Red Hat Summit

Connect

Automatisierung mit Disziplin

Policy Enforcement in Ansible Automation Platform meistern

2025-11-19

Darmstadt, Germany

Dr. Jason Breitweg

Senior Technical Account Manager
Red Hat

json@redhat.com

What are we going to talk about today?

- Why policy enforcement and why now?
- Our Policy as Code vision
- Policy enforcement in AAP
- Examples and demo
- What I'd like to see
- The future
- Resources
- Q & (hopefully) A

Ensure control, predictability and confidence

Enforce policies before automation runs

Operational control

Only execute automation during a specified maintenance window. Prohibit automation during peak workloads (e.g., Black Friday)

Control CloudOps

Control which automation and who can change, manage or update your cloud operations

Security

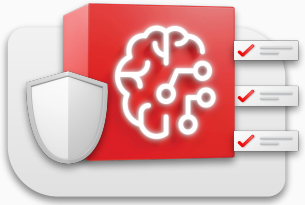
Validate variables in a workflow template to enforce criteria. E.g. Stop automation during an incident investigation window.

AIOps

Validate extra vars applied to AI-generated automation to ensure they meet specified criteria before automation execution.

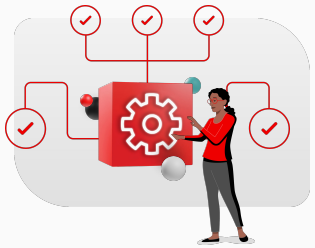
Policy enforcement is key to AIOps and AI infrastructure management

Standardize AI infrastructure operations



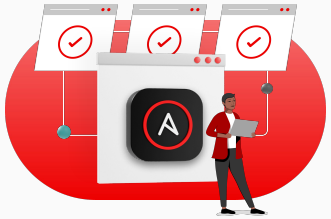
- Simplify AI infrastructure deployment, usage, and scaling.
- Provide lifecycle management for consistency and repeatability.
- Accelerate time to value for developers consuming the AI platform.

Enable AIOps



- Identify and address potential issues proactively.
- Resolve common issues and alerts via actionable advice from generative AI.
- Provide automated remediation, reducing MTTR.

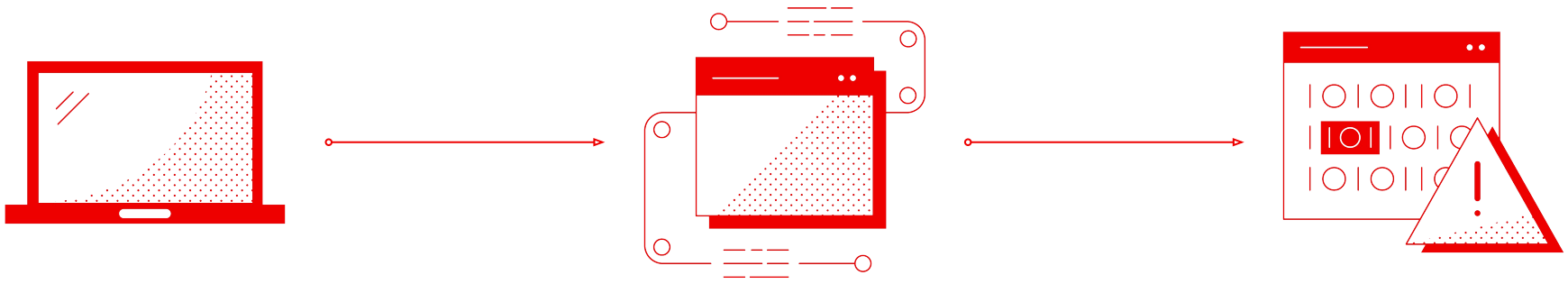
Enforce policy



- Implement safeguards around your AI solutions for better outcomes and control.
- Ensure AI is aligned to key internal and compliance policies.

