ATIX

The Linux & Open Source Company

Scaling Foreman Compute Provisioning with OpenTofu

A Generic, Extensible Approach to Host Provisioning

Manisha Singhal, Markus Bucher

Agenda

ATIX

1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

Foreman Compute Resources Today

- Foreman supports many platforms through dedicated plugins:
  - vSphere
  - Proxmox
  - AWS
  - [oVirt]
  - etc.

The Real Problem

ATIX

- Integrating new Platforms is hard
- Every new Platform requires its own Foreman Plugin & Fog Plugin
- Each plugin duplicates logic for provisioning and lifecycle
- Maintenance Overhead grows with platform count

Agenda

ATIX

1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

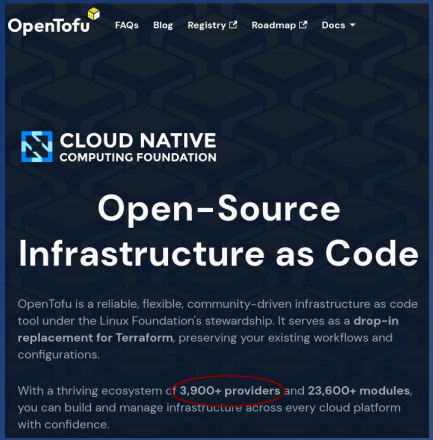
What ‘Another Plugin’ Costs

ATIX

- Foreman knowledge required
- Testing and maintenance
- Verify against real-life environment
- Small platforms rarely justify the effort

Meanwhile…

- Heterogeneous Infrastructure
- ‘Digital Sovereignity’ push Platforms
- many Providers exist
- OpenTofu already handles diverse Platforms declaratively



Agenda

ATIX

1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

Design Goal

ATIX

- Keep Foreman as the control plane
- Delegate provisioning to a generic backend
- Unified lifecycle
- Provider-agnostic
- Declarative
- Extensible without ‘Coding’

Why OpenTofu

ATIX

- Declarative provisioning model
- Mature ecosystem of providers
- Well-understood lifecycle (plan/apply/destroy)
- Open and extensible

Design Parameters

- Single plugin for multiple providers
- Use OpenTofu to provision hosts
- Foreman remains authority for
  - UI
  - API
  - Lifecycle

What This Plugin Is (and Is Not)

It is:

- Generic compute provisioning layer
- Extensible via configuration

It is **not**:

- Replacement for existing plugins

Traditional vs OpenTofu Model

Traditional:

- One plugin per platform
- Duplicated Logic
- High (Dev-)Cost

OpenTofu:

- One Plugin
- Multiple Providers
- Logic written once

Agenda

ATIX

1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

High-Level Architecture

- Foreman Core:
  - UI/API/lifecycle
- OpenTofu Plugin:
  - Host → OpenTofu map
  - Provisioning abstraction
- OpenTofu Engine:
  - Plan/Apply/State
- Providers:
  - Nutanix, Hetzner, others

Provisioning Flow

1. Host defined in Foreman
2. Plugin maps host → OpenTofu resource
3. OpenTofu plan & apply executed
4. Provider creates resource
5. Foreman tracks host

State & Lifecycle Ownership

- OpenTofu manages infrastructure state
- Foreman manages host lifecycle
- Clear separation avoids duplicate state machines

Extensibility Model

- New compute platform = new OpenTofu provider
- No Foreman code changes required
- Resource types defined via configuration

Agenda

ATIX

1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

Demo Overview

- Configure OpenTofu provider (via Compute Resource)
- Define host provisioning via Foreman
- OpenTofu Script
- Configuration extension

Live Demo

ATIX

Demo time

What This Enables

- One Foreman plugin to maintain
- Same provisioning workflow across platforms
- Faster onboarding of new compute providers
- Foreman scales with infrastructure diversity

Agenda

ATIX

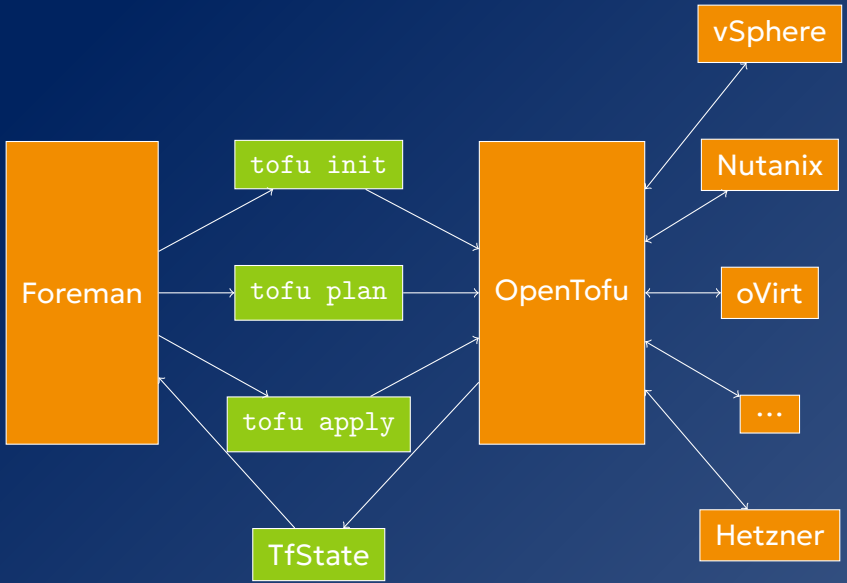
1. Foreman's current compute model
2. 'Issues' with the current model
3. Introducing the Foreman OpenTofu Plugin
4. Architecture & Extensibility
5. 'Live' Demo
6. Lessons learned & Roadmap

Challenges

ATIX

- Mapping declarative resources to hosts
- UX boundaries Foreman/OpenTofu
- Error handling & visibility
- TfState storage & storing information

Workflow visualized



Trade-offs

- Less Configuration Options in UI?
- OpenTofu Knowledge required?
- Redundant state → divergence possible
- Not all Providers map cleanly to Foreman Hosts
- OpenTofu (Provider-)Limitations
  - Power State
  - Parameter changes might result in Host reset do not offer Parameter or show Warning

Roadmap

- More Provider-Types (e.g. Hetzner)
- Better state visibility (ideally integrated in Host-Details/-Edit)
- Simplify adding new Provider (1 file per Provider)
- Feature parity with traditional Compute Resource Plugins
  - Image based Deployment (if applicable)
  - Snapshots
- React UI

Key Takeaways

- Plugin-per-Platform doesn’t scale
- OpenTofu is a generic backend
- Foreman remains Orchestration Control Plane
- ‘Digital Sovereignity’ might push certain Provider Types
- →
  - OpenTofu builds it
  - Ansible configures it
  - Foreman orchestrates it

Thanks

- Questions!?
- Contributions Welcome

ATIX-AG/foreman_opentofu