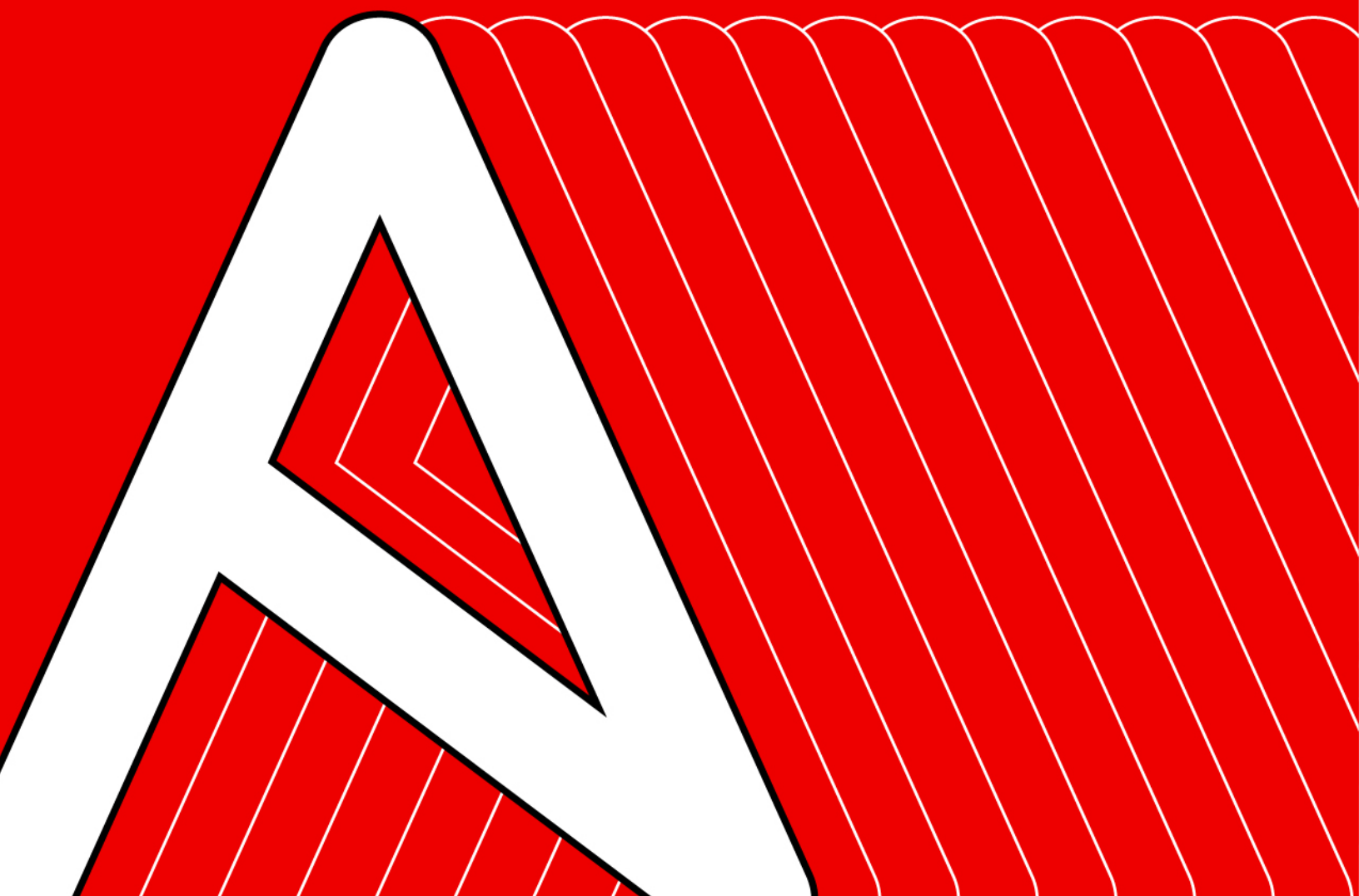




Event-driven automation for IT Ops

Act fast, consistently, and efficiently
using Event-Driven Ansible



Contents

3 Introduction: Extend what your automation can do

4 Chapter 1: Event-driven automation for IT operations

Advanced event-driven automation techniques

5 Chapter 2: Make your automation platform more powerful

How Event-Driven Ansible works

Event-Driven Ansible controller

Event source integrations

Have homegrown tools or need a custom source plug-in?

