
:warning: Deprecated in favor of ReDroid

As of April 2021, AinD is deprecated in favor of ReDroid.

While Anbox/AinD has got stuck in Android 7.1, ReDroid supports very recent Android versions: 8.1, 9, 10, 11, and Android S (12).

AinD: Android (Anbox) in Docker

AinD launches Android apps in Docker, by nesting Anbox containers inside Docker.

Unlike VM-based similar projects, AinD can be executed on IaaS instances without support for nested virtualization.

GHCR: ghcr.io/aind-containers/aind

:warning: Docker Hub image [aind/aind](#) is no longer updated. Please use ghcr.io/aind-containers/aind image on GHCR.

Purposes

- Anti-theft (see FAQ)
- Android compatibility (via cloud) for iOS and Windows tablets

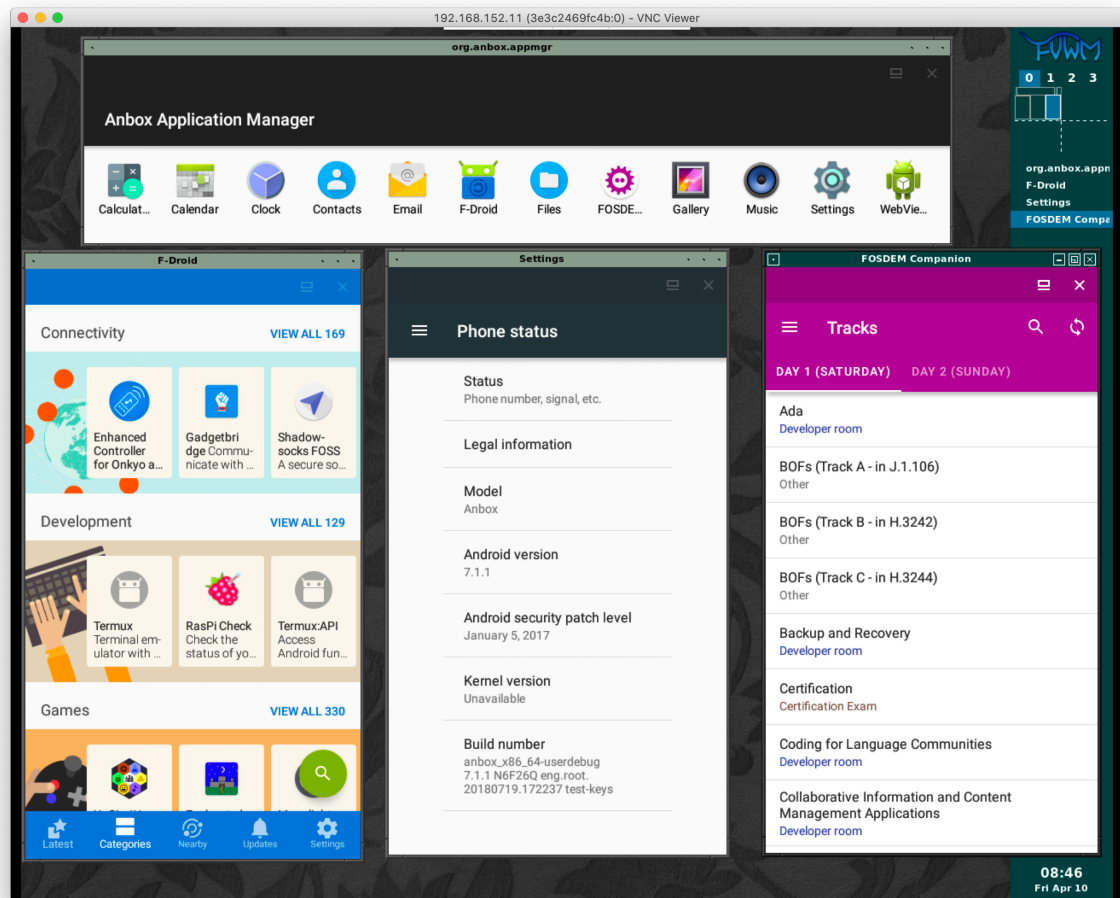
Non-goals

- Cloud gaming

Screenshots

```
1 $ docker ps -a
2 CONTAINER ID        IMAGE               COMMAND
   CREATED             STATUS              PORTS
   NAMES
3 950e3fa7d320        aind               "/docker-entrypoint..." 7
   minutes ago        Up 7 minutes       0.0.0.0:5900->5900/tcp    aind
4 $ docker exec aind ps -ef | tail -n 20
5 101023              323                138   0 11:18 pts/2    00:00:00 /system/bin/sdcard
   -u 1023 -g 1023 -m -w /data/media/emulated
```

