CI for CADR

Synopsis

Currently CADR lacks Continuous Integration. As of current investigation, we decide to use a self-hosted runner at GitHub Actions using VMs from AWS or Azure, which will implement the Debian 10 environment for testing successfully. If we decide to use other CI platforms later during the project period, Jenkins or GitLab CI/CD will also be a valid choice.

Project Plan

Current Status

As of now, a demo for CI using the GitHub Actions has already been worked out. https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/pull/198 The demo CI can be triggered manually, or by sending PRs as well as pushing directly to the master branch. The demo CI first builds the running podman environment, then with the podman environment, builds the CADR packages, and then uploads the built Debian packages to Artifacts. Finally test the CADR just built. However, tests in the CI have failed for lacking some dependencies. Moreover, GitHub Actions can do the CI with the containers' help, but the default runner has an issue that its disk space is too small to satisfy our test needs (Although it's OK to just test the regtest data).

Road of Discovery

At first, I ignored the fact that there already exists a dockerfile for CADR running (although it was not for building)

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/tree/master/docker , and setup my own dockerfile from scratch by adding dependencies when I encounter any errors.

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/commit/1580e9e02 5f7ccb597b35753e004a7d5327f10ef

My own dockerfile turned out to work fine on GitHub Actions for the building process, but failed for the test process.

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/actions/runs/20429636 52

Initially I thought it was some more dependency issues, since the test can work on my own computer!

```
Terminal - hollowman@bogon: ~/Desktop/cryptoanarchy-deb-repo-builder

File Edit View Terminal Tabs Help

(Reading database ... 147822 files and directories currently installed.)

Preparing to unpack .../bitcoind_0.21.1-2_amd64.deb ...

Unpacking bitcoind (0.21.1-2) ...

Selecting previously unselected package bitcoin-mainnet.

Preparing to unpack .../bitcoin-mainnet_0.21.1-2_all.deb ...

Unpacking bitcoin-mainnet (0.21.1-2) ...

Setting up bitcoind (0.21.1-2) ...

Setting up bitcoin-mainnet (0.21.1-2) ...

Processing triggers for man-db (2.8.5-2) ...

Processing triggers for libc-bin (2.28-10) ...

Checking package bitcoin-mainnet

iltcoin-mainnet has pruning enabled - 0K

-2022-03-26 18:18:50-- http://127.0.0.1:8331/

connecting to 127.0.0.1:8331... connected.

HTTP request sent, awaiting response... 200 0K

length: 930 [application/json]

saving to: 'STDOUT'

100%[=============]

930 --.-KB/s in 0s

2022-03-26 18:18:50 (239 MB/s) - written to stdout [930/930]

Thain is correct
```

When checking the logs I found that it's due to the unshare issue.

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5700123801?check_suite_focus=true

```
/etc/apt/sources.list.d/microsoft.list:1
764 Checking package bitcoin-mainnet
765 bitcoin-mainnet has pruning enabled - OK
766 cat: /var/run/bitcoin-mainnet/cookie: No such file or directory
    --2022-03-26 01:52:26-- http://127.0.0.1:8331/
768 Connecting to 127.0.0.1:8331... failed: Connection refused.
769 Bad chain:
770 cat: /var/run/bitcoin-mainnet/cookie: No such file or directory
771 --2022-03-26 01:52:26-- <u>http://127.0.0.1:8331/</u>
772 Connecting to 127.0.0.1:8331... failed: Connection refused.
773 make: *** [Makefile:72: test-here-basic-bitcoin-mainnet] Error 1
774 Error: Process completed with exit code 2.
     Setting up bitcoind (0.21.1-2)
     unshare: unshare failed: Operation not permitted
    Processing triggers for man-db (2.8.5-2) ...
     W: Target Packages (main/binary-amd64/Packages) is configured mult
```

Then I noticed that adding a --privileged parameter can fix the unshare issue for docker. But then a systemd issue just came after it. Finally I noticed the dockerfile in the codebase that already exists for CADR, and just as how it works, I made the systemd to run as the first programme.

```
VOLUME [ "/sys/fs/cgroup" ]
CMD ["/lib/systemd/systemd"]
and add --tmpfs /tmp --tmpfs /run --tmpfs /run/lock -v
/sys/fs/cgroup:/sys/fs/cgroup:ro
```

as additional parameters to share some host machine resources and make it as the daemon container. Operations are done using docker exec to attach to the container. But still it doesn't fix the systemd issue.