Connect telemetric data, observability, and automation

10 Chapter 3: Design your use cases

IT service management

Application healing

Infrastructure, cloud, and security

Network automation

Automation at the edge

12 Chapter 4: Implementing Event-Driven Ansible

Taking a *start small, think big* approach to Event-Driven Ansible

Think about your people

Consider your technology needs

Plan your automation strategy

The role of event-driven automation in an OpsAsCode model

16 Chapter 5: Test drive Event-Driven Ansible

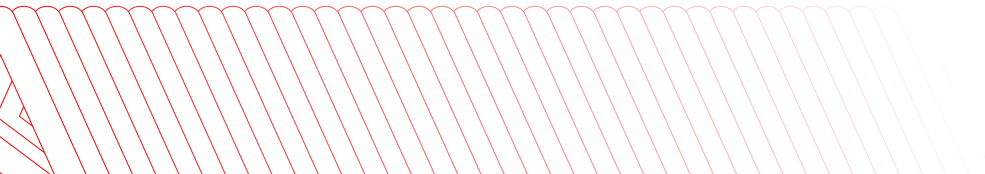
Interactive labs

In-depth webinar

Read more

Start your trial

Engage consulting services to jump-start event-driven automation



Extend what your automation can do

In today's rapidly changing IT landscape, businesses are constantly seeking innovative strategies to manage complexity and stay ahead of the curve. For many organizations, automation is essential to that strategy because it helps teams to be agile and responsive while freeing them from repetitive tasks to focus on higher-value projects. But for organizations under pressure to deliver innovations faster and respond more quickly, the potential for automation is expansive.

Event-driven automation is the next step for automation maturity and is quickly becoming a must-have for IT teams. It is the process of responding automatically to changing conditions in an IT environment to help resolve issues faster and reduce routine, repetitive tasks.

Event-driven automation can boost efficiency by providing a single, consistent, and accurate response to events. It helps connect data, analytics, and service requests to automated actions so that activities—such as responding to an outage or adjusting some aspect of your IT system—can take place in a single, rapid motion and in the same desired way each time a specific condition exists. Automating in this fashion helps IT teams manage how and when to target specific actions. It also helps manage the complexity of hybrid cloud and edge environments, while freeing teams to focus on other priorities.

Event-driven automation can help IT teams to:

- Select ideal tasks for an automated response, then allow IT domain experts—such as a network engineer—to flexibly apply automation to key needs.
- Build existing operational knowledge into automated decision-making and actions for consistency.
- Complete repetitive tasks efficiently and deliver services more quickly across any IT use case including networking, edge, infrastructure, DevOps, security, and cloud.
- Reduce low-level tasks and instead use valuable resources for other priorities such as delivering innovation.
- Address problems rapidly, before they become urgent issues.
- Document your environment and actions in automation rulebooks and playbooks as part of an Ops-As-Code model.

In this e-book, we will look more closely at the role of event-driven automation, why it is important for IT operations teams, and how to get started.



Event-driven automation for IT operations

IT operations continue to expand in scope and maturity. With budgets and skilled resources in high demand, organizations need the ability to respond faster and with greater consistency and accuracy, especially for key business applications and their underlying technologies.

Event-driven techniques have been used in the past, but they can be time consuming and slow to deliver when teams must code or integrate solutions. This makes the implementation of event-driven automation selective at best.

As organizations strive to use automation more strategically across hybrid cloud environments and edge locations, event-driven automation can help improve speed, efficiency, and resilience.

Advanced event-driven automation techniques

With the right automation tooling, event-driven automation allows IT systems to respond to specific triggers or events without manual intervention. For instance, if network traffic spikes beyond a certain threshold, automated processes can kick in to adjust bandwidth allocation, ensuring that operations remain smooth. If a potential security threat is detected, automated defenses can spring into action before requiring human intervention to lower risks as quickly as possible—even while you investigate the resolution.

