GNU social - Network Services Improvements

Bruno Casteleiro

```
Name: Bruno Miguel Alves Abrantes Carvalho Casteleiro
Website: www.github.com/brunoccast
Email: up201505347@fc.up.pt
Project Name: Network Services Improvements
```

Summary

GNU social is a communication software used in federated/descentralized social networks. In order to archive such descentratization, some communication standards were created such as OStatus and ActivityPub. Currently, GNU social is still using OStatus.

In 2018, a plugin that implements the ActivityPub specification was developed. Despite that, an implementation of ActivityPub standard is not sufficient for production usage. Currently, the plugin has an unfinished implementation of HTTP Signatures, making GNU social unable to federate with other pieces of software currently using ActivityPub, and it doesn't use queues.

Furthermore, GNU social its client API, the Twitter-like API, needs some of its functionality reviewed, specially concerning third-party interfaces. The current implementation still requires Tools - Actors with type Application - to use OAuth1 for authorization and it doesn't handle Bots - internal (non-federated) entity that allow Actors with type Person to use third-party tools to control their accounts - the proper way.

In a first stage, this proposal aims to replace the existing network system with a more sophisticated one that will support HTTP Signatures and to move existing ActivityStreams implementations into a plugin to be used by OStatus and the ActivityPub plugin.

The second and final part of the proposal is reserved for the migration of OAuth1 to OAuth2 and adding support to ActivityPub C2S.

Benefits

This will be greatly beneficial to GNU social, the fediverse, and, obviously, the whole community involved:

- The ActivityPub plugin will become fully compatible with other software, safe and robust, ready for GNU social to make full use of it.
- The current social API will improve, greatly benefiting third party tools:
    - The authorization model will be udpated, using the current best practices of OAuth2.
    - Revision on Tools support
- Bots will be properly handled with the introduction of a new API
- New interfaces (managment dashboards) for both Tools and Bots

Implementation plan and Timeline

May 6 - May 27

- Familiarize with the community
- Familiarize with the code (further insights in):
  - Social API
  - Tool support
  - Plugin and Event systems
  - Queues
- Familiarize with the documentation and both development and test system used
- Familiarize with the ActivityPub Standard

May 27 - June 24

ActivityPub plugin

- Fix HTTP Signatures
  - Use and adapt Pixelfed's own Guzzle adaptation
- Add support for the Group actor type
  - Add classes and actions for the relevant Activities: Add, Join, Leave and Remove
  - Make Postman publish the new activities
  - Address group-related GNU social events
- Use the GNU social queue system
  - Create necessary queue item handlers
  - Make Postman enqueue Activity logic to be (later) handled

Deduplication - Fediverse plugin

I propose the introduction of a new plugin - the Fediverse plugin - to deduplicate OStatus/ActivityPub. This plugin would gather together all the event handling required concerning notice "input/output", calling the necessary resources from both the OStatus and ActivityPub plugins in order to follow up execution.

This approach to deduplication would make code better organized while preventing new replication of event handling in a (possible) future introduction of a new protocol for the federated-web.

Additional refactoring

- Move code related to ActivityStreams (1.0 + 2.0) to a separated plugin

June 24 - June 28 - First evaluation

June 25 - July 22

There were some alternatives I could have proposed for this month work. The options I choosed are backed up on research and discussion on different channels: I've opened an issue in AndStatus git repository and started some discussion in both GNU social and W3C IRC channels.

Authorization Model

Compared to its predecessor, OAuth2 has better performance at scale, supports a better user experience for Tools and uses TLS to address the annoying requirements of signing requests. Since there is no strong reason to oppose the full adoption of TLS in GNU social, I'm proposing the implementation of this, obviously, advantageous version of the authorization standard.

