Obliterating history from Git

To obliterate history from the Git repo means removing it from three different sources: Gerrit, Forgejo, and GitHub.

A tool has been written, called gerrit-rewrite-branch, to rewrite Gerrit history completely, including the meta on past CLs.

To use it, build it as `--release` (it will stack overflow on debug mode), and find the following repos, and make backups of them:

- `/var/lib/gerrit/git/lix.git`
- `/var/lib/forgejo/repositories/*/lix.git`

To start off, stop Gerrit and find the Git repo for it. The tool requires four things: The email address to obliterate, and a replacement name + email address. It also needs a cutoff date for where to remove commits before. To find this, run `git cat-file -p {commit}` for a commit earlier than the oldest you want to remove, and note down the timestamp on the `committer` line.

Call the tool. It will churn for a while, and rewrite all previous Git commits, plus the Gerrit metadata of affected commits. As a bonus, run a `git gc --prune=now`.

Before turning on Gerrit, run `systemd-run -p DynamicUser=yes -p StateDirectory=gerrit -t gerrit reindex -d /var/lib/gerrit`. This ensures Gerrit is aware of the changes made outside of its existence.

For forgejo, no special steps are needed; just run the same tool over these repos plus all their forks, and prune the reflog and unreachable commits as well:

```
[root@lix:/var/lib/forgejo/repositories]# for i in */lix.git; do pushd $i; sudo -u git git reflog expire --expire=all --expire-unreachable=all --all; sudo -u git git gc --prune=now; popd; done
```

Once Gerrit and Forgejo are back up, run `ssh gerrit.lix.systems replication start --now --url github` to propagate the changes to GitHub.

Don't forget to ban the commits as well, using `ssh gerrit.lix.systems gerrit ban-commit lix {commits}`.

Revision #4
Created 4 May 2024 21:54:14 by Puck Meerburg
Updated 4 May 2024 22:57:31 by Puck Meerburg