6	110020	347	154	0	11:18	pts/2	00:00:00	com.android.
	systemui							
7	101001	397	154	0	11:18	pts/2	00:00:00	com.android.phone
8	user	403	154	0	11:18	pts/2	00:00:00	com.android.
	settings:CryptKeeper							
9	user	448	154	0	11:18	pts/2	00:00:00	com.android.
	settings							
10	110009	531	154	0	11:18	pts/2	00:00:00	android.ext.
	services							
11	110032	546	154	0	11:18	pts/2	00:00:00	com.android.
	clock							
12	110015	577	154	0	11:18	pts/2	00:00:00	com.android.
	provision							
13	110047	583	154	0	11:18	pts/2	00:00:00	com.android.smpush
14	110000	615	154	0	11:18	pts/2	00:00:00	org.anbox.appmgr
15	110011	642	154	0	11:18	pts/2	00:00:00	com.android.
	managedprovisioning							
16	110008	657	154	0	11:18	pts/2	00:00:00	android.process.
	media							
17	110003	675	154	0	11:18	pts/2	00:00:00	com.android.
	providers.calendar							
18	110002	694	154	0	11:18	pts/2	00:00:00	android.process.
	acore							
19	110027	744	154	0	11:18	pts/2	00:00:00	com.android.
	calendar							
20	110028	765	154	0	11:18	pts/2	00:00:00	com.android.camera2
21	110034	784	154	0	11:18	pts/2	00:00:00	com.android.email
22	110037	807	154	0	11:18	pts/2	00:00:00	com.android.
	gallery3d							
23	110013	822	154	0	11:18	pts/2	00:00:00	com.android.
	onetimeinitializer							
24	root	1003	0	0	11:25	?	00:00:00	ps -ef



Quick start

Tested on Ubuntu 19.10 (Kernel 5.3). May not work on other distros. If `modprobe ashmem_linux` or `modprobe binder_linux` fails, see <https://github.com/anbox/anbox-modules>.

```
1 sudo modprobe ashmem_linux
2 sudo modprobe binder_linux
```

Docker

VNC

```
1 docker run -td --name aind --privileged -p 5900:5900 -v /lib/modules:/lib/modules:ro ghcr.io/aind-containers/aind
2 docker exec aind cat /home/user/.vnc/passwdfile
```

NOTE: `--privileged` is required for nesting an Anbox (LXC) inside Docker. But you don't need to worry too much because Anbox launches "unprivileged" LXC using user namespaces. You can confirm that all Android processes are running as non-root users, by executing `docker exec aind ps -ef`.

Wait for 10-20 seconds until Android processes are shown up in `docker exec aind ps -ef`, and then connect to 5900 via `vncviewer`. The VNC password is stored in `/home/user/.vnc/passwdfile`. The password file can be also overridden by `docker run -v /your/own/passwdfile:/home/user/.vnc/passwdfile:ro`.

If the application manager doesn't shown up on the VNC screen, try `docker run ...` several times (FIXME). Also make sure to check `docker logs aind`.

Web mode (noVNC) To run the container with noVNC support, the environment variable `WEBMODE` can be set with the following command:

```
1 docker run -td --name aind --privileged -p 8080:8080 -e "WEBMODE=1" -v
  /lib/modules:/lib/modules:ro ghcr.io/aind-containers/aind
2 docker exec aind cat /home/user/.vnc/passwdfile
```

The container will be accessible via the browser at `http://localhost:8080/vnc.html`

Docker Compose

To bring the container up simply run

```
1 docker-compose up -d
```

the vnc password can be gotten with

```
1 docker-compose exec aind cat /home/user/.vnc/passwdfile
```

you can check how far it is with

```
1 docker-compose exec aind ps -ef
```

Kubernetes

```
1 kubectl apply -f kube/aind.yaml
2 kubectl port-forward service/aind 5900
```

The manifest contains the kernel module installer as `initContainers`.

