CENO and Ouinet: Ignore censorship with P2P-backed Web browsing

Ivan Vilata-i-Balaguer, Peter Jankuliak, Dmitriy Volkov
(eQualitie)

Topics on Internet Censorship and Surveillance (TICS) 2018,
24-28 March 2019

Why CENO/Ouinet?

- Censorship as *resource availability problem*
  (incl. traffic shaping/blocking, distributed denial-of-service)
- The Web we know is sensitive to the *availability of servers*

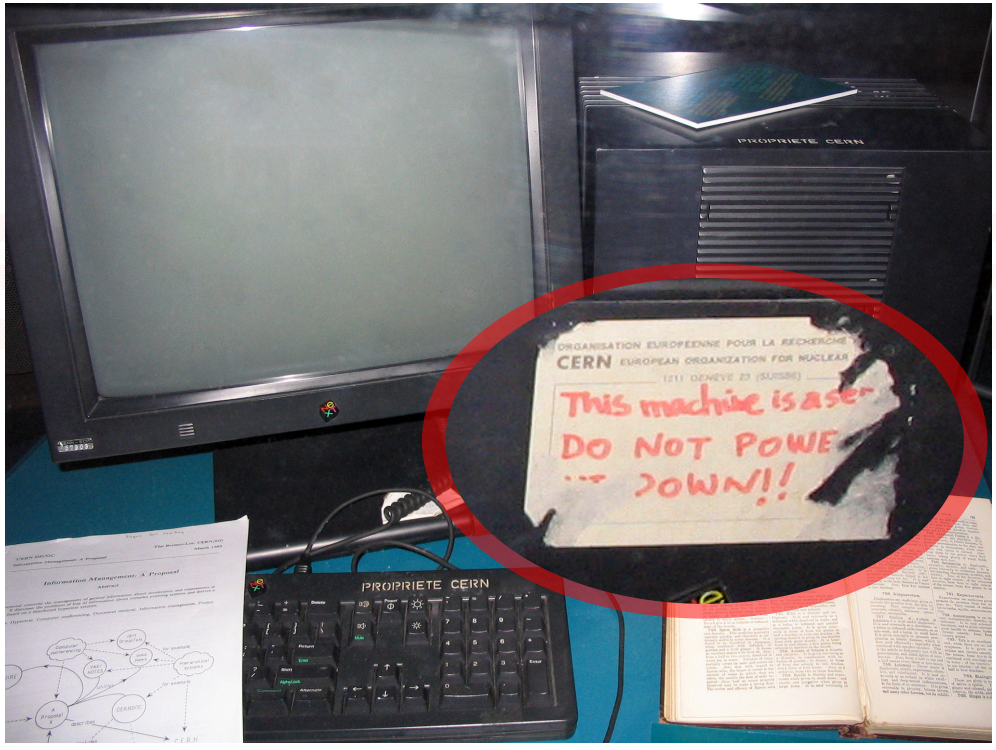


Figure: Tim Berners-Lee's first web server

Conventional circumvention: VPNs, proxies, Tor

- Commoditized via browser add-ons and Tor Browser
- Prone to censorship themselves
  (e.g. Russian Federal Law articles 10.1, 15.4)
- Exit node sees one's data
- Sometimes the user only has access to a *national intranet*

