Execution Environment Best Practices

CfgMgmtCamp 2024

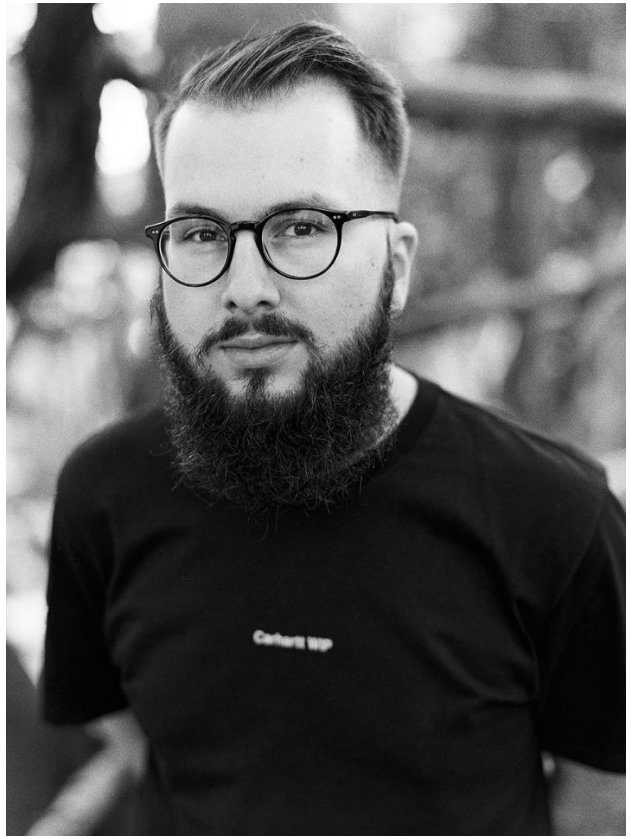
/ Agenda

- Introduction
- Execution Environment Basics
- Execution Environment Toolbelt
- Best Practices
- Automation
- Q&A

Introduction

SVA

/ Whoami



Niklas Werker (he/him)

- 🏢 SVA Systemvertrieb Alexander GmbH
- 📍 Cologne, Germany
- 👨‍💻 DevOps: Infrastructure Automation

- 3 years Automotive Engineering Background: Systems Engineering (Embedded Systems)
- 6 years IT Background

- nwerker @ 

/ About this talk & Call for action

The world is not black and white:

- Best Practices
- Good Practices
- Experiences from projects / scoping
- Food for thought

Call for action:

- GitHub Repository including this talk and Best Practices
- Contribute / Fork / PR / Discussion

/ Motivation

/ Why would you might want to use Ansible Execution Environments?

/ Are there any pitfalls in maintaining Execution Environments?

/ Why should we automate Execution Environment builds and how?

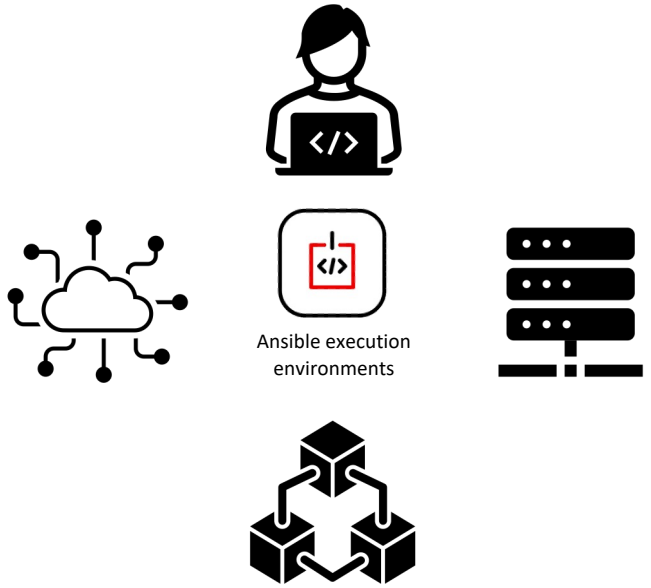
/ Lack of public community maintained Best Practices

