MAINTAINING OVER 40 ANSIBLE MODULES: 3 4 YEARS LATER
$ WHOAMI

Evgeni Golov
Senior Software Engineer at Red Hat
ex-Consultant at Red Hat
Debian Developer

♥ FOSS ♥

♥ automation ♥
FOREMAN + ANSIBLE = ♥

- Foreman has an API
- Everyone loves writing YAML instead of clicking in a GUI
- So we wrote modules, rewrote them again, refactored them and stuffed them into a collection
- This is the story of our journey
PRELUDE:
MOTIVATION / WTF
WHAT'S FOREMAN?

- lifecycle management tool for physical and virtual servers
- power management, provisioning, configuration
- Bare-Metal, VMware, RHV, OpenStack, GCE, Azure, etc
- huge plugin ecosystem (Katello, Monitoring, Ansible, ...)

5 / 38
WHAT'S ANSIBLE?

- "radically simple IT automation engine"
- huge number of modules for various usecases
- writing own modules is very easy
- integrates well with REST APIs
WHY AUTOMATING FOREMAN WITH ANSIBLE?

- We have daily tasks in our environment
- WebUI and hammer don't scale well
- Using Ansible as a declarative API client
CHAPTER 1: ANSIBLE/ANSIBLE
FOREMAN AND KATELLO
MODULES IN
ANSIBLE/ANSIBLE

• Ansible upstream since 2.3 (2016)
• one module to rule them all, thus cumbersome to use
• uses the (Satellite specific) nailgun library
• mostly Katello oriented
FOREMAN AND KATELLO
MODULES IN
ANSIBLE/ANSIBLE

- Turns out one maintainer for code in ansible/ansible is not enough
- Didn't have tests until 2018
- Deprecated since 2.8
- To be removed in 2.12
CHAPTER 2: A NEW REPOSITORY
FOREMAN-ANSIBLE-MODULES.GIT

- Started in June 2017
- A new repository under @theforeman organization
- Goal: central place for collaboration around Ansible modules for Foreman
- First step: split foreman and katello into "one module per entity"
  - started with 6 modules
- Centralized module_utils: July 2017
CHAPTER 3: TESTS
CHAPTER 3.1: TEST PLAYBOOKS

- First set of tests added in November 2017
- Playbooks that would use the modules against a live server
- Good start, but expensive test execution
- Doesn't play well with Travis CI and friends
CHAPTER 3.2: VCR BASED TESTS

- VCR (vcrpy) is a great way to record and replay HTTP requests/responses
- Allows recording "good" API interactions and replay them on Travis
- Added January 2018
- Ensured modules work on Python 2.7 + 3.5
- First PlaybookCLI, now ansible-runner
- Full coverage: August 2019
CHAPTER 3.3: CHECK MODE TESTS

- All our modules support check mode
- We re-run the VCR based tests with `--check`
CHAPTER 3.4: SANITY TESTS

- Ansible provides ansible-test for in-tree modules
- Since Ansible 2.9 it can also handle Collections
- We run ansible-test sanity --venv plugins/ across all supported Pythons
CHAPTER 3.6: EXPECTED CHANGE TESTS

- Our test playbooks execute every task twice
- The first execution is expected to have changed=True
- The second changed=False
- This ensures the modules are idempotent
CHAPTER 3.7: DIFF MODE TESTS

- Our modules return before/after diff data to Ansible
- We access that data in our test playbooks and analyze the content
CHAPTER 4: DOCUMENTATION
CHAPTER 4.1: BUILDING DOCUMENTATION

- All modules have DOCUMENTATION populated
- We use build-ansible.py document-plugins with a customized template
- Ansible internal, our use of it breaks sometimes
  - Would be great to have official tooling
  - Automatic builds on Galaxy?
- Need to figure out how to autopublish docs
CHAPTER 4.2: DOCUMENTATION FRAGMENTS

- Ansible 2.8 introduced documentation fragments
- We use them heavily to document common parameters (credentials etc)
- Fragments for return values would be cool
CHAPTER 5:
FOREMAN
ANSIBLE
MODULE
CHAPTER 5.1: FOREMANANSIBLEMODULE

- ForemanAnsibleModule is a sub-class of AnsibleModule
- Simplified definition of common parameters in argument_spec
- Import error handling
- Entity create/update/delete/compare helpers
- before/after diff handling
CHAPTER 5.2: FOREMANENTITY..., KATELLOANSIBLEMODULE, ...

- Further sub-classing useful
- ForemanEntityAnsibleModule adds a state parameter
- KatelloAnsibleModule makes organization required
CHAPTER 6: LIBRARIES
CHAPTER 6.1: NAILGUN

• We started with the nailgun library
• Originally developed by Satellite QE
• Targeted at Satellite environments
  ▪ no support for non-Satellite plugins
  ▪ released at the same cadence as Satellite
• Designed to test the Satellite API
CHAPTER 6.2: APYPIE

- nailgun was fine when we targeted Satellite environments
- Katello (and Foreman) were moving quicker
- Decided to write an own API library
  - using the published apidoc.json, thus mostly version agnostic
- Switching libraries was rather easy due to the abstraction we've built
  - And tests, tests will save you!
CHAPTER 7: USE THE FORCE
CHAPTER 7.1: USE THE FORCE OF THE ARGUMENT_SPEC

• Ansible supports complex (nested) argument specs
  ▪ `elements='dict',
    options=dict(...)
  ▪ Allows better checking of complex parameters
We always had a need to map from Ansible param names to Foreman API parameters.

This resulted in the introduction of the entity_spec argument_spec extended with Foreman specific data.

The plain argument_spec can be generated from it.
Many modules perform simple CRUD operations:
- take user input
- find matching entity
- create/update/delete based on input
- report

We used to have write code for that, now this is generated from the entity_spec
CHAPTER 8: COMMUNITY
CHAPTER 8: COMMUNITY

- Originally started as "my team needs this"
- Quickly gained contributions from ATIX
- Today: 35 contributors, many from Red Hat and ATIX
- Developers, Consultants, Ops, Customers
- Adding their usecases and features
CHAPTER 8: COMMUNITY

• Initial contribution was hard, duplicated code, hard to test
• Increased contribution when we moved to a more centralized codebase
• Having a collection and RPMs made consumption easier
• Recording VCR test results is still the biggest blocker
CHAPTER 9: OUTLOOK
CHAPTER 9: OUTLOOK

- `foreman_host` supporting *ALL* the parameters
- official roles to support central workflows
- *more* modules
- documentation autopublishing and versioning
- easier contribution
- collection defaults like `module defaults groups`?
THANKS!

✉️ evgeni@golov.de
🌐 die-welt.net
🐦 @zhenech
✉️ @zhenech@chaos.social
👥 @evgeni