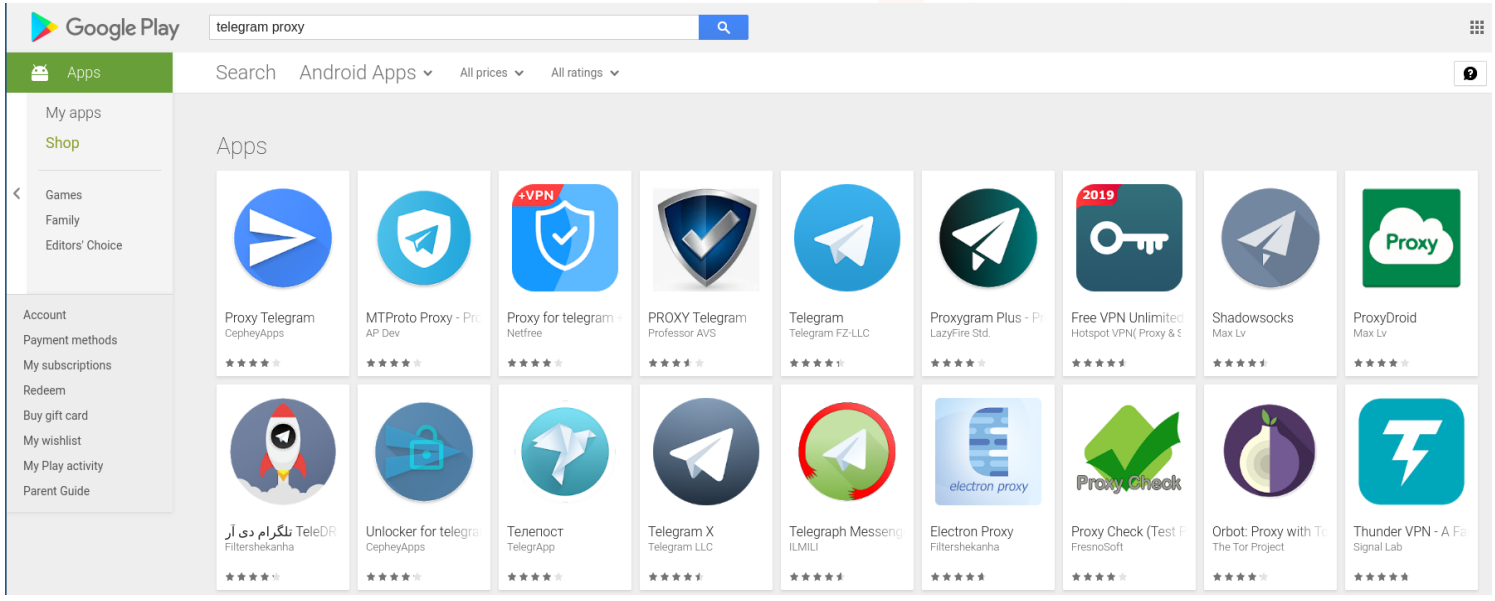


Figure: Shady proxy apps to circumvent Telegram blocking

Decentralization approach

- Decentralized techs avoid central source issues
- Much innovation in current decentralized techs:
  - Web-like: Beaker browser, ZeroNet, Freenet...
  - Storage: IPFS—InterPlanetary File System, Dat, BitTorrent...
  - Multi-hop proxying (anonymous or not):
    I2P—Invisible Internet Project, Tor, Lantern, Psiphon...
  - Obfuscation: Pluggable Transports, OBFS...
  - Curated content distribution: Kiwix, Internet-in-a-Box,
    Toosheh/Knapsack for Hope...
- They still require users to learn new concepts and abilities
  - Barrier for non tech-savvy users
- None of them allows *browsing the common Web naturally* under *total restriction of international traffic*

Enter Ouinet and CENO

Ouinet brings together:

- Decentralized tech strengths
- Familiarity with existing Web concepts and tools

Enable a *censorship-resistant and easy to use Web browser*
via a *distributed cache*
that *stays up when network is split*.

What is Ouinet?

Ouinet: a technology for users to cooperate on interference-resistant Web browsing:

- github.com/equalitie/ouinet
- Decentralized transport and caching for the Web
  - It feels like normal browsing
  - It avoids server reachability issues
- Supported by users' cooperation
  - A CDN—content delivery network for content accessed by users
- Available as a library for your app

CENO—Censorship.no!: a browser using Ouinet to allow users to freely browse the Internet

See Ouinet README and censorship.no for more details.

Cooperative browsing

Ouinet-powered apps auto-seed visited content (configurable).

- Avoid later *upstream availability* issues
  - Get content *from other users*
  - Maybe slightly outdated (but better than nothing)
- Avoid *slow* international connections
- Avoid *expensive* international connections (non-neutral pricing)
- Better for recent content popular in the region

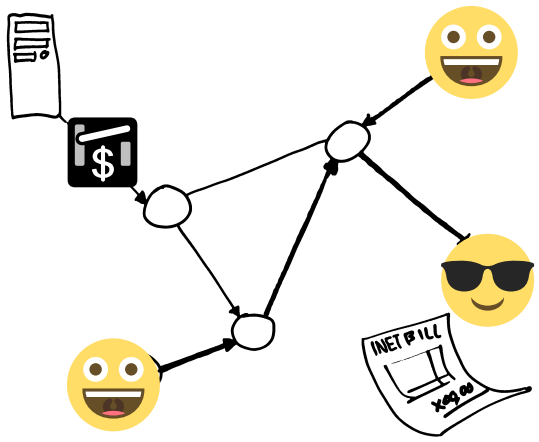
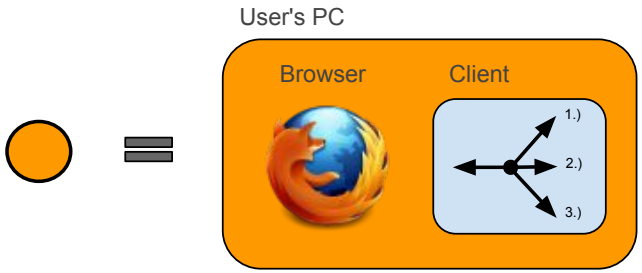


