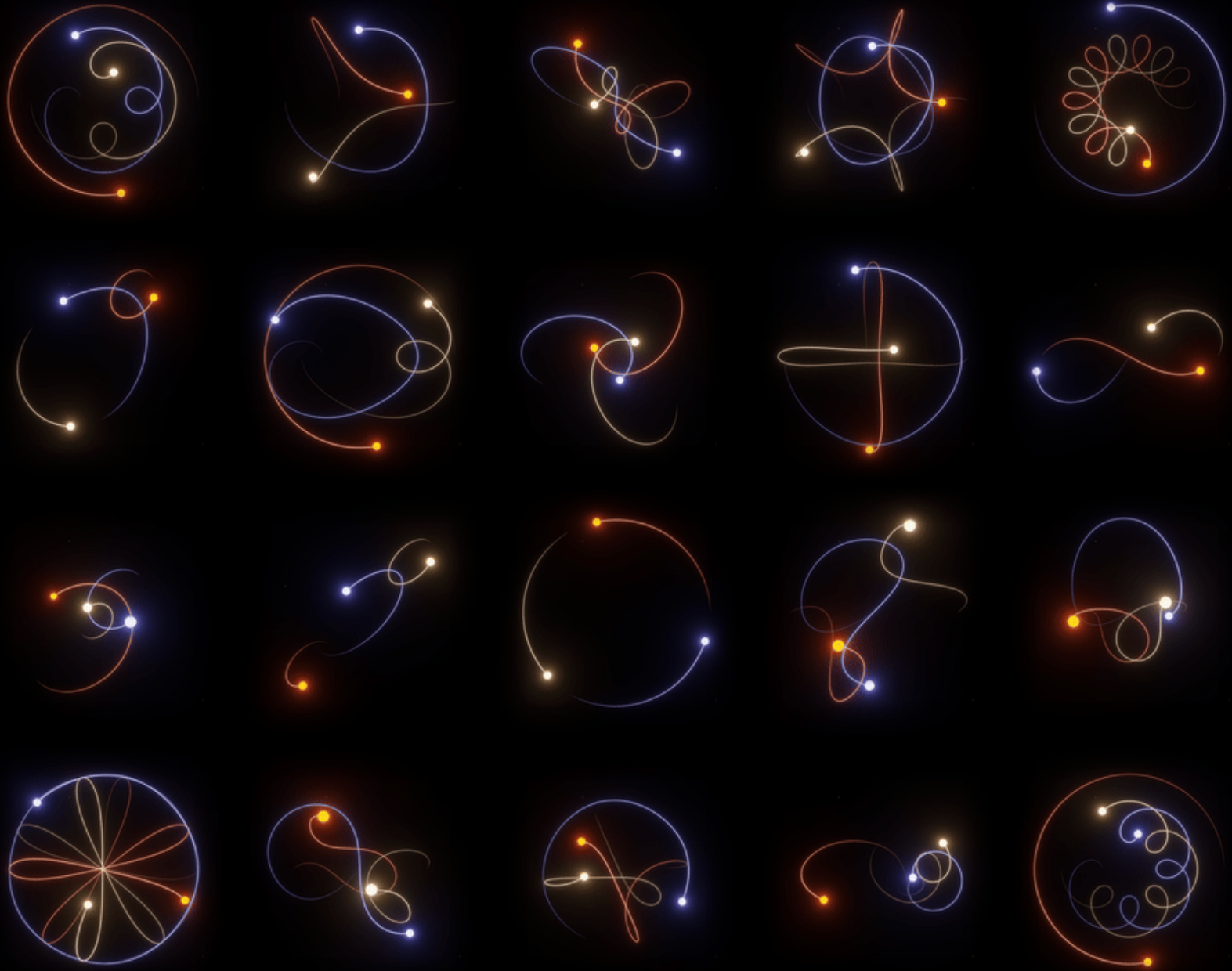


# The 3 AVD Containers Problem

Three Containers - Endless Possibilities  
Petr Ankudinov

Jan 2024



# Origins

- Arista AVD collection can be [installed manually](#)
  - Very feasible in many cases and not going anywhere
  - Do this at your own risk. You may encounter weird problems, especially if environment has some history
- [AVD all-in-one container](#)
  - Was never officially documented or advertised, but quite actively used anyway. Around 60K total downloads so far.
  - Can be used as dev container (with some modifications) or standalone
  - It was never integrated with AVD CI and must be manually updated on every release
  - A lot of complexity to maintain
  - No plans to support it long term
- [Ansible Automation Platform](#)
  - Out of scope. For customers heavily relying on RedHat support and internal Ansible ecosystem

# Motivation

- Better integration with AVD with automated image build on every release
- Must be **documented** and known to AVD users
- Reuse work done by Microsoft. It's not perfect for every use case, but quite a few developers are working on dev container features. Their contribution is appreciated and must not be wasted
- Better VSCode integration

**WARNING:** AVD dev containers are in the preview phase. They are working well, but breaking changes can happen and they must not be advertised to customers as fully supported solution.

