further details.

However, please note that Android will only allow a mobile device to act as a proper bridge while you are actively using it, as power-saving features will throttle the operation of CENO otherwise.

---

**Technical note:** This is mainly due to Android's [Doze mode](#) slowing down the operation of the native Ouinet library. Unfortunately, disabling battery optimization for CENO does not seem to exclude Ouinet from it. Your particular device may also include its own power-saving features which may interfere with CENO; please check [Don't kill my app!](#) for your device's brand.

---

Thus if you intend to have CENO acting as a permanent, always-reachable bridge, besides a properly configured Wi-Fi network you will need to:

1. Have your device plugged in to power at all times.
2. Have the device's screen on at all times.

One convenient way of doing this without much power consumption and obnoxious, permanent lighting is using Android's screen saver: enable it under *Settings / Display / Screen saver* (or *Daydream* in some versions), pick the *Clock* widget, choose *When to start screen saver* in the menu and select *While charging* or *Either*. A very dimmed down clock will appear on a black background while the device is not active.

Please note that you should not use the power button to lock the device as this will turn the screen off. Instead, just wait for the device to lock itself with the screen on.

If that setup is not an option for you, do not desist yet! If you have a computer with good connectivity that stays on most of the time, please continue reading.

## Running a bridge on a computer

If your computer supports [Docker containers](#), you can run a pre-configured CENO client on it to act as a bridge. If Docker is not yet installed, please follow the instructions to [install the Docker Engine](#) in your platform. For Debian derivatives like Ubuntu or Linux Mint, you can just run: `sudo apt install docker.io`

To deploy a CENO client container you only need to run the following command on a terminal (it looks scary but you can just copy and paste it as is on the command line):

```
sudo docker run --name ceno-client \
  -dv ceno:/var/opt/ouinet --network host \
  --restart unless-stopped equalitie/ceno-client
```

If your computer is not based on GNU/Linux, the command needs to be slightly different:

```
sudo docker run --name ceno-client \
  -dv ceno:/var/opt/ouinet \
  -p 127.0.0.1:8077-8078:8077-8078 -p 28729:28729/udp \
  --restart unless-stopped equalitie/ceno-client
```

The command will start a container named `ceno-client` that will run on every boot unless you explicitly tell it to stop. Please check the [CENO Docker client documentation](#) for more information on how to manipulate the container.

---

**Note:** This client has no *CENO Settings*: when instructed below to access that page, open instead the [client front-end](#), which contains mostly equivalent information.

---

# Enabling UPnP on your Wi-Fi router

UPnP is the easiest way of making your CENO Browser (or computer client) reachable to the CENO network. The [CENO Settings](#) page will indicate the UPnP status on your local network.

---

**Note:** Enabling UPnP on the Wi-Fi router may expose devices on your network to external interference. Please make yourself [aware of the risks](#) and also consider using alternative methods as explained below.

---

A status like the one below indicates that UPnP is not enabled on your WiFi router:

---

## UPnP status

inactive

## Reachability status

undecided

---

The status below indicates that UPnP is likely working and CENO is currently verifying connectivity:

---

## UPnP status

enabled

## Reachability status

undecided

---

The status below indicates that UPnP is working and you can bridge connections for other CENO users:

---

## UPnP status

enabled

## Reachability status

likely reachable / reachable

---

There are many Wi-Fi routers on the market and each has their own particular features. Herein a list of some manufacturers' instructions for enabling UPnP:

- [Linksys](#)
- [D-Link](#)
- [Huawei](#)
- [Xfinity](#)
- [TP-Link](#)

## Using port forwarding as an alternative to UPnP

Instead of enabling UPnP on your router, you can create a port forwarding rule to make sure that connections from the CENO network are forwarded to your device. You will need to login to the router's administrative interface and locate the *port forwarding* option. To see which IP address you need to forward the connections to and the relevant port, open the *CENO Settings* page and look under the *Local UDP endpoints*.

---

### Local UDP endpoints

192.168.1.132:28729

---

The port forwarding must be for the UDP protocol (not TCP). CENO chooses a random port on first run and keeps it for subsequent runs, but your device's local network IP address may change from time to time. Thus you should periodically review the *CENO Settings* page to see that your device is reachable to the CENO network.

---

**Technical note:** Alternatively, you can make sure that the router always assigns the same IP address to your device (e.g. via a static DHCP lease for the device's MAC address).

---

# Troubleshooting

This section will give you some hints about what to do when different, known problems arise with CENO and Ouinet. Please bear in mind that these are experimental projects, and that their operation is subject to a multitude of factors beyond our control, like the particular configuration and status of network infrastructure, as well as which content other users have retrieved and the characteristics of their connections.

If problems still persist, please report them to [cenoers@equalit.ie](mailto:cenoers@equalit.ie). We will try to help you with them.

## There is no CENO entry in the app menu

The CENO Extension may still be loading. Please be patient.

## All widgets are grayed-out in the *CENO Settings* page

The CENO Extension has loaded, but it has not yet been able to retrieve status from Ouinet. Since it may take a moment for Ouinet to be ready, please be patient.