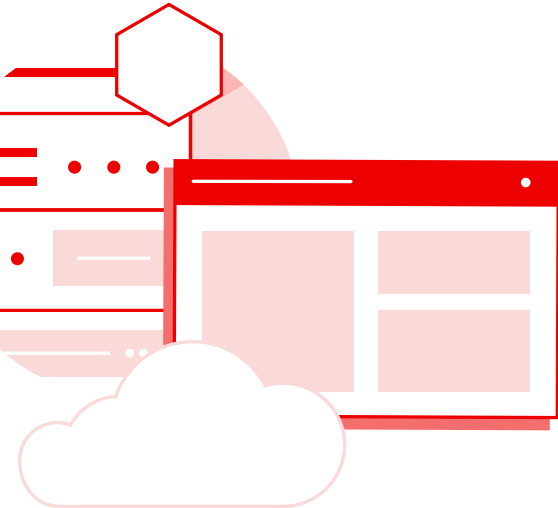

Use cases across the IT organization are broad and what is needed is a solution that makes it fast and easy to create event-driven automation scenarios without extensive coding, integration or specialized skills.

As IT operational teams grapple with the ever-increasing demands of modern businesses, event-driven automation stands out as a tangible way to do more with less, help teams to innovate, and enjoy a better work-life balance.



Make your automation platform more powerful

The best automation platforms are those that can extend across a broad range of IT domains, capable of automating within and across complex processes. The introduction of event-driven automation extends what your automation platform can do with the ability to automatically respond to changing IT conditions.



“Event-Driven Ansible enables us to more easily connect various events to automated responses. This leads to faster, more reliable, and more consistent automation, which means our site-reliability engineers have more time to spend on other priorities.”¹

Tech Lead, Platform Automation and Observability
Major brand retailer



Organizations using the latest version of Red Hat® Ansible® Automation Platform can access [Event-Driven Ansible](#) as part of their subscription, making this trusted enterprise automation platform even more powerful to handle new types of automation needs.

Event-Driven Ansible uses [YAML](#) to write rulebooks containing conditional rules that implement decisioning about the specific actions you want to take. Rulebooks can, at your discretion, be set up to reference existing playbooks or templates based on these rule decisions, helping you extend the use of your existing automation. This provides the event-handling capability needed to automate time-consuming tasks and respond to changing conditions in any IT domain in an easy implementation format.

¹ Red Hat video. “[AnsibleFest at Red Hat Summit Keynote: The automation moment—and beyond](#),” accessed August 2023.

How Event-Driven Ansible works

Event-Driven Ansible processes events containing discrete intelligence about conditions in your IT environment, determines the appropriate response to the event, then executes automated actions to address or remediate the event. These 3 components are the building blocks of Event-Driven Ansible: **event sources, rules, and actions.**



Figure 1: How Event-Driven Ansible works with sources, rules contained in rulebooks, and actions.

Event sources:

Intelligent sources of events are sent to Event-Driven Ansible via an event source plug-in. Event-Driven Ansible receives the event source and processes each event against the ruleset in the Ansible Rulebook. Event source plug-ins are either developed by Red Hat or partners, where Red Hat certifies or validates (as appropriate) these partner plug-ins and related automation content. In the *ansible.eda* collection, Red Hat provides source plug-ins for technologies including those described in the chart. These plug-ins are in addition to those provided by Red Hat's ecosystem of partners. A current list of certified and validated Content Collections for Event-Driven Ansible can be found [here](#).

alertmanager	Receives events via webhook from alertmanager
AWS CloudTrail	Provides a method for receiving events from AWS CloudTrail for cloud infrastructure
AWS SQS	Provides a method of receiving events via an AWS SQS queue.
azure_service_bus	Receives events from an Azure service
file	Loads facts from YAML files and reloads if there are any changes
journald	Provides a method to tail the systemd journald logs as a source of events
kafka	Receives events from a Kafka topic
range	Generates events with an increasing index within a range
tick	Generates events with an increasing index that never ends
url_check	Polls a set of URLs and sends events with the URL status
watchdog	Watch a file system and send events when a file changes
webhook	Provides a webhook and receives events

Figure 2: Current capabilities included in the *ansible.eda* collection.

