

# How to use this deck

## Name:

Ansible Automation Platform - Technical Deck

## Purpose:

These additional slides are for crafting technical conversations with a customer after an account executive has established a sales pitch.

**It is not recommended to present the complete deck** but to select slides based on the audience. The presentation length should be between **45 - 60 minutes**.

These can be focused on individual platform components, persona workflows, or used for deep dives.

## Last updated:

9 August 2023 (check history for older versions)

## What this deck is for?

Technical level conversation with customers covering Ansible Automation Platform 2.

## What this deck is not for?

Business level discussions, or to be sent to customers without a discussion.

## Google Slides source link (Red Hat internal):

[https://docs.google.com/presentation/d/1sa\\_O12EIRG-fdChArYJf9HZe6wKDfqnIAiH3\\_udL39s](https://docs.google.com/presentation/d/1sa_O12EIRG-fdChArYJf9HZe6wKDfqnIAiH3_udL39s)

## Feedback / Suggestions

Please send all feedback and suggestions to [ansible-feedback@redhat.com](mailto:ansible-feedback@redhat.com). Use the slide deck name as the reference.

## Owner:

Ansible MBU, [ansible-pmm-tmm@redhat.com](mailto:ansible-pmm-tmm@redhat.com)

## Ansible Deck Finder:

<https://ansible.github.io/slides/>

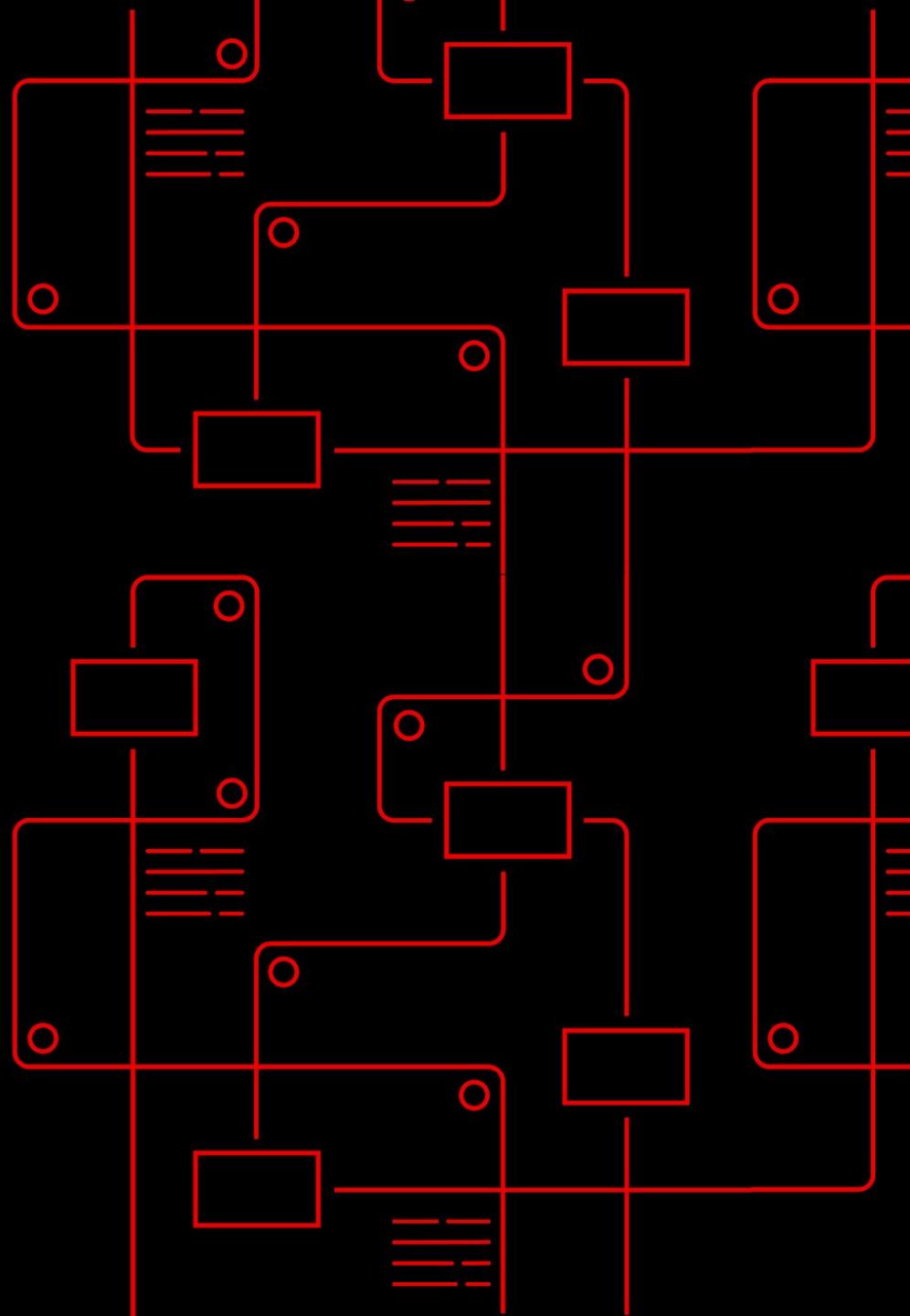
Note: the back of the deck contains a short appendix with playbook example slides adjusted for better color balance for visual accessibility purposes.



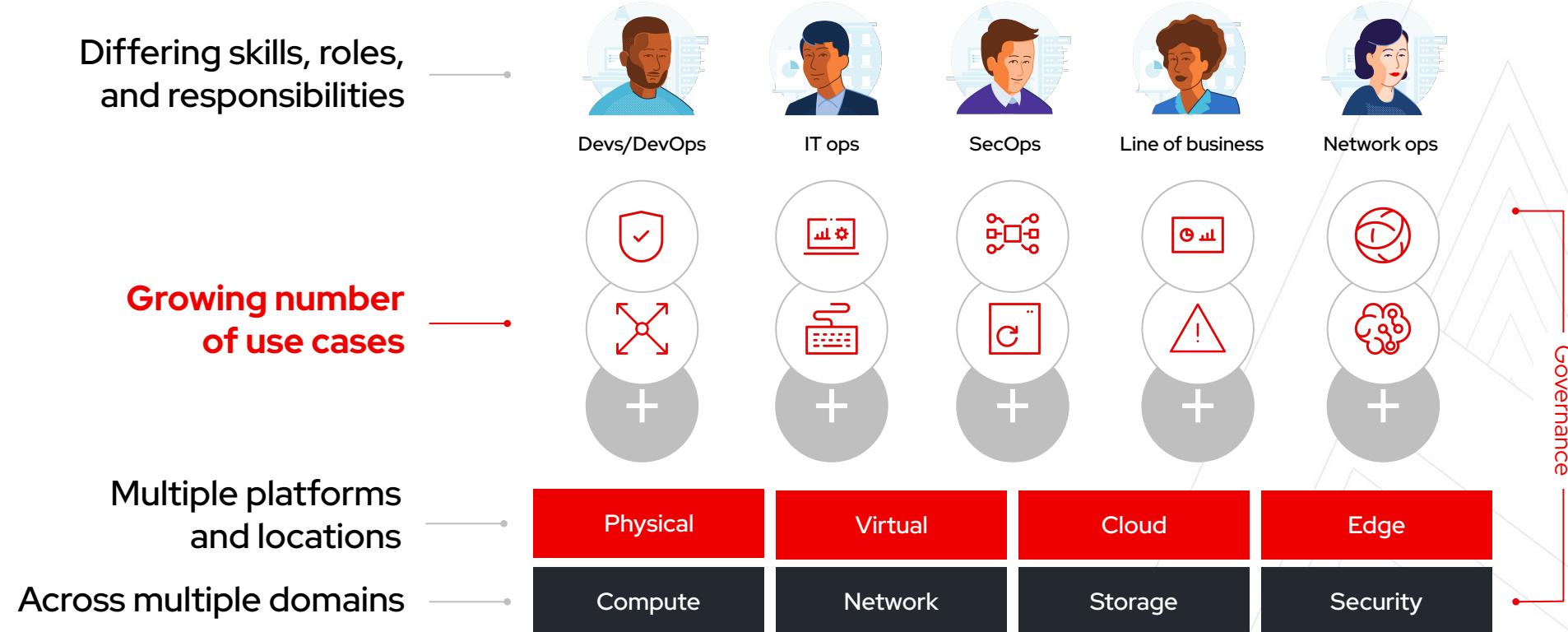
# Red Hat Ansible Automation Workshop



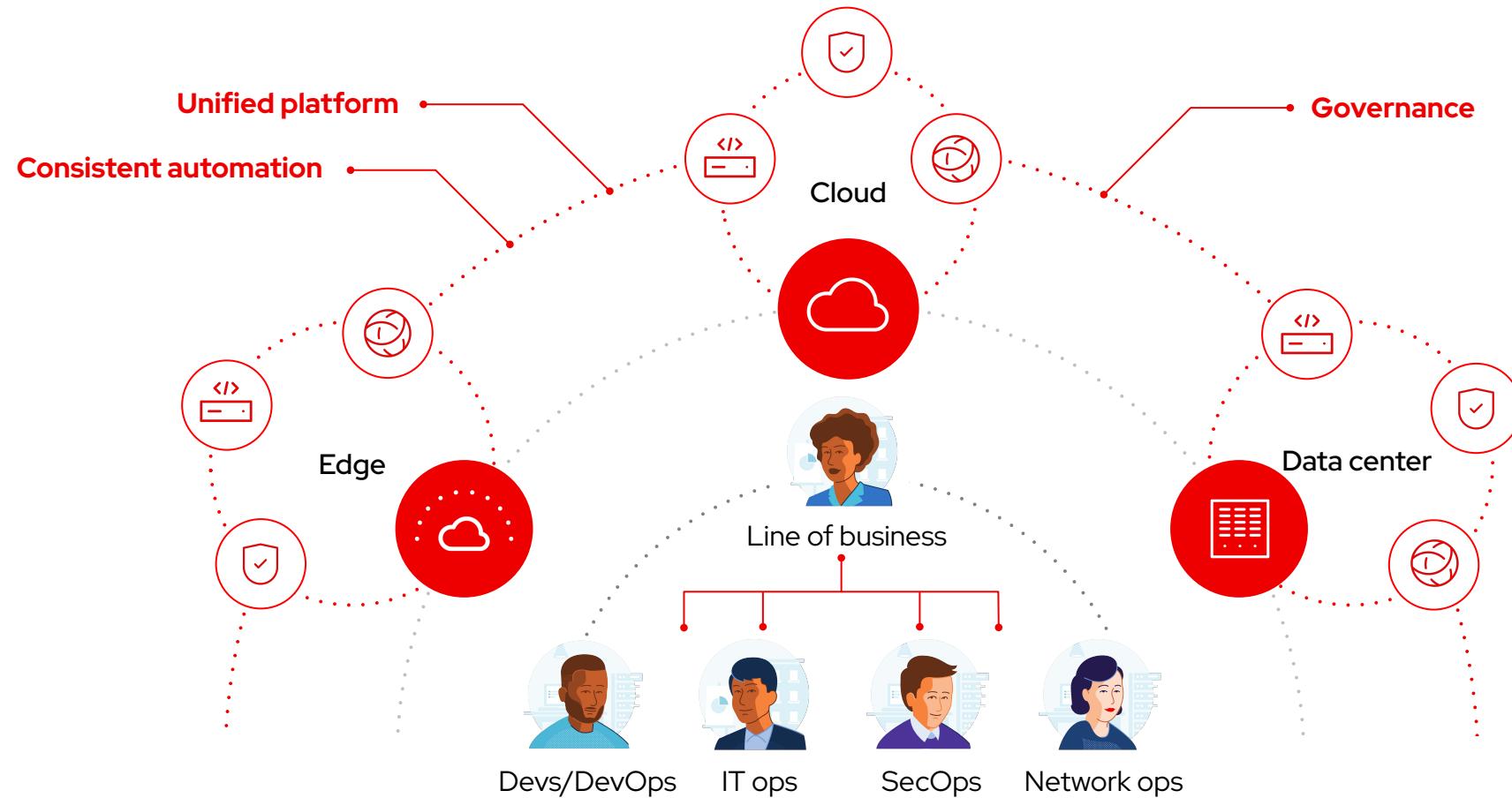
**Anyone can automate...  
but an enterprise needs to  
coordinate and scale**



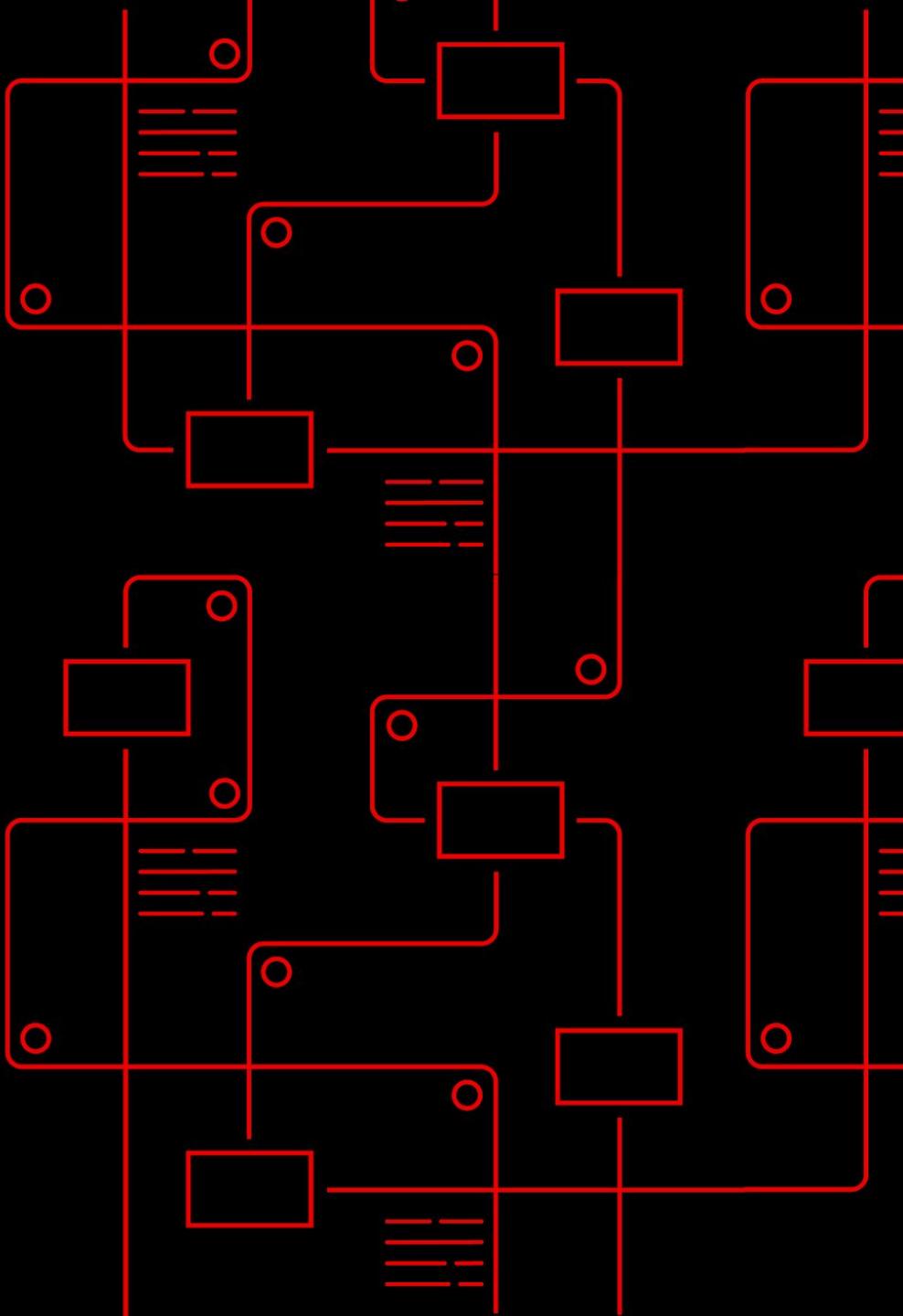
# Many organizations share the same challenge.



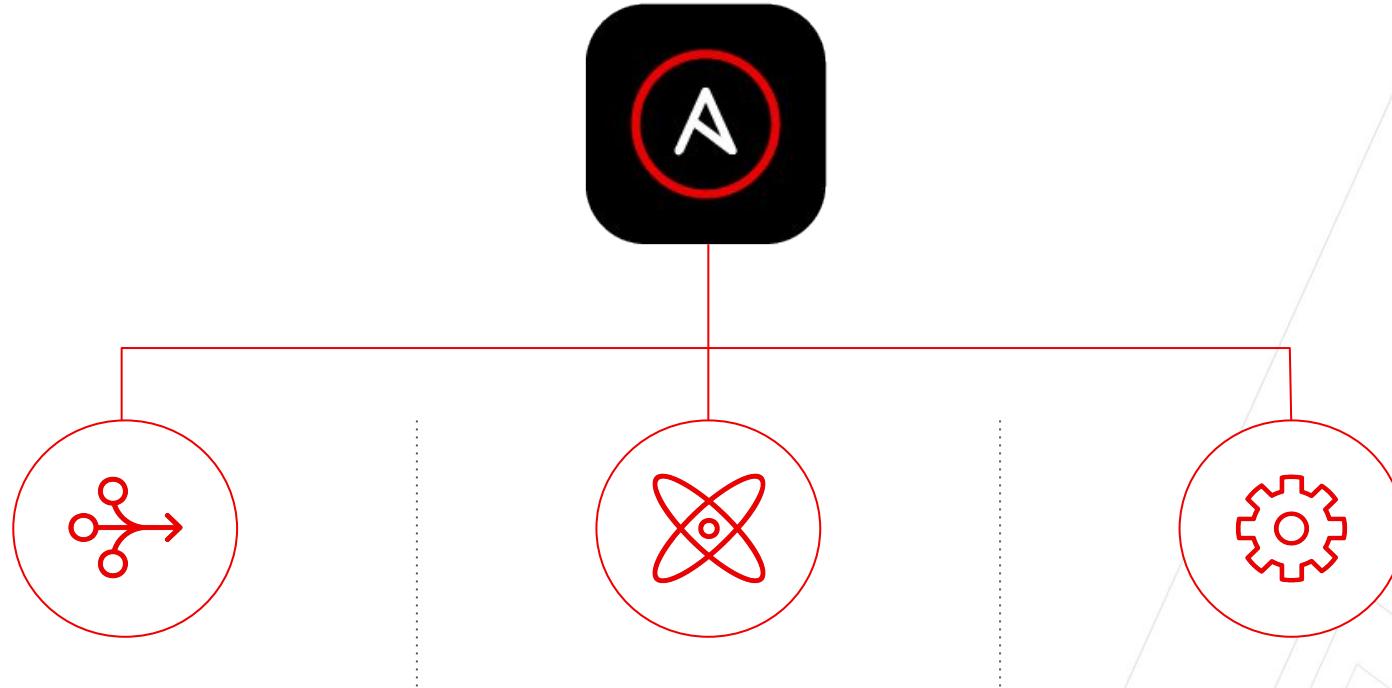
# The solution? Break down the silos.



# Why Red Hat® Ansible® Automation Platform?



# Ansible is the **de facto** automation language.



**Simple**  
Human-readable language  
with quick adoption

**Powerful**  
Universal language spanning  
multiple IT domains

**Agentless**  
Easily integrate with  
hybrid environments

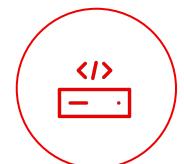
## Supported and certified **content you can trust.**

**150+**

Certified Content  
Collections

**55+**

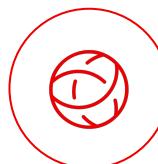
Certified technology  
partners



Infrastructure



Cloud



Network



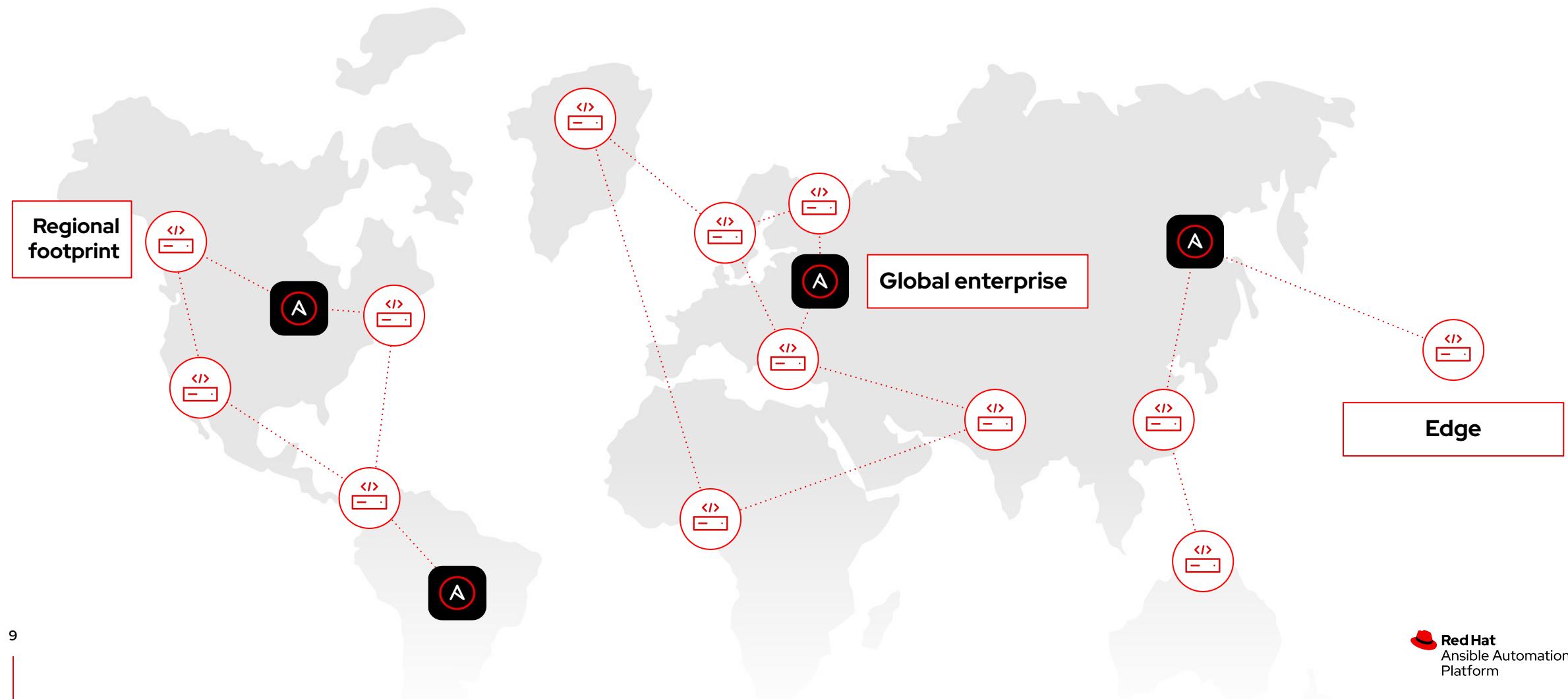
Security



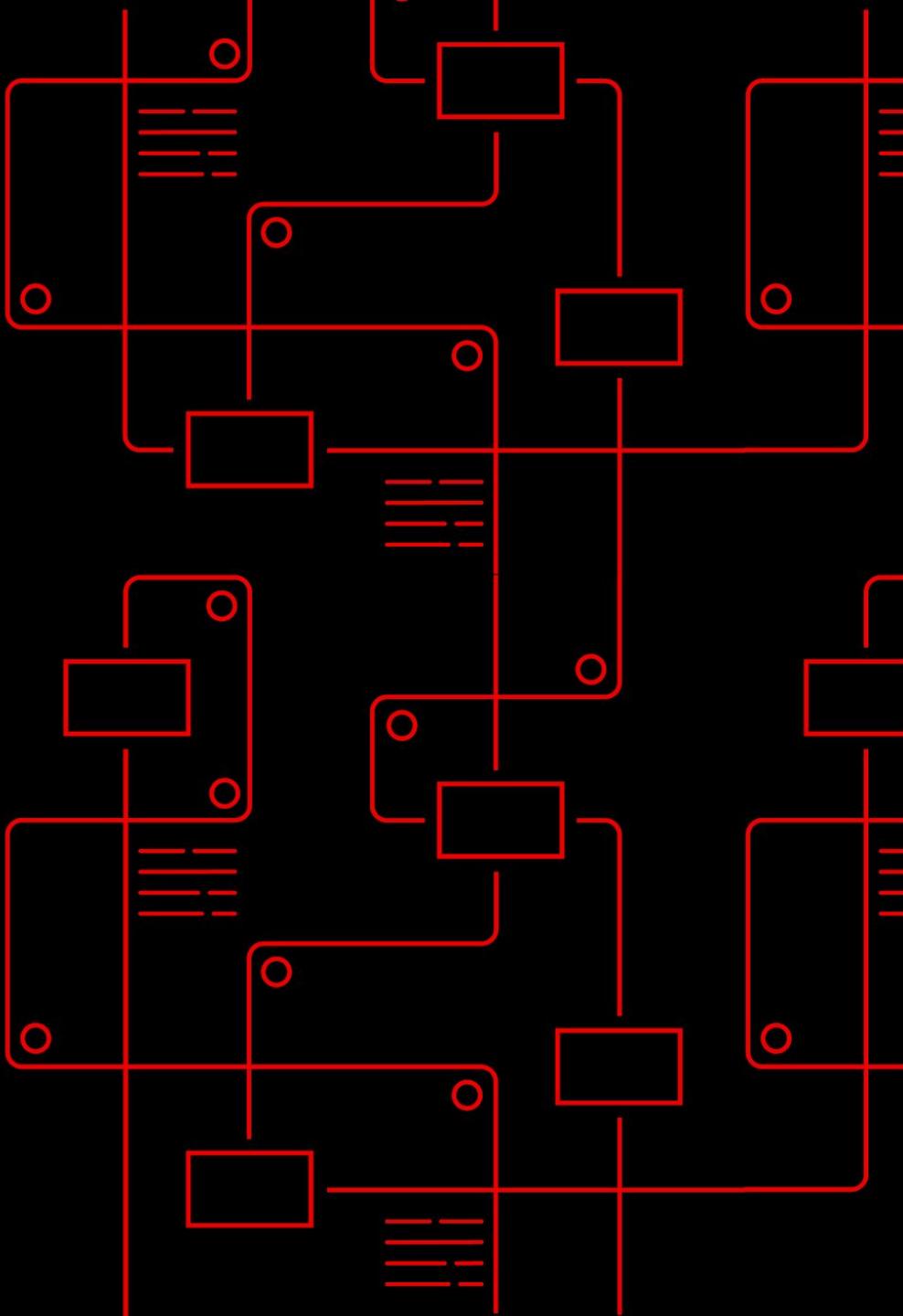
Edge



# The flexibility to scale, wherever that may be.



# What is Red Hat® Ansible® Automation Platform?



# A platform for the **entire automation team**.

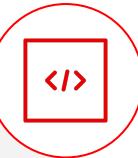


## Content creation

Content creation tooling

Portable distribution and reliable execution

Large ecosystem of certified automation



## Operations

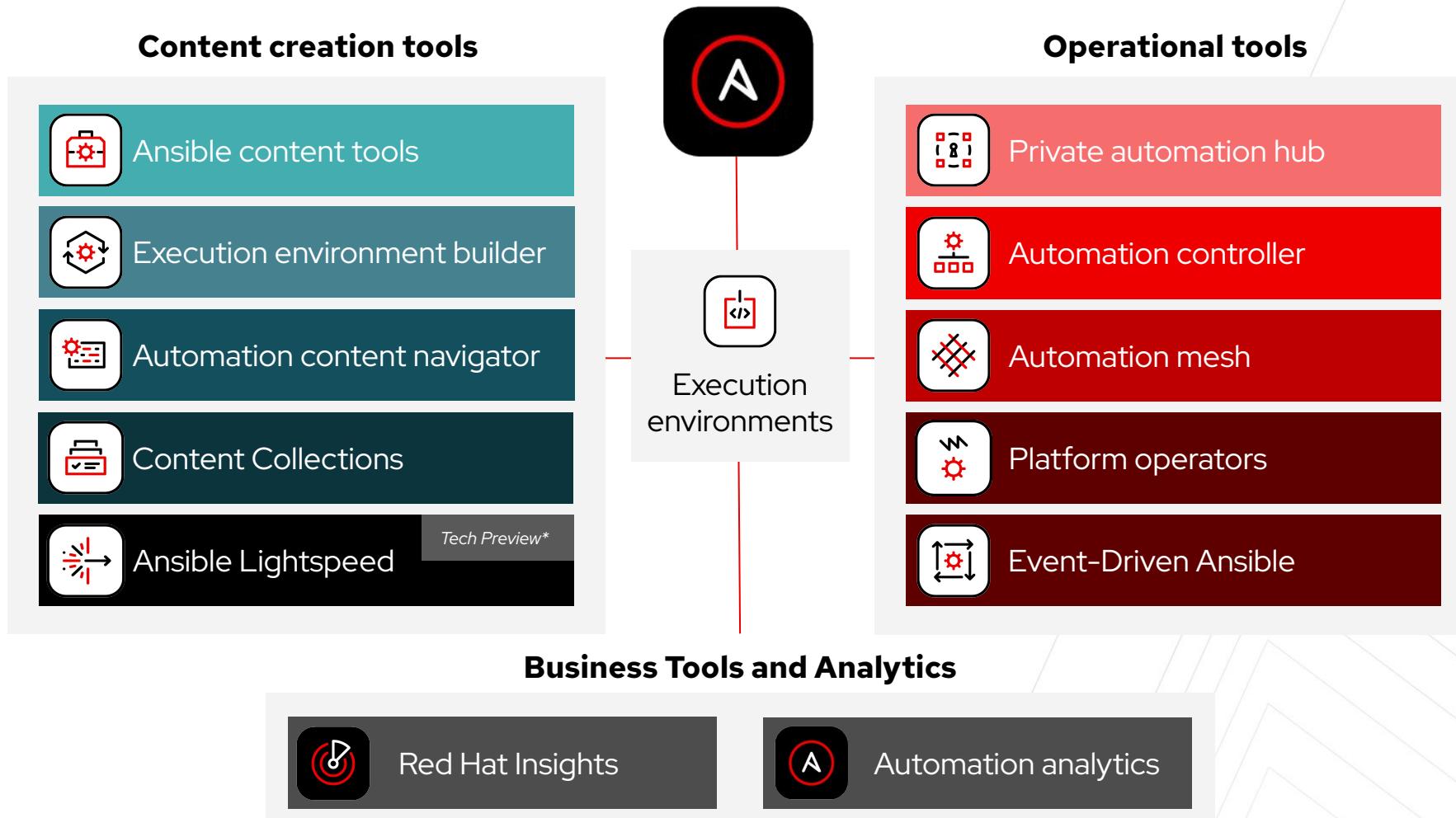
Enterprise features:  
WebUI, API, role-based access control (RBAC), auditing and workflows for managing at scale