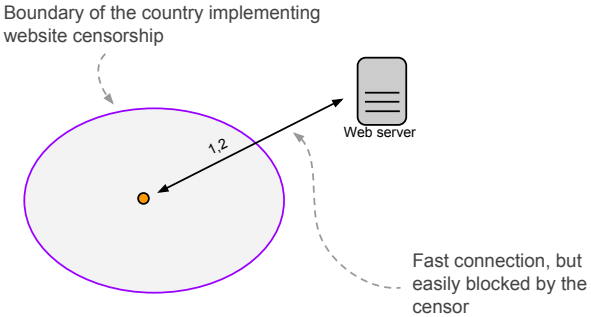
Figure: Cooperative delivery of content

How does Ouinet work?

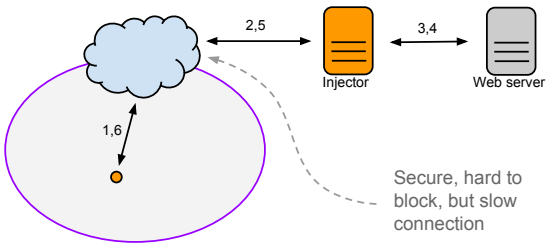


Client is a program that runs on user's PC and acts as a local HTTP/S proxy to the browser.

1.)

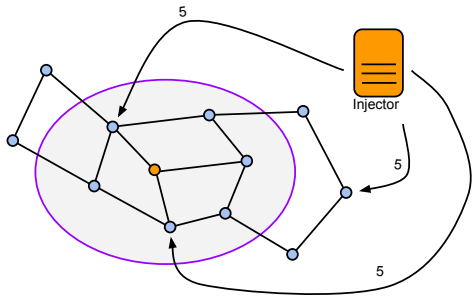


2.)



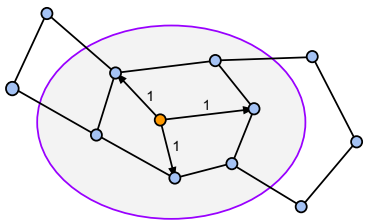
Client sends a request over a secure channel (e.g. TOR, I2P) to the *injector* which then forwards the request to the origin web server. Response from the server follows the same path in reverse.The link between the client and injector - while secure - is slower.

&



While the injector is sending a response back to the user, it also uploads it to a *distributed cache* (e.g. BitTorrent, IPFS, …).

3.)



&

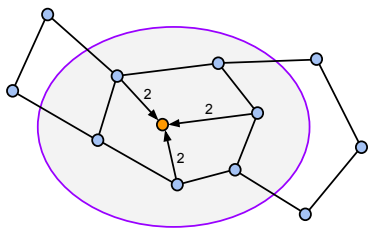


Figure: Direct access vs. injection and cache retrieval

Ouinet technical features

- Pluggable architecture
  - Easily replace non-working modules
  - Existing, well-stablished projects and standards
- Distributed caching modules
  - Distributed back-end for HTTP proxy-cache
  - Degraded operation on unreachable origin
  - IPFS—InterPlanetary File System, BitTorrent BEP44, BitTorrent BEP5 (in progress)
- P2P—peer-to-peer transport modules
  - P2P routing towards origin via proxies and injectors
  - They bridge access to the Web
  - Plain TCP, TLS—Transport Layer Security, I2P—Invisible Internet Project, Pluggable Transports 2.0, µTP—Micro Transport Protocol (in progress)
- Flexible trust, cooperation between users
  - Choose injectors to validate URL mappings