# AVD Dev Containers

```
python:3.{9-11}-slim-bullseye
|
|         <--- trigger CI on every AVD release, pre-release, merge
|
L> base      :python3.{9-11}, latest
|
|   L> dev    :python{9-11}, latest
|   L> universal :python3.{9-11}-avd-devel, python3.{9-11}-avd-v4.5.0, latest
```

- Common use cases:
  - `base` - not to be used directly, base for all other images
  - `dev` - AVD contributors and testing new features/branches. AVD collection is not pre-installed
  - `universal` - AVD collection is pre-installed, ready to use
- All containers are multi-platform - linux/arm64, linux/amd64

# Demo 01: Universal Basic Use Case

- Add following `.devcontainer/devcontainer.json` to your inventory:

```
{  
  "image": "ghcr.io/aristanetworks/ansible-avd/universal:python3.11-avd-v4.5.0"  
}
```

- Use VSCode to open your inventory or another tool supporting dev containers. For ex., devcontainer CLI:

```
devcontainer up --workspace-folder /Users/pa/Documents/VSCode/github/avd-dev-container-toi/demo-01  
devcontainer exec --workspace-folder /Users/pa/Documents/VSCode/github/avd-dev-container-toi/demo-01 ansible --version  
devcontainer exec --workspace-folder /Users/pa/Documents/VSCode/github/avd-dev-container-toi/demo-01 ansible-galaxy collection list  
devcontainer open /Users/pa/Documents/VSCode/github/avd-dev-container-toi/demo-01
```

- The commands above are replaced with `make demo={demo-number}`
- Ready to go. Build some configs using AVD, etc.

## Demo 02: Fixing Deprecation Warnings

- Install `passlib` inside the container:

```
{
  "image": "ghcr.io/aristanetworks/ansible-avd/universal:python3.11-avd-v4.5.0",
  // fixing deprecation warnings
  "onCreateCommand": "pip3 install passlib",
  // you can also add some VSCode extensions
  "customizations": {
    "vscode": {
      "extensions": [
        // excalidraw, drawio and tldraw
        "pomdtr.excalidraw-editor",
        "hediet.vscode-drawio",
        "tldraw-org.tldraw-vscode"
      ]
    }
  }
}
```

- Check [dev container metadata reference](#) for more customization options

## Demo 03: Advanced Customization With Dockerfile

- You can define your own Dockerfile to customize the container

```
{
  "build": {
    "dockerfile": "Dockerfile",
    "args": {
      "_AR_FILE_PATH": "/support/download/EOS-USA/Active Releases/4.30/EOS-4.30.3M/ceOS-lab/ceOS-lab-4.30.3M.tar.xz",
      "_ARTOKEN": "${localEnv:ARTOKEN}"
    }
  },
  "onCreateCommand": "docker import /home/vscode/tmp/ceos-lab-temp ceos-lab:latest; rm -rf /home/vscode/tmp/"
}
```

```
FROM ghcr.io/aristanetworks/ansible-avd/universal:python3.11-avd-v4.5.0

ARG _ARTOKEN
ARG _AR_FILE_PATH

RUN bash -c "$(curl -sL https://get.containerlab.dev); \
  pip3 install passlib cookiecutter; \
  sudo apt-get update; sudo apt-get -y install xz-utils

# add ceos-lab image is _ARTOKEN is not empty
RUN if [ ! -z $_ARTOKEN ]; \
  then \
    ARTOKEN=$(echo -n "$_ARTOKEN" | base64) \
    && ARSESSION=$(curl -X "POST" "https://www.arista.com/custom_data/api/cvp/getSessionCode/" -H 'Content-Type: application/json' -d "{\"accessToken\": \"$_ARTOKEN\"}" | sed -n 's|.*"session_code": "([^\"]*)".*|\1|p') \
    && FILE_PATH=$_AR_FILE_PATH \
    && DOWNLOAD_URL=$(curl -X "POST" "https://www.arista.com/custom_data/api/cvp/getDownloadLink/" -H 'Content-Type: application/json' -d "{\"sessionCode\": \"$_ARSESSION\", \"filePath\": \"$_FILE_PATH\"}" | sed -n 's|.*"url": "([^\"]*)".*|\1|p') \
    && mkdir -p /home/vscode/tmp \
    && curl "$DOWNLOAD_URL" --output /home/vscode/tmp/ceos-lab-temp \
  ; fi
```

- This is usually helpful for advanced use cases, like installing cLab with images, etc.