Hosted and private content management solutions

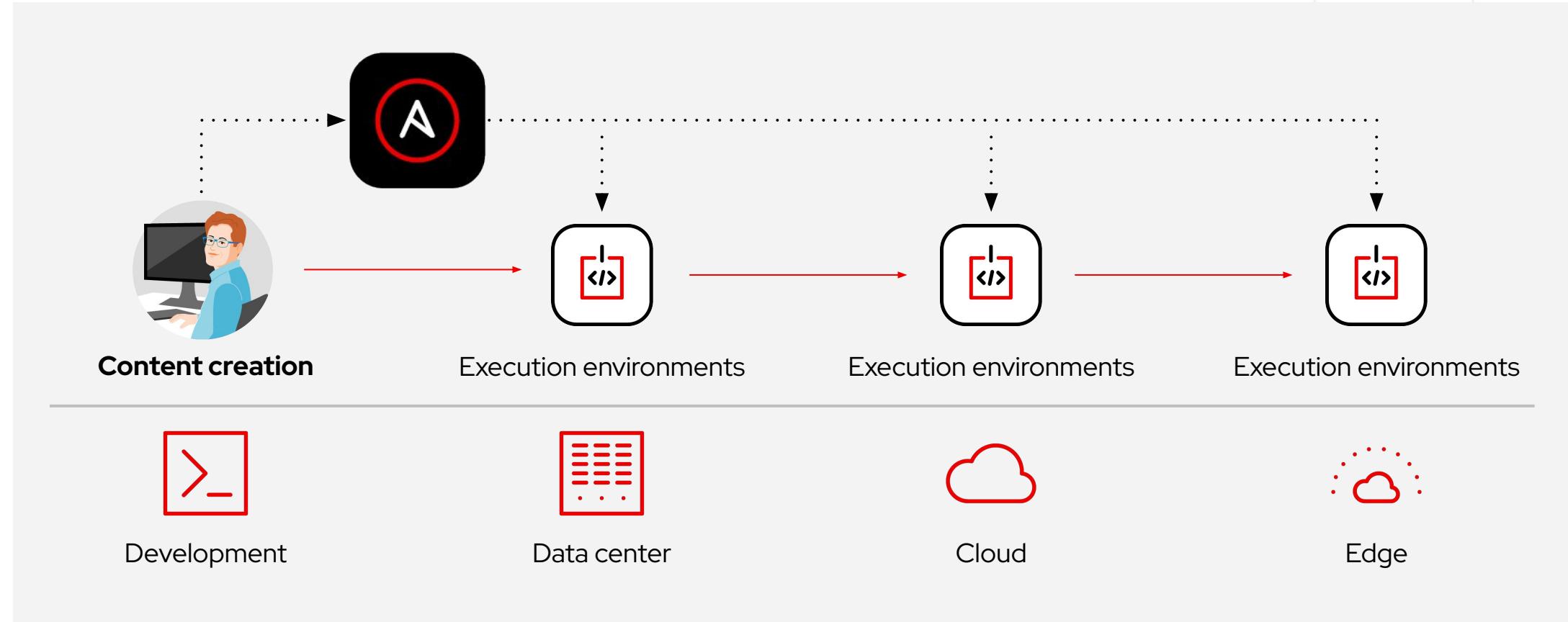
Integrates with your environment



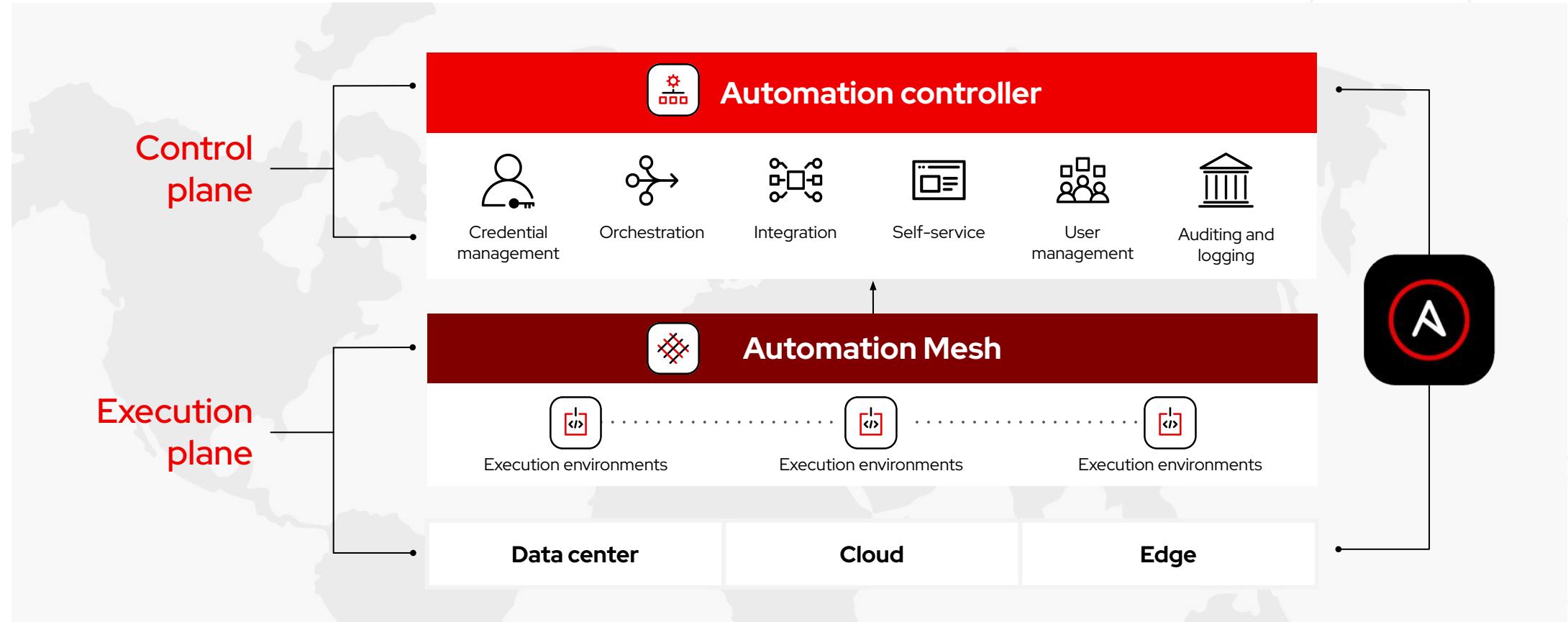
# An integrated solution for the enterprise.



## Built for consistency. **Portability** is reliability.



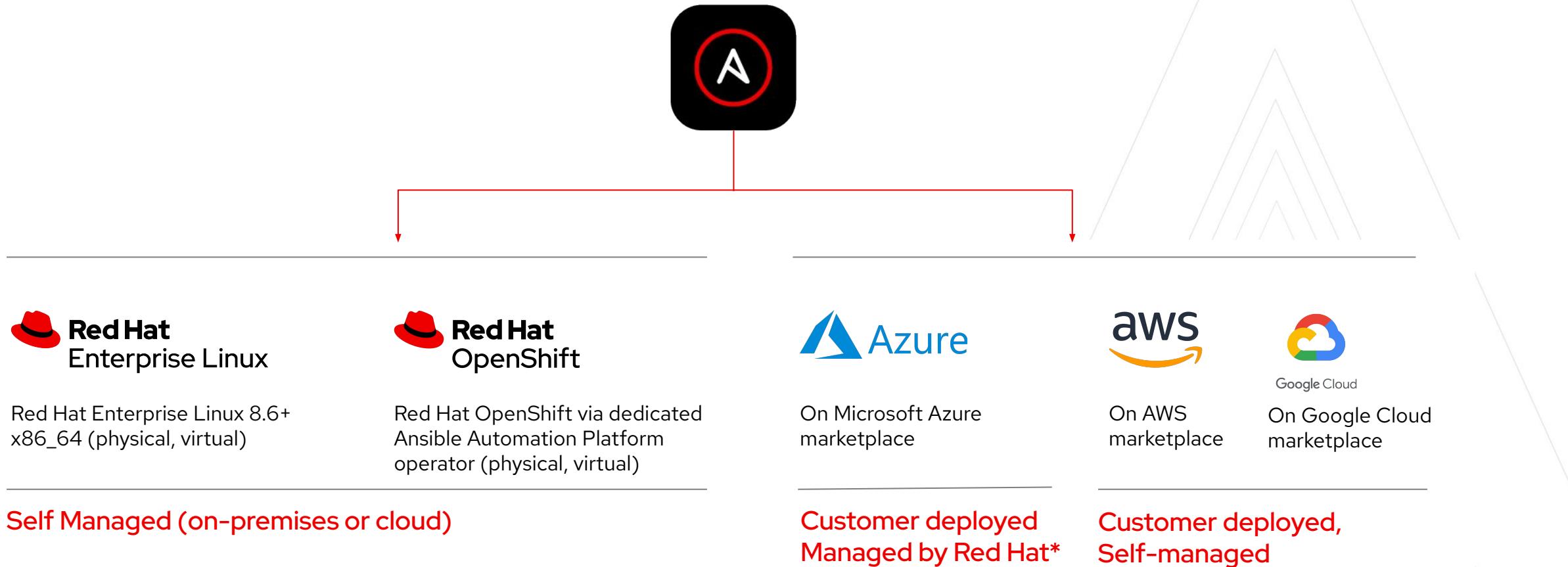
# A distributed architecture built for scale.



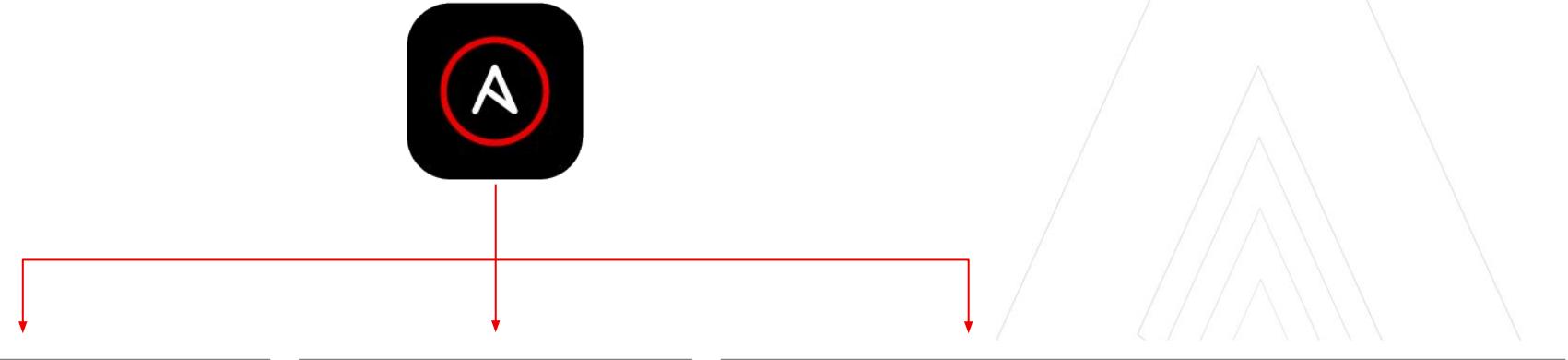
# The capabilities you need across your IT footprint.



# Ansible Automation Platform hosting options.



# Ansible Automation Platform across architectures.



**intel**<sup>®</sup>

Red Hat Enterprise Linux 8.6+  
Red Hat Enterprise Linux 9.0+

**AMD**

Red Hat Enterprise Linux 8.6+  
Red Hat Enterprise Linux 9.0+

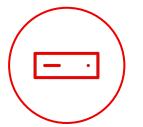
**arm**

Red Hat Enterprise Linux 8.6+  
Red Hat Enterprise Linux 9.0+

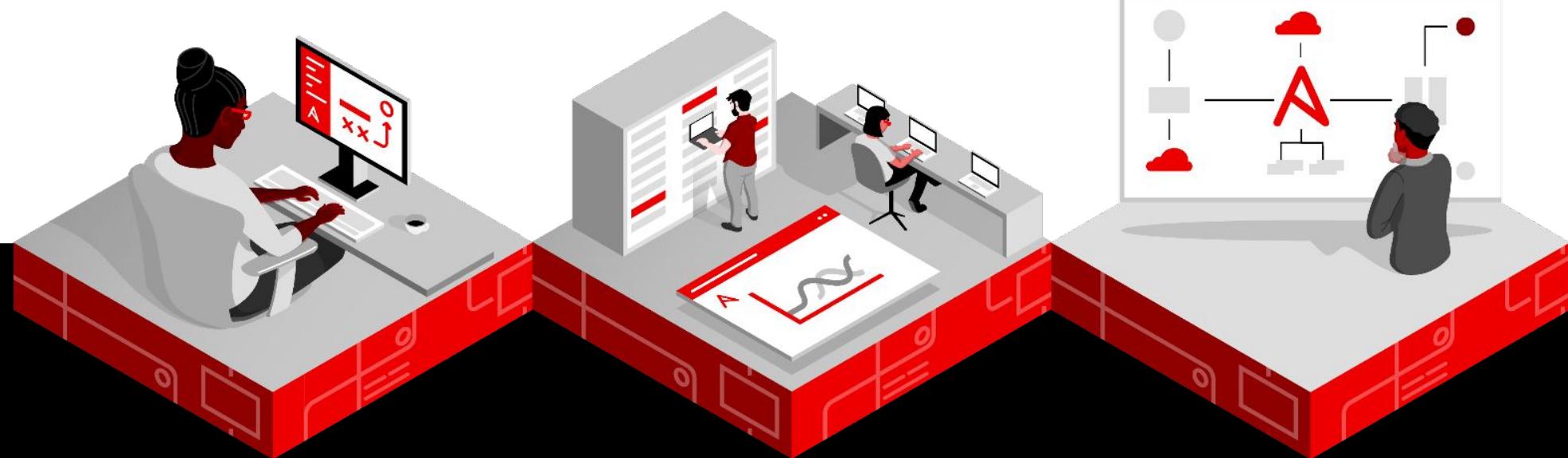
**IBM Power**    **IBM Z**

\* Red Hat Enterprise Linux 8.6+  
\* Red Hat Enterprise Linux 9.0+  
ppc64le

\* Red Hat Enterprise Linux 8.6+  
\* Red Hat Enterprise Linux 9.0+  
s390x



## Enabling your automation team **to consistently...**



**Create**

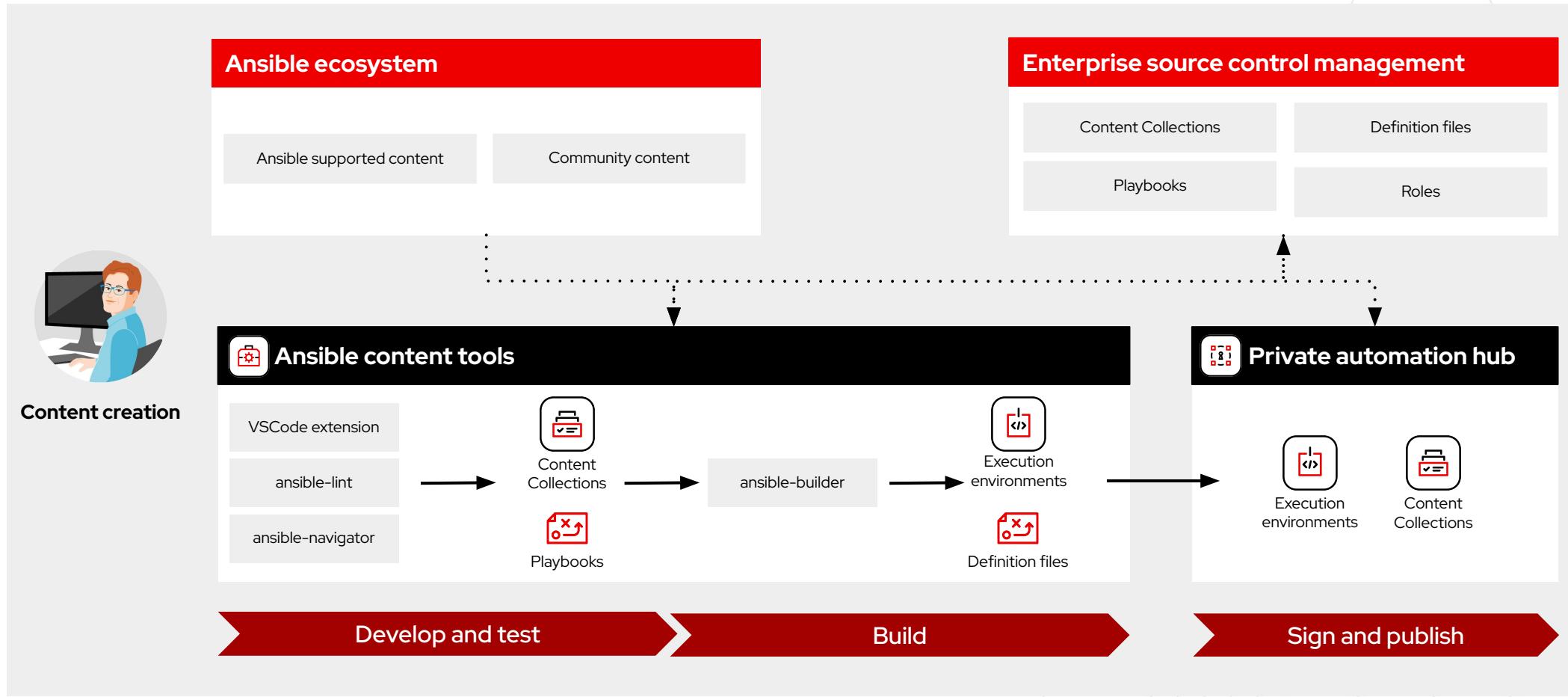
**Manage**

**Scale**

# Create



# The automation content life cycle. **Create.**



# Ansible Core (`ansible-core`)



## What is it?

- ▶ The main building block for Ansible
- ▶ Simple YAML syntax to develop Ansible Playbooks
- ▶ Provides CLI tools to develop, test and run playbooks
- ▶ Pluggable architecture that allows extensions through Content Collections

```
---  
- name: Shutdown VM guest  
hosts: localhost  
gather_facts: false  
tasks:  
  - name: Turn off specified VM guest  
    vmware.vmware_rest.vcenter_vm_guest_power:  
      state: shutdown  
      vm: 1021343  
      vcenter_hostname: vcenter.demoreddhat.com  
      vcenter_username: admin  
      vcenter_password: tedlasso
```

# Ansible playbooks

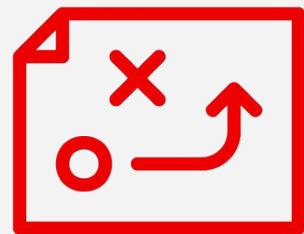
```
---
- name: Install and start apache
  hosts: web
  become: true

  tasks:
    - name: Ensure the httpd package is installed
      ansible.builtin.package:
        name: httpd
        state: present

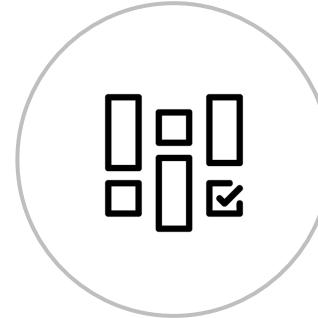
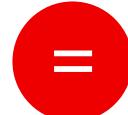
    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```

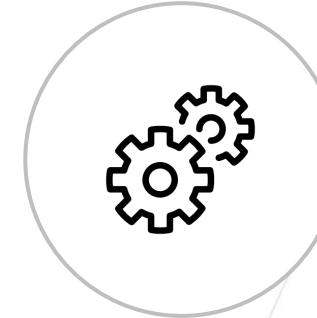
# What makes up an Ansible playbook?



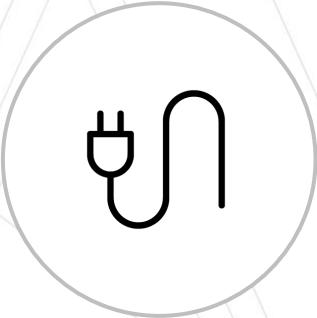
Playbook



Plays



Modules



Plugins

# Ansible plays. What am I automating?



## What are they?

- ▶ Top level specification for a group of tasks
- ▶ Will tell that play which hosts it will execute on and control behavior such as fact gathering or privilege level

## Building blocks for playbooks

- ▶ Multiple plays can exist within an Ansible playbook

```
...  
- name: Ensure the httpd package is installed  
  hosts: web  
  become: true
```

# Ansible modules. The “tools in the toolkit”.

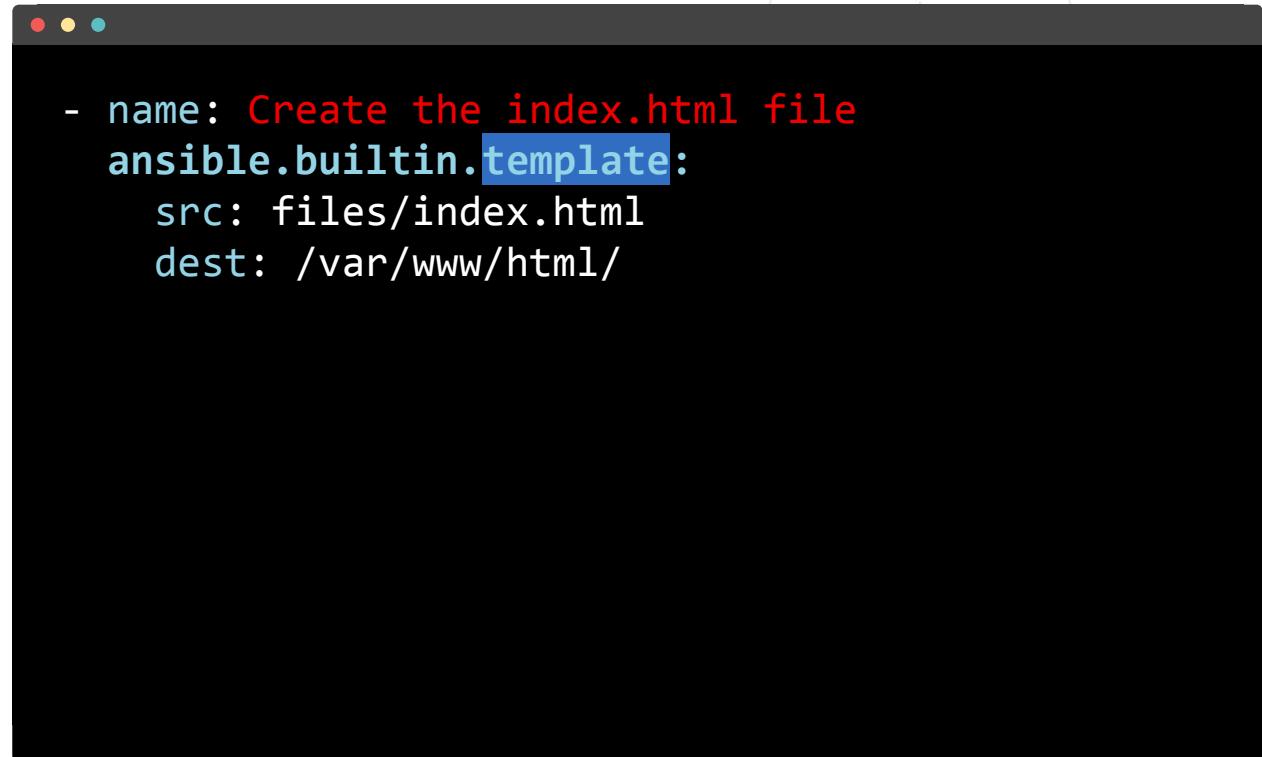


## What are they?

- ▶ Parameterized components with internal logic, representing a single step to be done
- ▶ The modules “do” things in Ansible

## Language

- ▶ Usually created in Python, or Powershell for Windows setups, but can be developed in any language



A terminal window showing Ansible code. The code is a single command:

```
- name: Create the index.html file
  ansible.builtin.template:
    src: files/index.html
    dest: /var/www/html/
```



# Ansible plugins. The “extra bits”.

## What are they?

- ▶ Plugins are pieces of code that augment Ansible's core functionality
- ▶ Ansible uses a plugin architecture to enable a rich, flexible, and expandable feature set

```
● ● ●

Example become plugin:
---
- name: Install and start apache
  hosts: web
  become: true

Example filter plugins:
{{ some_variable | to_nice_json }}
{{ some_variable | to_nice_yaml }}
```

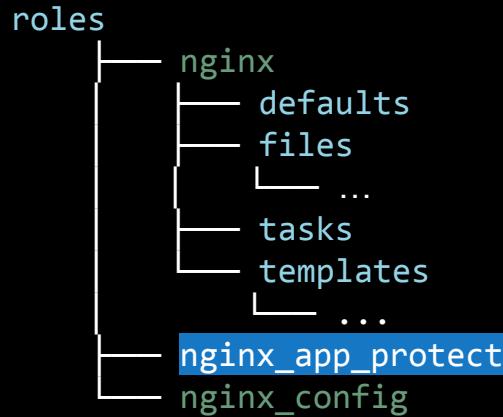
# Ansible Roles. Reusable automation actions.



## What are they?

- ▶ Group tasks and variables of your automation in a reusable structure
- ▶ Write roles once, and share them with others who have similar challenges in front of them

```
---  
- name: Install and start apache  
hosts: web  
ansible.builtin.roles:  
  - common  
  - webservers
```



### deploy-nginx.yml

