Unhosted Adventures

Official Handbook of the No Cookie Crew

Location: Hacker Beach

Follow: atom twitter identica diaspora facebook

Author: Michiel B. de Jong

Episode 1: Personal servers and ~~unhosted~~Unhosted web apps

Hosted software threatens software freedom

The Linux operating system and the Firefox browser have an important thing in common: they are free software products. They are, in a sense, owned by humankind as a whole. Both have played an important role in solving the situation that existed in the nineties, where it had become pretty much impossible to create or use software without paying tribute to the Microsoft monopoly which held everything~~everthing~~ in its grip.

The possibility to freely create, improve, share, and use software is called ~~Software Freedom~~software freedom. It is very much comparable to for instance ~~Freedom of Press~~freedom of press. And as software becomes ever more important in our society, so does ~~Software Freedom~~software freedom. We have come a long way since the free software movement was started, but in the last five or ten years, a new threat has come up: hosted software.

When you use your computer, some software runs on your computer, but you are also using software that runs on servers. Servers are computers just like the ones we use every day, but usually racked into large air-conditioned rooms, and stripped of their screen and keyboard, so their only interface is a a network cable. Whenever you "go online", your computer is communicating over the ~~internet~~Internet to one or more of these servers, which produce the things you see online. Unfortunately, most servers are controlled by big companies ~~like~~such as Google, Facebook and Apple~~, and t~~. These companies have full control over most things that happen online.

This means that even though you can run Linux and Firefox on your own computer, humankind is still not free to create, improve, share and use just any software. In almost everything we do with our computers, we have to go through the companies that control the hosted software that currently occupies most central parts of our global infrastructure. The companies themselves are not to blame for this, they are inanimate entities whose aim it is to make money by offering attractive products. What is failing are the non-commercial alternatives to their services.

One server per human

The solution to the control of hosted software over our infrastructure is quite simple: we have to decentralize the power. Just like freedom of press can be achieved by giving people the tools to print and distribute underground pamphlets, we can give people their freedom of software back by teaching them to control their own server.

Building an operating system like Linux, or a browser like Firefox is quite hard, and it took some of the world's best software engineers many years to achieve this. But building a web server~~webserver~~ which each human can choose to use for their online software needs is not hard at all: from an engineering perspective, it is a solved problem. The real challenge is organizational.

The network effect

A lot of people are actually working on alternative "personal servers" like this. Many of them are already functional. You can go to one of these projects right now, follow the install instructions, and you will have your own personal server, independent from any of the hosted software monopolies. But there are several reasons why not many people do this yet. They all have to do with the network effect.

First, there is the spreading of effort, thinned out across many unrelated projects. But this has not stopped most of these projects from already publishing something ~~workable~~that works. So let's say you choose the personal server from project X, you install it, and run it.

Then we get to the second problem: if I happen to run a personal server from project Y, and I now want to communicate with you, then this will only work if these two projects have taken the effort of making their servers compatible with each other.

Luckily, again, in practice most of these personal server projects are aware of this need, and cooperate to develop standard protocols that describe how servers should interact with each other. But these protocols often cover only a portion of the functionality that these personal servers have, and also, they often lead to a lot of discussion and alternative versions. This means that there are now effectively groups of personal server projects. So your server and my server no longer have to be from the same project in order to understand each other, but they now have to be from the same group of projects.

But the fourth problem is probably bigger than all the previous ones put together: since at present only early adopters run their own servers, chances are that you want to communicate with the majority of other people, who are (still) using the hosted software platforms from web 2.0's big monopolies. With the notable exception of Friendica, most personal server projects do not really provide a way to switch to them without your friends also switching to a server from the same group of projects.

Simple servers plus ~~unhosted~~Unhosted apps

So how can we solve all those problems? There are too many different applications to choose from, and we have reached a situation where choosing which application you want to use dictates, through all these manifestations of the network effect, which people you will be able to communicate with. The answer, or at least the answer we propose in this blog series, is to move the application out of the server, and into the browser.