Finally I switched to podman and the systemd issue got fixed (it supports --systemd=true) because I occasionally found this article

https://developers.redhat.com/blog/2019/04/24/how-to-run-systemd-in-a-container when I Googled the issue.

However, a new issue occurs

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5708913219?che ck_suite_focus=true suggesting failed to override dbcache.

```
/etc/apt/sources.list.d/microsoft.list:1

690 W: Target Packages (main/binary-amhf/Packages) is configured multiple times in /etc/apt/sources.list.d/microsoft-prod.list:2 and /etc/apt/sources.list.d/microsoft.list:1

691 W: Target Packages (main/binary-all/Packages) is configured multiple times in /etc/apt/sources.list.d/microsoft.prod.list:2 and /etc/apt/sources.list.d/microsoft.list:1

692 Checking package bitcoin-regtest

693 bitcoin-regtest has pruning enabled - OK

694 --2022-03-27 10:38:43-- http://127.09.0.1:18442/

695 Connecting to 127.0.0.1:18442... connected.

696 HTTP request sent, awaiting response... 200 OK

697 Length: 1069 (1.0K) [application/json]

698 Saving to: 'STDOUT'

699

610 OK 100% 227M=0s

611

612 2022-03-27 10:38:43 (227 MB/s) - written to stdout [1069/1069]

613

614 Chain is correct

615 dbcache=450

616 Failed to override dbcache

617 make: *** [Makefile:72: test-here-basic-bitcoin-regtest] Error 1

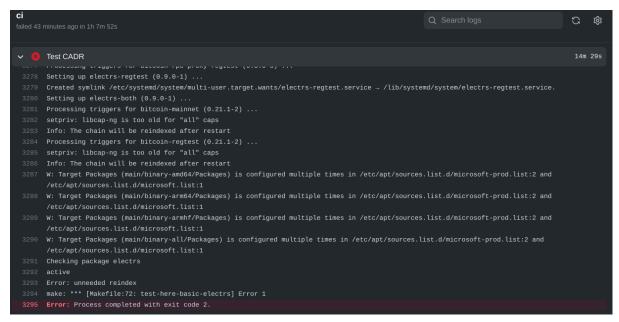
618 Error: Process completed with exit code 2.
```

Can't see any errors from the workflow logs

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5710646832?che ck suite focus=true, even though the missing bc dependency has been fixed and I can confirm that the command here

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/blob/master/pkg_sp_ecs/bitcoin-@variant.sps#L88 works correctly in the container running on my PC. Moreover when I manually skip the test just mentioned, the test further shows that there is an unneeded reindex

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5710055525?check_suite_focus=true,



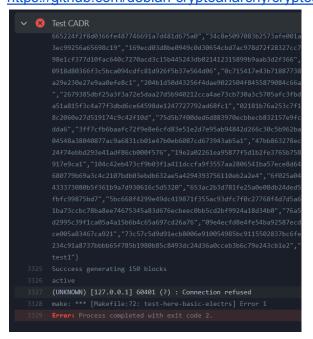
which suggests that it's related to issue 108

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/issues/108 maybe? Have opened an issue here:

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/issues/202 I can skip that test with this commit

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/commit/4defba982028 724de70b9ac26d27a59cb150c4c6, but further error suggesting electrs is not available https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5713299561?che ck suite focus=true Can be fixed with this PR

https://github.com/debian-cryptoanarchy/cryptoanarchy-deb-repo-builder/pull/203



Even if those issues are skipped, another platform will still be needed since the full test requires too much space