```
---
```

```
- name: Install NGINX Plus
  hosts: all
  tasks:
```

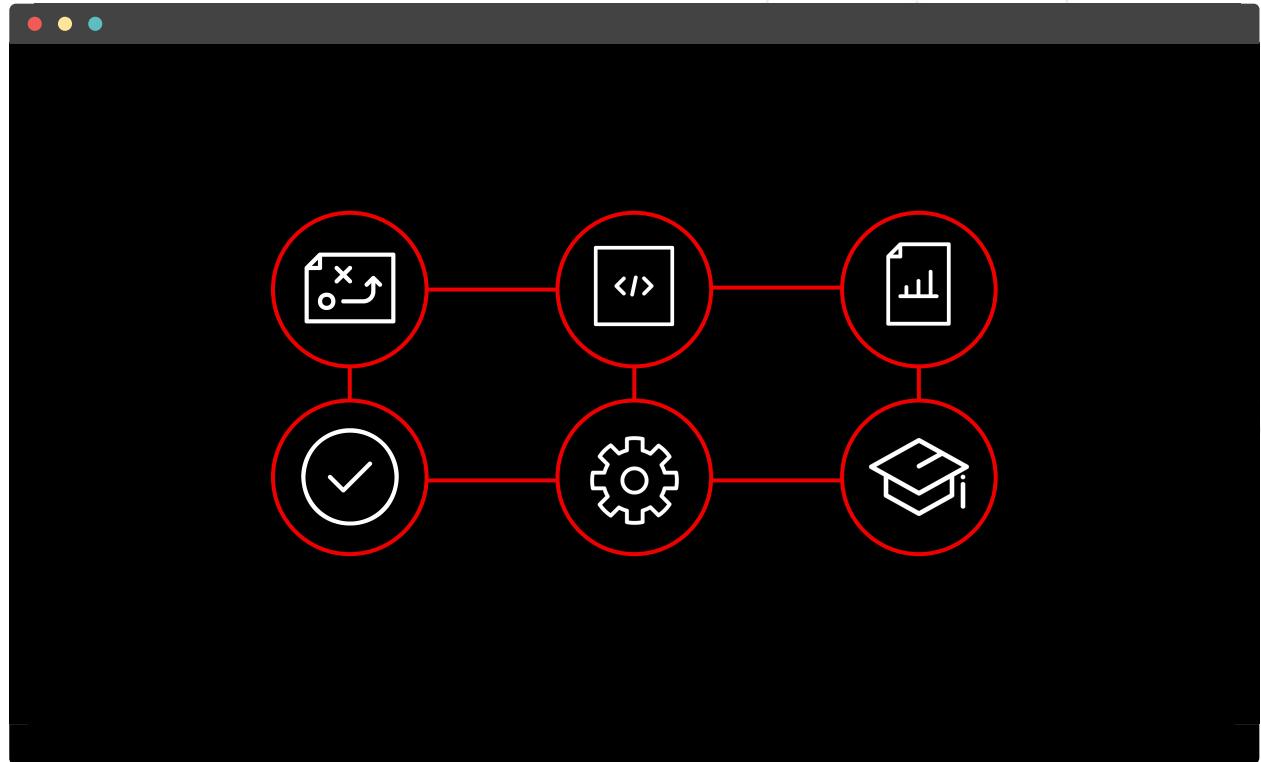
```
  - name: Install NGINX App Protect
    ansible.builtin.include_role:
      name: nginx_app_protect
  vars:
    nginx_app_protect_setup_license: false
    nginx_app_protect_remove_license: false
    nginx_app_protect_install_signatures: false
```

# Content Collections.

## Simplified, consistent content delivery.

### What are they?

- ▶ Contains automation content, including modules, multiple roles ,and playbooks
- ▶ Portable, reusable, and versioned enabling better collaboration





```
nginx_core
├── galaxy.yml
├── meta
└── playbooks
    └── deploy-nginx.yml
    ...
plugins
README.md
roles
└── nginx
    ├── defaults
    ├── files
    │   └── ...
    ├── tasks
    └── templates
        └── ...
    └── nginx_app_protect
    └── nginx_config
```

### deploy-nginx.yml

```
---
- name: Install NGINX Plus
  hosts: all
  tasks:
    - name: Install NGINX
      ansible.builtin.include_role:
        name: nginxinc.nginx
      vars:
        nginx_type: plus

    - name: Install NGINX App Protect
      ansible.builtin.include_role:
        name: nginxinc.nginx_app_protect
      vars:
        nginx_app_protect_setup_license: false
        nginx_app_protect_remove_license: false
        nginx_app_protect_install_signatures: false
```

# Automation hub. Trusted automation content.

[console.redhat.com](https://console.redhat.com)

## What is it?

- ▶ Hosted source of trusted Red Hat and Certified Partner Content Collections
- ▶ Integrated documentation and examples
- ▶ Configurable as primary content collection source for your automation environment
- ▶ Access to hosted automation hub and content included in subscription

The screenshot shows the 'Approval dashboard' interface. At the top, there's a search bar with filters for 'Repository' (set to 'published') and a search icon. Below the search bar, there are buttons for 'repository' and 'published' with an 'X' icon, and a 'Clear all filters' link. The main area is a table with columns: Namespace, Collection, Version, Date created, and Status. A single row is visible: 'kyleabenson' under Namespace, 'fest2020collection' under Collection, '1.1.1' under Version, '3 hours ago' under Date created, and 'Approved' with a green checkmark under Status. At the bottom of the table, there are navigation links for '1-1 of 1' and page numbers '1 of 1'.

Namespace	Collection	Version	Date created	Status
kyleabenson	fest2020collection	1.1.1	3 hours ago	Approved

# Ansible VS Code extension. Simplifying content creation



## What is it?

- ▶ Syntax highlighting of keywords such as module names
- ▶ Live validation of your code while you type
- ▶ Integration with ansible-lint\*
- ▶ Autocompletion on play, block or task contents etc
- ▶ Documentation references as you code

The screenshot shows a portion of an Ansible playbook in the main editor area:

```

- name: query incident number and creation time
  set_fact:
    incident_list: '{{ incident_list + [{"number": item.number, "opened_at": item.opened_at}] }}'
  loop: "{{ incidents.records }}"
  when: incidents

- name: Create a problem from incident
  problem
  
```

A tooltip is displayed over the word "problem", listing available completions:

- (servicenow.itsm.problem)
- (servicenow.itsm.problem\_info)
- (servicenow.itsm.problem\_task)
- (servicenow.itsm.problem\_task\_info)
- (ansible.builtin.proxmox)
- (ansible.builtin.portage)
- (ansible.builtin.proxmox\_kvm)
- (ansible.builtin.profitbricks)
- (ansible.builtin.portinstall)
- (ansible.builtin.proxysql\_backend\_servers)
- (ansible.builtin.profitbricks\_nic)
- (ansible.builtin.proxmox\_template)

Details about the selected completion are shown in the tooltip:

**servicenow.itsm**

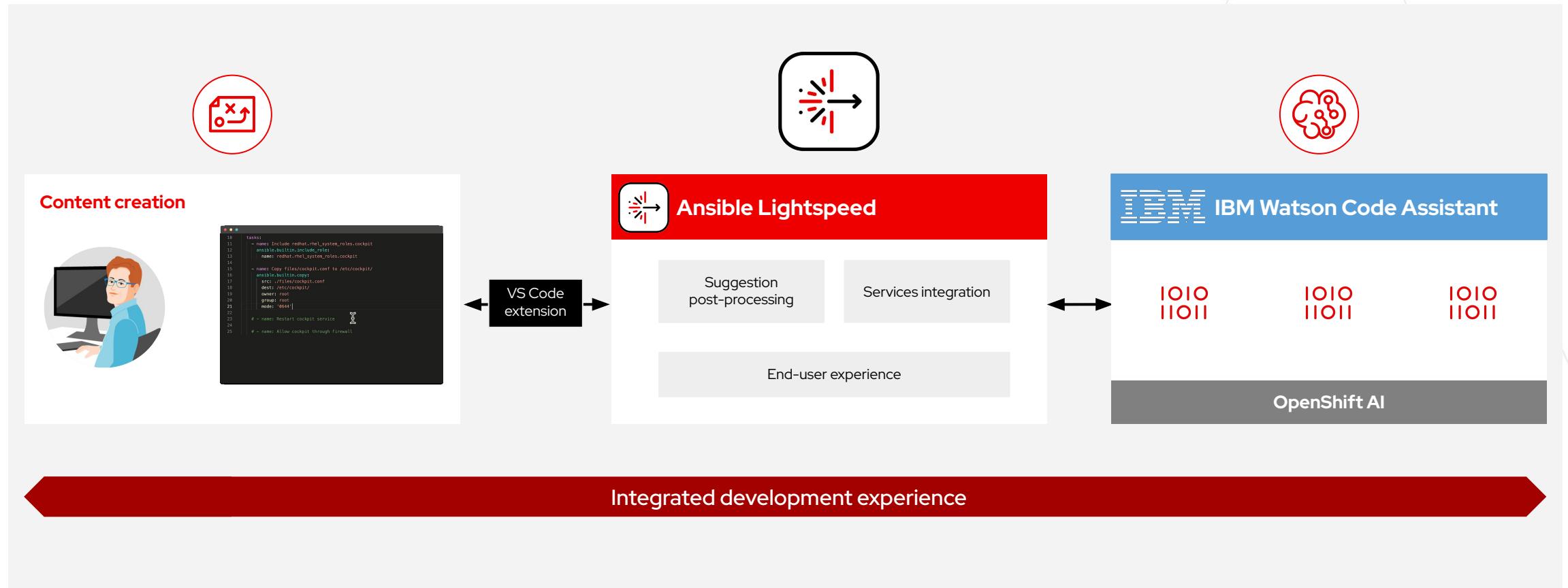
Manage ServiceNow problems

**Description**

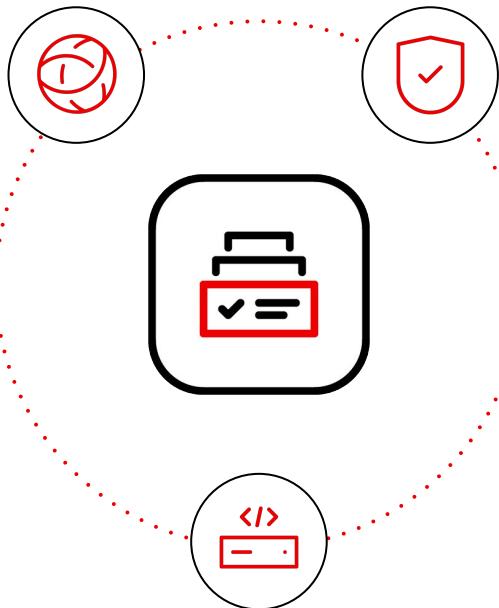
- Create, delete or update a ServiceNow problem.
- For more information, refer to the ServiceNow problem management documentation at [https://docs.servicenow.com/bundle/paris-it-service-management/page/product/problem-management/concept/c\\_ProblemManagement.html](https://docs.servicenow.com/bundle/paris-it-service-management/page/product/problem-management/concept/c_ProblemManagement.html).

# Ansible Lightspeed. Generative AI the Ansible way.

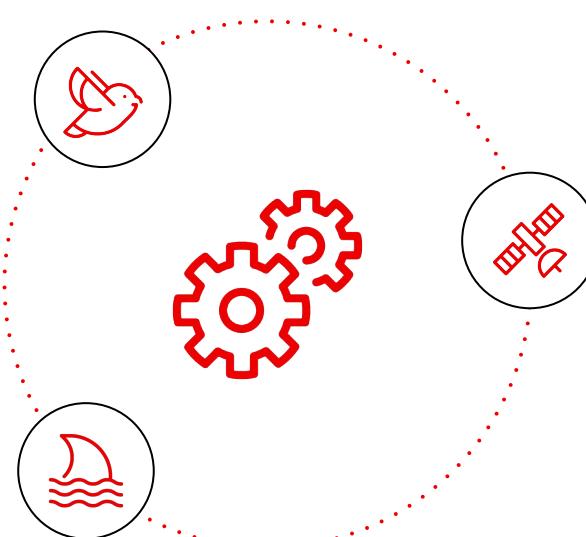
Technical Preview \*



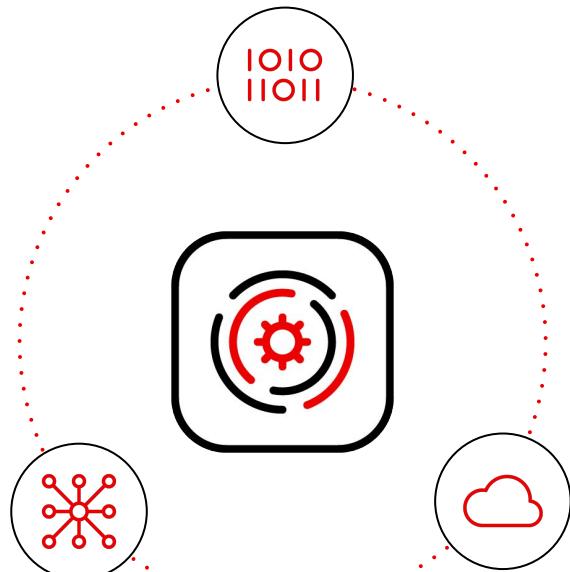
# Many technologies, different life cycles. How do I keep it all aligned?



Collections

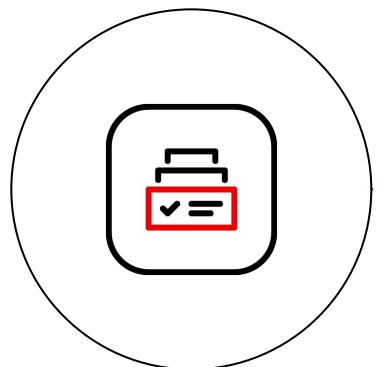


Dependencies

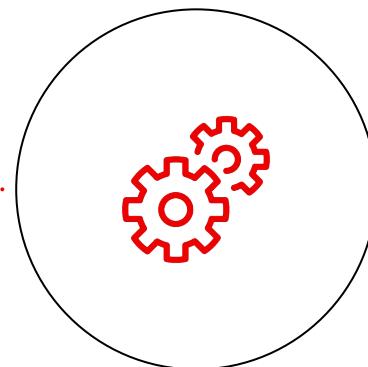


Runtime

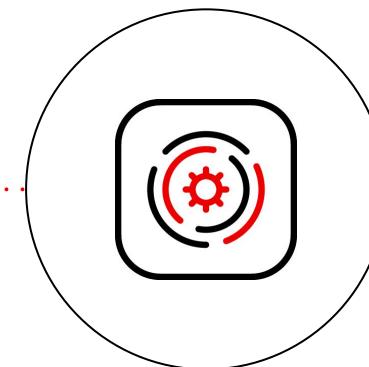
# Automation execution environments. Reuse and scale automation content.



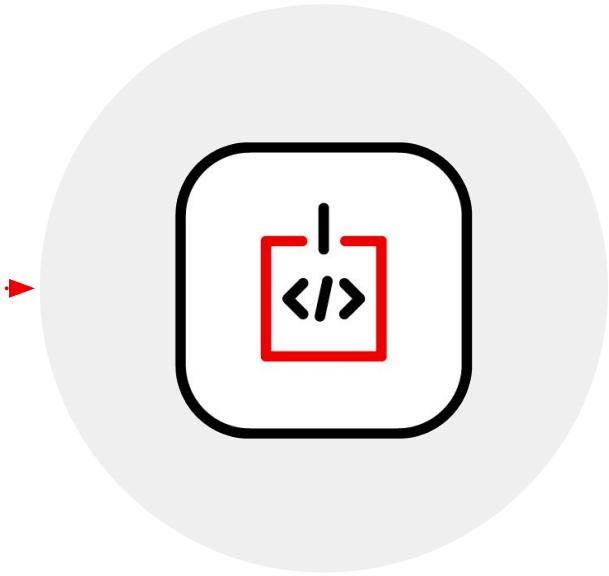
Collections



Dependencies



Ansible Core

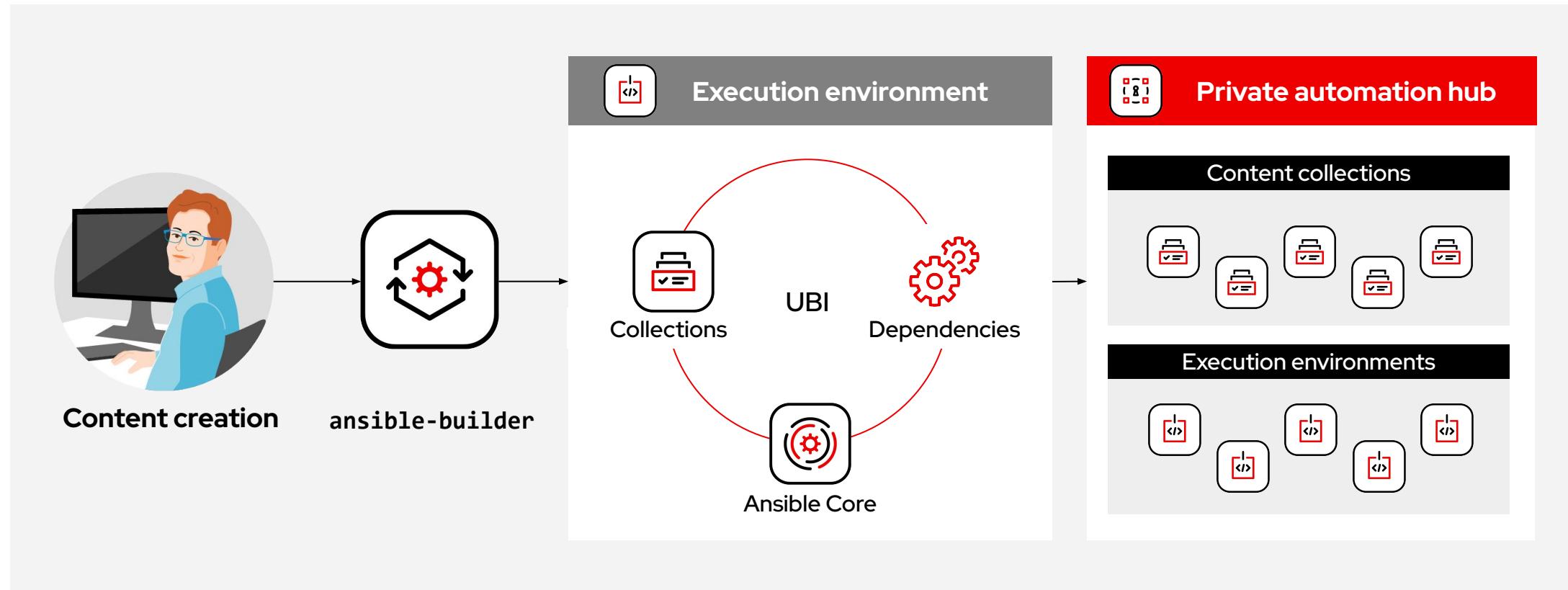


Automation  
execution environments

Red Hat Universal Base Image (UBI)

# Execution environment development.

## Build, collaborate, sign, publish.



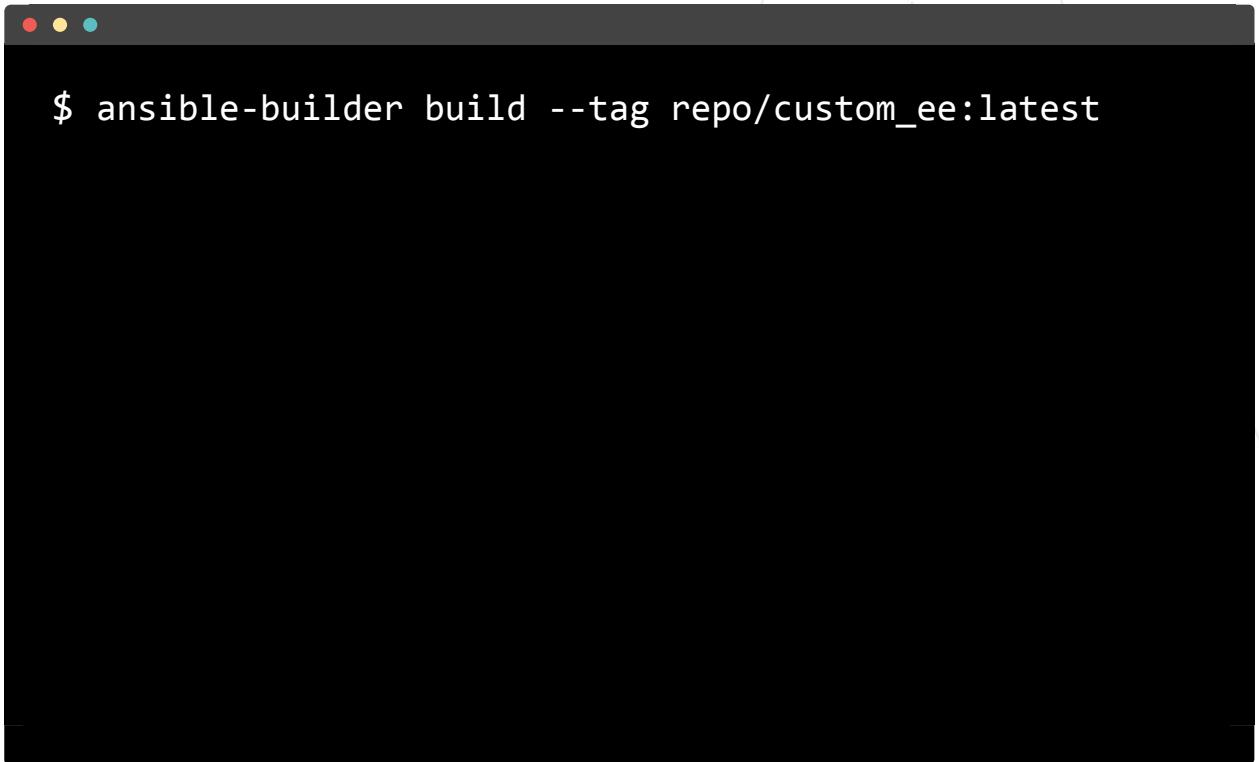


# Execution environment builder. Build.

ansible-builder

## What is it?

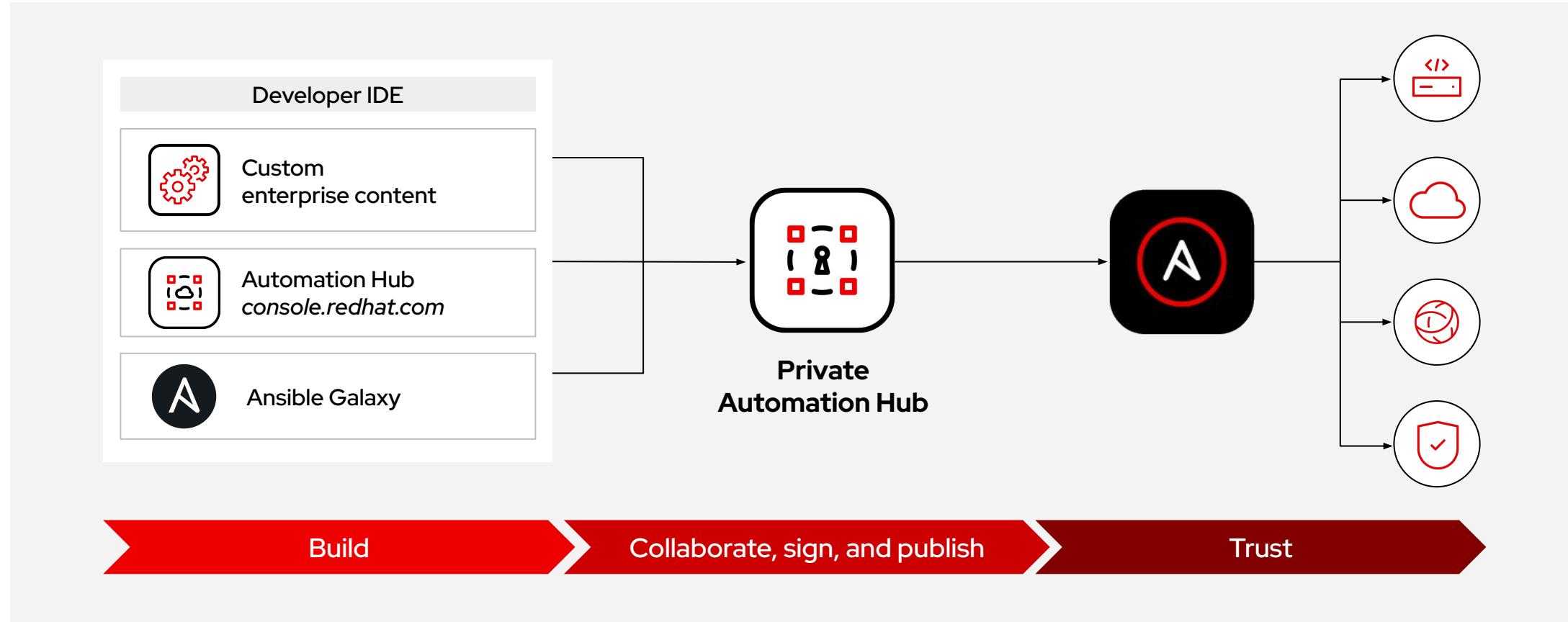
- ▶ Easily build custom execution environments with the exact Ansible content needed
- ▶ Manage, track and version execution environments
- ▶ Share execution environment build artifacts with other teams



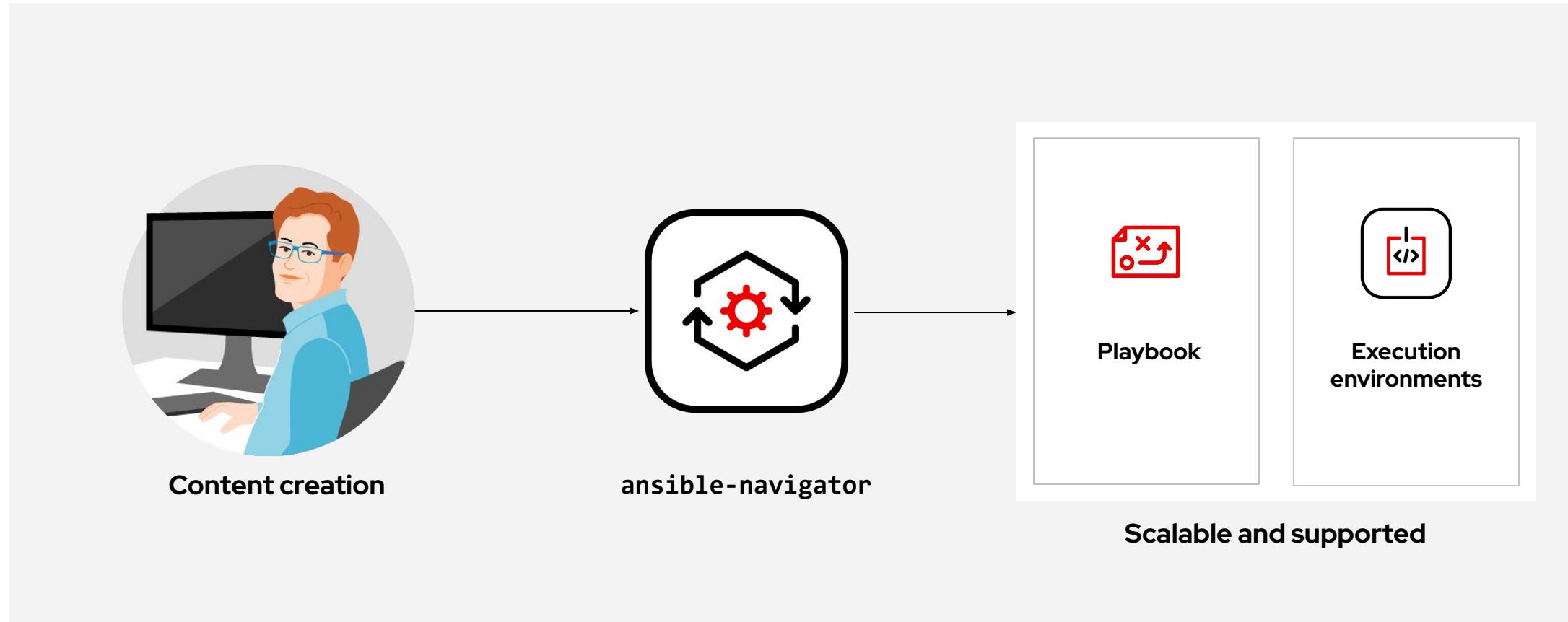
A screenshot of a terminal window with a dark background. At the top, there are three small colored dots (red, yellow, blue). Below them, the command is displayed in white text:

```
$ ansible-builder build --tag repo/custom_ee:latest
```

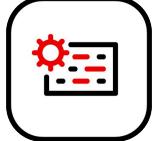
# Private automation hub. **Collaborate, sign, and publish.**



# Develop, test and run **containerized Ansible content**.



# Ansible content navigator (ansible-navigator).



## What is it?

It is a command line utility and text-based user interface (TUI) for running, testing and developing Ansible automation content

- ▶ Review EEs
- ▶ Develop collections
- ▶ Develop playbooks
- ▶ Troubleshoot problems

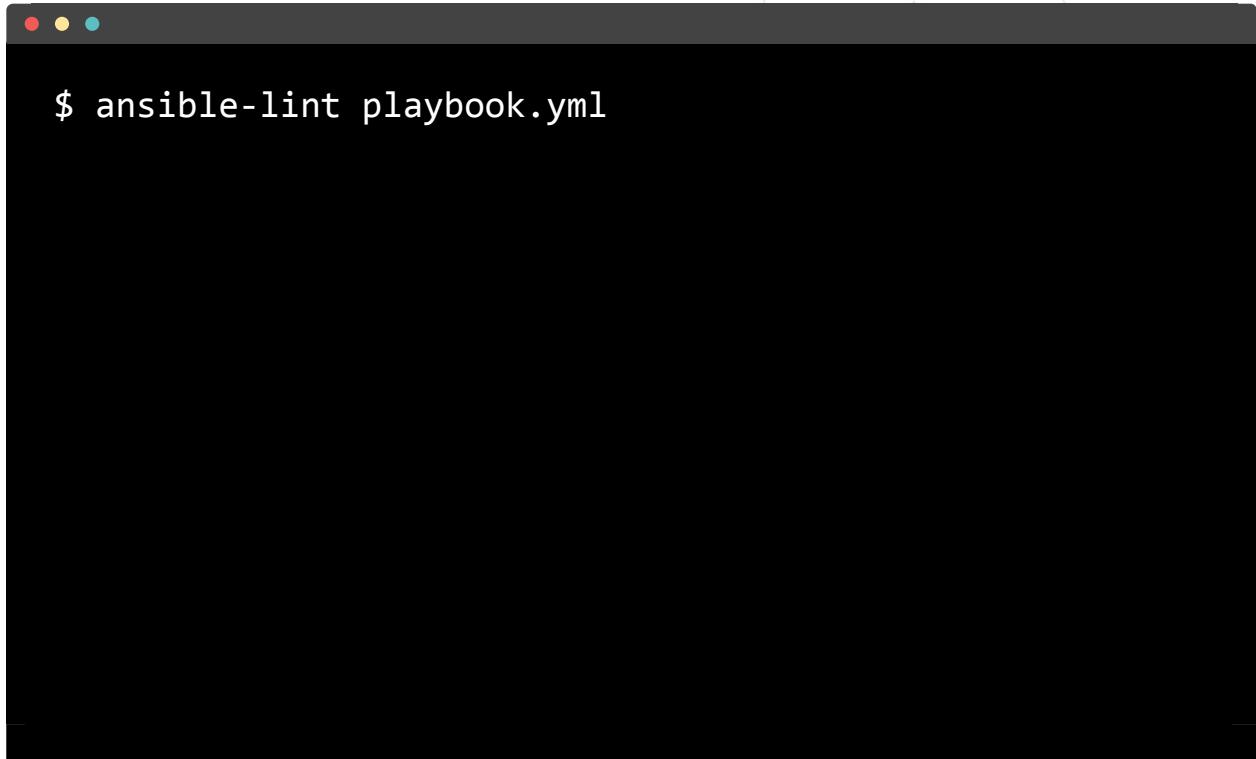
A screenshot of a terminal window with a black background and light gray text. The window has three colored window control buttons (red, yellow, blue) at the top. The text inside the terminal shows a command line: '\$ ansible-navigator run playbook.yml -i inventory.ini'. The terminal is set against a background featuring abstract white line art.

# Ansible lint (ansible-lint).

## What is it?

Command-line tool for linting playbooks, roles and collections aimed towards any Ansible users.

- ▶ Promote best practices and and patterns.
- ▶ Develop consistent code across teams and scale using an opinionated strategy.
- ▶ Integrate into larger development workflows and CI tools.
- ▶ Helps upgrade playbooks to later Ansible Core versions.

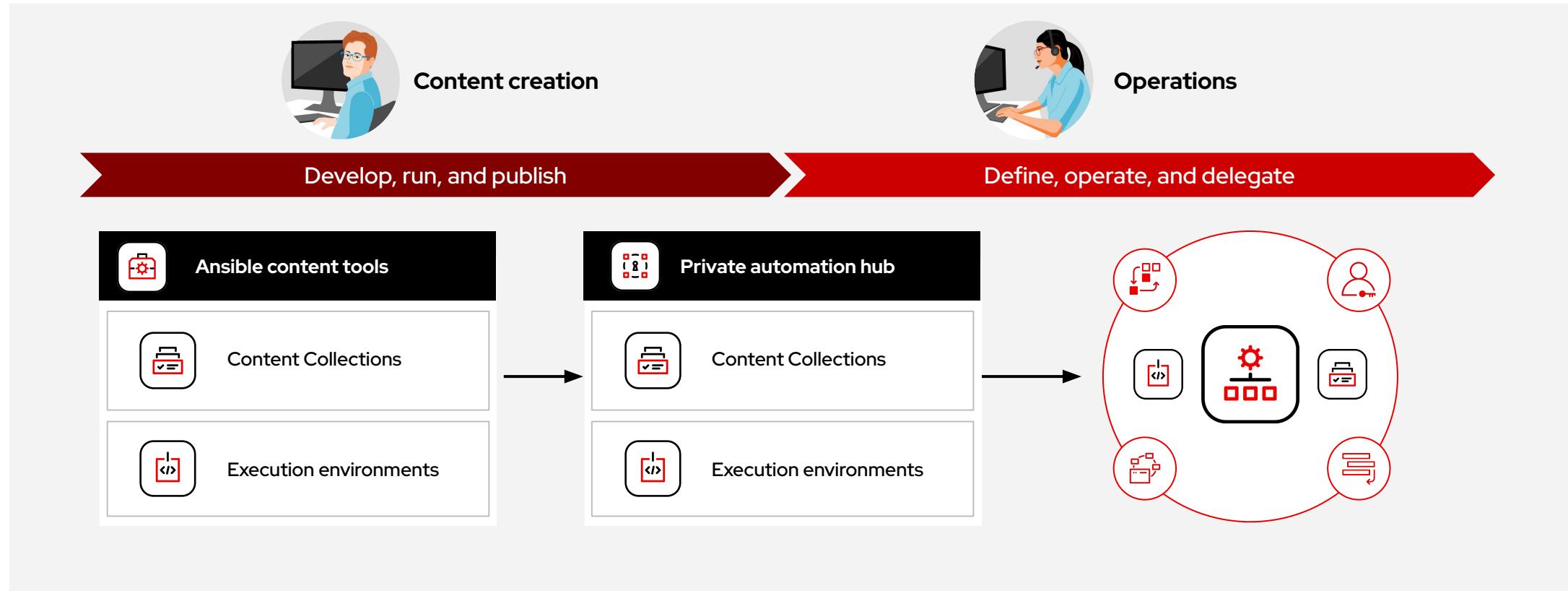


```
$ ansible-lint playbook.yml
```

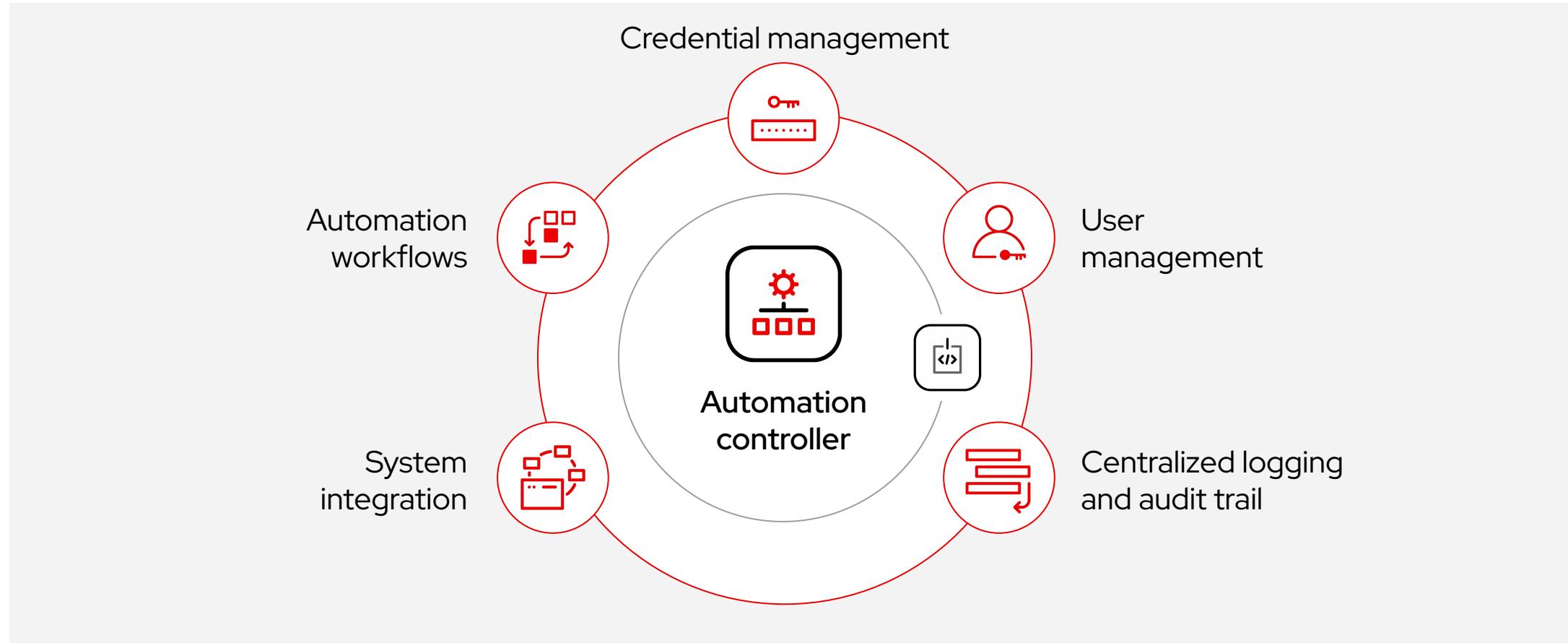
# Manage



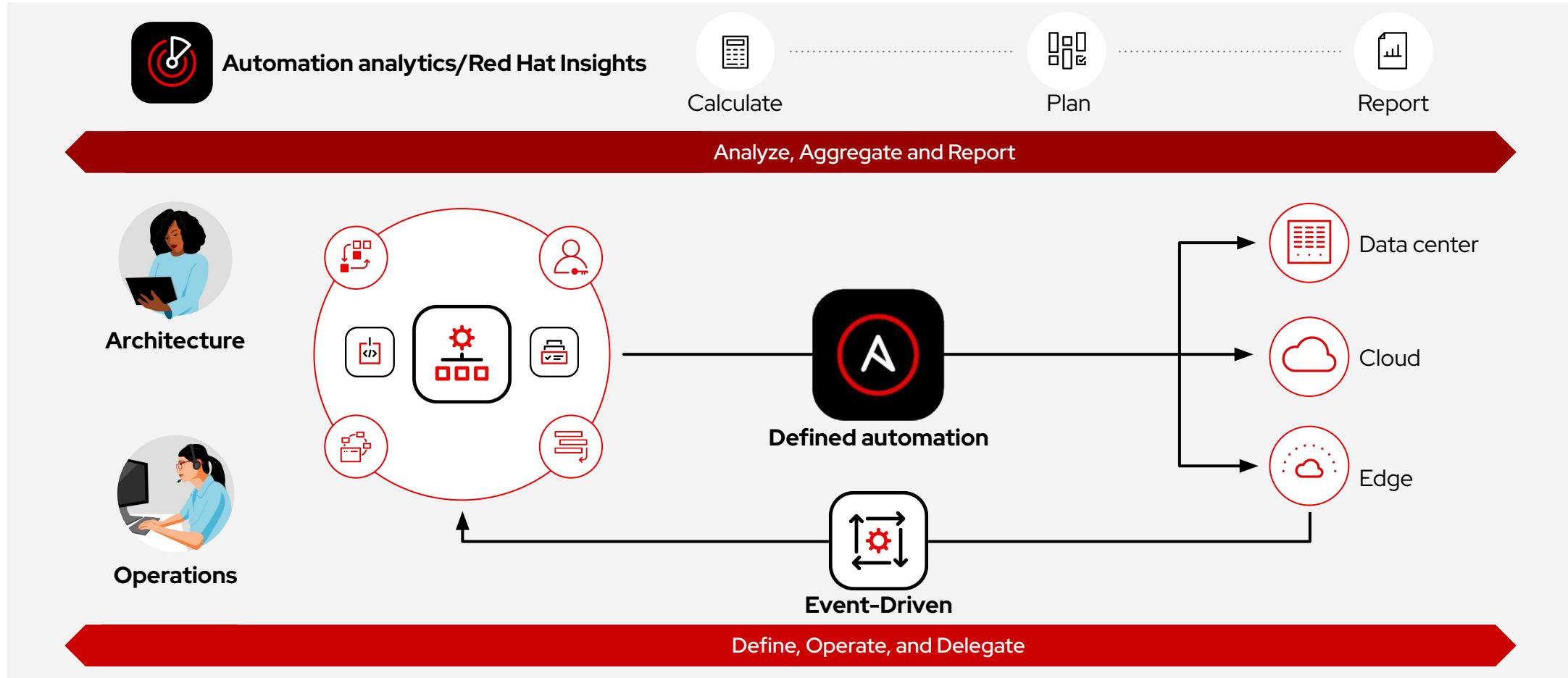
# Creation to operations handover. **Consistent and portable.**



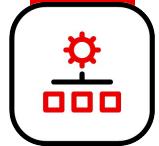
# Enterprises need more for their automation.



# The automation life cycle. **Manage.**



# Automation controller. Define, operate, and delegate.

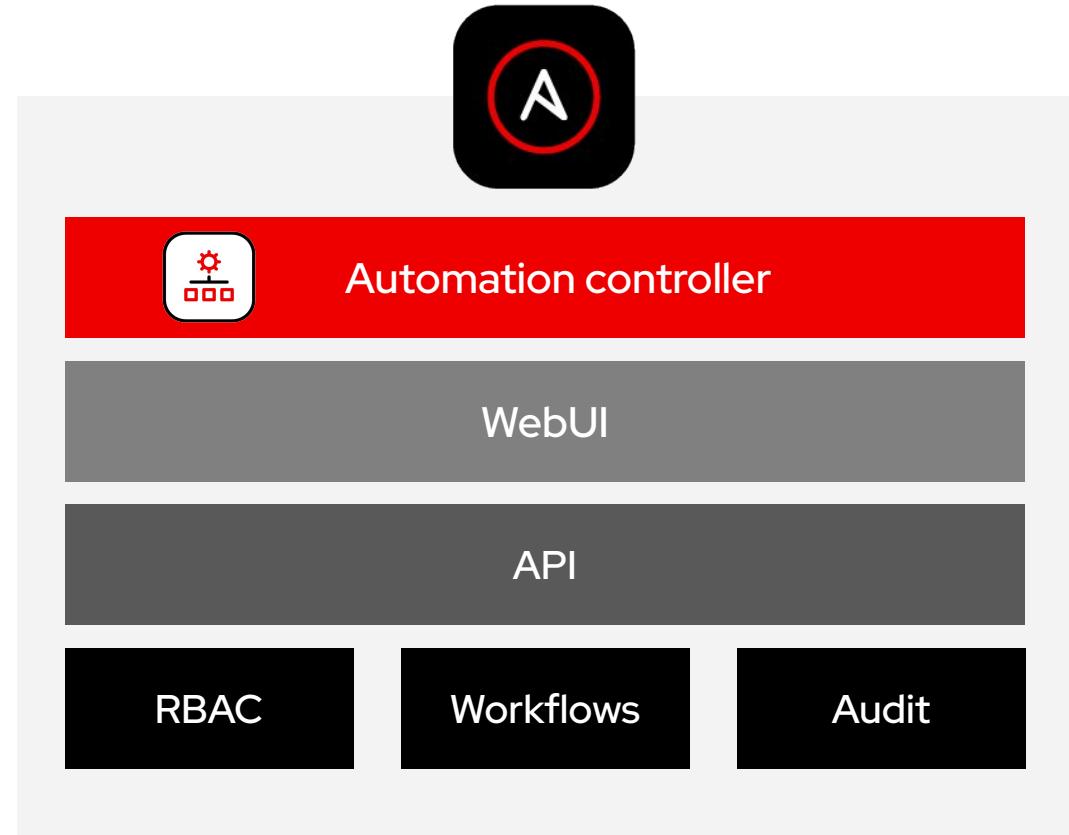


## What is it?

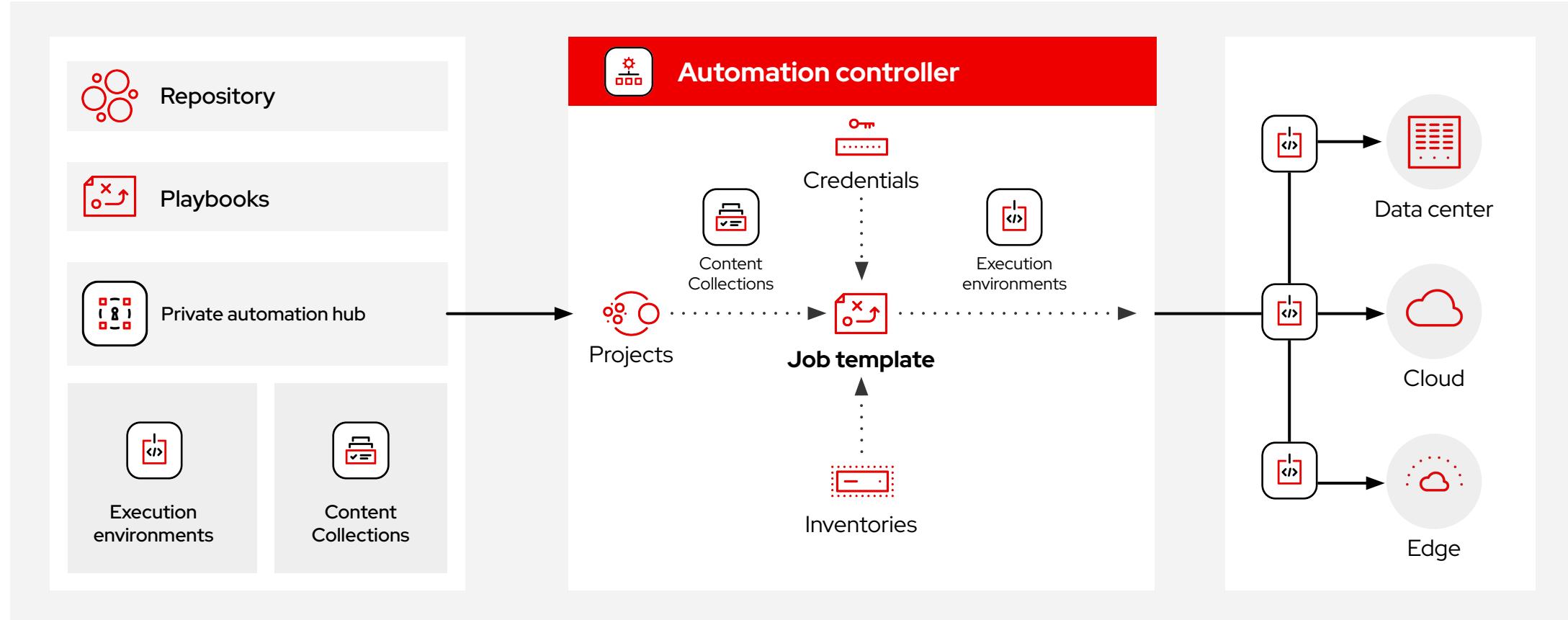
Automation controller is the Ansible Automation Platform control plane which enables users to define, operate, and delegate automation across their enterprise

## Automation controller provides:

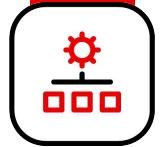
- ▶ WebUI and API
- ▶ Role-based access control
- ▶ Powerful workflows
- ▶ Centralized logging
- ▶ Credential management
- ▶ Push-button automation



# Defining how your content runs in automation controller.



# Using execution environments in automation controller.



## How do I add execution environments (EE)?

EEs can be added via the controller WebUI, API or command line

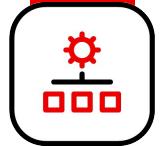
- ▶ Specify the EE image source
- ▶ Pull policy determines when and how EEs are downloaded when running automation
- ▶ Assign registry credentials to pull the EEs

## How do I use execution environments?

- ▶ Select the EE you want to use in your automation job
- ▶ The default EE is used if none is specified

The screenshot shows a modal dialog titled "Create new execution environment". It contains fields for "Name" (Security EE), "Image" (quay.io/ansible/security-ee:latest), and a "Pull" dropdown set to "Always pull container before running". There are also fields for "Description", "Organization", and "Registry credential". A note at the bottom says "Leave this field blank to make the execution environment globally available." At the bottom are "Save" and "Cancel" buttons.

# Projects. Adding your automation content to controller.



## What is it?

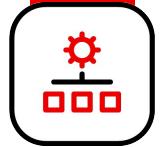
Logical collection of your playbooks:

- ▶ Multiple source types supported
- ▶ Source Control Management (SCM) integration and update strategies
- ▶ Red Hat Insights integration
- ▶ Role-based access control (RBAC) and schedules

The screenshot shows the 'Projects' page of the Red Hat Ansible Automation Platform. The interface has a red header bar. Below it, there's a search bar with a dropdown menu set to 'Name', a search icon, and a blue 'Add' button. To the right of the search bar are 'Delete' and three-dot buttons. The main area is titled 'Projects' and contains a table with two rows of data. The columns are labeled 'Name' (with an upward arrow), 'Status', and 'Type'. The first row shows a project named 'Demo Project' with a status of 'Successful' and a type of 'Git'. The second row shows a project named 'Example' with a status of 'Successful' and a type of 'Git'. Each project row has a small expand/collapse icon and a checkbox.

Name ↑	Status	Type
> Demo Project	Successful	Git
> Example	Successful	Git

# Inventories. What do I want to run my automation on?



## What is it?

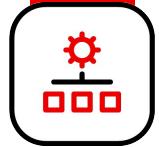
Collection of endpoints against which jobs may be launched

- ▶ Multiple inventory sources supported
- ▶ Dynamic endpoint discovery
- ▶ Logically group endpoints by metadata or user-defined filters
- ▶ Granular RBAC permissions

The screenshot shows a web-based management interface for inventories. At the top, there's a search bar labeled 'Name' with a dropdown arrow, a magnifying glass icon, and a blue 'Add' button. To the right of the search bar are 'Delete' and a page navigation indicator '1 - 4 of 4'. Below the header is a table with the following columns: Name, Status, Type, Organization, and Action. There are four rows in the table, each representing an inventory source:

Name	Status	Type	Organization	Action
AWS Inventory	Disabled	Inventory	ACME Corp	
Azure Inventory	Disabled	Inventory	ACME Corp	
Data center	Disabled	Inventory	ACME Corp	
Remote Office A	Disabled	Inventory	ACME Corp	

# Credentials. Securing resource and endpoint access.



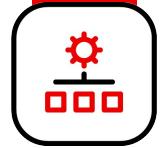
## What is it?

- ▶ Securely manage credentials needed for automation resources
- ▶ Multiple credential types supported
- ▶ Integrate external secret management systems
- ▶ Create custom credential types and plugins
- ▶ Use RBAC to govern access
- ▶ Actual credential never exposed

The screenshot shows a web-based interface titled 'Credentials'. At the top, there is a search bar labeled 'Name' with a dropdown arrow, a magnifying glass icon, and a blue 'Add' button. To the right of the search bar are 'Delete' and three-dot menu buttons. Below the header, the word 'Name' is followed by an upward arrow, indicating a sorting option. A table lists four credential entries:

Name	Type
Ansible Galaxy	Ansible Galaxy/Automation Hub API Token
Automation Controller	Red Hat Ansible Automation Platform
AWS Credential	Amazon Web Services
Azure Credentials	Microsoft Azure Resource Manager

# Job Templates. Bringing it all together.



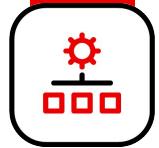
## What is it?

- ▶ Define and standardize running automation
- ▶ Reusable and shareable
- ▶ Leverage agile practices, such as GitOps and event-driven automation

The screenshot shows the 'Job template with slicing' configuration page. The page has a red header and a white body with various input fields and dropdown menus. The fields include:

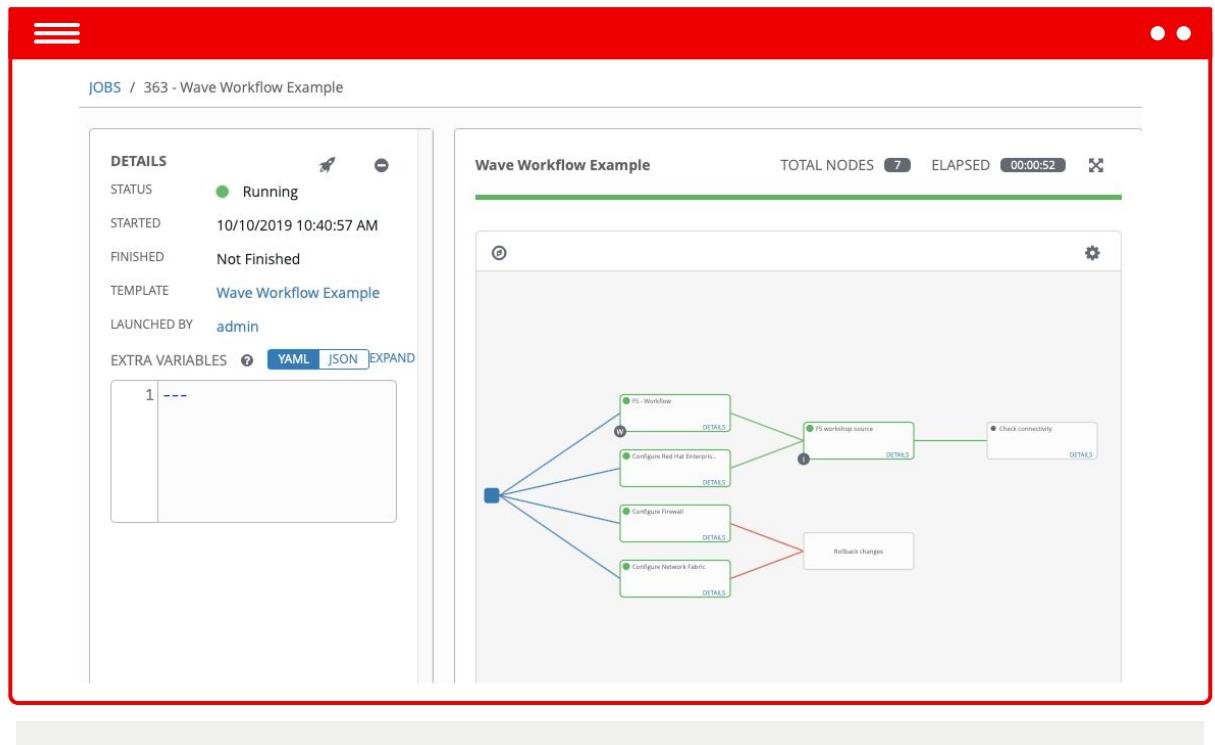
- DETAILS** tab selected.
- NAME**: Job template with slicing
- DESCRIPTION**: (empty)
- JOB TYPE**: Run
- INVENTORY**: King PLC
- PROJECT**: Project from Git
- PLAYBOOK**: gen\_host\_status.yml
- CREDENTIALS**: (empty)
- FORKS**: 0
- LIMIT**: (empty)
- VERBOSITY**: 0 (Normal)
- JOB TAGS**: (empty)
- Skip TAGS**: (empty)
- LABELS**: (empty)
- INSTANCE GROUPS**: (empty)
- JOB SLICING**: 1
- TIMEOUT**: 0
- SHOW CHANGES**: (switch is off)
- OPTIONS** section:
  - ENABLE PRIVILEGE ESCALATION
  - ENABLE PROVISIONING CALLBACKS
  - ENABLE WEBHOOK
  - ENABLE CONCURRENT JOBS
  - ENABLE FACT CACHE

# Workflows. Solve complex problems.

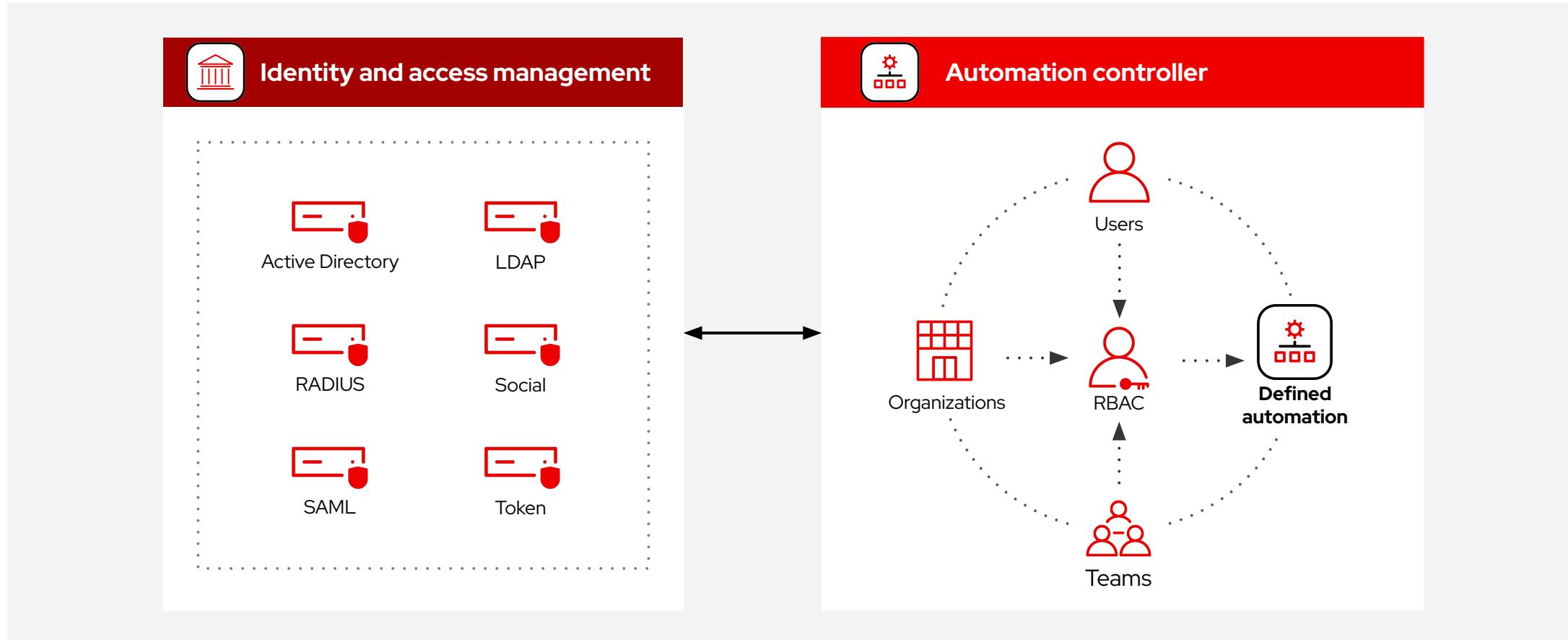


## What is it?

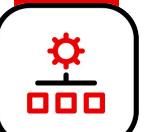
- ▶ Workflows enable the creation of powerful holistic automation, chaining together multiple pieces of automation and events
- ▶ Simple logic inside these workflows can trigger automation depending on the success or failure of previous steps
- ▶ Add approvals to your workflows to enhance governance
- ▶ Integrate other systems, such as ITSM to fit with your existing controls and processes



# Securing your content in automation controller. **Delegate.**



# Role-Based Access Control. Who can use my automation?



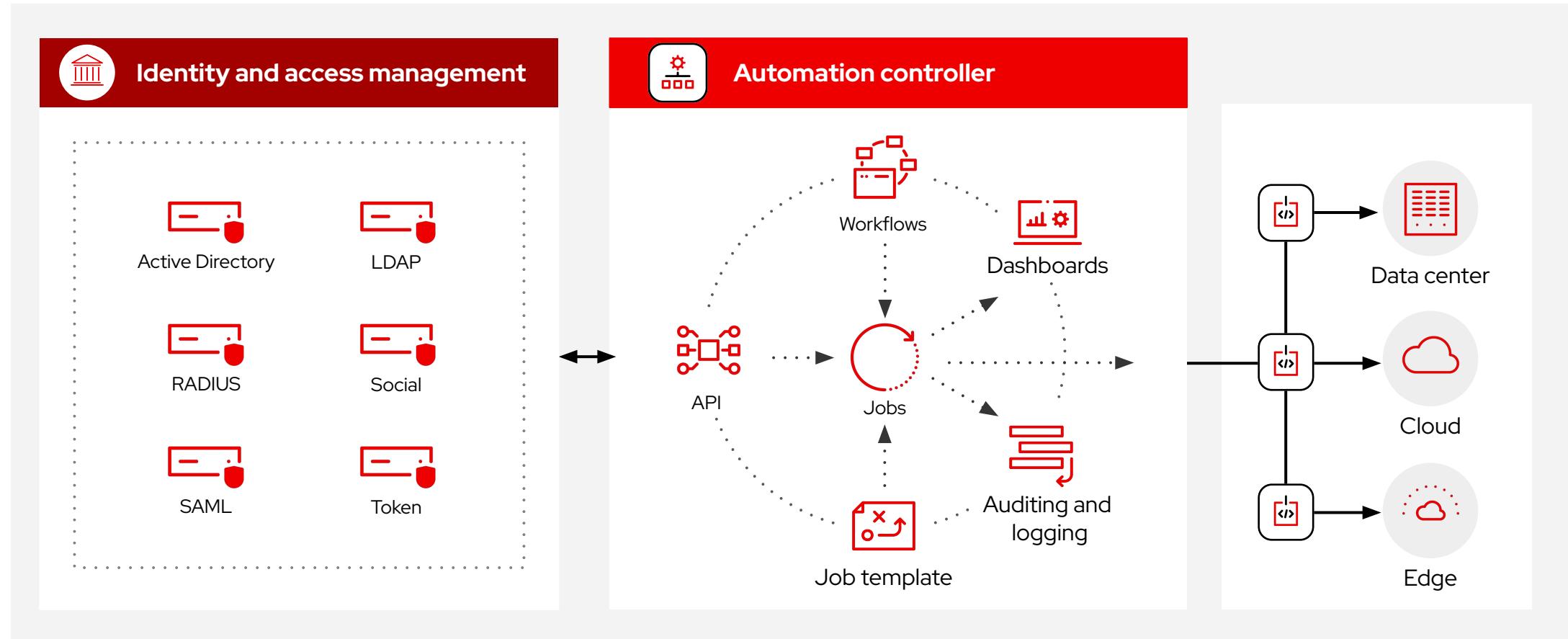
## What is it?

Securely govern access to your automation

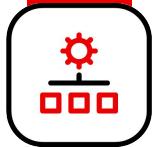
- ▶ Logically group controller objects and grant users and teams read, execute, edit permissions
- ▶ Use predefined roles to grant access
- ▶ Integrates with your existing enterprise authentication systems

Username	First name	Last name	Roles
admin			User Roles System Administrator
austin78	Austin	Austin	User Roles Member System Auditor
jgarcia	Jerry	Jerry	User Roles Member
jdoge	Josie	Josie	User Roles Project Admin

# Launching your content from automation controller.



# Automation jobs. Executing your defined automation.



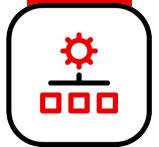
## What is it?

- ▶ Controller launching an instance of defined automation
- ▶ Relaunch automation jobs
- ▶ Use Job Details to view job outputs
- ▶ Troubleshoot automation execution using filtered views

The screenshot shows the 'JOBS' page with a red border. At the top, there's a search bar and a 'KEY' button. Below the header, a list of jobs is displayed:

- 110 - Cleanup Activity Stream (Management Job)
- 109 - Project from Git (SCM Update)
- 108 - Cleanup Job Details (Management Job)
- 101 - WF in WF (Workflow Job)
- 102 - Job template with slicing (Playbook Run)
- 100 - New Workflow Job Template (Workflow Job)
- 87 - Demo Job Template (Playbook Run)
- 88 - Demo Project (SCM Update)

# Automation controller API. Integrate and elevate your automation.



## What is it?

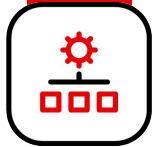
- ▶ The API provides programmatic access to the automation via a defined interface
- ▶ Underneath it is still powered by the same bits and pieces which are at the core: workflows, inventories, etc
- ▶ It offers simple integration into other tools like ITSM, SOAR, etc

The screenshot shows a web-based REST API interface for the Red Hat Ansible Automation Platform. The top navigation bar includes a user profile for 'colin', a 'Log out' button, and several icons. The main content area has a red header bar with the text 'REST API / Version 2 / Execution Environment List'. Below this, a sub-header reads 'Execution Environment List'. On the right side of the sub-header are 'OPTIONS' and 'GET' buttons. A code block below the sub-header contains the following JSON response:

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
X-API-Node: localhost
X-API-Product-Name: Red Hat Ansible Automation Platform
X-API-Product-Version: 4.1.0
X-API-Time: 0.034s

{
  "count": 5,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 5,
      "type": "execution_environment",
      "url": "/api/v2/execution_environments/5/",
      "related": {
        "created_by": "/api/v2/users/1/",
        "modified_by": "/api/v2/users/1/",
        "activity_stream": "/api/v2/execution_environments/5/activity_stream/",
        "unified_job_templates": "/api/v2/execution_environments/5/unified_job_templates/",
        "copy": "/api/v2/execution_environments/5/copy/"
      }
    }
  ]
}
```