Basics

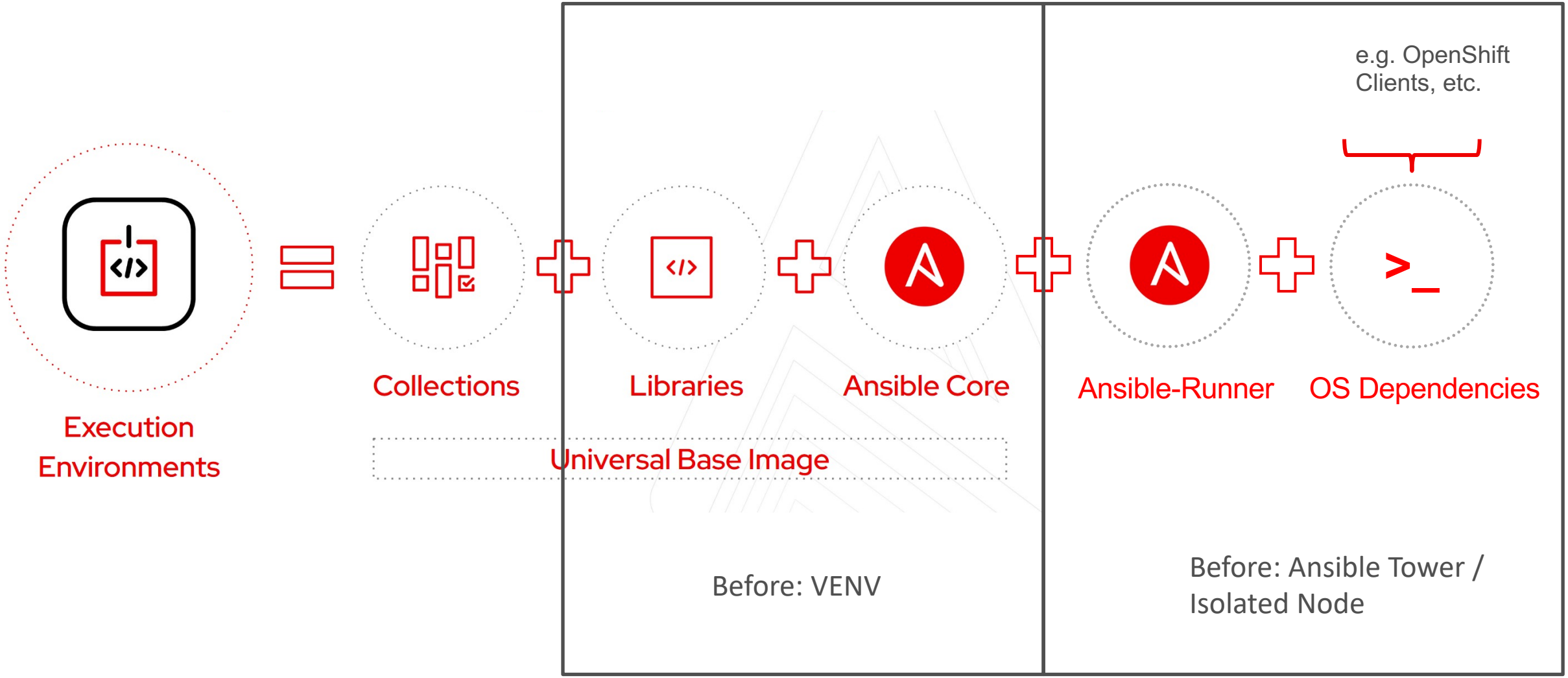
SVA

/ The Big Picture

- Ansible Runtime w/ needed dependencies independent of environment
  - AWX / Ansible Automation Platform
  - Notebook (it works on my machine)
  - Cloud
  - Edge
- OCI Compliant Image & OCI Compliant Runtime
- Dependency Management (Collection level metadata)
- Number of Tools to build / handle or run Execution Environments
  - ansible-builder
  - ansible-navigator
  - ansible-runner
  - OCI: podman / buildah or docker



/ Execution Environments



Execution Environment Toolbelt

SVA

/ Execution Environment Toolbelt

- Number of Tools to build / handle or run Execution Environments
  - ansible-builder
  - ansible-navigator
  - ansible-runner
- Dependencies:
  - Container Engine (podman / docker)
  - Python >= 3.8

/ Execution Environment Definition

execution-environment.yml

```yaml
---
version: 3

dependencies:
  ansible_core:
    package_pip: ansible-core==2.14.4
  ansible_runner:
    package_pip: ansible-runner
  galaxy: requirements.yml
  python:
    - six
    - psutil
  system: bindep.txt

images:
  base_image:
    name: docker.io/fedora:39

additional_build_steps:
  prepend_final: |
    RUN whoami
    RUN cat /etc/os-release
  append_final:
    - RUN echo This is a post-install command!
    - RUN ls -la /etc

