ATIX

ATIX

The Linux & Open Source Company

Director's Cut

A new role for Ansible in Foreman

Jan Bundesmann, Thorben Denzer

Agenda

ATIX

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

ATIX Professional Services

Linux Platform Operations

Infrastructure Automation

Container Platforms & Cloud Solutions

DevOps

Cloud Native Solution

ATIX Product(s)

orcharhino

- Vendor independent datacenter management
- Patch management
- Configuration management
- Pizza

Cast

Thorben
Software Engineer @ ATIX AG

Jan
Software Quality Engineer @ ATIX AG

Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Every story has its beginning

Our premise: What if we could…

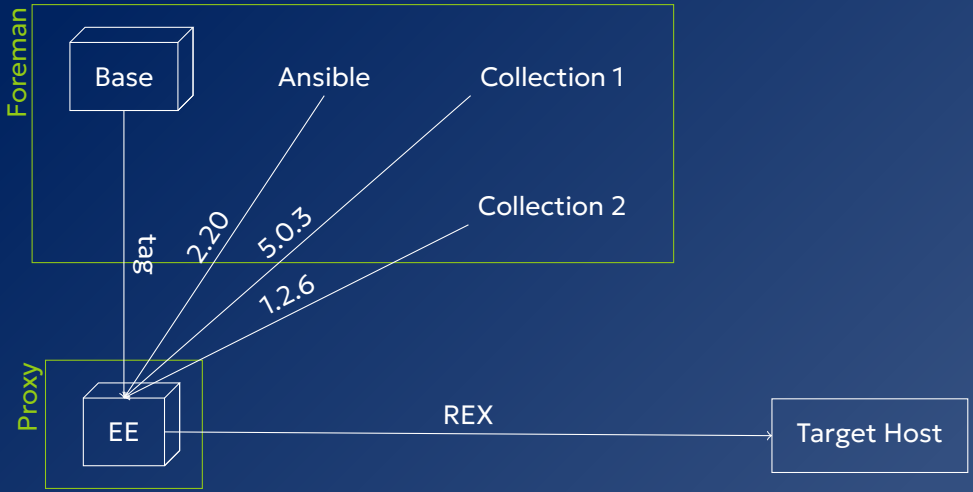
1. use a new Ansible version than the one installed on Foreman?
2. avoid managing Ansible content on the terminal?
3. manage different versions of Ansible content?
4. focus on collections instead of roles?

Audience wishes + consultants' opinions + engineers' concerns

Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Design of Ansible Director



Agenda

ATIX

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Motivation

- Execution Environments encapsulate Ansible and dependencies
- Re-usable and well-defined
- ⇒ Checks the first box of our premise!

Execution Environments

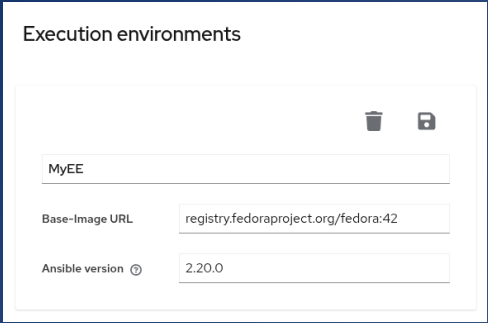
- Container image containing Ansible, content and dependencies
- Implications:
  - Host filesystem stays (relatively) untouched
  - Decouples Ansible version from the Smart Proxy Server
  - Content is isolated between runs
- `ansible-navigator` spawns a container from the execution environment image
- Ansible runs inside the container against selected hosts

Automated container builds in the background

- Foreman keeps track of Execution Environment inputs and
- automatically rebuilds a base-image if inputs change
- → Environments fully reproducible, content-wise
- Actual execution environment is built prior to Ansible run

Distribution und storage

- (Base-) Execution Environments are defined in Foreman
- Definitions are translated and built on the Smart Proxy Server
- Built images are stored in Katello (pulp_container) registry for re-usability

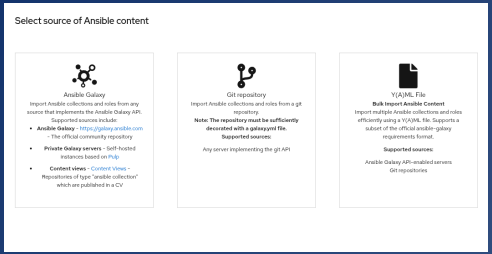


Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Overview

- Manage both Ansible Roles and Collections
- Multi-version support for Ansible content
- Add Ansible content to organizations
- Simplified Ansible content management:
  - Importing and deletion via web-UI
  - Import from Ansible Galaxy and Git