# Automation controller logging aggregation.



## What is it?

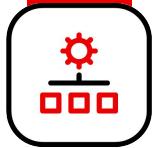
- ▶ Provides the capability to send detailed logs to several kinds of 3rd party external log aggregation services
- ▶ Use Red Hat Enterprise Linux rsyslog to aggregate and redirect logs
- ▶ Native integration exists for multiple 3rd party logging systems
- ▶ Multiple detailed logs are gathered for job events, system facts, users, activity streams, and controller features

The screenshot shows a web-based interface for managing jobs. At the top, there's a red header bar with a search bar and a key icon. Below the header, a sidebar on the left has a 'JOBS' section with a count of 32. The main area displays a list of jobs, each with a green circle icon, a job ID, a name, and a category. The categories include Management job, SCM Update, Workflow job, Playbook Run, and Demo Project. The jobs listed are:

- 110 - Cleanup Activity Stream Management job
- 109 - Project from Git SCM Update
- 108 - Cleanup Job Details Management job
- 101 - WF in WF Workflow job
- 102 - Job template with slicing Playbook Run
- 100 - New Workflow Job Template Workflow job
- 87 - Demo Job Template Playbook Run
- 88 - Demo Project SCM Update

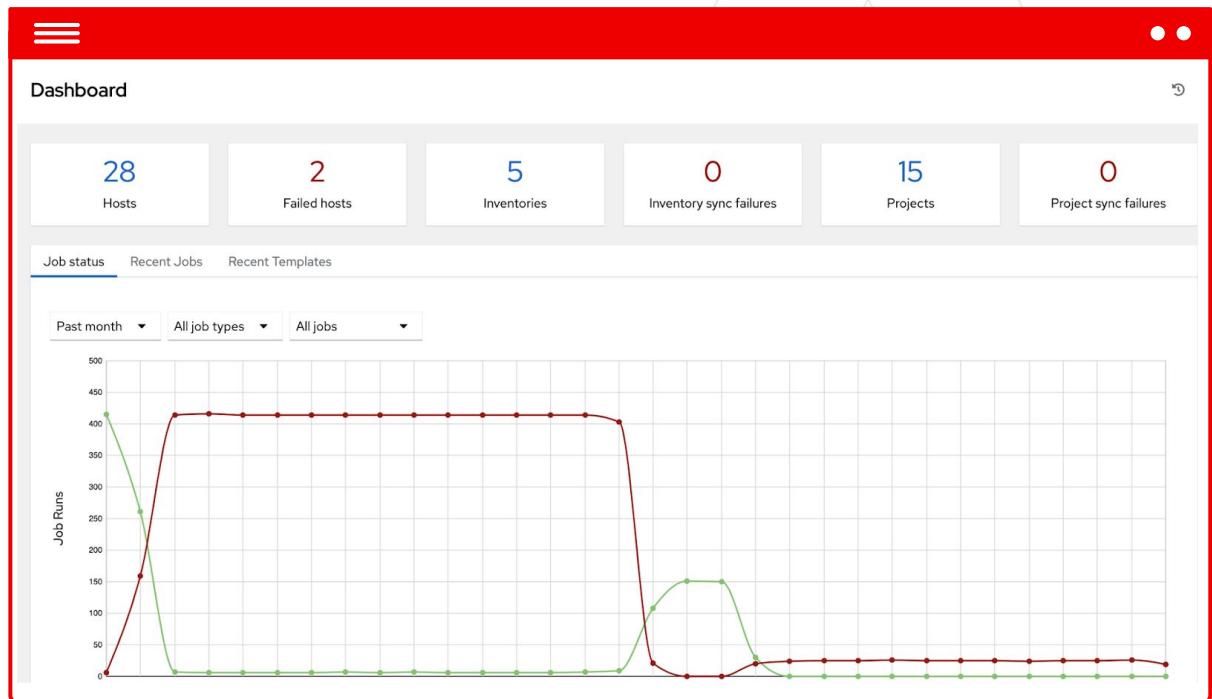
# Automation controller dashboard.

## View your automation status.

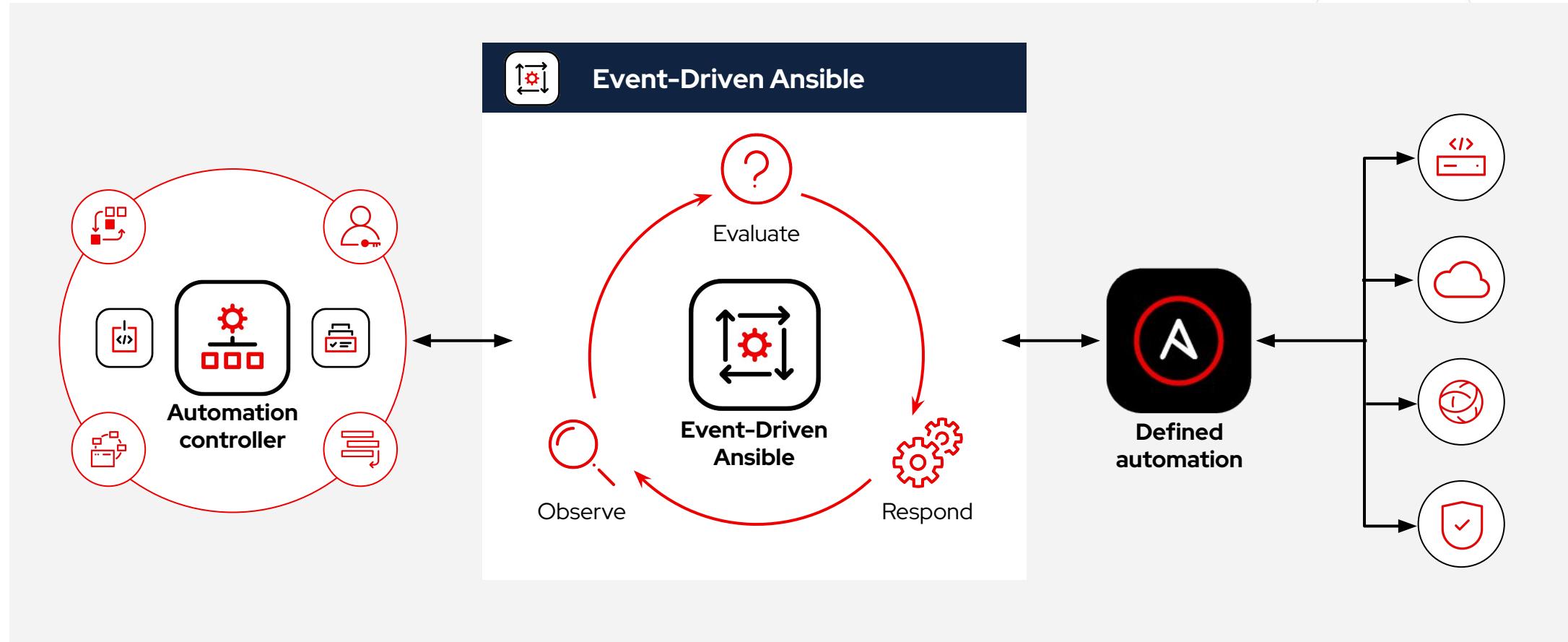


### What is it?

- ▶ Overview of automation controller, the jobs happening, the nodes connected and what worked and failed
- ▶ Provides the ability to quickly spot irregularities and drill into more details



# Event-Driven Ansible. Observe, evaluate, respond.

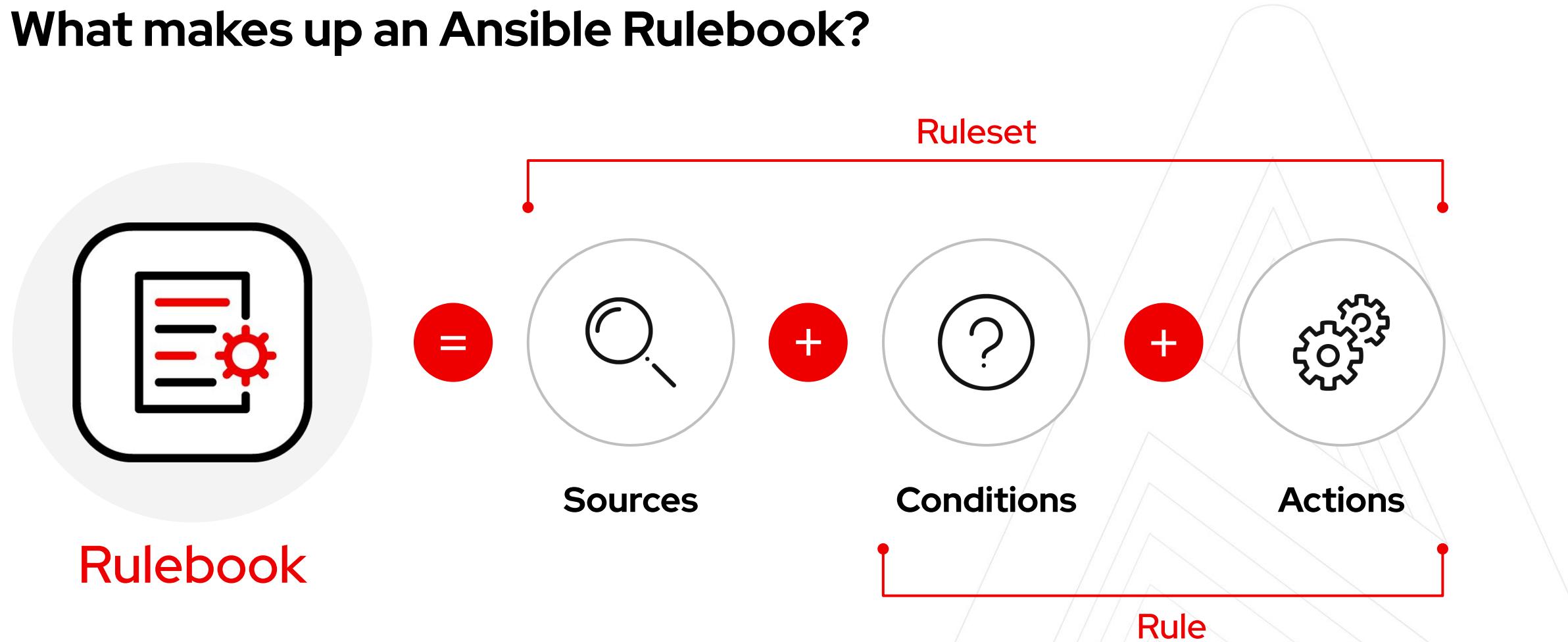


# Ansible Rulebooks

```
---
- name: Capture alertmanager alerts
  hosts: all
  sources:
    - ansible.eda.alertmanager:
        host: 0.0.0.0
        port: 5050
        data_alerts_path: alerts
        data_host_path: labels.instance
        data_path_separator: .
        skip_original_data: true

  rules:
    - name: SELinux was disabled
      condition: event.alert.labels.alertname == "selinux disabled" and
                  event.alert.status == "firing"
      action:
        run_job_template:
          name: Apply baseline
          organization: Default
          job_args:
            limit: "{{ event.meta.hosts }}"
```

# What makes up an Ansible Rulebook?



# Event-Driven Ansible rulesets. Event sources and rules.



## What are they?

- ▶ Top level specification that defines event sources and rules.
- ▶ Defines properties such as Ansible inventory target hosts, gathering facts, and more.

## Building blocks for Rulebooks

- ▶ Multiple rulesets can exist within an Ansible Rulebook.

A screenshot of a terminal window with a black background and light gray text. The window shows a portion of an Ansible YAML file. A blue rectangular highlight covers the line '- name: Capture alertmanager alerts' and the line 'hosts: all'. Above the highlighted lines, there are three ellipsis dots (...).

```
...  
- name: Capture alertmanager alerts  
  hosts: all
```

# Event-Driven Ansible sources. How are events gathered?



## What are they?

- ▶ Event data is gathered from multiple sources using source plugins.
- ▶ One or multiple event source plugins can be configured in a ruleset.

## Source plugins

- ▶ Puts events in the queue to be passed to rules engine.

## Event filters

- ▶ Enables data transformation and clean-up before being passed to rules engine.

```
...
- name: Capture alertmanager alerts
  hosts: all
  sources:
    - ansible.eda.alertmanager:
        host: 0.0.0.0
        port: 5050
        data_alerts_path: alerts
  filters:
    - ansible.eda.json_filter:
        include_keys: ['alert']
```

# Event-Driven Ansible rules. Evaluate and automate.



## What are they?

- ▶ Rules determine actions taken on events based on conditions using simple YAML structure.

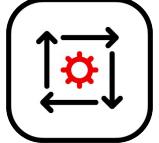
## Conditions

- ▶ Evaluates event data using “If-Then-Else” structure.
- ▶ Single condition or multiple conditions supported.

## Actions

- ▶ Triggered once rule conditions are met
- ▶ Processed sequentially once conditions are met.

```
rules:  
  - name: SELinux was disabled  
    condition: event.alert.labels.alertname ==  
               "selinux disabled"  
    action:  
      run_job_template:  
        name: Apply baseline  
        organization: Default  
        job_args:  
          limit: "{{ event.meta.hosts }}"
```



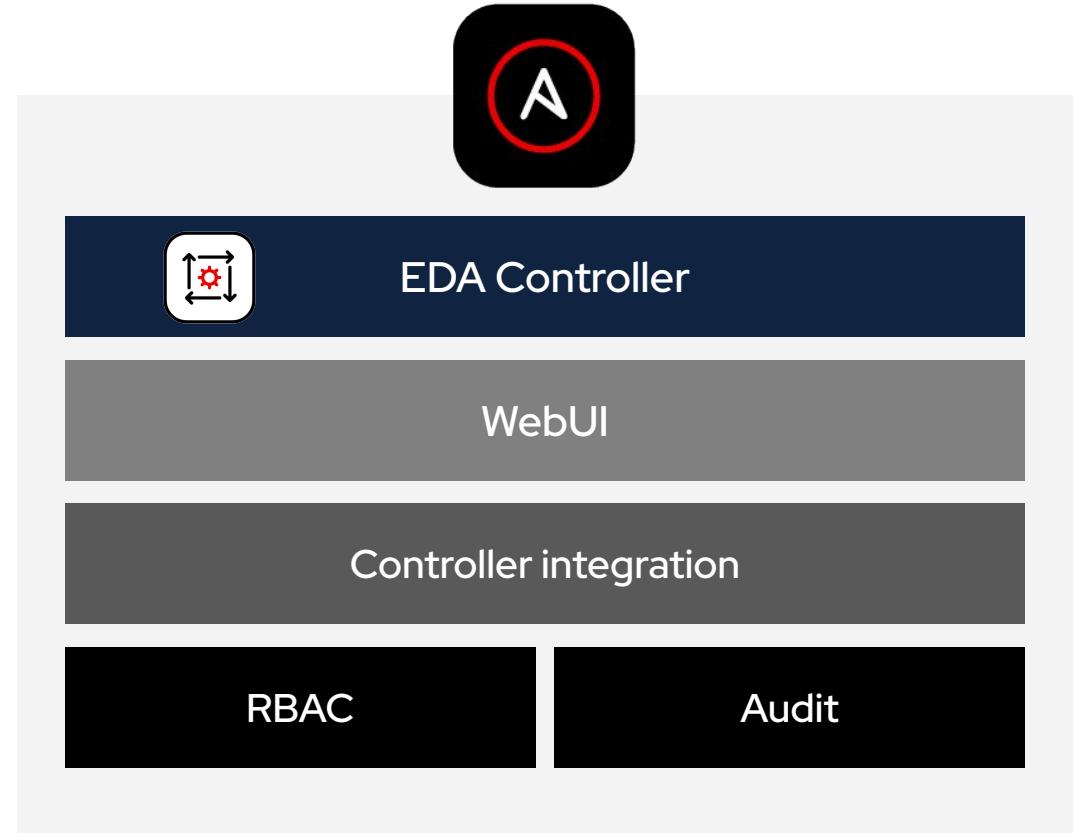
# Event-Driven Ansible controller. Define and manage.

## What is it?

- ▶ EDA Controller enables users to centrally manage Event-Driven Ansible across the enterprise.

## EDA controller provides:

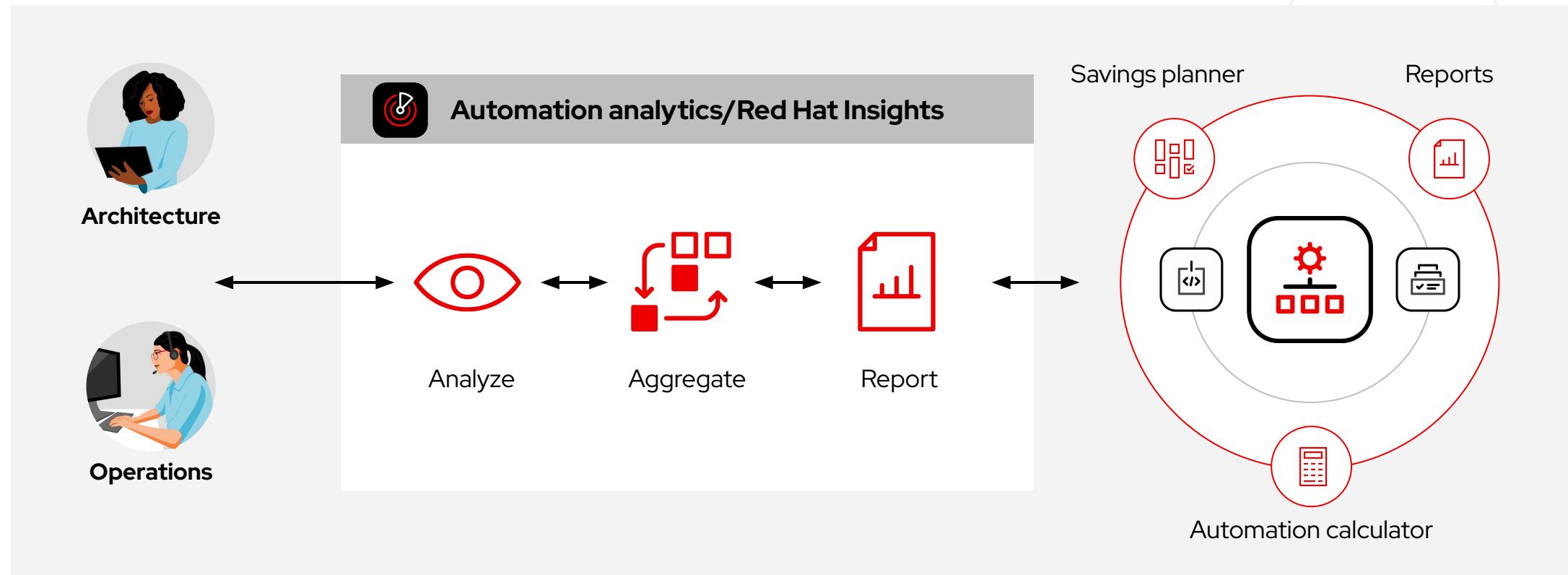
- ▶ WebUI.
- ▶ Rulebook activation.
- ▶ Role-based access control.
- ▶ Auditing trail.
- ▶ Secure automation controller integration.



# Red Hat Insights for Ansible Automation Platform.

## Analyze, aggregate and report.

[console.redhat.com](https://console.redhat.com)



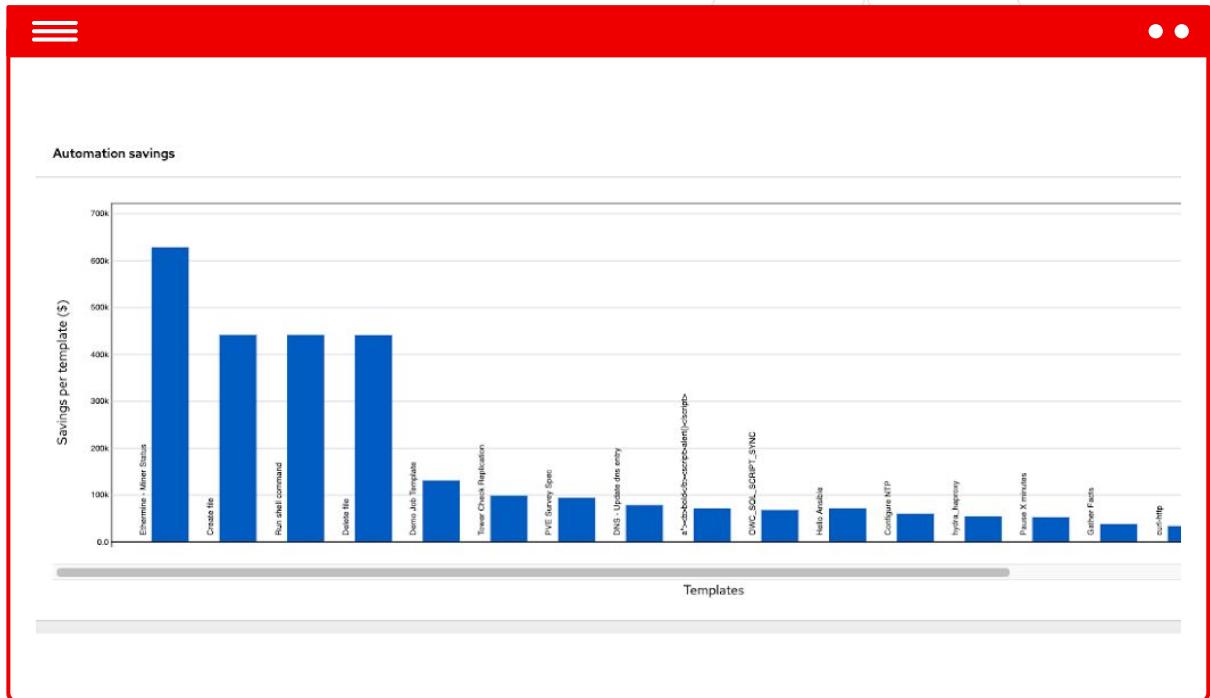
# Automation calculator. Analyze and aggregate.