```

requirements.yml

```yaml
---
collections:
  - name: azure.azcollection
    version: 1.4.0
  - name: ansible.windows
  - name: community.general
    version: 3.8.0
  - name: community.windows
  - name: awx.awx
    version: 19.0.0
  - name: t_systems_mms.icinga_director
  - name: netbox.netbox
  - name: ansible.netcommon
```

bindep.txt

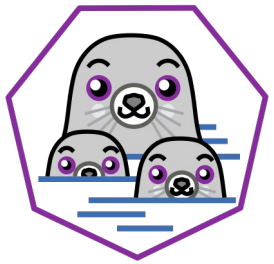
```
python3-lxml [(platform:redhat platform:base-py3)]
python-lxml [(platform:redhat platform:base-py2)]
openssh-server [platform:redhat]
openssh [platform:suse]
lolcat
```

/ Execution Environment Toolbelt

ansible-builder	ansible-navigator	ansible-runner	OCI
• preferred Tool for building EEs • Home of the execution environment definition	• swiss army knive • preferred tool for running automation and inspecting images	• Indented purpose: M2M communication • CLI capabilities shouldn't necessarily be used	• Podman / Buildah preferred • Docker: No feature parity with Podman out of the box

ANSIBLE BUILDER

ANSIBLE RUNNER

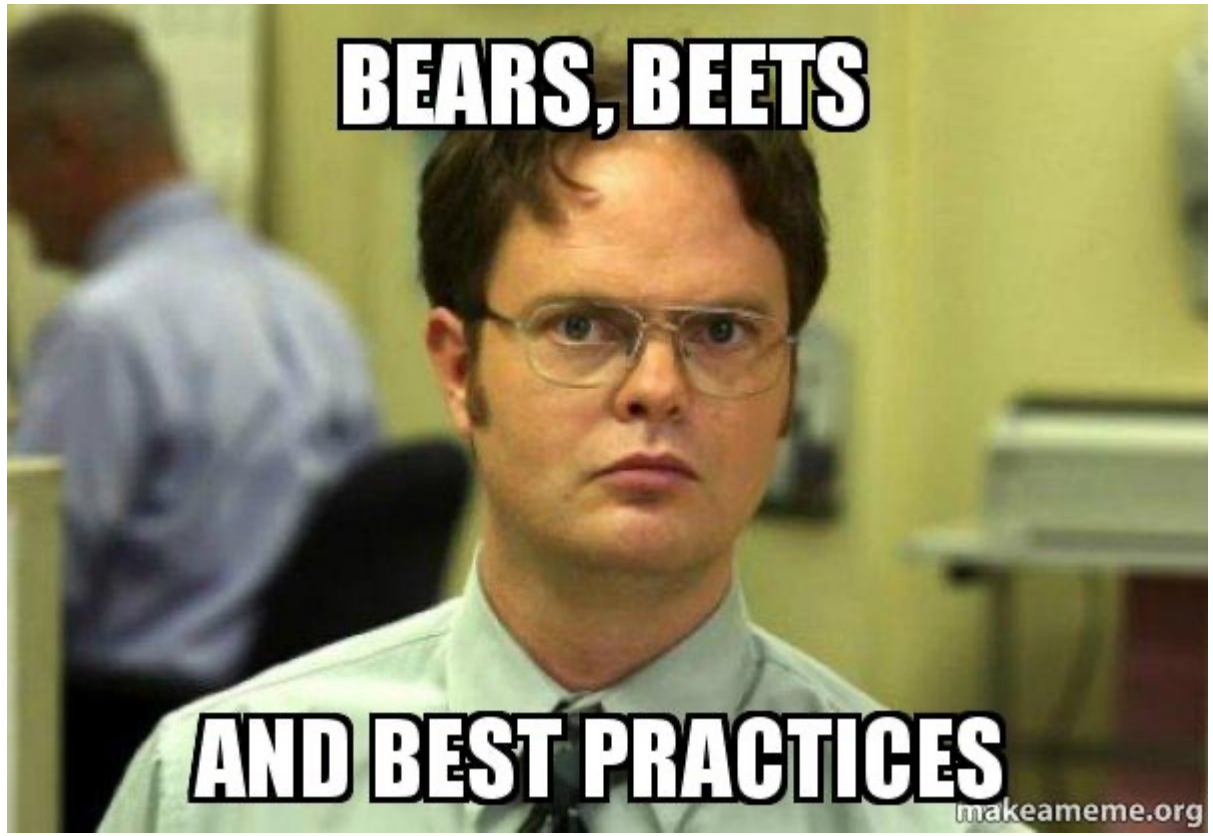


Best Practices

/ Best Practices

Categorized into:

- Execution Environment Best Practices
- Ansible Best Practices
- Container Best Practices
- Security Best Practices