Foundation of automation → Self-healing infrastructure → Policy checks →

The automated Policy as Code vision



Create

During automation discovery, creation or development

Manage

Using Ansible as a centralized entry point for automation before and during automation

Scale

Using Ansible as a centralized point for all post automation policy logging, reporting and auditing

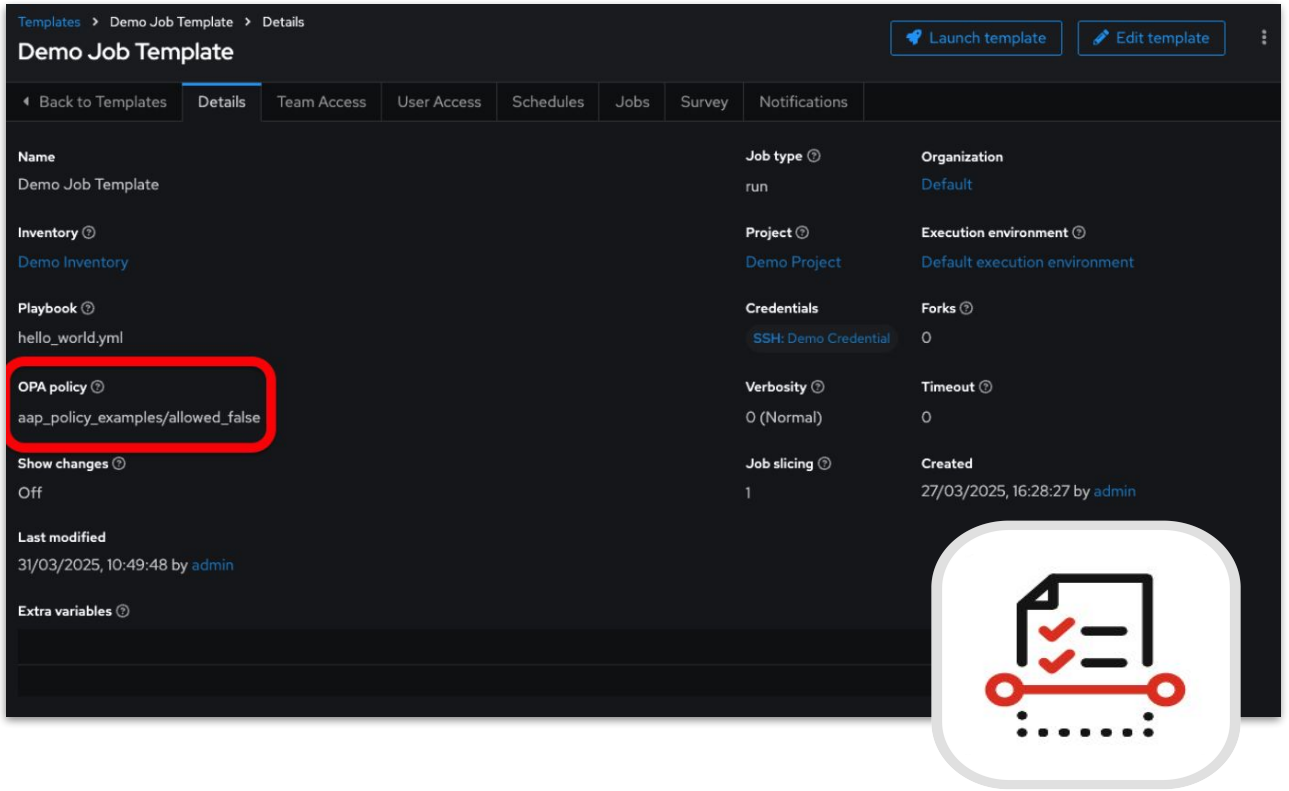
New Policy enforcement

New to Ansible Automation Platform

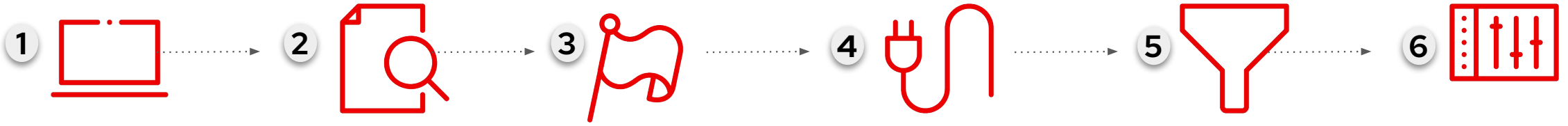
Introducing the policy enforcement feature

New to Red Hat Ansible Automation Platform

- **Enforce operational policies in automation in the job template, at the inventory level and/or at the Ansible organization level *before* automation is allowed to run**
- Allows (or stops) automation based on whether the criteria in the policy are met (or not)
- Policies can be set at the organization, inventory or job template level
- Work with Open Policy Agent (OPA) based policies where you will specify which variables and which values to validate before automation runs