## What is it?

Track and measure the total value of automation efforts by time and cost savings

- ▶ Automation calculator looks at the total ROI of your automation
- ▶ Calculations are based on automation data across your organization
- ▶ Job template savings are compared to costs of performing the tasks manually
- ▶ Measure the ROI of your automation investment and identify automation that contributes the most savings



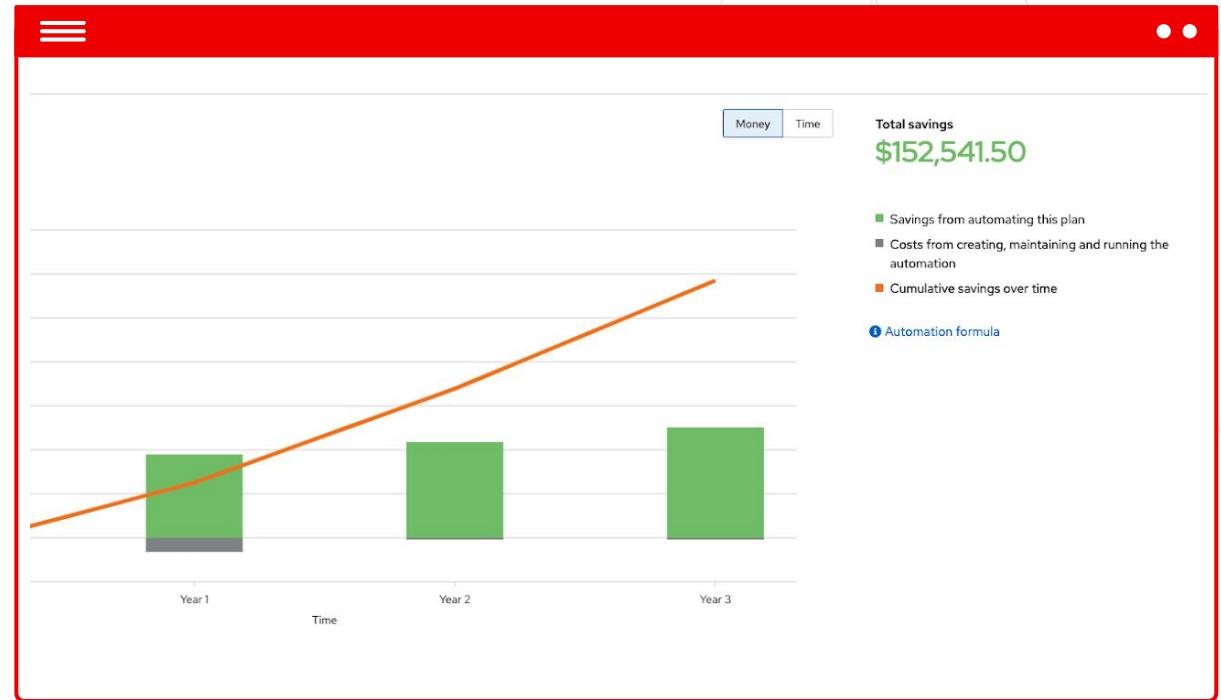
# Automation Savings Planner. Analyze and aggregate.



## What is it?

Prioritize automation by projecting the 1- to 3-year savings by time and money for each automation task

- ▶ Predict ROI and time savings based on a given automation task
- ▶ Use your automation data from your organization to prioritize tasks and projects based on business value



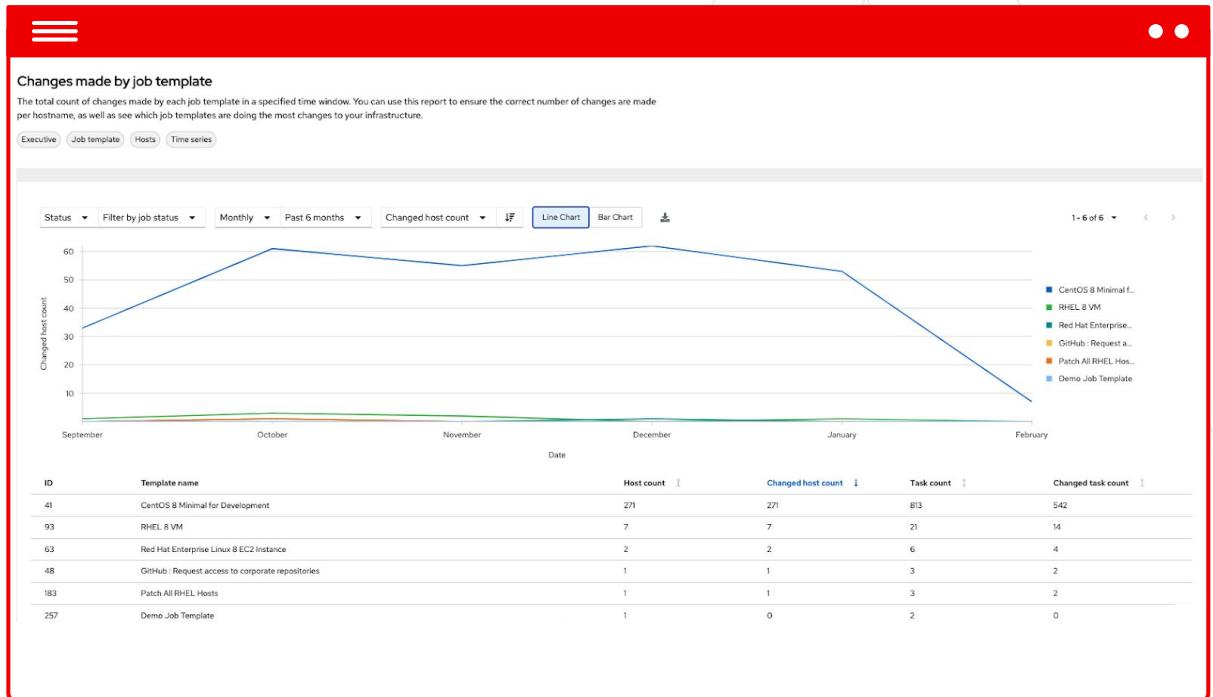
# Reports. The holistic view of your automation platform.



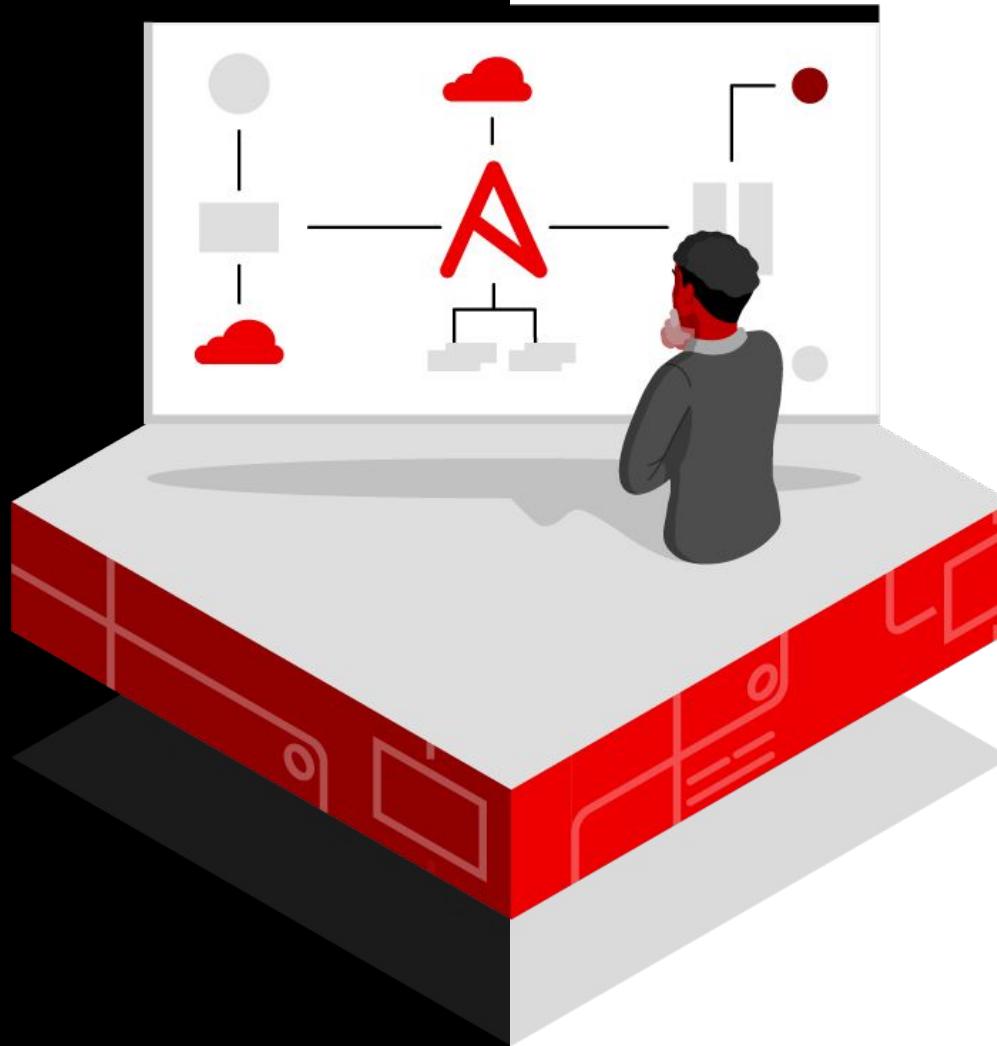
## What is it?

Unified, visual dashboards of Ansible Automation Platform key metrics across clusters

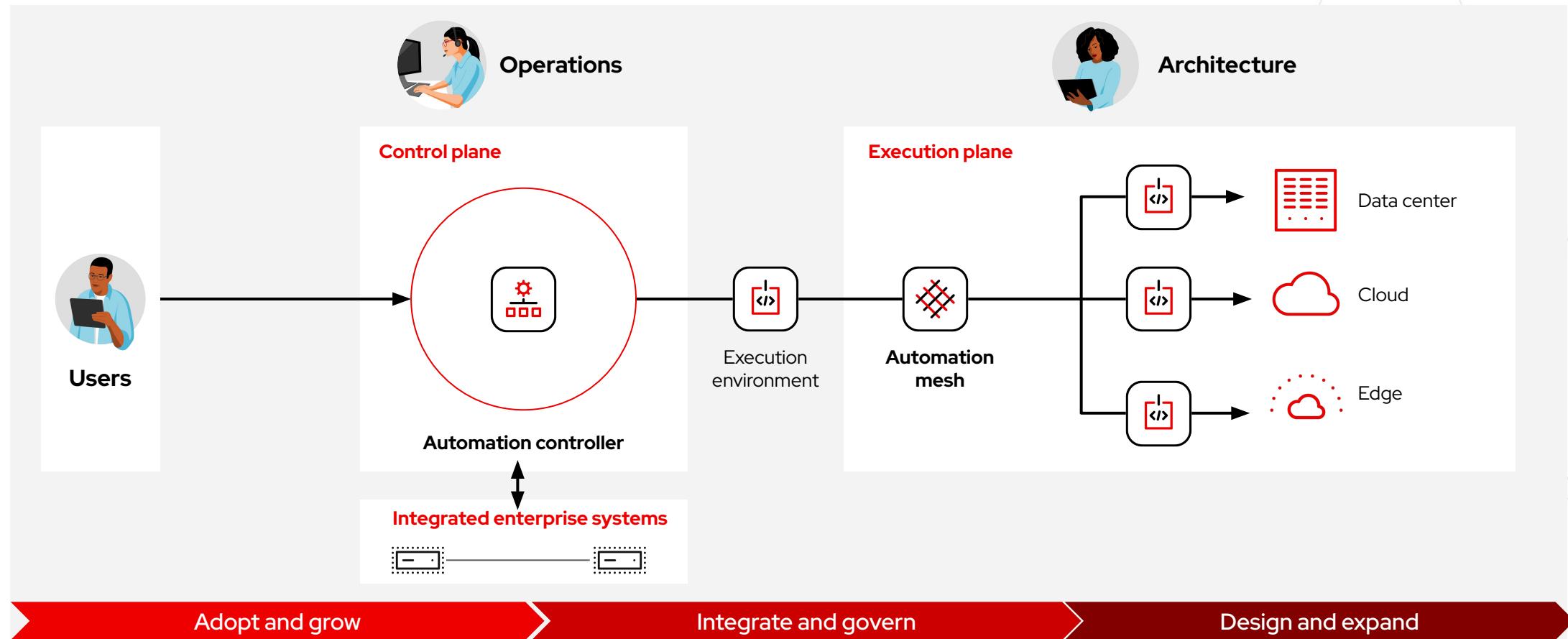
- ▶ Reveal most used Ansible Playbooks, modules, and deployment pass/fail rates
- ▶ Filter information based on automation controller clusters in real-time
- ▶ Use historical data to predict and improve the automation practice
- ▶ Measure the value of your Ansible Automation Platform subscription
- ▶ Export reports and share with your organization



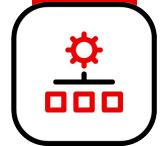
# Scale



# The automation content life cycle. **Scale.**



# Automation controller approvals. Integrate and govern.



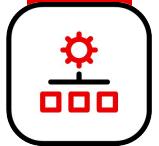
## What is it?

- ▶ Adds human interaction to the automation for administration and governance
- ▶ Available at the operational level on the Automation controller UI

The screenshot shows a web-based interface for workflow approvals. At the top, a navigation bar includes 'Workflow Approvals' and a back arrow. Below it, a sub-header reads 'Deploy to Prod?'. A 'Details' section contains the following information:

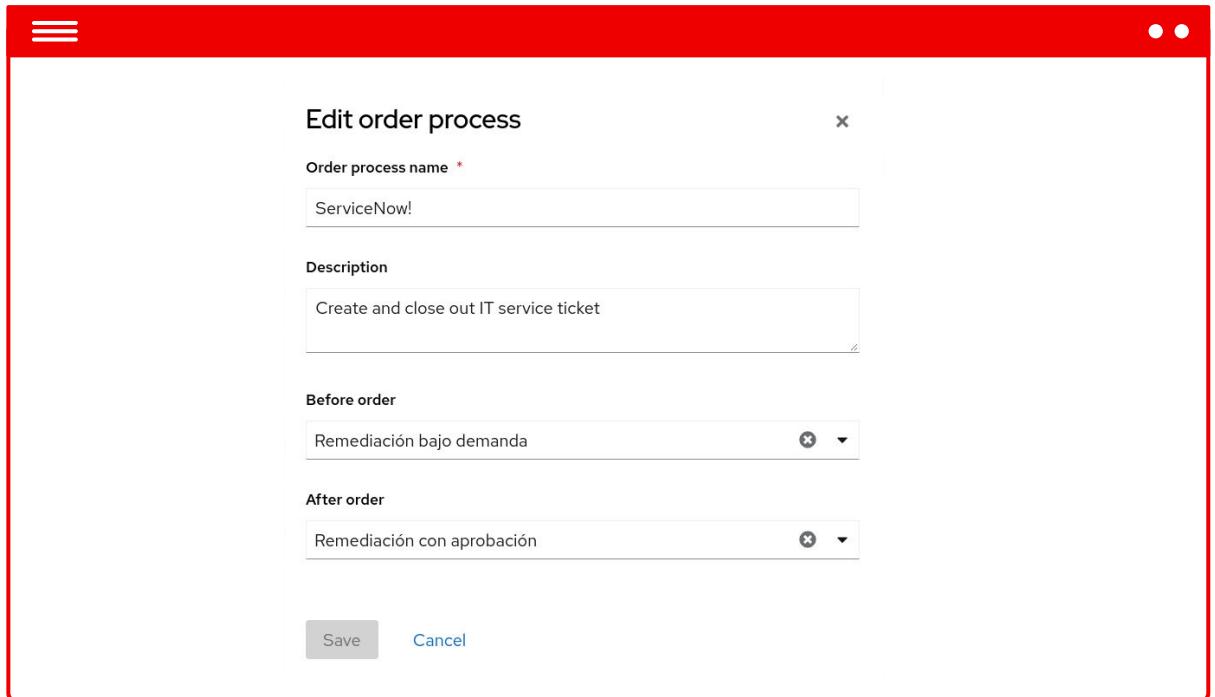
Name	Deploy to Prod?	Description	Expires
Workflow Job	5 - DevOps Workflow	Workflow Job Template	DevOps Workflow
Last Modified	8/5/2022, 5:11:01 PM	Elapsed	00:00:13
		Approve	Deny

# IT service management integration (ITSM). Integrate and govern.



## Incorporate automation into your ITSM

- ▶ Integrate high level workflows in existing ITSM toolsets with the automation platform.
- ▶ Apply organization governance and integrate your automation to larger business processes
- ▶ Have the automation platform reach out to the ITSM system whenever things are changing, including data transmission between the tools



Edit order process

Order process name \*

ServiceNow!

Description

Create and close out IT service ticket

Before order

Remediación bajo demanda

After order

Remediación con aprobación

Save Cancel

# Automation mesh. Design and expand.

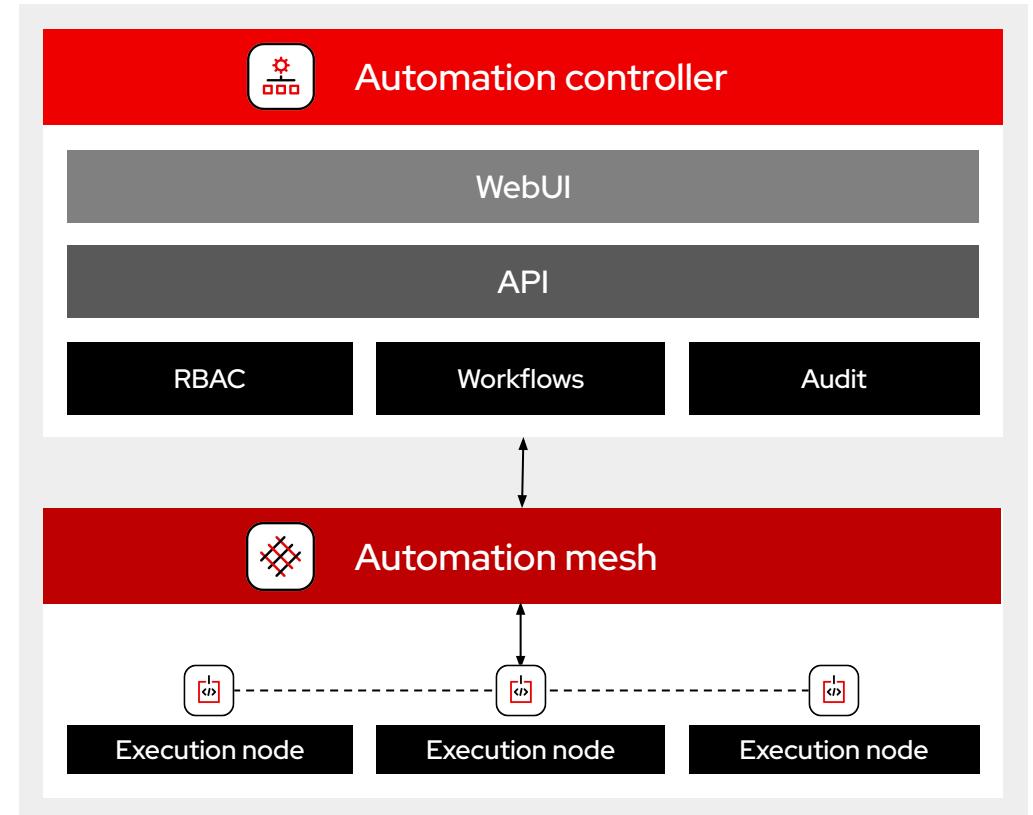


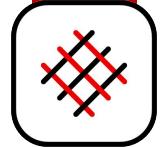
## What is it?

- ▶ Simple, reliable framework to scale automation independently with little or no downtime
- ▶ Distributed and localized execution across worker nodes
- ▶ Scale across segmented and remote networks with native resiliency
- ▶ Flexible design choices to deliver automation across networks and design redundant topologies

## Improved communication and security

- ▶ Worker node health checks
- ▶ Bi-directional, secured (TLS) communication between nodes
- ▶ Centralized, secured management with automation controller





# Automation mesh node types. The execution plane.

## What is it?

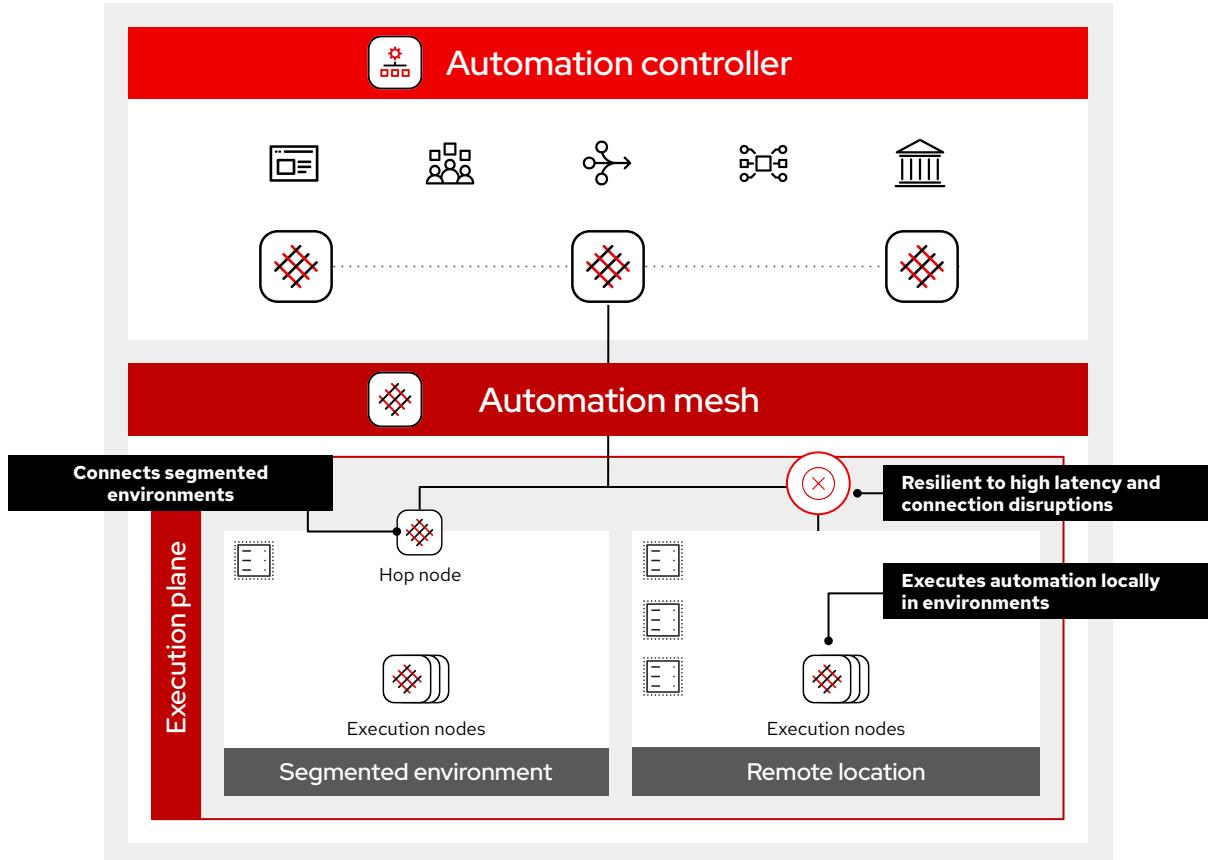
- ▶ Dedicated capacity to running playbooks
- ▶ Flexible designs possible across geographies and networks
- ▶ Resilient to high latency and connection disruptions
- ▶ Run automation without direct connection to controller

## Execution node

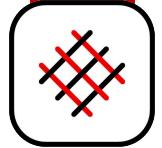
- ▶ Dedicated capacity to run automation and no controller runtime functions executed

## Hop node

- ▶ Dedicated to route traffic across execution nodes not directly connected to controller and cannot run automation



# Automation mesh node types. The control plane.



## What is it?

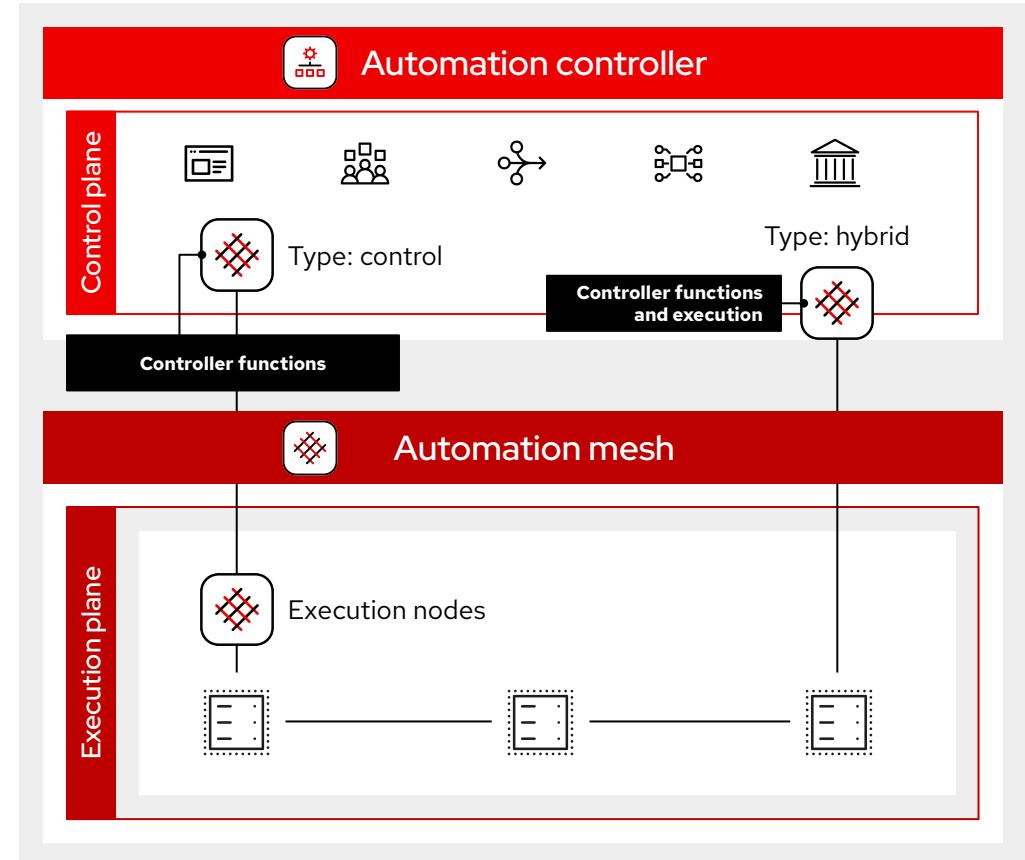
- ▶ Dedicated capacity for automation controller runtime functions
- ▶ Scale with little or no downtime
- ▶ Reduced operational overhead

## Hybrid nodes

- ▶ Default node type for controller nodes
- ▶ Perform controller functions and execute automation

## Control nodes

- ▶ Capacity is dedicated to controller functions
- ▶ Automation execution capability is disabled