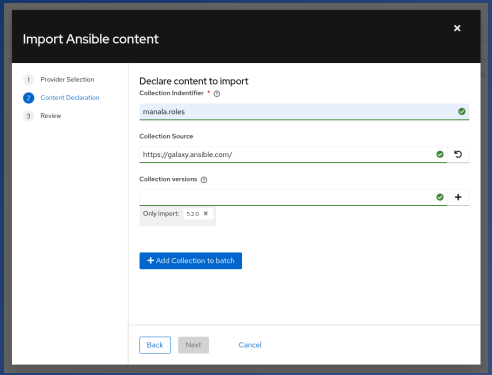


Ansible Galaxy

- Any galaxy-API compliant server
  - https://galaxy.ansible.com
  - Pulp with pulp_ansible
  - Katello Products

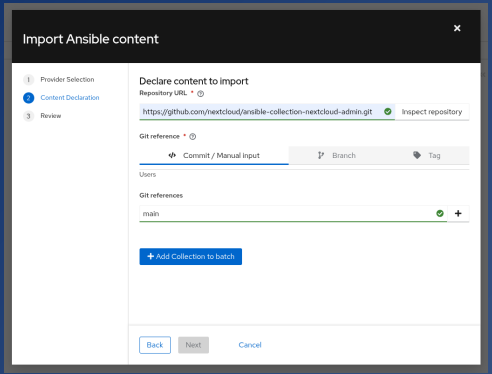
Side note:

- Granular importing only possible for collections
- Roles are downloaded completely



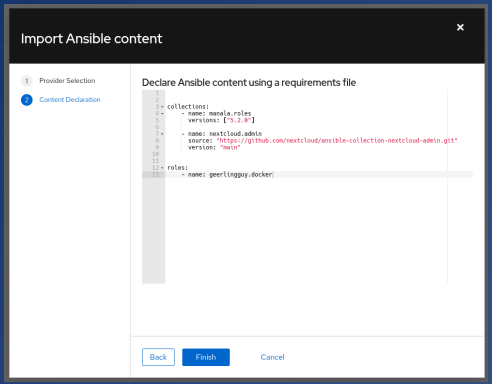
Git

- Any (public) git-repository as source
- Collections must have necessary metadata
- Any valid git reference may be used
  - “Dynamic” references (branches/pulls) are treated differently



YAML to combine those

- Editor in UI
  - Functionality exposed via API endpoint
- Subset of requirements.yaml; source can be
  - Galaxy
  - Git repository
  - Versions can be declared
    - Ignored for Roles
    - Multiple Versions allowed for collections



Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Similar to Katello environments, yet fully decoupled

- Library: Default meta-environment containing everything
- LCEs bundle a subset of content + Execution Environment
- Paths define “flow” of content
- First environment in path can manage arbitrary content
- Later environments receive content by promotion
- Content is “snapshotted” on promotion
  - Snapshots…
    - …are atomic and immutable
    - …use hashes to identify differences
    - …are created and destroyed automatically
- Special treatment of dynamic git references:
  - Dynamic references can be synced before use
  - References are pinned to latest commit on promotion

Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Demo

Agenda

1. Prequel
2. Storyboard
   - Execution Environments
   - Content Management
   - Lifecycle Management
3. Demo
4. Critical Review

Testing

1. Unit tests for “services”
   - Running locally
   - Planned
     - GitHub workflows
     - `.gitlab-ci.yml` publicly available
2. Installability tests
3. Systems tests via robottelo
   - API / CLI / UI functionality
   - End-to-end tests how to act on target hosts

Comparison

Criterion	foreman_ansible	foreman_ansible_director
Maturity	well established	newcomer
Requirements	foreman	katello, container runtime
Ansible content	roles	roles, collections
Content management	manually	automatic
Versions	as Foreman Proxy	as you like
Organizations	same content	distinct content
Lifecycles	in theory	inherent feature

Achievements and future objectives

- Add authentication / secret management
- Playbooks
- Address hosts in multiple subnets
- Increase test coverage
- Support for offline installations
- (Rollbacks)
- Production ready approx. Q1 2026

Production

- 16k lines of code
  - 9000 lines of TypeScript
  - 7000 lines of Ruby
- Planned end of 2024; In dev since 2025-01-01
- ~40 DynFlow actions
- 2 Repositories
  - https://github.com/ATIX-AG/foreman_ansible_director/
  - https://github.com/ATIX-AG/smart_proxy_ansible_director/