/ Collection level Metadata

- Dependencies will be automatically installed, since defined on Collection level
- No dependency management and resolving conflicts by hand
- Listing Collections in requirements.yml should be enough
- Done via following files on Collection Level
  - meta/execution-environment.yml (if naming best practices are not followed)
  - requirements.txt
  - bindep.txt

- Community participation: Please open GitHub issues when Collections used by you have no Collection level Metadata yet

/ Base images

Base Image	Description	Ressources
ansible-runner	Default Base Image (deprecated)	Quay.io
awx-ee	Default Base Image AWX incl. common collections & dependencies	Quay.io GitHub Repository
ee-29-rhel8	Base image AAP, legacy Ansible 2.9 for migration purpose	Red Hat Registry
ee-minimal	Base image AAP, based on supporteted ansible-core Version	Red Hat Registry
ee-supported	Base image AAP based on ee-minimal incl. Red Hat supported Collections	Red Hat Registry
community-ee-minimal	Upstream of ee-minimal: latest or tagged ansible version	GitHub Registry GitHub Repository
community-ee-base	community-ee-minimal + ansible.posix, ansible.utils, ansible.windows	GitHub Registry GitHub Repository

/ Base images

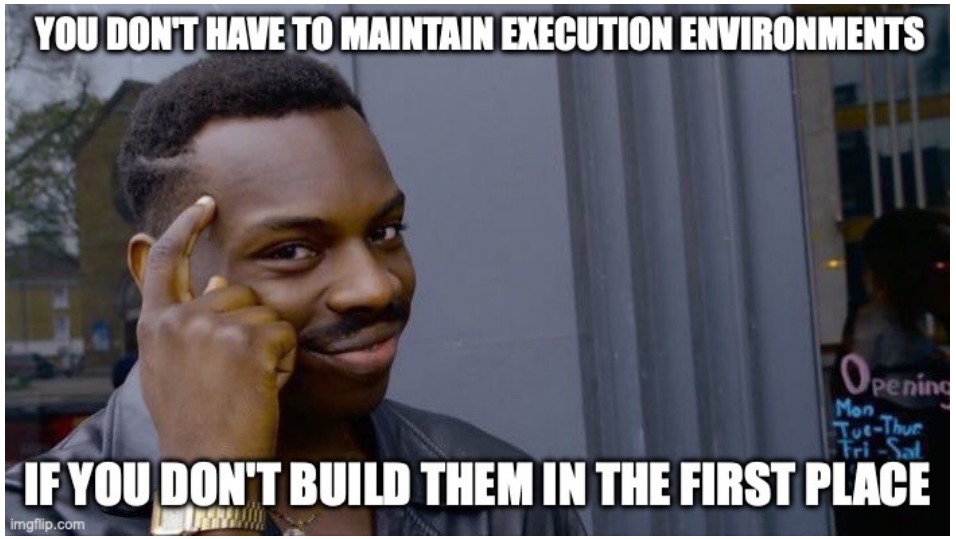
- Don‘t use the default ansible-runner image
  - deprecated / ansible-core 2.12 & CentOS 8 stream image
- Use Image according to your needs
  - awx-ee can be used as base image but shouldn‘t necessarily
  - Only: if there is a big overlap in the included collections and dependencies
  - Image size: 1.74 GB
    - fedora:39 in comparison: 264 MB
- To enterprise customers: use what you pay for (especially when it comes to support cases)
  - ansible-automation-platform-21 (ee-minimal / ee-supported etc.)
  - ansible-automation-platform-22
  - ansible-automation-platform-23
  - ansible-automation-platform-XX

/ Execution Environment definition

- Use default naming
- Evolution of EE defnition Spec:
  - v1:
    - All ansible-builder versions
  - v2:
    - ansible-builder version 1.2 upwards
    - Adds podman container image signing
  - v3:
    - Ansible-builder version 3.0 upwards
    - Allows for granular choice of ansible-core, ansible-runner & python version
    - Deprecates ansible-runner Base image
    - Adds build options
    - Fine grained additional build steps (8 instead of previously 2)

/ Execution Environment Best Practices

- Scope:
  - Highly dependent on use case, automated domain and user group
  - Maintain as little as possible as much as necessary
- Align release management und testing with your ansible workflow
- Use Collection overloading with care, maybe dependencies will break
  - Instead build a new Execution Environment
- No unnecessary mounting of paths to consume binaries
- No unnecessary delegation of tasks to evade EE
- Support:
  - Use of correct base images
  - Use of correct tool versions