# Automation mesh. Management and visibility.

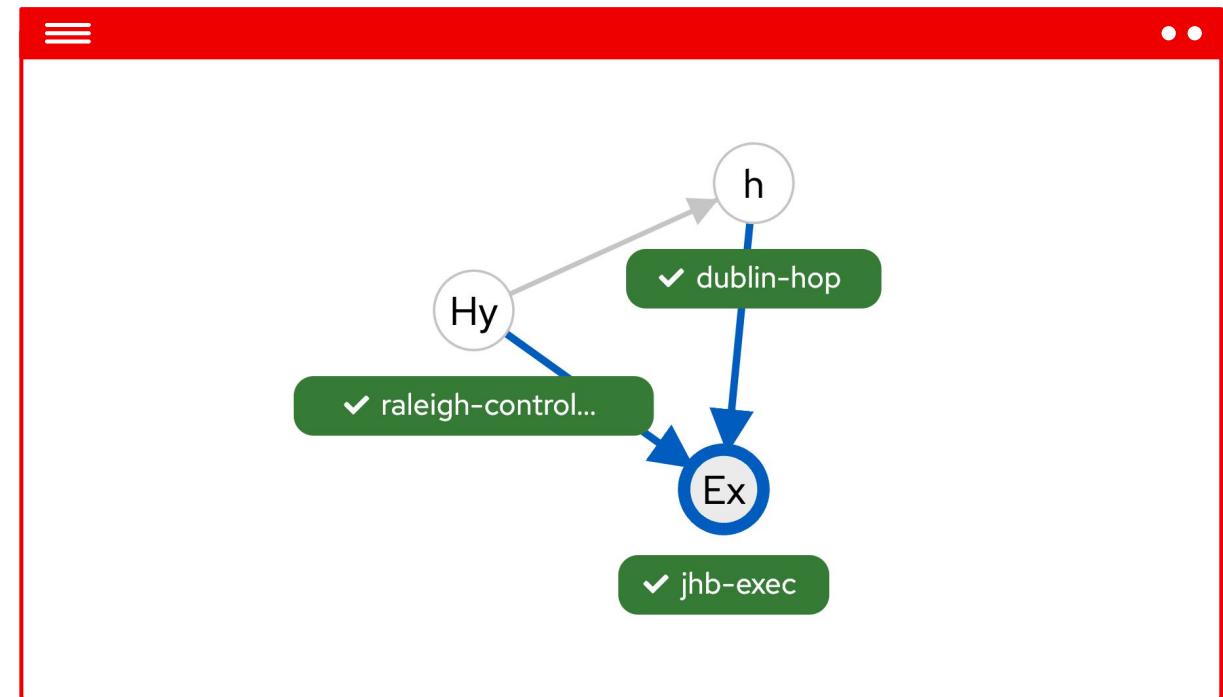


## Instance groups

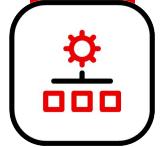
- ▶ Logically group and assign mesh execution capacity
- ▶ Predefined controlplane and default instance groups
- ▶ Configure instance groups via API, WebUI or installer

## Topology Viewer

- ▶ View existing mesh topologies across environments

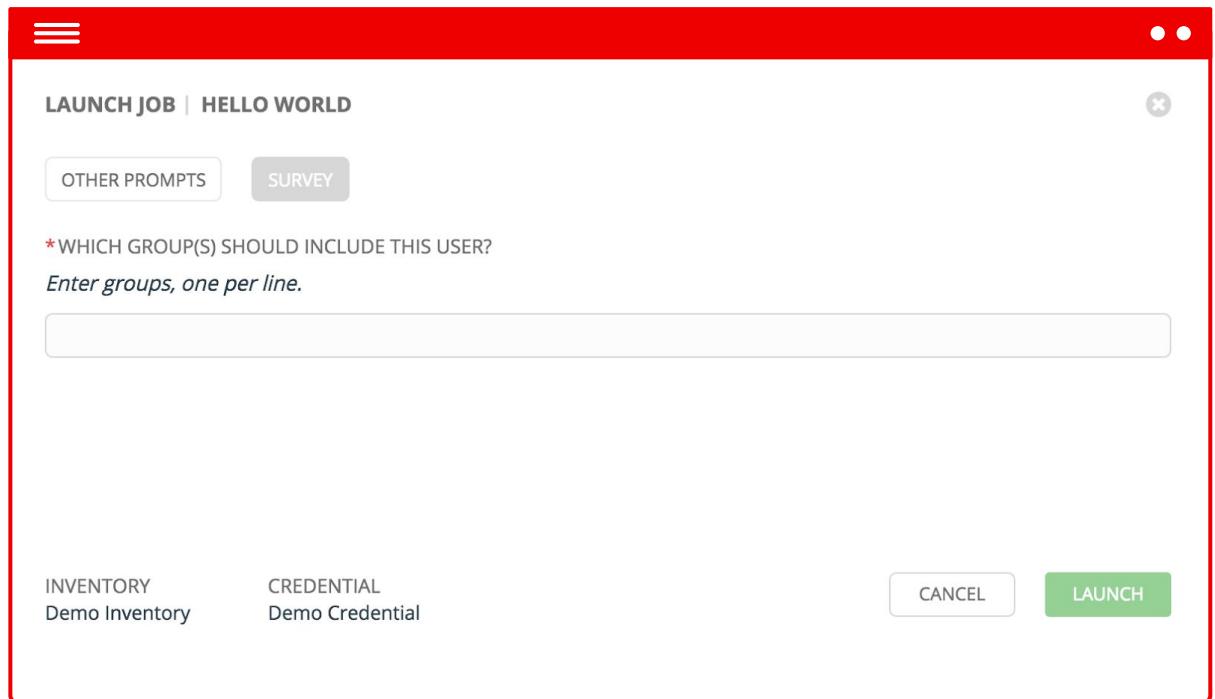


# Automation controller surveys. Adopt and grow.

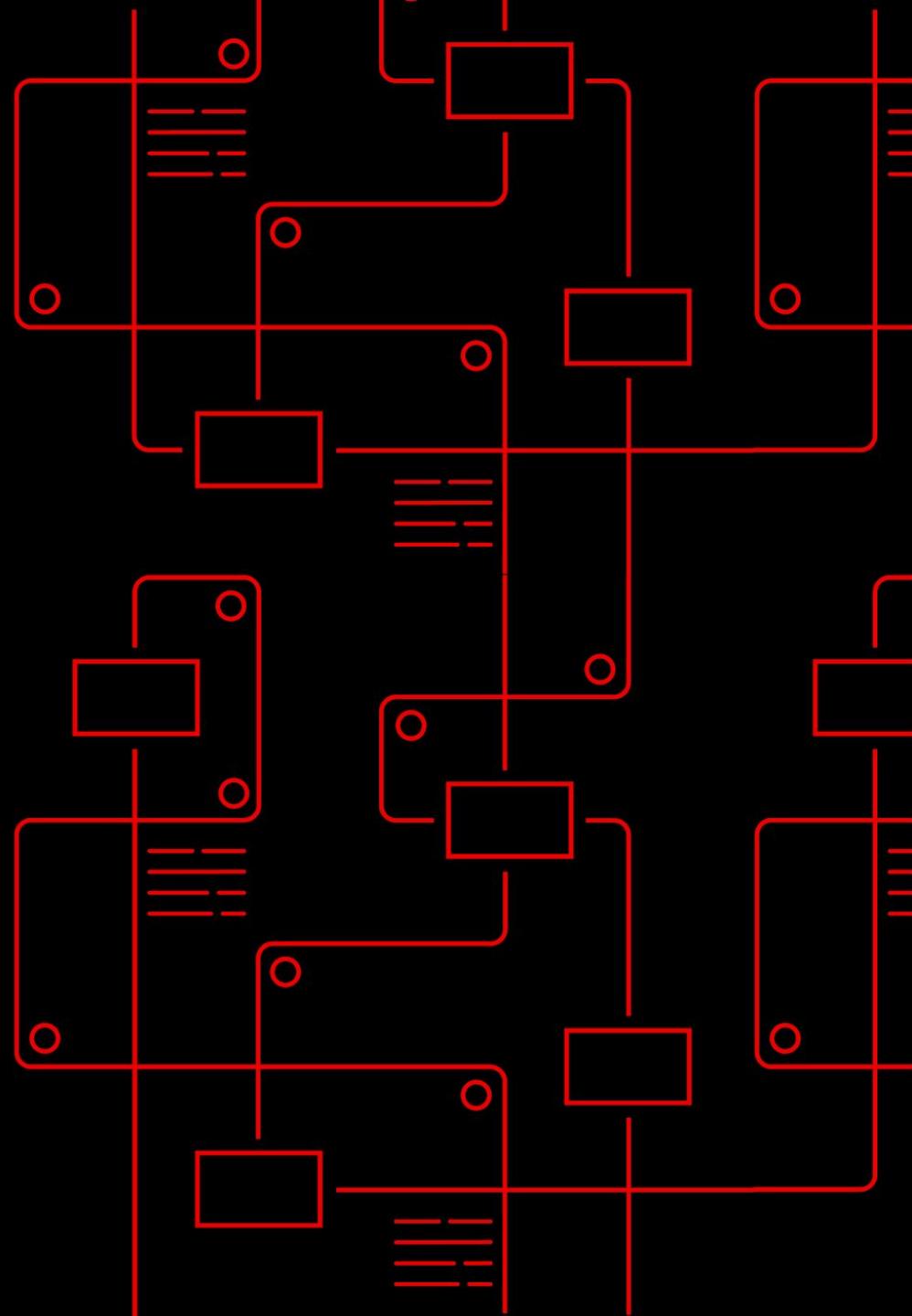


## What is it?

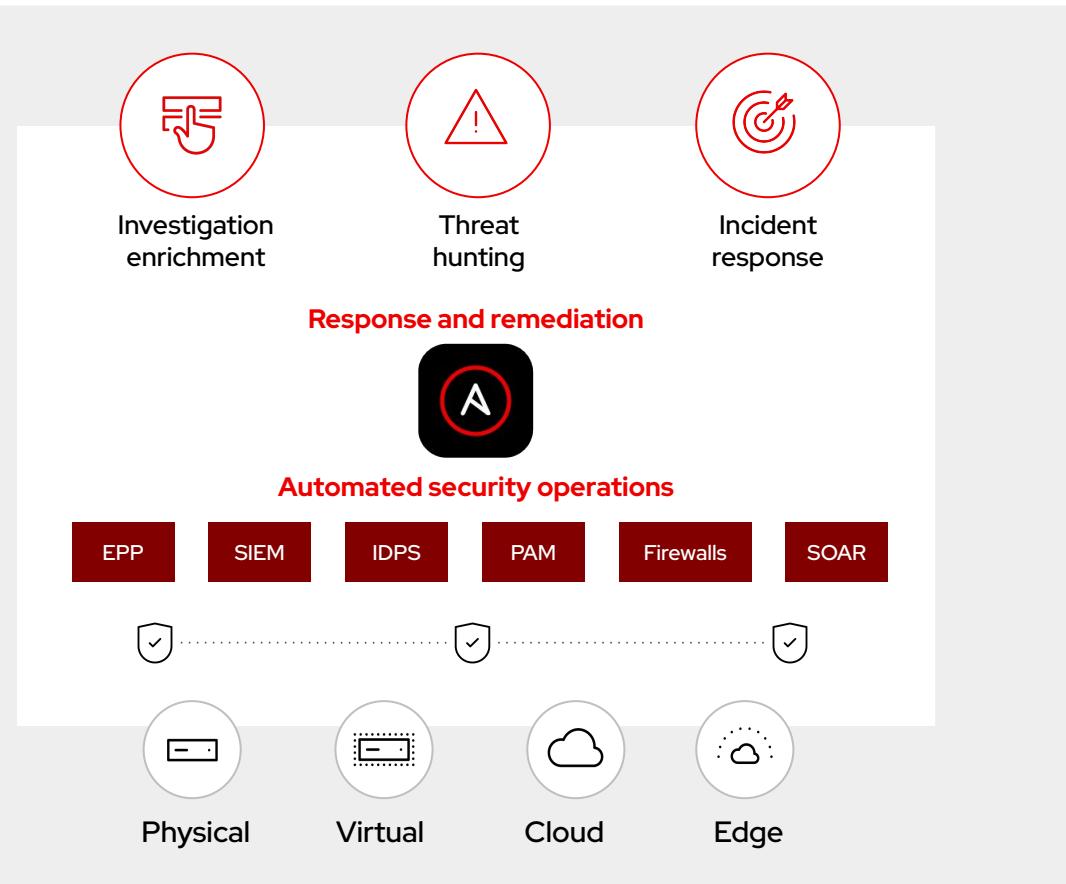
- ▶ User-friendly, self-service interface in automation controller
- ▶ Abstracts complexity using question and answer format
- ▶ Best suited for teams directly accessing automation and close to the automation practice
- ▶ Access and execution governed using controller features



# Use cases



# Ansible security automation. Response and remediation.



## Investigation enrichment

Enabling programmatic access to log configurations such as destination, verbosity, etc

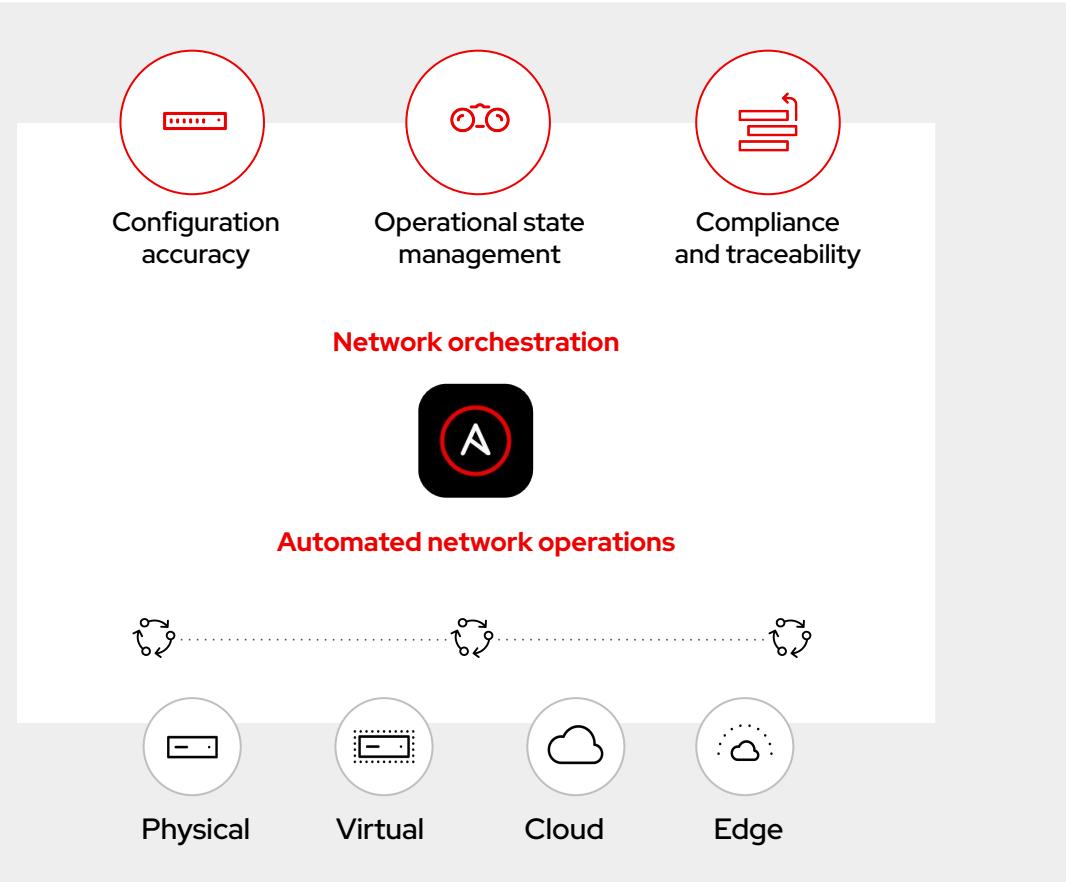
## Threat hunting

Automating alerts, correlation searches, and signature manipulation

## Incident response

Creating new security policies to whitelist, blacklist, or quarantine a machine

# Ansible network automation. Next-gen network operations.



## Configuration accuracy

- ▶ Config backup and restore
- ▶ Scoped configuration management

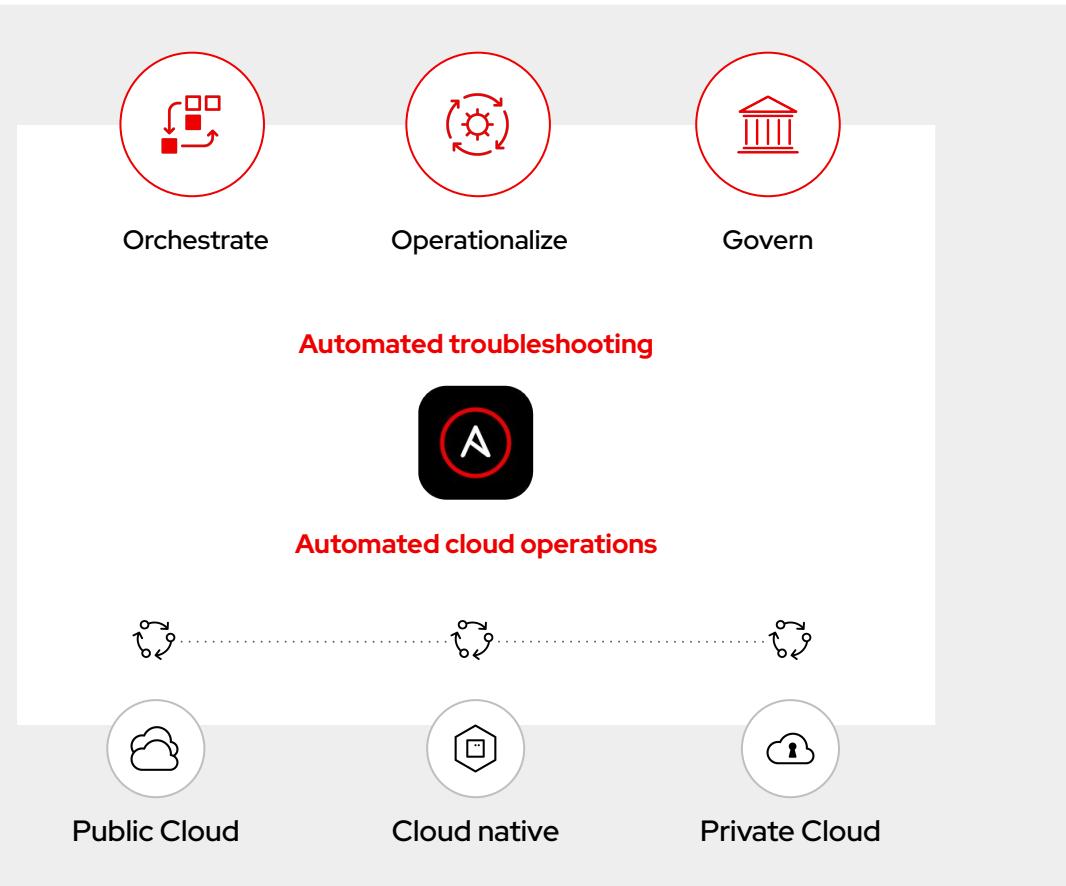
## Operational state management

- ▶ Dynamic documentation
- ▶ Automated NetOps

## Compliance and traceability

- ▶ Operational state validation
- ▶ Network compliance

# Ansible hybrid cloud automation. Tame your clouds.



## Orchestrate

- ▶ Deployment and retirement
- ▶ Infrastructure coordination
- ▶ Cloud migration

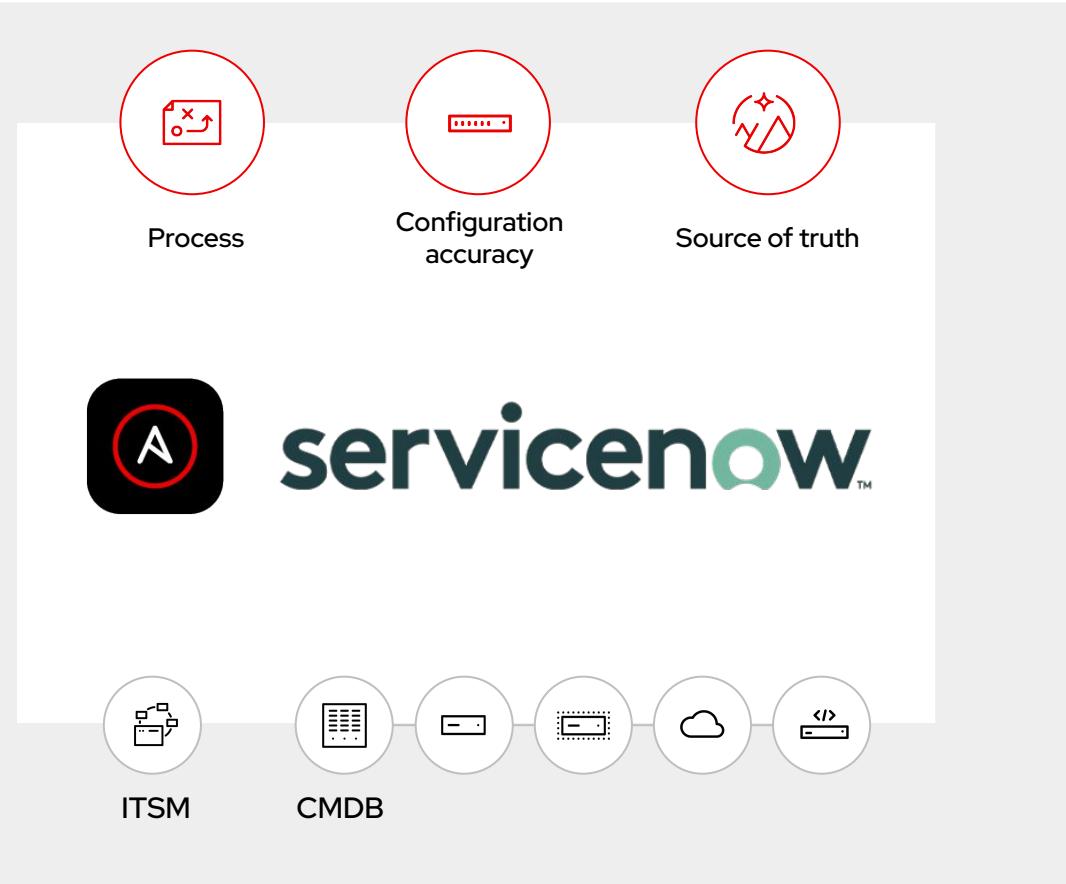
## Operationalize

- ▶ Infrastructure visibility
- ▶ Cloud operations
- ▶ Automated troubleshooting

## Govern

- ▶ Business continuity
- ▶ Cost management
- ▶ compliance

# Ansible Automation Platform for ServiceNow solution.



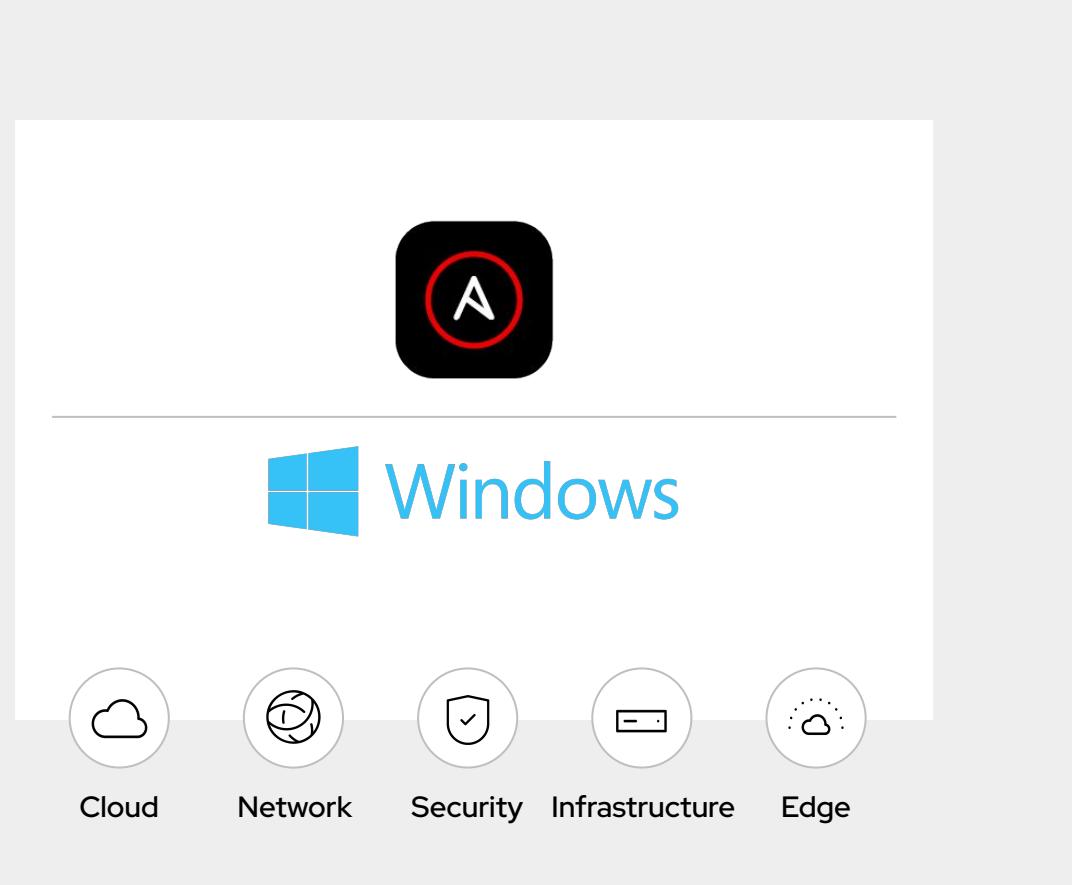
## IT Service Management (ITSM)

- ▶ Create and update records
  - Incident, problem, problem task, change request
- ▶ Assign items to user accounts
- ▶ Attach files to records
- ▶ Advanced queries of record types
- ▶ Support for custom mappings (modified choice lists)

## Configuration Management Database (CMDB)

- ▶ Advanced queries of configuration items
- ▶ Update configuration items after automated changes
- ▶ CMDB as inventory source for automation
- ▶ Batch modifications of configuration items

# Ansible Automation Platform for Windows.



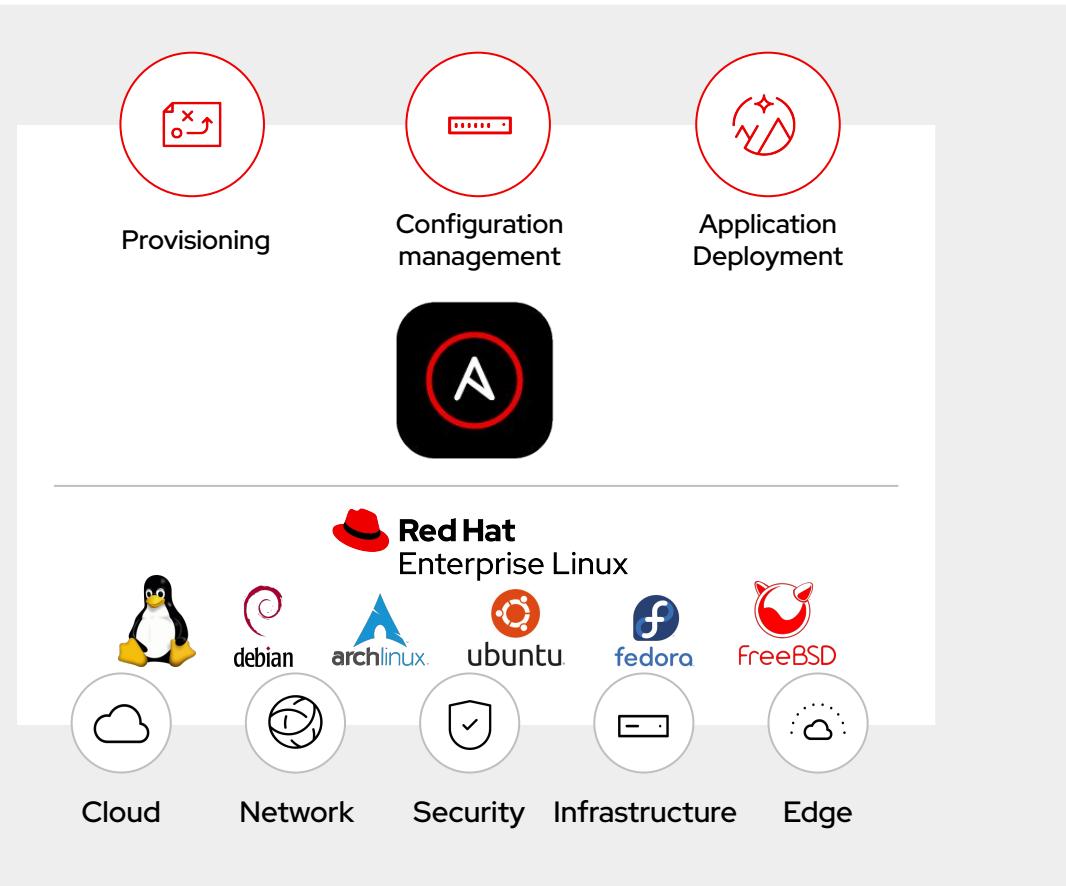
## Day 1 configuration

- ▶ Install and uninstall MSIs, .exes
- ▶ Chocolatey package manager integration
- ▶ Start, stop, and manage Windows services
- ▶ Template and apply registry updates
- ▶ Flexible authentication support

## Day 2 operations

- ▶ Manage and install Windows updates across reboots
- ▶ Create and manage local users
- ▶ Create and manage domain controller/member server state
- ▶ Manage certificates
- ▶ Fetch files from remote hosts
- ▶ Push and execute Powershell scripts
- ▶ Leverage Powershell DSC resources

# Ansible Automation Platform for Linux.



## Provisioning

- ▶ Work seamlessly with bare metal, virtualized and cloud infrastructure
- ▶ Easily patch, upgrade and maintain Linux servers
- ▶ Automation can handle reboots and ad-hoc changes