The manifest is known to work on: - Google Kubernetes Engine (GKE) 1.16.8-gke.8 (ubuntu) [Apr 14, 2020] - Kubernetes 1.16.8, Ubuntu 18.04.4, Kernel 5.3.0-1012-gke, Docker 19.03.2 - n2-standard-8 - Google Kubernetes Engine (GKE) 1.16.8-gke.8 (ubuntu_containerd) [Apr 14, 2020] - Kubernetes 1.16.8, Ubuntu 18.04.4, Kernel 5.3.0-1012-gke, containerd 1.2.10 - n2-standard-8 - Azure Kubernetes Service (AKS) 1.17.3 [Apr 14, 2020] - Kubernetes 1.17.3, Ubuntu 16.04.6, Kernel 4.15.0-1071-azure, MS-Moby 3.0.10+azure - Standard DS2 v2 - kind 0.7.0 [Apr 14, 2020] - Kubernetes 1.17.0, Ubuntu 19.10, Kernel 5.3.0-46-generic, containerd 1.3.2 - **NOTE:** Requires `docker exec kind-control-plane mount -o remount,rw /sys`

Tips

Troubleshooting

- `docker logs aind`
- `docker exec -it aind systemctl status anbox-container-manager`
- `docker exec -it aind ps -ef`
- `docker exec -it aind cat /var/lib/anbox/logs/console.log`

adb

```
1 docker exec -it aind adb shell
```

To run adb on the host:

```
1 socat TCP-LISTEN:5037,reuseaddr,fork 'EXEC:docker exec -i aind "socat
  STDIO TCP-CONNECT:localhost:5037"' &
2 adb connect localhost:5037
3 adb shell
```

Apps

Pre-installed Apps

- Firefox
- F-Droid
- Misc accessories like Clock and Calculator

Installing apk packages

APK files mounted as `/apk.d/*.apk` are automatically installed on start up.

You can also use F-Droid. To use F-Droid, enable “Settings” -> “Security” -> “Allow installation of apps from unknown sources”.

FAQ

Isn't encrypting the phone with strong passcode enough for anti-theft? Why do we need aind?

People in the real world are likely to set weak passcode like “1234” (or finger pattern), because they want to open email/phone/twitter/maps/payment apps in just a few seconds.

aind is expected to be used in conjunction with encryption of the client device, and to be used only for sensitive apps, with a passcode that is stronger than the passcode of the client device itself.

TODOs

- Map different UID range per containers
- Better touch screen experience
- Redirect camera, notifications, ...

Similar projects

- Anbox: Desktop only and single-user/single-instance only. aind is built using Anbox.
- Android container in Chrome OS: ChromeOS/ChromiumOS-only
- Docker-Android: VM-based
- kubedroid: VM-based
- Anbox Cloud: Proprietary
- Intel Celadon in Container (CIC): Proprietary

License

- aind itself (e.g. Dockerfile, Kubernetes manifests, and start-up scripts) is licensed under the terms of the Apache License, Version 2.0.
- The Anbox patches (`./src/patches/anbox/*.patch`) are licensed under the terms of the GNU General Public License, Version 3, corresponding to Anbox itself.

Binary image

- The [ghcr.io/aio-libraries/aio-containers](https://github.com/aio-libraries/aio-containers) image on GitHub Container Registry (built from [./Dockerfile](#)) contains the binaries of several free software.
 - Anbox ([/usr/local/bin/anbox](#)): the GNU General Public License, Version 3
 - Firefox ([/apk-pre.d/fennec-*.apk](#)): the Mozilla Public License 2
 - F-Droid ([/apk-pre.d/FDroid.apk](#)): the GNU General Public License, Version 3
 - Android image ([/android.img](#), fetched from <https://build.anbox.io/>): see <https://source.android.com/s>
• For build instruction, see <https://github.com/anbox/anbox/blob/master/docs/build-android.md>
 - For other packages, see [/usr/share/doc/*/copyright](#).