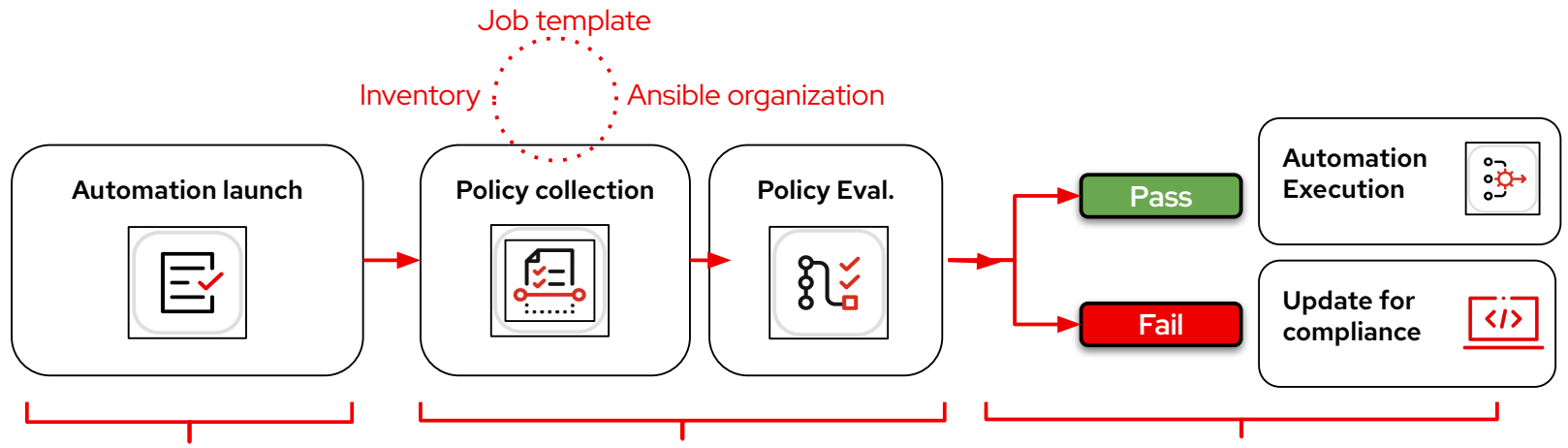
Getting started with policy enforcement



1. Set up Open Policy Agent server
   OPA docs
2. Write OPA policies or start with these
   examples
3. Install Ansible Automation Platform 2.5+
4. Set up policy settings in Ansible to connect OPA server
5. Add OPA policy link to desired enforcement point (job template, inventory, Ansible org) or start with
   examples
6. Policy checks run at job launch. Watch for job success and failures in Ansible controller

Policy enforcement at automation runtime

How it works



Automation job is initiated in a job template.

Automation is validated against specified policies stored externally and applied to job template, inventory or Ansible org.

If it passes, automation executes.

If it fails, automation must be updated for compliance before it successfully runs.

Decision maker (a human) controls what policies are written and where they will be applied (inventory, org or job template)

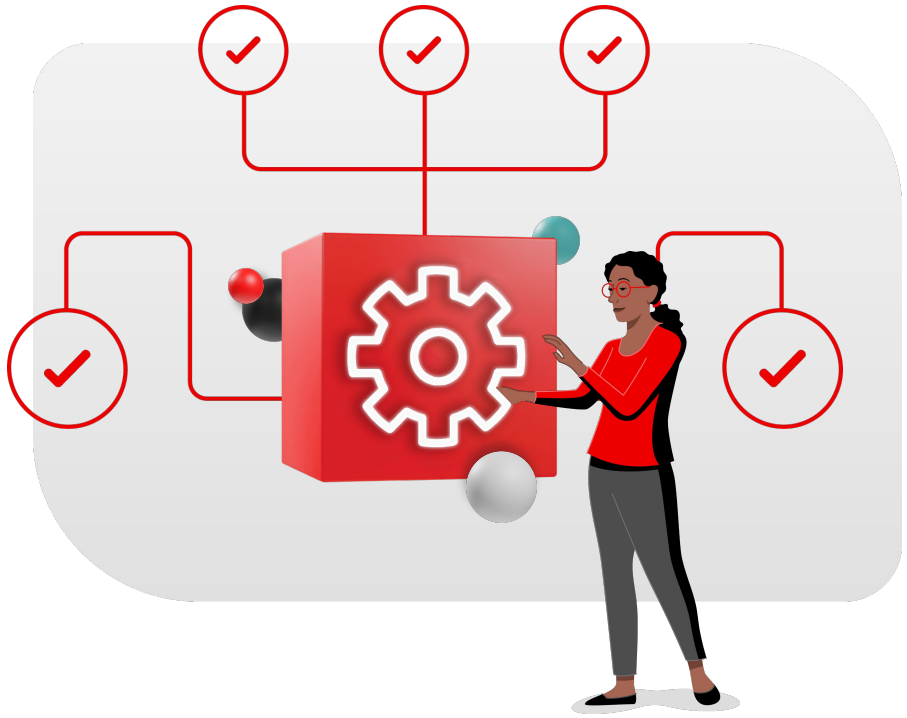
Get started

- ✓ **Write** external policies to be enforced.
- ✓ **Apply policies** at the job template, Ansible organization or inventory levels
- ✓ **Run** policies and watch for pass / fail results in Ansible controller.