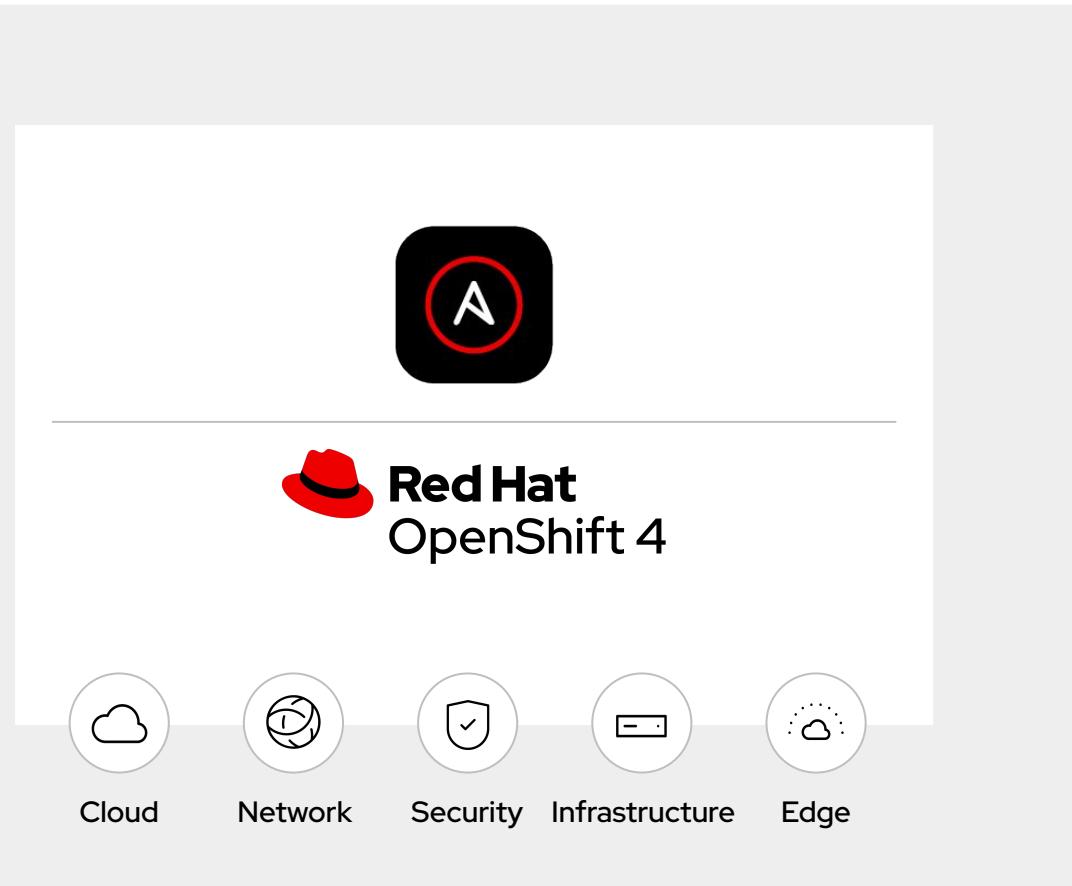
## Configuration Management

- ▶ Built-in support for Linux file and user management
- ▶ Full integration of Jinja2 templating library
- ▶ Support for Idempotence across numerous Linux modules

## Application Deployment

- ▶ Full support of dnf, yum and apt packaging tools
- ▶ Start, stop, and manage Linux services
- ▶ Check operational state and verify application deployments

# Ansible Automation Platform for Red Hat OpenShift.



## Infrastructure coordination

- ▶ Coordination of existing off-cluster IT infrastructure and services with cloud-native systems

## Lifecycle management

- ▶ Automation of Red Hat OpenShift infrastructure and applications lifecycle management

## Day 2 configuration

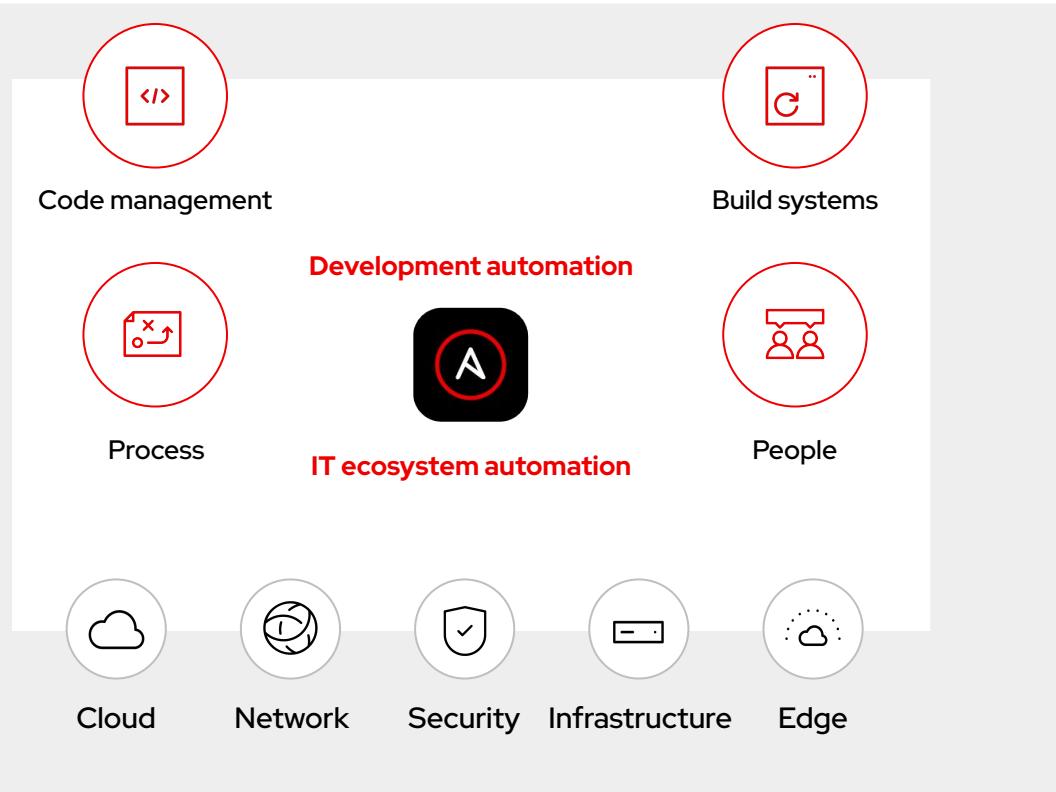
- ▶ Repeatable “last mile” automation for Red Hat OpenShift cluster configuration and management

## Business continuity

- ▶ End-to-end business continuity and disaster recovery (DR) automation of Red Hat OpenShift clusters

# Ansible Automation Platform and DevOps.

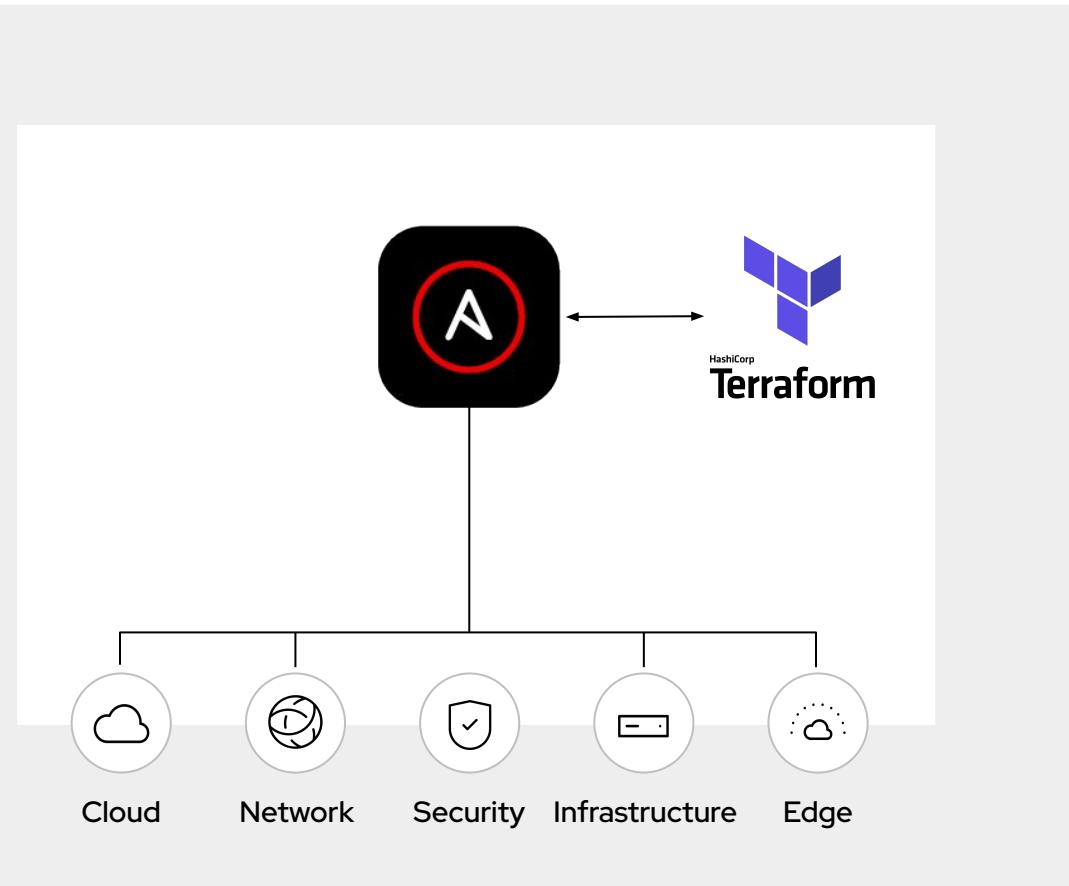
## Deliver consistently, reliably, and rapidly.



### Accelerate DevOps throughout your IT ecosystem

- ▶ **Align to business goals** with outcome-based IT process automation.
- ▶ **Orchestrate and integrate** using the API and webhooks
- ▶ **Simple, pervasive automation language** used across the IT ecosystem.
- ▶ **Extend capabilities** using Ansible Content Collections.
- ▶ **Improve CI/CD security and governance** with approvals, auditing, and RBAC
- ▶ **Speed up development cycles** with consistent dev, test, and prod environments

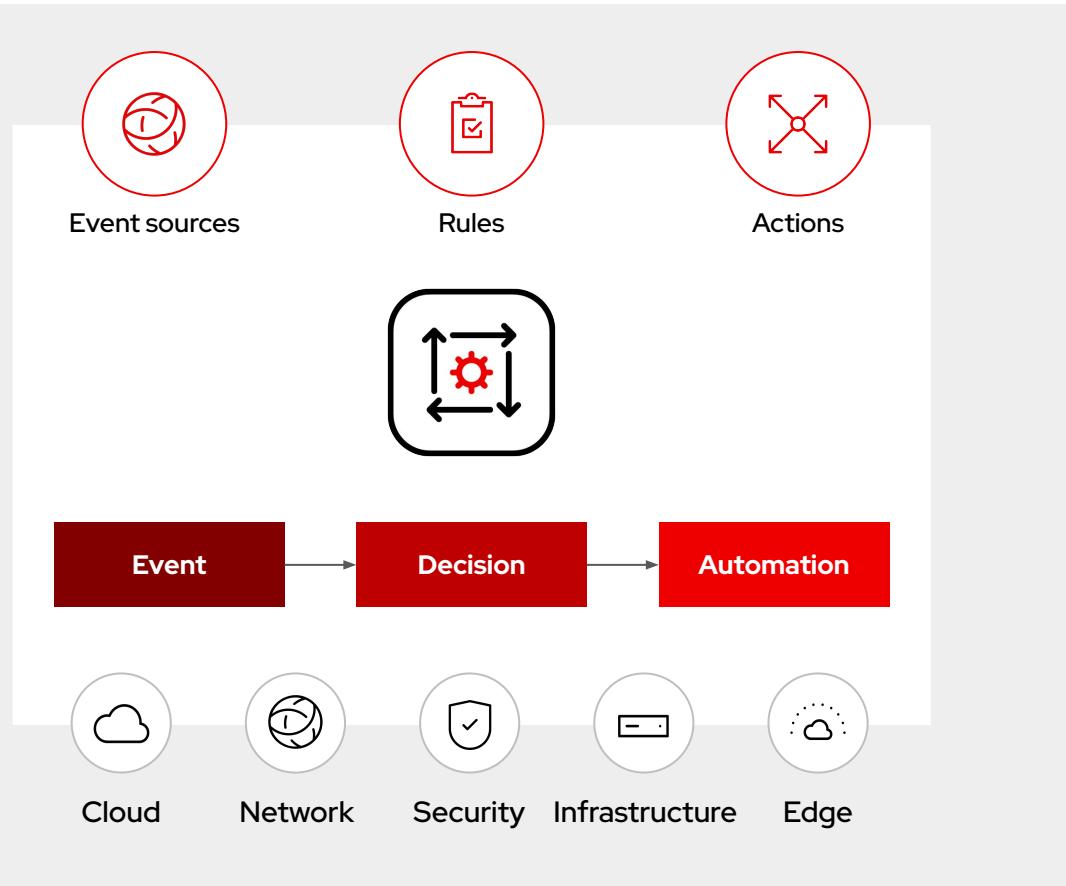
# Everything as code. Everything is possible.



## Infrastructure and configuration as code

- ▶ **Ansible creates** infrastructure manifests and triggers the provisioning and deprovisioning of infrastructure.
- ▶ **Dynamic inventory** management provides access to newly provisioned infrastructure without manual intervention.
- ▶ **Combine tool chains** to deliver infrastructure as code and configuration as code. Allowing for complete management of infrastructure life cycles with post-provisioning tasks.

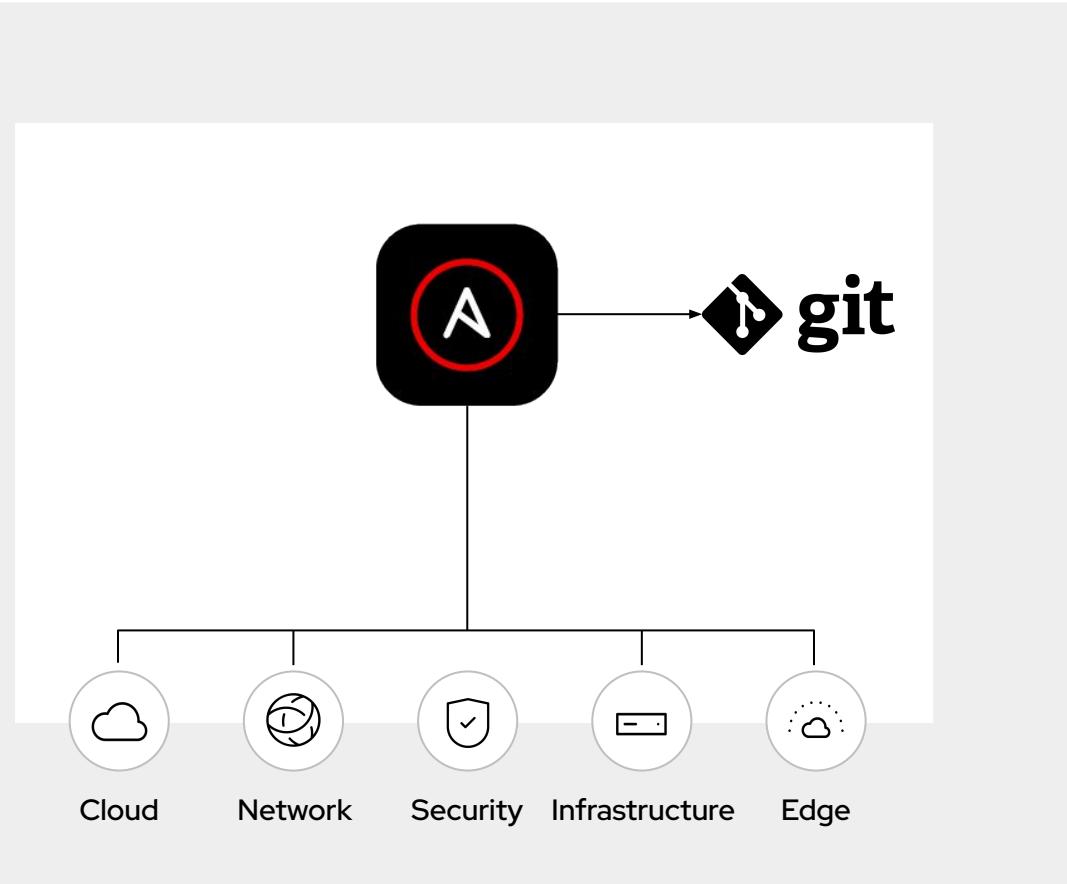
# Event-Driven Ansible. Observe Evaluate Automate



## Remediation and Observation from events

- ▶ **Source** plugins provide Event-Driven Ansible the ability to listen for events which can be processed through rulebooks.
- ▶ **Rules** in the form of Rulebooks allow us to create event conditions which once met will trigger an action.
- ▶ **Actions** give us the ability to trigger playbooks, modules, notifications and further event triggers based on the conditions that have been met by a specific event.

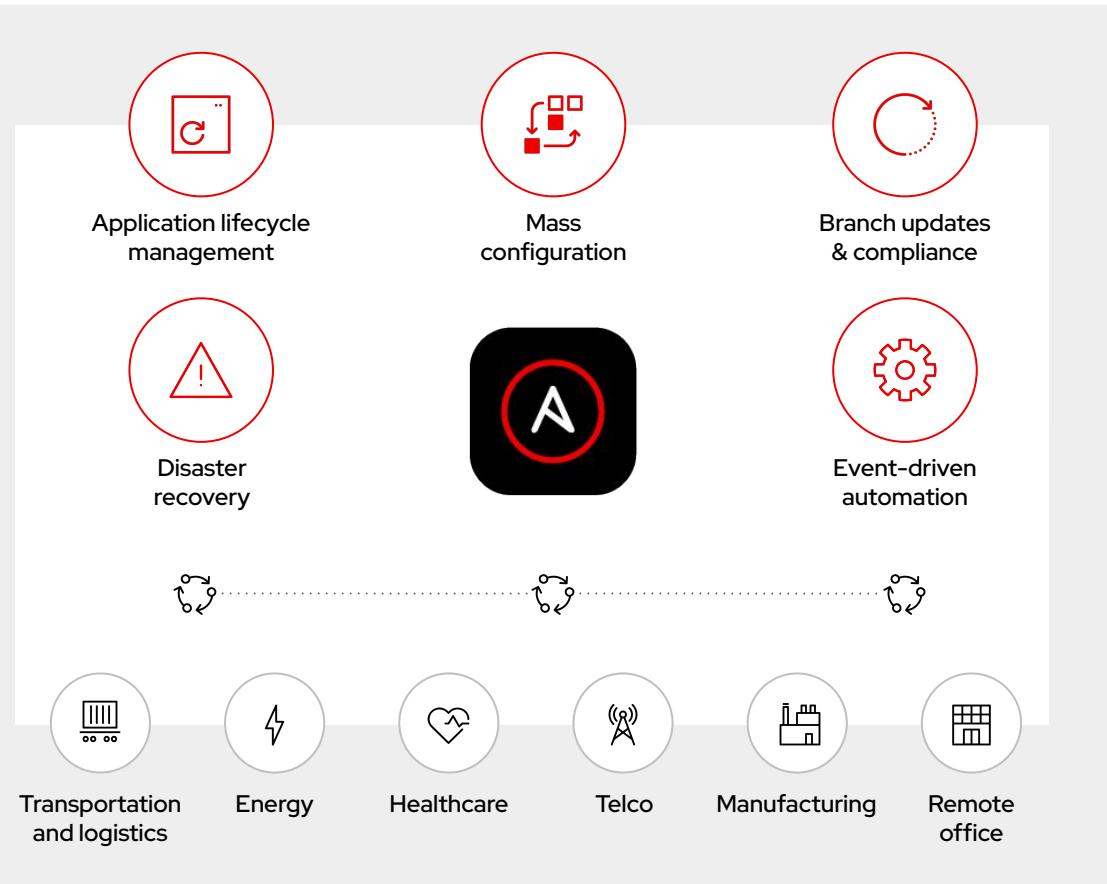
# GitOps. IaC, merge requests, and CI/CD.



## Ansible Automation Platform and GitOps

- ▶ **GitOps** is a framework that uses Git repositories as a single source of truth (SoT) to deliver infrastructure as code.
- ▶ **Infrastructure as Code (IaC)** is stored within a Git repository as the SoT.
- ▶ **Merge Requests (MRs)** is the change mechanism for any updates to the infrastructure.
- ▶ **Continuous Integration/Continuous Delivery (CI/CD)** automates the infrastructure updates.
- ▶ **Ansible triggers and automates** the IaC updates via Ansible workflows while acting as a CI/CD tool and providing complete management of the entire lifecycle day-2 operations.

# Automating edge enterprise **use cases**.



## Why Ansible for Automation at the Edge?

- ▶ **Automate scale and complexity** with a consistent platform from the datacenter to the edge across heterogeneous estates
- ▶ **Facilitates IT/OT convergence**
- ▶ **Provides predictability and repeatability** to automate anything with programmatic API or Linux OS



# Where to go next



## Learn more

- ▶ Workshops
- ▶ Documents
- ▶ Youtube
- ▶ Twitter



## Get started

- ▶ Self-paced labs
- ▶ Evals
- ▶ [console.redhat.com](https://console.redhat.com)



## Get serious

- ▶ Red Hat Automation Adoption Journey
- ▶ Red Hat Training
- ▶ Red Hat Consulting

# Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



[linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)



[youtube.com/c/AnsibleAutomation](https://youtube.com/c/AnsibleAutomation)



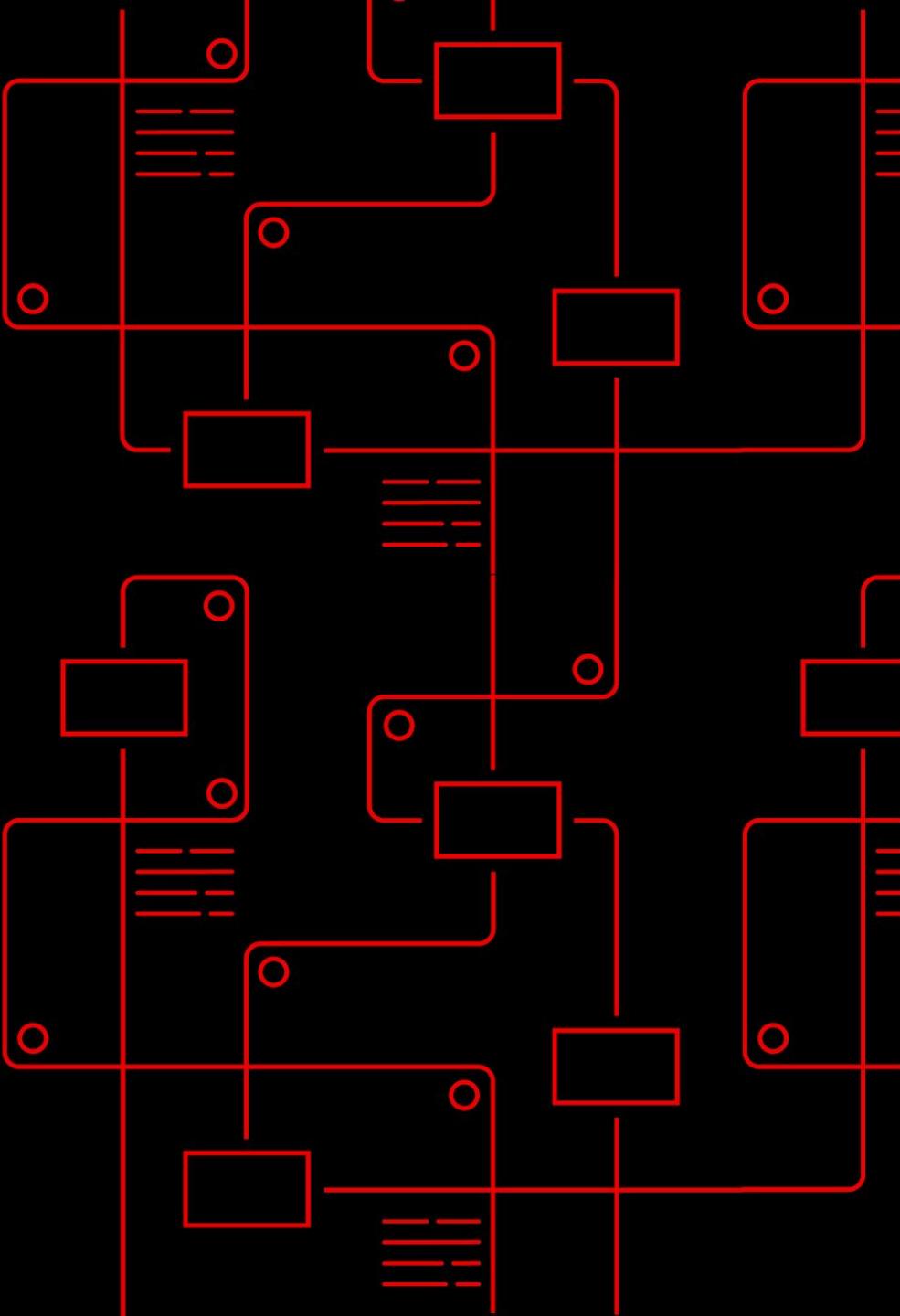
[facebook.com/redhatinc](https://facebook.com/redhatinc)



[twitter.com/ansible](https://twitter.com/ansible)

# Appendix:

## Alternate playbook example slides (colors adjusted for accessibility)



# Ansible playbooks

```
---
- name: Install and start apache
  hosts: web
  become: true

  tasks:
    - name: Ensure the httpd package is installed
      ansible.builtin.yum:
        name: httpd
        state: present

    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```

# Ansible plays. What am I automating?



## What are they?

- ▶ Top level specification for a group of tasks
- ▶ Will tell that play which hosts it will execute on and control behavior such as fact gathering or privilege level

## Building blocks for playbooks

- ▶ Multiple plays can exist within an Ansible playbook

```
...  
- name: Ensure the httpd package is installed  
  hosts: web  
  become: true
```



# Ansible modules. The “tools in the toolkit”.

## What are they?

- ▶ Parametrized components with internal logic, representing a single step to be done
- ▶ The modules “do” things in Ansible

## Language

- ▶ Usually created in Python, or Powershell for Windows setups, but can be developed in any language

```
● ● ●  
- name: Create the index.html file  
  ansible.builtin.template:  
    src: files/index.html  
    dest: /var/www/html/
```

# Ansible plugins. The “extra bits”.



## What are they?

- ▶ Plugins are pieces of code that augment Ansible's core functionality
- ▶ Ansible uses a plugin architecture to enable a rich, flexible, and expandable feature set

```
● ● ●

Example become plugin:
---
- name: Install and start apache
  hosts: web
  become: true

Example filter plugins:
{{ some_variable | to_nice_json }}
{{ some_variable | to_nice_yaml }}
```

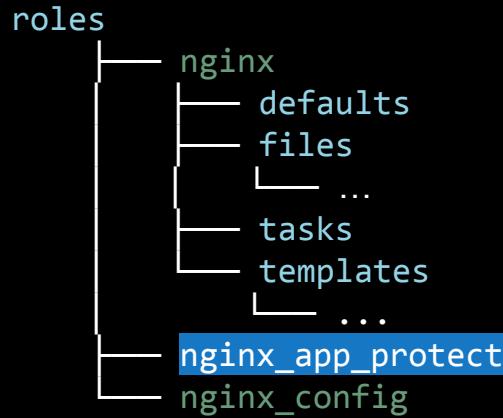
# Ansible Roles. Reusable automation actions.



## What are they?

- ▶ Group tasks and variables of your automation in a reusable structure
- ▶ Write roles once, and share them with others who have similar challenges in front of them

```
...  
- name: Install and start apache  
hosts: web  
ansible.builtin.roles:  
  - common  
  - webservers
```



### deploy-nginx.yml

```
---
```

```
- name: Install NGINX Plus
  hosts: all
  tasks:
    - name: Install NGINX App Protect
      ansible.builtin.include_role:
        name: nginx_app_protect
  vars:
    nginx_app_protect_setup_license: false
    nginx_app_protect_remove_license: false
    nginx_app_protect_install_signatures: false
```



```
nginx_core
├── galaxy.yml
├── meta
└── playbooks
    └── deploy-nginx.yml
    ...
plugins
README.md
roles
└── nginx
    ├── defaults
    ├── files
    │   └── ...
    ├── tasks
    └── templates
        └── ...
    └── nginx_app_protect
    └── nginx_config
```

### deploy-nginx.yml

```
---
- name: Install NGINX Plus
  hosts: all
  tasks:
    - name: Install NGINX
      ansible.builtin.include_role:
        name: nginxinc.nginx
      vars:
        nginx_type: plus

    - name: Install NGINX App Protect
      ansible.builtin.include_role:
        name: nginxinc.nginx_app_protect
      vars:
        nginx_app_protect_setup_license: false
        nginx_app_protect_remove_license: false
        nginx_app_protect_install_signatures: false
```