Rules and rulebooks:

Event-Driven Ansible requires the use of an Ansible Rulebook. The rulebook contains rulesets and conditions that need to be met in order to trigger an action. Conditional statements are used to filter events and determine the desired action. These actions could include responding to an event with an Ansible Playbook, module, workflow or job templates from Event-Driven Ansible controller. Once an event matches a condition in the ruleset, the corresponding defined action can take place. The structure of the rulebook requires at least 1 event source and rule to be defined. Rulebooks are built in YAML and adhere to a specific structure as illustrated in figure 3.

```
---
- name: Port State Event from switch
  hosts: switch

  sources:                                ## event sources defined
  - ansible.eda.kafka:
      host: 192.168.11.49
      port: 9092
      topic: network

  rules:                                  ## rule conditions defined
  - name: Port is down
    condition: event.fields.admin_status == "DOWN"
    action:                                     ## action defined
      run_playbook:
        name: bring-interface-up.yml
```

Figure 3: Sample rulebook showing sources, rules, and actions taken when conditions are met for that rule.

Actions:

Once a rule has been matched, the action associated with that rule is triggered. Current actions that can be used are listed in figure 4.

debug	Debugs the running rulebook
post_event	Post an event to a running rule set in the rules engine
print_event	Write the event to stdout
retract_fact	Remove a fact from the running rule set in the rules engine
run_job_template	Execute a job template on automation controller
run_module	Execute an Ansible Module
run_playbook	Execute an Ansible Playbook
run_workflow_template	Execute a workflow template on automation controller
set_fact	Post a fact to the running rule set in the rules engine
shutdown	Shut down the rulebook
run_workflow_template (coming soon)	Execute workflows on automation controller

Figure 4: Current actions that can be triggered based on an associated rule.

Event-Driven Ansible controller

The Event-Driven Ansible controller provides the integration and user experience (UX) for Event-Driven Ansible. When the appropriate action is determined, it's sent to the automation controller via an API. In the Event-Driven Ansible controller, rulebooks are

activated so they can listen to event sources and conditionally respond with an appropriate action to events received from that source and when conditional rules are met. Once a rule is matched and a `run_job_template`

or `run_workflow_template` action is triggered, the event variables are sent to the automation controller as extra variables for the template or workflow to start. Figure 5 illustrates the process from event source to action.

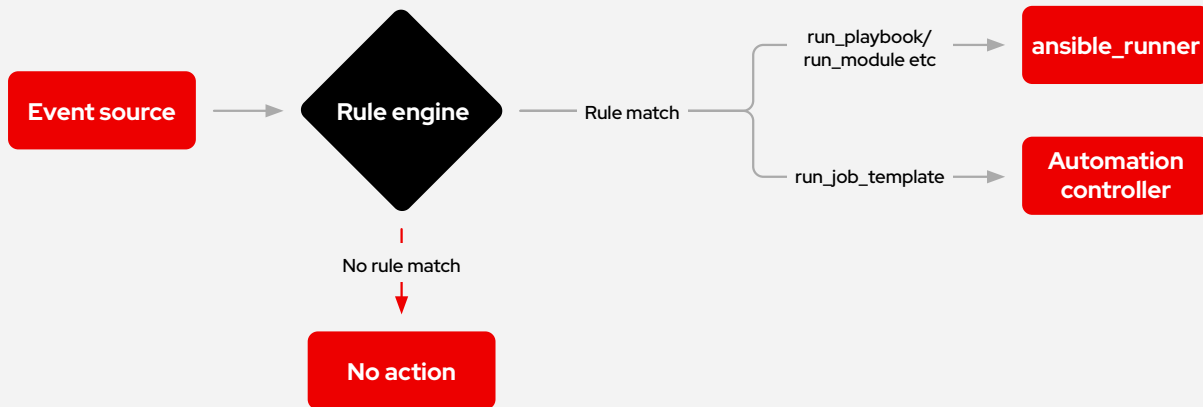


Figure 5: Components and architecture of Event-Driven Ansible from event source to automated action.

