A GNU Hurd development environment

Dr. Arne Babenhauserheide

<2021-09-11 Sa>

I am finally working on Hurd-Projects again. Here I am describing my setup for hacking on the Hurd.

This is a **work in progress**.

Contents

Setting up a VM

Firstoff: see the setup instructions for qemu. Here I only describe what I need myself.

```
wget https://cdimage.debian.org/cdimage/ports/latest/hurd-i386/debian-hurd.img.tar.gz
tar -xz < debian-hurd.img.tar.gz
```

Simple tooling: start the VM and connect to it

I structure my hurd setup into two scripts: `run-hurd.sh` and `login-to-hurd.sh`.

start

My Hurd VM is started with ncurses interface to avoid qemu interference with my Keyboard layout. If you use quert*, leave out the `--curses`.

```
qemu-system-x86_64 --enable-kvm -m 5G -drive cache=writeback,file=$(echo debian-hurd-
  --device rtl8139,netdev=net0 --netdev user,id=net0,hostfwd=tcp:127.0.0.1:10022-:22
```

Note the hostfwd: this provides ssh login via port 10022 (copied from the Guix Hurd setup).

login

Login is simple:

```
ssh -p 10022 root@localhost || echo "if ssh cannot connect,
  run inside the vm
  ssh arne@192.168.2.105 -p 22 -- cat '~/.ssh/id_rsa.pub' >> ~/.ssh/authorized_keys"
```

Note the error handling: For this to work, you must add your public ssh key to the authorized keys. That provides minimal security.

Development tooling

login to and from the VM

I want simple synchronization between VM and Host, so I'm settting up SSH keys on both sides. The ssh-setup for the VM is above. The following allows me to login from the Hurd VM to my host machine:

```
echo -e "\n\n\n" | ssh root@localhost -p 10022 -- bash -c \
  'ssh-keygen >/dev/null ; cat ~/.ssh/id_rsa.pub' >> ~/.ssh/authorized_keys
echo
ssh root@localhost -p 10022 -- ssh arne@192.168.2.105 echo test
```

If this prints test, the automatic round-trip works

To avoid using my username and host in here all the time, let's add aliases.

```
# host to hurd alias

echo '
Host hurd localhost
    HostName localhost
    User root
    Port 10022
' >> ~/.ssh/config

# hurd to host alias
ssh hurd -- 'cat >> ~/.ssh/config' <<EOF
Host host 192.168.2.105
    HostName 192.168.2.105
    User arne
EOF

ssh hurd -- ssh host echo test
```

Install the tooling

Login to the vm as above. I keep all specific stuff in `~/Dev`. To install a fully working development environment, use:

- Mercurial

  This is what I use for version control of my own stuff. Also my git install failed.

  ```
  mkdir -p ~/Dev
  cd ~/Dev
  apt install python3-dev
  wget https://www.mercurial-scm.org/release/mercurial-5.9.1.tar.gz
  tar xf mercurial-5.9.1.tar.gz
  cd mercurial-5.9.1
  python3 setup.py install --user
  echo 'export PATH="${PATH}:${HOME}/.local/bin"' >> ~/.bashrc
  source ~/.bashrc
  ```

- Git

  To work with the Hurd git repos. My git install failed (didn't find openssl), so I'll be using hg-git.

  ```
  cd ~/Dev
  apt install python3-dulwich
  hg clone https://foss.heptapod.net/mercurial/hg-git
  hg conf --edit # opens with vim:
  # - move to the bottom
  # - press o to add and edit a line
  # - write hggit = ~/Dev/hg-git/hggit
  # - press ESCAPE :wq ENTER to save and exit
  ```

  Update: After asking on IRC, paulusASol told me a working method to get git:

  ```
  cd ~/Dev
  # get an old snapshot of git-man
  wget https://snapshot.debian.org/archive/debian/20210607T032400Z/pool/main/g/git
  dpkg -i ./git-man_2.32.0-1_all.deb
  # install git
  apt install git
  ```

- The Hurd repositories

  ```
  cd ~/Dev
  for i in glibc gnumach hurd incubator libpthread mig procfs unionfs web; do
    # this uses my savannah useraccount.
    # It needs ~/.ssh/id_rsa.pub added to my account.
    hg clone git+ssh://arnebab@git.savannah.gnu.org:/srv/git/hurd/$i.git;
  ```

```
done
```

Create a working translator

```
# get dependencies
apt install autoconf automake mig
# Enter the Hurd translator project
cd hurd/
# build the translators
autoreconf -i
./configure --without-parted
make
# Test the Hello World translator
# -c creates the file.
# Without it, you need to touch the file before.
settrans -c ~/hello $(realpath trans/hello)
cat ~/hello
# Hello, world!
# remove the translator again
settrans -g ~/hello
rm ~/hello
```

You can replace the translator during development with

```
settrans -g ~/hello $(realpath trans/hello)
```

Sync with code on the host system for more convenient hacking

```
# in the Hurd
cd ~/Dev/hurd
hg clone . ssh://host/path/to/hurd-dev
echo "[paths]\ndefault = ssh://host/path/to/hurd-dev" >> .hg/hgrc
```

Now you can `hg pull -u` and `hg push` to sync changes.

You can use any programming tool supported by the host system. Just commit it and pull from the hurd to try it out.

Create a non-root user for testing permissions-stuff

```
useradd -m user
```

Start hacking

You have the sources, and you can hack: Ready to try whatever you want to do.