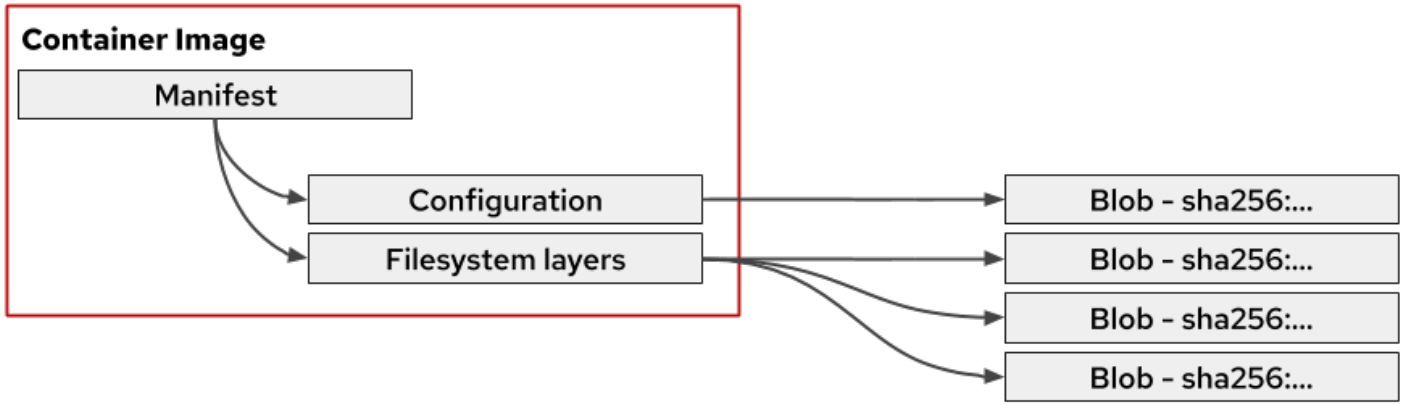


/ Ansible Best Practices

- Latest greatest collection versions, where appropriate:
  - Especially in public cloud collections a lot of breaking changes
  - New features and robustness
  - Avoid technical debt building up

/ Container Best Practices

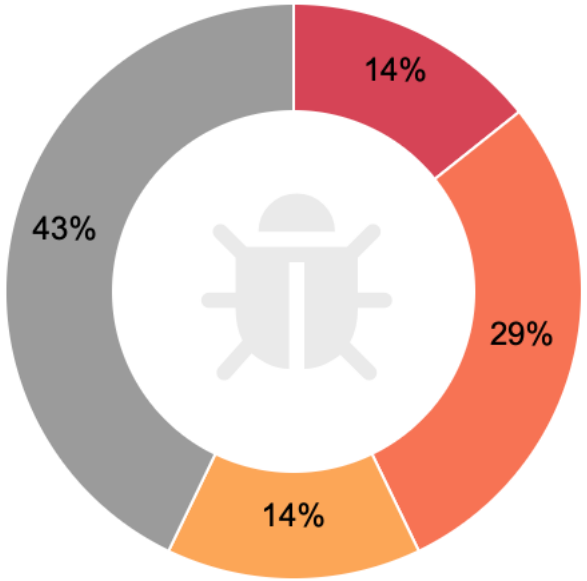
- Frequently (preferably automated) builds with updated base image
- Slim / storage friendly images
  - Avoid including unused assets into the image
  - Think in Layers and Blobs when it comes to “Additional Build Steps“
- Lifecycle: Regulary cleanups
  - Build environment
  - Dangling images on hosts
  - Unused images in registry
- Tagging
  - Semver x.v.z
  - Calver YYYYMMDD
- Multi-arch builds




```
additional_build_steps:
  append_final:
    - RUN this wont change that often && but this will change like all the time
    - RUN I'm just a useless build step for debugging reasons
```

/ Security Best Practices

- Container Best Practice: Rebuild images scheduled
  - Because of bug and security fixes in base image and dependencies
- Security Scans at build and at rest
  - CVEs
  - Secrets
  - Robustness
  - Tools: Trivy
  - Bult into container registries: Harbor, quay.io, GitLab
- Secure container registry
- Seperate build and production environment
- Reduce container privileges during runtime (podman rootles ftw)
- Sign images (e.g. with podman and ansible-builder >= 1.2.0)



Automation

SVA

/ Automation (CI/CD, etc.)

No „defacto“ standard. Use what fit's your usecase:

Ansible:

- redhat_cop.ee_utilities Collection

CI/CD:

- Automating execution environment image builds with GitHub Actions
- How to Build Ansible Execution Environments with OpenShift Pipelines
- Tekton Task
- DIY:
  - Best practice: ansible-builder creates build context, buildah builds the image

Q&A

/ Resources