Get to know event-source integrations

Event source plug-ins within Event-Driven Ansible act as a bridge between Ansible and event-generating applications and services, such as observability and monitoring tools. Event-Driven Ansible works with sources of intelligence from within your IT environment. This comes from both included collections and validated content created by Red Hat and partner-developed content. These plug-ins are listed in figure 6.

Current partner certified and validated Ansible Content Collections

- CrowdStrike*
- Dynatrace*
- Instana*
- LogicMonitor*
- Palo Alto Networks*
- Red Hat Insights*
- Turbonomic*
- Zabbix

*Currently includes event source plug-in.

Current ansible.eda certified Ansible Content Collection included in subscription

- AWS SQS
- AWS CloudTrail
- Azure Service Bus
- GCP Pub/Sub
- Kafka (AMQ Streams)
- Prometheus/Alertmanager
- Webhooks
- watchdog (file system watcher)
- url_check (url status check)
- range (event generation plug-in)
- file (loading facts from yaml)
- journald
- tick

Figure 6: Current Ansible Content Collections and integrations for Event-Driven Ansible.

Learn the details about these [content collections](#)

Have homegrown tools or need a custom source plug-in?

We recognize that you use a wide array of tools to operate and monitor a range of systems. Some of these tools may be purpose-built or incorporate components for which there is not currently an event-source plug-in. Fortunately, you can create custom event-source plug-ins of your own to incorporate the key sources of events in your business domain. In addition, we will also continue to work with partners across the industry to deliver plug-ins and other automation assets you need.

[Learn how to create your own custom plug-ins](#)

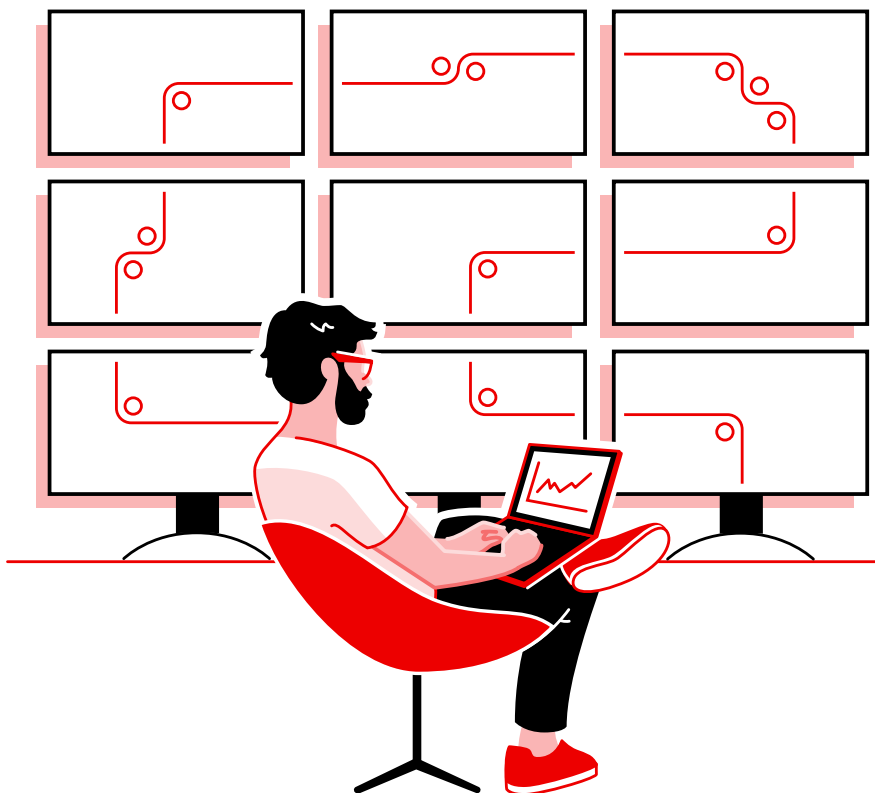
Connect telemetric data, observability, and automation

An observability tool provides a centralized platform for aggregating and visualizing telemetric data that has been collected from application and infrastructure components in a distributed environment.