If the Settings page stays like that after more than two minutes, Ouinet may have encountered some issue while starting.

Try to visit some page, if possible one which is usually available. If you get an error like "Failed to retrieve the resource (after attempting all configured mechanisms)", CENO may be experiencing some issues with general connectivity (like being unable to join the BitTorrent network). If you are on a mobile connection, try again with Wi-Fi.

If you get an error like "The proxy server is refusing connections" when visiting the page, try to stop other applications which may be offering some service to the device, then restart CENO.

---

**Technical note:** This may happen if another application is already listening on TCP ports 127.0.0.1:8077 or 127.0.0.1:8078 .

---



## A page shows bogus content (like a block message)

CENO was able to contact a web server directly and retrieve content from it. Unfortunately, someone intercepted the connection and directed it to an illegitimate server.

This usually means that the site is blocked by your access provider or country. However, this particular error can only happen when you are accessing the site over insecure HTTP (instead of the more secure HTTPS protocol), as CENO cannot detect the tampering by itself.

Thus, one way to get to the legitimate site is to try to access it over HTTPS by tapping the address bar and replacing `http://` with `https://` at the beginning of its URL. Of course, this will only work if the site supports HTTPS. If it does not, and you get "Failed to retrieve the resource" from CENO, you will need to go to the [Settings page](#), disable Origin access, and try again.

Since such setting is not remembered, and modifying the URL every time can get tiresome, you may instead configure CENO to always use HTTPS for all sites via the embedded [HTTPS Everywhere](#) extension: choose *HTTPS Everywhere* in the app's main menu, then enable *Encrypt All Sites Eligible* (EASE) as shown below. This setting is remembered by CENO.



Version: 2021.4.15

Rulesets version for EFF (Full):  
2021.7.28

Rulesets version for DuckDuckGo  
Smarter Encryption: 2021.7.28

**HTTPS Everywhere is ON**



**Encrypt All Sites Eligible is ON**



Unencrypted requests are currently blocked

## Settings for this site

Change your preferences for encrypted connections

**Disable on this site**

If you still need to access some particular site over plain HTTP when EASE is on, you can visit the site (even if access fails), then from that tab open HTTPS Everywhere settings and tap on the *Disable on this site* button shown above to set an exception for the site.

## Accessing some content shows "Failed to retrieve the resource"

This means that CENO tried all available mechanisms to access the content, but none of them succeeded.

You should make sure that the following requirements are fulfilled for CENO to work:

- You are running a recent version of the CENO Browser. Obsolete versions may not be

able to communicate with newer injectors or other clients. Check the [installation instructions](#) to know where to get new versions.

- All access mechanisms in the [Settings page](#) are enabled. Otherwise CENO will not be able to circumvent some connectivity issues when accessing content.
- Your device has a working connection to the network, i.e. your normal Web browser is able to open some Web sites. CENO and Ouinet cannot work when all network connectivity is shut down (although users may still find a common Wi-Fi access point to do device-to-device sharing).

If that is the case, it is worth explaining what may be happening for all access mechanisms to fail, to give you an idea of your chances for getting content using CENO.

## Origin access

Your CENO Browser cannot directly reach the content's origin server. Either the server is having some difficulties itself (e.g. it is down or under some attack), or someone is interfering with your connection to it.

This is the main use case for CENO and the other mechanisms should compensate for it.

## Proxy/Injector access

Someone is interfering with your connection to Internet addresses in the injector swarm. Since this is expected to happen eventually as CENO (and Ouinet) gain traction, CENO resorts to reaching injectors via other clients acting as bridges.

There are different reasons for CENO being unable to reach such clients:

- No other client is reachable by you. If only a few Ouinet clients are online, it is entirely possible that none of them are in a network that can be reached from the outside. This should become less likely as CENO and Ouinet get popular and more clients with diverse connectivity come online.

The extreme case here is that connections between Ouinet clients are detected as such by a censor and blocked. This is quite unlikely (since it may entail blocking all BitTorrent traffic) and currently beyond what Ouinet supports, but we plan to make it more resistant against these attacks.

- No other client can reach an injector. Since it is unlikely that all injectors are simultaneously down, that may mean that you can only reach Ouinet clients which are themselves affected by similar network interference as yours.

This can happen when all traffic leaving or entering the country you are in is interrupted. CENO would resort in this case to getting and using the content from the distributed cache, even if it is stale.

Again, as CENO and Ouinet become more popular, the chances increase that there are at least a few clients that do have some kind of access to international sites. Even if access is precarious, a single Ouinet client able to inject some content into the country could suffice to enable it to spread (over the distributed cache) without further need to access the outside world.

Finally, we do run some Ouinet clients with good connectivity in (hopefully) censorship-free countries to try to avoid those two situations, but please note that clients with such stable Internet addresses could get blocked as well.

## **Distributed cache**

Keep in mind that an absolute requirement to be able to retrieve any content from the distributed cache is that it has already been injected by some other CENO or Ouinet user. This means that popular content is more likely to eventually get injected and replicated in a natural manner, while more obscure content is less so, unless someone takes care of using CENO or some other Ouinet client to inject and keep seeding it (which may further expose them to some [risks](#)).