Request flow in Ouinet

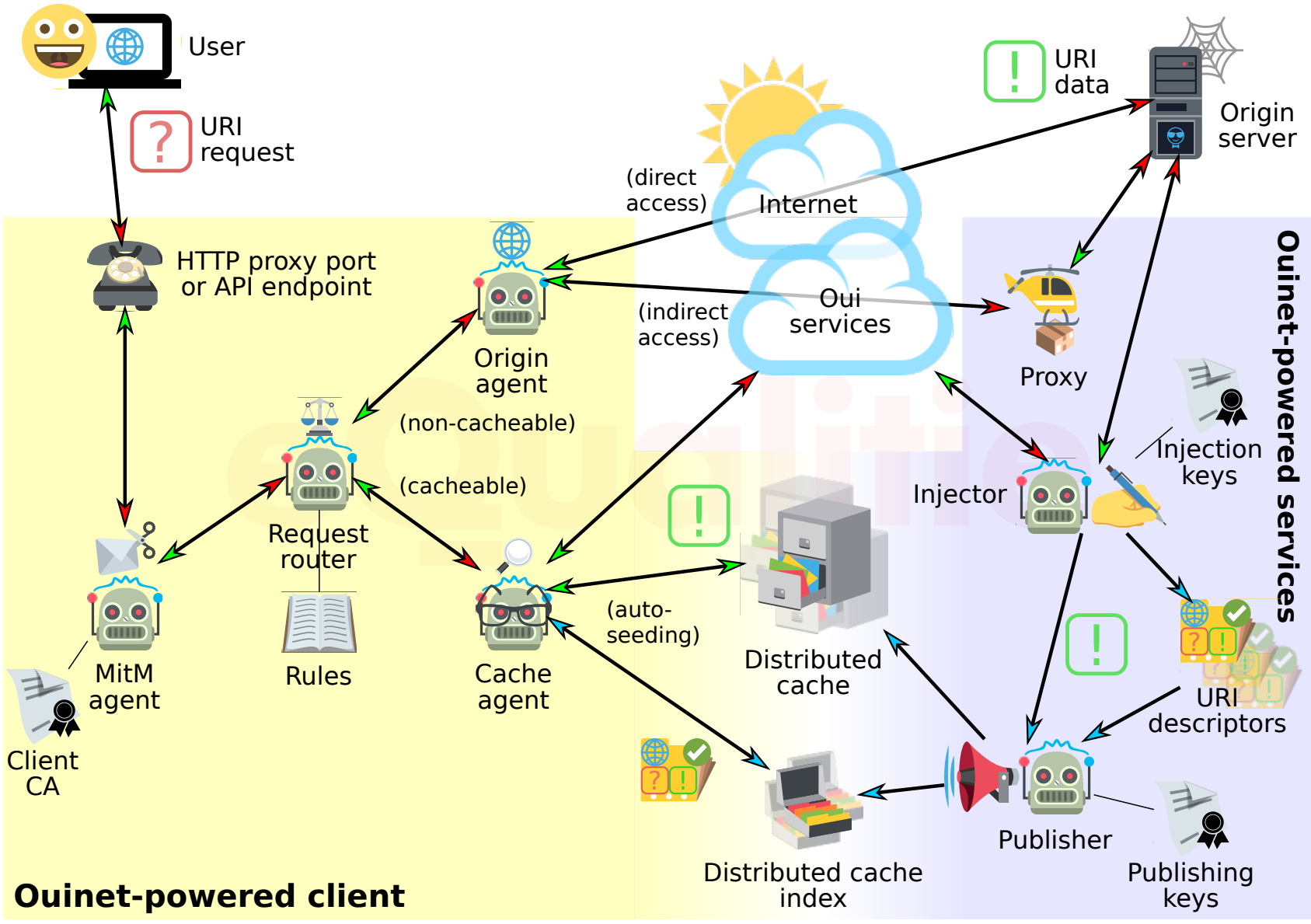


Figure: Different paths followed by requests and responses

Request routing and caching

Client's **request router** decides where to send requests:

- Directly to the *Origin*
- Indirectly using a *Proxy* (reachable over P2P transport)
- Lookup the *Cache* (and seed the content)
- Ask the *Injector* to add it to the cache (then seed it)

Standard HTTP proxy-cache mechanisms to decide when to use the cache.

Different **cache indexes** to lookup URLs in the cache (and certify content):

	fast	not enumerable	reinsertable
B-tree	+	-	-
BEP44	-	+	+
BEP5 (in progress)	+	+	+

Ouinet-based tools

The CENO browser:

- github.com/equalitie/ouifennec
- Rebranded Fennec + embedded Ouinet + WebExtension user interface
- Uses eQualitie's injectors (by default)

Content uploader:

- github.com/equalitie/ouinet-upload
- Helper tool to help publish file collections
- BitTorrent insertion data is exported to disc
- Content and insertion files are circulated
- Untrusted users can reinsert data in isolated country

The future

Ouinet:

- More caching and transport back-ends
  - Resilient, geography-aware, popular
- Caching and privacy
- Speed, resource usage
- Reliability
  - Multiple injectors, better transports
- Usable, autonomous content injection
- Adoption (apps & publishers)

CENO browser:

- More user interface work for Ouinet features
- More user testing
- More documentation and outreach
- Usage statistics

Thank you!

Image credits:

- TimBL's server CC-BY-SA Coolcaesar
- EmojiOne icons CC-BY EmojiOne Inc. (by Inkscape Open Symbols)