These tools have taken a prominent role because of hybrid cloud and other complex architectures where it can be difficult to understand how critical applications are behaving when they are spread among various systems. Observability tools provide the

intelligence that organizations need to respond quickly to any problem, such as application, network, and infrastructure health. Observability tools will often drill into issues to determine the root cause, such as an application not performing well because of a problem in the infrastructure or cloud environment.

There are many observability tools, each with different strengths in what they observe. For example, some focus on cloud applications while others focus on networks. There are additional management tools for the platform that also provide data. Event logs or homegrown tools can also collect important information about conditions in your environment and contribute to an effective observability posture. For these tools that do not have plug-ins or direct integrations, we have the ability to use webhooks, Kafka or Prometheus Alertmanager as integration points for Event-Driven Ansible.



Design your use cases

To understand the potential of Event-Driven Ansible, it can be helpful to think of what it can do in specific use cases. The following are 7 practical applications for Event-Driven Ansible that can apply to nearly any enterprise organization, and are good starting points for assessing where it can be applied to your IT environment.



IT service management

Tasks such as ticket enhancement, remediation, and user management are ideal starting points for Event-Driven Ansible. It provides the flexibility to automate a variety of tasks across your IT environment by connecting analytics to automated actions, and improving the resilience and responsiveness of IT, while freeing teams to focus on more valuable work. For example, there is no need to “drop everything” to enhance a service ticket with facts such as configuration information that aids in troubleshooting.



Application healing

Event-Driven Ansible helps ensure that applications remain in top operational form. For example, if an observability tool such as Dynatrace or IBM Instana is watching key applications and discovers that a cloud instance needs to be scaled up to handle traffic, this event can be sent to Event-Driven Ansible, which finds the corresponding Ansible Rulebook and matches the condition with the defined action. This action could, for instance, add more cloud resources via an existing playbook to ease existing constraints. Rulebooks run the actions you specify. These actions can include reapplying a configuration, resetting the router, or creating a service ticket depending on the nature of the issue or event. Event-Driven Ansible triggers the instructions in the rulebook, and in our example, would run the playbook so cloud resources are increased and the application functions as expected and at scale.



Infrastructure, cloud, and security

IT infrastructure, whether in a cloud, hybrid cloud, multicloud, or on-premise, is becoming ever more complex, and automation is an efficient tool to combat this rising complexity. Infrastructure use case scenarios in this area may include addressing drift, resetting servers, managing certificates, adjusting storage pools, enhancing tickets with configuration information, responding to security risks and much more.

Cloud-native use cases may include building up and tearing down cloud resources in a configuration-as-code model to respond to changing workload demands, addressing security exposures (including closing down affected cloud resources and load balancing to utilize other resources), reapplying cloud configurations from a source of truth, and managing operations for hybrid (cloud and on-premise) workloads.



Network automation

IT networks require fact gathering and notifications to monitor and maintain healthy operations. Event-Driven Ansible can help by automatically creating and enhancing tickets with configuration information, gathering other troubleshooting facts, and creating notifications as needed, or doing basic device reset actions. With these inputs, remediation can be planned according to your specification, from basic to advanced.