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5713729925?check suite focus=true

```
ci
failed 1 hour ago in 1h 43m 23s

V Test CADR

5463 Unpacking 1nd (0.14.1-2) ...

5464 dpkg: error processing archive /tmp/apt-dpkg-install-3y43dx/05-lnd_0.14.1-2_amd64.deb (--unpack):

5465 cannot copy extracted data for './usr/bin/lnd' to '/usr/bin/lnd.dpkg-new': failed to write (No space left on device)

5466 dpkg-deb: error: paste supprocess was killed by signal (Broken pipe)

5467 Selecting previously unselected package Ind-auto-unlock.

5468 Preparing to unpack .../06-lnd-auto-unlock_0.14.1-2_all.deb ...

5469 Unpacking Ind-auto-unlock (0.14.1-2) ...
```

Only testing the regtest part will be fine with no space issue.

https://github.com/HollowMan6/build-cryptoanarchy-deb-repo-builder/runs/5750416035?check suite focus=true

Future Plans

Find all dependencies for the tests to run smoothly.

In order to speed up the CI process, as well as use other alternative platforms, the dockerfile in the demo can also be moved to debcrafter such that whenever debcrafter has a new push, a docker (podman) image can be persistently created for running CADR CI.

The demo mentioned above then can pull the docker (podman) image directly from the docker registry and thus don't need to rebuild it every time the CI runs.

Learn how to create GitHub self-hosted runners on AWS or Azure.

Other CI platforms can be considered. If we decide to use Jenkins, we can specify to use the docker (podman) image as the running environment

https://www.jenkins.io/doc/book/pipeline/docker/ and it will be pretty much like GitHub Actions but with more flexibility. The GitLab CI/CD will also work just like Jenkins https://docs.gitlab.com/ee/ci/pipelines/pipeline architectures.html by specifying the custom docker (podman) images. You can also check one of my projects that uses python docker (podman) image for GitLab CI/CD as an example:

https://gitlab.com/HollowMan6/LZU-Auto-COVID-Health-Report/-/blob/main/.gitlab-ci.yml#L1

Jenkins support Jenkins agents, https://www.jenkins.io/doc/book/using/using-agents/ while GitLab CI/CD support GitLab Runner https://docs.gitlab.com/runner/, which are all equivalent to GitHub self-hosted runners.

More CI platforms will be investigated during the project period, and will compare the pros and cons in detail, then finally choose one platform to implement.

Project Timeline

May 23, 2022 - Jun 5, 2022: Find the missing dependencies to run tests in the container environment to fix the tests running for the demo mentioned above.

Jun 6, 2022 - Jun 19, 2022: Decide which cloud service provider to use.

Jun 20, 2022 - July 3, 2022: Create the self-host runners for CI on GitHub Actions to run successfully.

July 4, 2022 - July 8, 2022: Mid-term evaluations.

July 11, 2022 - July 24, 2022: Write CI on Jenkins and GitLab CI/CD, check if they need self-host runners. If needed, create the runners and check their effectiveness.

July 25, 2022 - August 7, 2022: Also Investigate other possible CI platforms, Write CI on those platforms and check their effectiveness, whether they need self-hosted runners, form a detailed report. Finally decide which CI to use by cost-effectiveness.

August 8, 2022 - August 14, 2022: Time for documentation as well as any lacks.

August 15, 2022 - August 22, 2022: : Final evaluations.

Future Deliverables

I can help resolve issues as much as I can, pack more packages, write more documentations or tests so that eventually help stabilize CADR.

Benefits to Community

With CI the development for CADR will be much easier, any potential developers can now contribute directly to the repo and the reviewer can check the building result of CI to determine whether the PR is valid, thus relieving the workload of maintainers a lot. With CI, new contributors will also feel comfortable as I learnt that many of them still don't know how to build CADR even with detailed documentation.