Please note that some content which is not considered safe for sharing will never be injected, no matter how many people retrieve it using public browsing. This includes content marked as private by the origin server, content that requires authentication, and some traffic exchanged by certain dynamic Web applications.

Also note that the clients holding copies of injected content need to be reachable by you. The same observations described in the previous point for reaching bridge clients do apply here.

## **Others cannot retrieve content seeded by my device**

First, make sure that your device is still seeding the content by going to the [CENO Settings page](#), only leaving the *Shared by other CENO users* source box checked, then accessing that content again: it should load (at least partially), and pushing the CENO address bar icon should only show non-zero values under *Shared by other CENO users* or *Shared by you*.

If the content does not load, it could be that CENO has already removed it, since it

automatically cleans up stale content (older than a week by default) from your local cache. Enable the *Via the CENO network (public)* source in the Settings page and access the content again. Please allow a couple of minutes to pass for the device to announce the content in the distributed cache index. Make *Shared by other CENO users* the only checked box again and access the content once more; if it still does not load, it may be that the particular content is not deemed safe for sharing by Ouinet.

If the previous step works, but another device with only the *Shared by other CENO users* source enabled still shows "Failed to retrieve the resource...", there are two possible scenarios. If both devices are in the same network (e.g. on the same Wi-Fi access point), it could be that the network does not allow direct communication between devices connected to it. This happens in some public Wi-Fi networks, so try using a private one.

If the devices are in different networks, it could be due to a variety of reasons. One of them is that the network of the first device does not allow incoming connections: if you open its *CENO Settings* page, under *Reachability status* it should say *reachable* or *likely reachable*. Otherwise seeding may not be possible from that network as it is.

---

**Technical note:** If your device reports *undecided* reachability and you can change the configuration of the access point, you may create a permanent port forwarding rule towards your client. See [here](#) for further instructions.

---

# Annex: The Ouinet client front-end

The Ouinet client (as run by e.g. the CENO Browser) offers a front-end page with some information and actions which may be useful for debugging the client. Many of them are also offered by the CENO Extension via the [CENO Settings page](#), though others are only available here.

The front-end is accessible using any plain Web browser running on the same device (you can use CENO too). Its default address is <http://127.0.0.1:8078/>. If you open it, you will see something like the figure below.



08:02

127.0.0.1:8078



[Install client-specific CA certificate for HTTPS support](#). This certificate will only be used by your Ouinet-enabled applications in this device. Verification of HTTPS content coming from the cache will be performed by injectors or publishers that you have configured your Ouinet client to trust. Verification of HTTPS content coming from the origin will be performed by your Ouinet client using system-accepted Certification Authorities.

Auto refresh: disabled

### Request mechanisms

Origin access: enabled

Proxy access: enabled

Injector proxy: enabled

Distributed Cache: enabled

### Logging

Log level: INFO

Log file: disabled

### Ouinet client

State: started

Version: 0.11.3 Release HEAD 808123cbd04ad80b5245b8e7731452fddeb7a7e5

Protocol: 6

Now: 2021-07-29T12:01:34.002490

### Network

Local UDP endpoints:

- 192.168.1.132:28729

UPnP status: inactive

Reachability status: likely reachable

Injector endpoint:

hen5-ed25519-zh6vlt6dghu6swbhia2i66icminonv53tstxyvi6acu64sc62fng/v6/injectors

Injector pubkey (Base32): zh6ylt6dghu6swhhje2j66icmjnonv53tstxxvj6acu64sc62fnq

Content cached locally if newer than 604800 seconds (i.e. not older than 2021-07-22T12:01:34.006467).

Approximate size of content cached locally: 0.00 MiB

Purge cache now

[See announced groups](#)

Static cache is not enabled.

The items shown in the page include:

- A link to enable the client as a certificate authority (CA) at your browser, since the client needs to intercept HTTPS traffic.

You only need this to use a plain browser for testing the Ouinet client, in which case you will also have to configure its HTTP/HTTPS proxies to `127.0.0.1:8077`, and manually enable the [CENO Extension](#) for injection to work. We very strongly recommend using a *separate, specific browser profile* for this purpose.

Please note that none of this needs to be done for the CENO Browser, since it is already configured like that.

- Buttons to enable or disable the different mechanisms used by the client to access content.
- Selectors to choose different log levels, like the default `INFO` (informational messages, warnings and errors) or `DEBUG` (verbose output useful for reporting errors). The log file can also be enabled and retrieved from here.

When enabling the log file, the log level is automatically set to `DEBUG` (though you may change it again from here). When disabling the log file, the original log level is restored.

- Global client state and version information. Useful when reporting errors.
- Information about client connectivity and injector addressing. The default `bep5`



method looks up Internet addresses in a BitTorrent injector swarm, as explained [here](#).

- The public key used to verify signatures from injectors in the distributed cache.
- Information on your local cache like the maximum content age, approximate size of the cache, a button to purge it completely, and a link to the list of announced cache entries.
- The directory of the external, static cache if enabled (CENO does not currently use this).