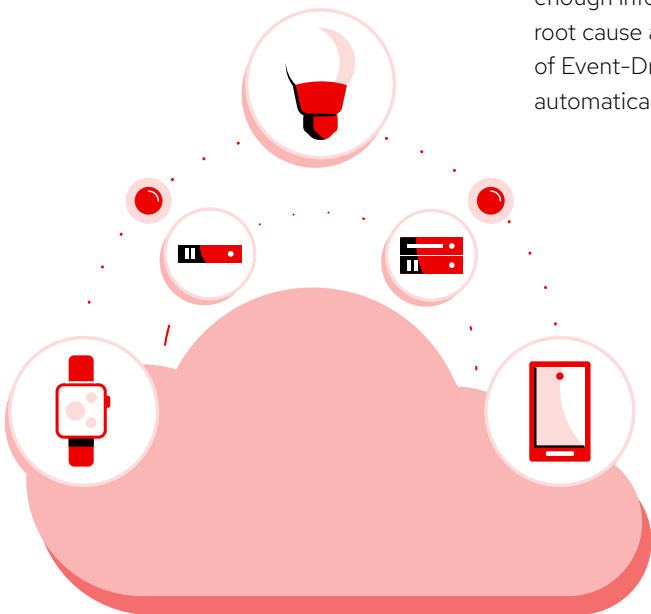
When first starting out with Event-Driven Ansible, remediation might be as simple as rerouting traffic automatically when a routing protocol is intermittent or down. After that, you might progress to more advanced remediation measures that can automatically report and reapply configurations to manage drift, or isolate or shut down affected devices for certain security risks while investigation takes place. What's more, this can happen at any time—even at 2 AM, while the network engineer stays asleep.



Automation at the edge

The introduction of Event-Driven Ansible is especially applicable to managing edge environments. A common scenario in IT management is the speed at which a technician gets to look at a ticket. The longer it takes, the more production services are impacted, and downtime keeps adding up and this is true all the way to the edge. Frequently, open tickets do not provide enough information to provide effective root cause analysis. With the addition of Event-Driven Ansible, you can now automatically provide the necessary

information such as configuration information that is needed so that corresponding tickets can be addressed faster and with less toil, so outages at the edge are resolved more quickly. There are also action-reaction scenarios that can be delivered with Event-Driven Automation. For example, a smart button is pressed and a camera that takes and sends a picture, or proactive actions such as restarting access points at hospitals when the quality of the WiFi signal declines below a predefined threshold.



Implementing Event-Driven Ansible

Now that you understand the big picture of where you want to apply event-driven automation, we can start your team's learning journey. Similar to implementing new automation instances, Red Hat recommends a start small, think big approach to explore what Event-Driven Ansible can do. In this section, we will discuss some simple-yet-progressive ways you can learn before you apply the advanced use cases we discussed above.

Taking a start small, think big approach to Event-Driven Ansible

The following examples can help you understand and gradually implement Event-Driven Ansible, so you can grow your automation sophistication.

Gather facts

When a ticket comes in, the Event-Driven Ansible rulebook may define an action to reach out to the affected device and gather and add the configuration information to a trouble ticket. So, when the support person responds to the ticket, this information they require is already present and they are able to address the issue faster. This simple step saves time, reduces toil and churn, and is a great use case that has low impact while you learn.

Generate a service ticket

When a condition is identified by your observability tool, Event-Driven Ansible can automatically generate a ticket in an IT service management (ITSM) solution or post a notification to an internal messaging system, such as Slack or a packaged application. For example, in the event a security certificate is about to expire, your rulebook can create an alert and automatically generate a service ticket.

Send a notification

Take the automatic ticket generation a step further where you can also send a notification to the right person on your team to address the event. For example, if a network or edge device is not responding, Event-Driven Ansible can create a ticket and notify the right person, which can accelerate response time.

Do basic remediation

The next step is basic remediation which might include resetting or rebooting a system and sending a notification if necessary. For example, if some part of the network or an edge device is not responding, Event-Driven Ansible can automatically create a ticket along with performing a basic reboot. If the basic reboot does not work, a person can be paged or notified as part of the automation sequence.

Perform advanced remediation

With the previous steps mastered, you are ready to bring in multiple event sources and correlate them, to orchestrate the response that best suits your needs. For example, in the event that a basic reboot does not work, Event-Driven Ansible, based on your prewritten Ansible rulebook, can read a second event and find out which neighboring device is available and can reroute the network traffic. An important thing to remember is that Event-Driven Ansible can execute your already developed automation as an action according to rulebook conditions. This allows you to supplement your automation by integrating Event-Driven Ansible with existing playbooks.

Think about your people

Your team is essential to the success of any new technology that you implement. As you begin to learn and implement with Event-Driven Ansible, there are training, resources, and best practices that can help ensure success.