Moreover, OAuth2 is really simple to understand, the general flow consists in the following simple steps:

1. Some Tool request authorization to some of its user's resources
2. If authorized, the Tool receives and authorization grant
3. The Tool uses its identity and authorization grant to request an access token
4. If the identity is authenticated and the grant is valid, an access token is issued
5. The Tool can finally request the desired resource from the provider API by presenting its access token for authentication

Summing up:

- Migrate OAuth1 to OAuth2
  - Using OAuth 2 server PHP as the implementation
- Use authorization code flow (without secret code) for Tools authentication
- Ensure API functionality
  - Review and test API actions that require authentication

ActivityPub C2S

While the implementation of OAuth2 on top of the current Twitter-like API would suffice to support Bots, discussion showed some advantages in adopting the AP C2S API:

- Bots (actors of type Application) are not that different of users, therefore it makes sense to use ActivityPub
    - This is, more than an advantage, semantic-sugar and code organization
- Queue usage (like the S2S API)
    - Contrary to Tools, the system could benefit from using the queue system for Bots since there is no requirement for immediate answer-issuing

Moreover, the ActivityPub S2S API should be finished by now, so implementing the C2S API should be fast and simple to do.

Summing up:

- Implement requirements for Outbox Delivery (S2S)
    - target and deliver to specific object fields: to, bto, cc, bcc
- Handle Activity reception in the Outbox
    - Fairly straightforward since both the S2S API and the Outbox are implemented
- Use OAuth2 bearer tokens for authentication of C2S interactions

Interfaces:

- Introduce user Dev Dashboard for Applications
    - Review current Application new/edit form and move it to this dashboard
    - Clone current Application Settings to this dashboard
- Review (non-dev) Application Settings
    - Change to "list all" behavior (not user-proprietary Apps only)
    - Add option to revoke App access permission
- Introduce user Dev Dashboard for Bots
    - Add new/edit form
    - Add settings

July 22 - July 26 - second evaluation

July 23 - August 19

At this stage I should have all the proposed work completed. If I miss something from last month, or if perhaps I need to do some more testing or documentation, then I'll be using (at least) the first part of this month to finish it.

Moreover, and since part of my proposal relates to frontend work (dashboards), I think it would make sense to use the remaining time I have to work on enhancing the asthetics of GNU social. For this I would most likely look into PixelFed's MicroUI.

Summing up:

- Finish proposed work (if required)
- Rework GNU social frontend

August 19 - 26 August - Code submisson and final evaluation

- General review, documentation and testing
- Tech-report write up
- Final merge and evaluation

Deliverables

GNU social's codebase:

- Code refactor, addition and removal in different endpoints
- New API
- 2 new plugins (+documentation)
- Documentation and testing

ActivityPub Plugin:

- Code refactor and addition

Communication

GNU social's IRC will be used to communicate with the community and the mentors, but I'm open to use whatever is more convenient. My nick on freenode is tenma. I'll be adapting my work schedule as I expect it to be hard to maintain as a constant throughout the entire summer. Being so, I'll make sure to always communicate any changes so Diogo knows my work and communication hours.

My changes/additions to the code will be hosted on a fork of Diogo's fork. This will be done to prevent accidental commits, as a significant part of the community has switched to Diogo's repository.

Qualification and related relevant experience

Since exposed to the concept of federated software and, in particular, to the many federated social networks already up and running, I rapidily began to not only use them but to also grow a big interest in the all the "movement" surrounding this pieces of software. Having the opportunity to contribute and help growing nodes of this network of federated communication software, like GNU social, is therefore something I'm really motivated to do.

Currently I'm in my first year of the masters degree in Computer Science, at the Sciences faculty of the University of Porto (Portugal). Although never figuring the main path of my university journey, this 4 academic years alone presented me to many important web-dev endpoints required for a work like the one I'm proposing, like databases or frontend-dev (HTML,CSS and JS). Back in high-school I also developed, as the final course project, a mobile social network that required a lot of database handling and PHP programming (backoffice + API).

For working in GNU social, I will have to keep learning more about the project itself, to get confortable with the ActivityPub standard and to get the rust off of my PHP programming.