Browsers are now very powerful environments, not just for viewing web pages, but actually for running entire software applications. We call these "~~unhosted~~Unhosted web applications", because they are not hosted on a server, yet they are still web applications, and not desktop applications, because they are written in ~~html~~HTML, ~~css~~CSS and ~~javascript~~JavaScript, and they can only run inside a browser's execution environment.

Since we move all the application features out of the personal server and into the browser, the actual personal server becomes very simple. It is basically reduced to a gateway that can broker connections between ~~unhosted~~Unhosted web apps and the outside world. And if an incoming message arrives while the user is not online with any ~~unhosted~~Unhosted web app, it can queue it up and keep it safe until the next time the user connects.

Unhosted web apps also lack a good place to store valuable user data, and servers are good for that. So apart from brokering our connections with the outside world, the server can also function as cloud storage for ~~unhosted~~Unhosted web apps. This makes sure your data is safe if your device is broken or lost, and also lets you easily synchronize data between multiple devices. We developed the remotestorage protocol for this, which we recently submitted as an IETF Internet Draft.

The ~~unhosted~~Unhosted web apps we use can be independent of our personal server. They can come from any trusted source, and can be running in our browser without the need to choose a specific application at the time of choosing which personal server to install. This separation of concerns is what ultimately allows people to freely choose:

- Which personal server you run
- Which application you want to use today
- Who you interact with

No Cookie Crew

This blog is the official handbook of a group of early adopters of ~~unhosted~~Unhosted web apps called the No Cookie Crew. We call ourselves that because we have disabled all cookies in our main browser. We ~~will~~ keep a secondary browser on the side, with cookies (as well as Flash) still enabled, but will only use it when we have to, and only to interact with what we consider "second parties". To give an example: if you publish something "on" Twitter, then Twitter acts as a third party, because they, as a company, are not the intended audience. But if you send a user support question to Twitter, then you are communicating "with" them, as a second party. Other examples of second-party interactions are online banking, online check-in at an airline website, or buying an ebook from Amazon. But if you are logging in to e-Bay to buy an electric guitar from a guitar dealer, then e-Bay is more a third party in that interaction.

There are situations where going through a third party is unavoidable, for instance if a certain guitar dealer only sells on e-Bay, or a certain friend of yours only posts on Facebook. For each of these "worlds", we will develop application-agnostic gateway modules that we can add to our personal server. Usually this will require having a "puppet" account in that world, which this gateway module can control. Sometimes, it is necessary to use an API key before your personal server can connect to control your "puppet", but these are usually easy to obtain. To the people in each world, your puppet will look just like any other user account in there. But the gateway module on your personal server allows ~~unhosted~~Unhosted web apps to control it, and to "see" what it sees.

In no case will we force other people to change how they communicate with us. Part of the goal of this exercise is to show that it is possible to completely log out of web 2.0 without having to lose contact with anybody who stays behind.

To add to the fun, we also disabled Flash, Quicktime, and other proprietary plugins, and will not use any desktop apps.

Whenever we have to violate these rules, we make a note of it, and try to learn from it. And before you ask, at this moment, the No Cookie Crew still has only one member. Logging out of web 2.0 is still pretty much a full-time occupation, because most things you do require a script that you have to write, upload and run first. But if that sounds like fun to you, you are very much invited to join! Otherwise, you can also just follow from a distance by following this blog.

This blog

Each blog post in this series will be written as a tutorial, including code snippets, so that you can follow along with the ~~unhosted~~Unhosted web apps and personal server tools in your own browser and on your own server. We will be using mainly nodejs on the server-side. We will publish one episode every Tuesday. As we build up material and discuss various topics, we will be building up

the official Handbook of the No Cookie Crew. Hopefully, any web enthusiast, even if you haven't taken the leap yet to join the No Cookie Crew, will find many relevant and diverse topics discussed here, and food for thought. We will start off next week by building an ~~unhosted~~Unhosted text editor from scratch. It is the editor I am using right now to write this, and it is indeed an ~~unhosted~~Unhosted web app. There are various ways to follow us, so you will get notified each time a new episode comes out: atom, mailing list, irc channel, twitter, identica, diaspora facebook. So stay tuned! :)