Flash Proxies

Contents

Status

As of 2017, the flash proxy project is deprecated. It was deployed in Tor Browser between 2013 and 2016, but has since been superseded by newer and more effective pluggable transports. If you want to help support a newer circumvention system designed along the same principles as flash proxy, please see Snowflake.

Overview

Flash proxies are a way of providing access to a censorship circumvention system such as Tor. A flash proxy is a miniature proxy that runs in a web browser. It checks for clients that need access, then conveys data between them and a Tor relay.

Tor has bridge relays, but in some cases even these can be blocked despite the fact that their addresses are handed out only a few at a time. The purpose of this project is to create many, generally ephemeral bridge IP addresses, with the goal of outpacing a censor's ability to block them. Rather than increasing the number of bridges at static addresses, we aim to make existing bridges reachable by a larger and changing pool of addresses.

"Flash proxy" is a name that should make you think "quick" and "short-lived." Our implementation uses standard web technologies: JavaScript and WebSocket. (In the long-ago past we used Adobe Flash, but do not any longer.)

If your browser runs JavaScript and has support for WebSockets then while you are viewing this page your browser is a potential proxy available to help censored Internet users.

How to Help

Copy and paste this HTML into your web site or blog. An example is at the bottom of this page.

```
<iframe src="//crypto.stanford.edu/flashproxy/embed.html" width="80" height="15" frameborder="0" scrolling="no"></iframe>
```

There is an options page (the same page you get by clicking on the badge) with which users can choose whether they want to be a proxy. By default, if a user has not made a choice, they will be a proxy. If you want only people who have explicitly clicked "yes" to be a proxy, add the `cookierequired` parameter. If a user has selected "no," they will never be a proxy, regardless of the presence of `cookierequired`.

```
<iframe src="//crypto.stanford.edu/flashproxy/embed.html?cookierequired" width="80" height="15" frameborder="0" scrolling="no"></iframe>
```

Browser plugins

Cupcake is an always-on flash proxy plugin for Chrome. Click here to install from the Chrome web store.

Tor Flashproxy Badge is an always-on flash proxy plugin for Firefox. Click here to install from Mozilla Add-Ons.

Wikipedia

Customize your Wikipedia skin to include a flash proxy badge.

Research Paper

This paper contains a fuller description of the system and the results of performance experiments.

- Evading Censorship with Browser-Based Proxies (PDF)
  In the Proceedings of the 12th Privacy Enhancing Technologies Symposium (PETS 2012), LNCS 7384, pp. 239–258, 2012.

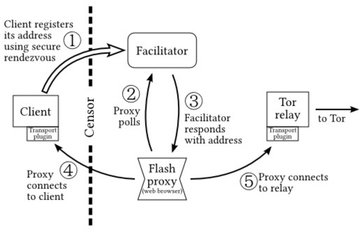
Video

An overview of the flash proxy system and its state of development as of February 2013.

- HTML abstract
- Video stream

How It Works

In addition to the Tor client and relay, we provide three new pieces. The Tor client contacts the **facilitator** to advertise that it needs a connection. The facilitator is responsible for keeping tack of clients and proxies, and assigning one to another. The **flash proxy** polls the facilitator for client registrations, then begins a connection to the client when it gets one. The **transport plugins** on the client and relay broker the connection between WebSockets and plain TCP.



A sample session may go like this:

1. The client starts Tor and the client transport plugin program (`flashproxy-client`), and sends a registration to the facilitator using a secure rendezvous. The client transport plugin begins listening for a remote connection.
2. A flash proxy comes online and polls the facilitator.
3. The facilitator returns a client registration, informing the flash proxy where to connect.
4. The proxy makes an outgoing connection to the client, which is received by the client's transport plugin.
5. The proxy makes an outgoing connection to the transport plugin on the Tor relay. The proxy begins sending and receiving data between the client and relay.

The whole reason this is necessary is because the client cannot communicate directly with the relay. (Perhaps the censor has enumerated all the relays and blocked them by IP address.) In the above diagram, there are two arrows that cross the censor boundary; here is why we think they are justified. The initial connection from the client to the facilitator (the client registration) is a very low-bandwidth, write-only communication that ideally may happen only once during a session. A careful, slow, specialized rendezvous protocol can provide this initial communication. The connection from the flash proxy to the client is from an IP address the censor has never seen before. If it is blocked within a few minutes, that's fine; it wasn't expected to run forever anyway, and there are other proxies lined up and waiting to provide service.

Doesn't the censor win just by blocking the facilitator? Doesn't this shift the problem from bridge-blocking to facilitator-blocking? The short answer to these questions is **no**. We assume that the censor has blocked the facilitator. For more details, see the FAQ.

From the user's perspective, only a few things change compared to using normal Tor. The user must run the client transport plugin program and use a slightly modified Tor configuration file. Complete details are in our README.

How to Use It

Flash proxy is built into Tor Browser. Follow this link to download Tor Browser.

- Tor Browser download page

Next, read the flash proxy howto to learn how to configure port forwarding. See the manual configuration in the rest of this section if the browser bundle doesn't work.

Some Technical Information

Limitations on outgoing connections

It is a restriction of WebSockets that they cannot receive TCP connections, only open them. That is the reason for the client transport plugin: it allows Tor to receive connections instead of making them.

Badge colors

The badge changes color depending on its state.

Dark blue means the proxy is running but no client is being served.

Light blue means a client is currently being served.

Gray means that the badge has disabled itself. This can be because it has detected it is running on a mobile device, or the browser doesn't support WebSocket (this happens on Internet Explorer 9).

Black means that there was an internal error and the proxy is no longer running.

Source Code

All the programs making up the flash proxy system are free software and their source code is visible. To get a copy of everything, run this command:

```
git clone https://git.torproject.org/flashproxy.git
```

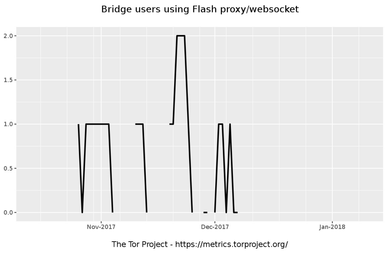
Or browse the code through gitweb.

History

Flash proxies began as a project in Stanford's CS294s class in spring 2011. David Fifield, Nate Hardison, and Jonathan Ellithorpe were members of the project team. They and Emily Stark, Roger Dingledine, Phil Porras, and Dan Boneh wrote a research paper on the subject. Development continues as part of the Tor Project.

Users

Estimated average number of concurrent users. See the metrics site for more control over the graph and historical measurements.



Contact

David Fifield <dcf@cs.stanford.edu>

Try the issue tracker and tor-dev mailing list.

More Information

- Research paper (PDF)
- README
- FAQ
- Bug reports and open tasks (including closed tickets)