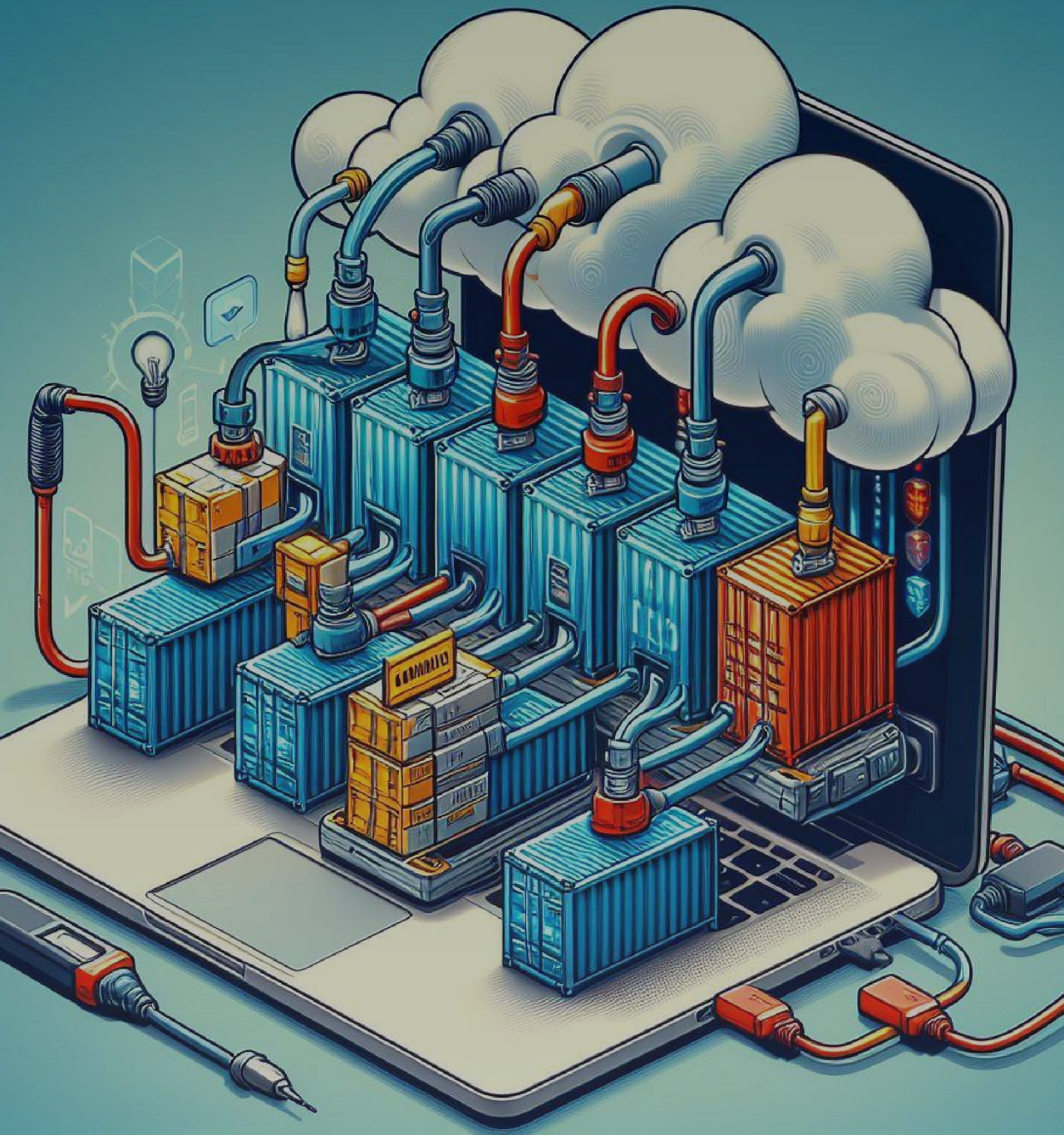
## Demo 04: Installing AVD Collection From Any Branch

- The `entrypoint.sh` script provide with AVD container can install collections automatically when container is created
  - If `AVD_GITHUB_REPO` and `AVD_BRANCH_NAME` env variables are defined, AVD collection will be installed from the specified Github repository
  - If env variables are not defined, AVD collection will be installed from the mounted repository (contributor mode)
- VSCode overrides container entrypoint. If `"onCreateCommand": "/bin/entrypoint.sh true"` is not defined - the container will start without any AVD installation
- An updated `devcontainer.json` example

```
{
  "image": "ghcr.io/aristanetworks/ansible-avd/dev:python3.11",
  "containerEnv": {
    "AVD_GITHUB_REPO": "aristanetworks/ansible-avd",
    "AVD_BRANCH_NAME": "devel"
  },
  // Run entrypoint script manually as it's ignored by dev container CLI otherwise.
  // The dev entrypoint is used to install ansible collections and requirements, as they are not included with the dev version.
  // "true" is required to exit "onCreateCommand" without entering ZSH.
  "onCreateCommand": "pip3 install passlib; /bin/entrypoint.sh true"
}
```

- In a real case scenario that will be someones fork and a different branch, as there is a pre-build `universal` image for AVD devel branch. It's only used for this demo as it's guaranteed to exist.





## Roadmap

- Testing and bugfixes
- Integrating with Github Codespaces for 1-click PR review
- Reviewing dev workflow and updating shortcuts to match container setup
- Testing and supporting standalone container use (with dev container supporting tools)
- Security scanning
- This TOI will be converted to a workshop and moved to [Netdevops Community](#) over time to support [AVD docs](#)

So far there is a single contributor for AVD containers. Help of any kind is very welcome!

Questions?