Policy enforcement examples

What can you do with policy enforcement today?

Control *where* a policy applies	Regulate when automation runs
Control *which variables* can be passed into automation	*Prevent mismatches* between credentials and inventory
Manage who automates which inventory	*Enforce* naming conventions



Enforce policies around extra vars

Consistently control what automation is allowed and not allowed with less effort

Let's take an example.

Variables and extra vars make playbooks more flexible and reusable.

Common extra vars may include:

- Environment variables (dev, test, prod, staging, etc.)
- Inventory variables
- User variables
- Configuration data

And more ...

Here are the extra vars from a playbook:

```
{
  "extra_vars": {
    "environment": "prod"
  },
  "created_by": {
    "teams": [
      { "name": "dev_team"
    ]
  }
}
```

Enforce policies around extra vars

Set your policy to include extra vars you wish to control

Set and enforce policies around what is allowed and not allowed when it comes to extra vars.

Notice the policy is written in Rego and is stored in a external policy server.

This policy says that the **dev_team can only automate on the dev and staging environments** and prod_team can only automate on the staging or production environments.

You write a policy that details who can operate on which environment. :

```
package aap_policy_examples

import rego.v1

# Define allowed values for specific keys in extra_vars based on teams
valid_extra_var_values_by_team := {
        "dev_team": {"environment": ["dev", "staging"]},
        "prod_team": {"environment": ["prod", "staging"]},
}

# Default response allowing extra_vars unless violations occur
default team_based_extra_vars_restriction := {
        "allowed": true,
        "violations": [],
}
```

.
.
.

Full policy can be viewed on GitHub

Enforce policies around extra vars

Gain control the easy way

Add your policy path to job templates, inventories or Ansible orgs.

Policy enforcement checks automation against the policy before it is allowed to run.

A dev user with this playbook, will get the message written in the job output because there is a mismatch between team and allowed environments.

Full example policy here.

Dev_team requests automation with these extra vars:

```
{
  "extra_vars": {
    "environment": "prod"
  },
  "created_by": {
    "teams": [
      { "name": "dev_team" }
    ]
  }
}
```

```
{
  "allowed": false,
  "violations": [
    "extra_vars contain disallowed values for keys: [\"environment\"]. Allowed extra_vars for your teams ([\"dev_tea
  ]
}
```

Dev_team is denied the ability to run automation on the production environment.

Policy input data

Full list available here

- id
- name
- created
- created_by
- credentials
- execution_environment
- extra_vars
- forks
- hosts_count
- instance_group
- inventory
- job_template
- job_type
- job_type_name
- labels
- launch_type
- limit
- launched_by
- organization
- project
- scm_branch
- scm_revision
- workflow_job
- workflow_job_template

Demo

Improvements

Polishing policy enforcement

I'd like to see...

- Policy hub
- RBAC
- Multiple policy capabilities
- Policy discovery
- Policy management

What's coming...

- Policy enforcement during the development lifecycle
- Reporting mechanisms to support compliance status and audit tracking

Resources

Become a policy enforcement master

Red Hat

- Implementing policy enforcement docs
- Automated policy as code with Red Hat Ansible Automation Platform overview
- Automated Policy-as-Code. Start Small. Think Big. blog article
- Online walkthrough
- Policy enforcement with Ansible Automation Platform video

OPA

- Open Policy Agent
- Learn the policy language
- Rego Playground
- OPA online courses

Examples

- https://github.com/ansible/example-opa-policy-for-aap
- https://gitlab.com/ansible-hetzner/opa_policies

Questions?

Jetzt Session bewerten!

Einfach QR-Code scannen, Session aus der Liste wählen und bewerten. **Vielen Dank!**

red.ht/rhsc-darmstadt-feedback

Red Hat Summit

Connect

Thank you


linkedin.com/company/red-hat


facebook.com/redhatinc


youtube.com/user/RedHatVideos


twitter.com/RedHat