Build domain expertise

Ansible Rulebooks are at the heart of using Event-Driven Ansible to automate recurring operational logic and processes, and since Ansible Rulebooks are written in YAML, they are ideal for domain experts who can easily write automation that solves specific problems and makes their jobs easier and less time consuming.

Self-paced labs for domain experts

Get the training you need on
[Event-Driven Ansible implementation](#)

Start an event-driven automation community of practice

Communities of practice are a great way to transform how your organization thinks about automation. Appointing a role, such as an automation architect or lead and starting a community of practice, can bring teams together to share ideas, content, experiences, questions, and best practices to advance the use and growth of Event-Driven Ansible across your organization.

Foster a culture of change

The secret to successful event-driven automation is not just a technology change, but also a change in mindset across the organization. Work toward building an automation-first mindset by focusing on goals and outcomes instead of tools. This might include showing your teams how Event-Driven Ansible gives them more time for interesting engineering priorities and helps reduce the need for extended work days.

Codify and share results

Sharing results is an excellent way to raise awareness, demonstrate legitimacy, and gain buy-in from senior executives. Consider the case of a major insurance provider for example. The team began looking for a solution to improve efficiency, reduce management costs, and enhance the user experience. After successfully launching small instances of Event-Driven Ansible, the team monitored and measured the impact of these changes to get CIO buy-in and make the case for an enterprise-wide rollout in the future. They worked closely with platform, networking, cloud, and applications teams to grow the use of automation across the business.



Consider your technology needs

What technologies are needed for event-driven automation? At the highest level, all teams will need to determine which use cases to automate, determine event sources, then write rulebooks to execute the automation. If you are an Ansible Automation Platform customer, you can optionally call existing playbooks within your new Ansible Rulebooks to build on your trusted automation. If you are a new Ansible Automation Platform customer, you may want to start with creating a playbook, then creating rulebooks that call these playbooks.

From there, you can grow your use of this automation to more sophisticated automated actions and broader applications across your operation—from network to infrastructure to cloud and DevOps, and more.

Red Hat Consulting and training services, as well as services from partners, are available to help you effectively implement automation so you can realize benefits even more quickly.

For more information see our [**Red Hat Consulting solution datasheet**](#) [**Event-Driven Ansible with Ansible Automation Platform**](#) or contact a Red Hat partner.

Plan your event-driven automation strategy

When thinking about your event-driven automation strategy, there are a few key questions that you can ask that can help accelerate the planning process.

Understand your organization's opportunity for event-driven automation:

- Are you using homegrown code to do event-driven automation scenarios today?
- What happens when the owner of those automation instances moves roles or leaves your company?
- Do you respond to the same tickets over and over again? What if responses could be automated? How confident are you that your response is consistent and accurate each time?
- Have you only implemented event-driven automation on a few selected use cases because it entails manual coding, time, and even consulting costs? What if these barriers were removed so that all teams can use these techniques?
- Do you have existing Ansible Playbook automation that requires manual initiation?
- How would your operation be transformed for the better if certain tasks are automated completely?
- Have you lost revenue when a key application is slow to respond or could not scale in some way to meet demand?
- Do you have a plan for managing the proliferation of devices at the edge?
- Can you use event-driven automation to help you manage a complex cloud or multicloud implementations?

- Are you running late on delivering key innovations? What if you could free up time to work on those?
- Are there efficiencies that can be realized by creating a standardized response to events that happen in disconnected instances?
- How would employee satisfaction change if there was better work-life balance?
- How much time are you spending on routine low-level tasks like managing certificates or checking URLs? What if this work could be automated?

Event-Driven Ansible can address all of the questions above because it is based on simple constructs and YAML that domain subject matter experts can easily use and update. This democratizes the use of event-driven techniques that can help an organization expand into a more mature automated enterprise IT operation.

The role of Event-Driven Ansible in Ops-as-Code

Innovation is often the driving force behind decision making, but what happens when you get to Day 2 operations? You need solutions that, once built, can excel at managing the life cycle of needs for your solutions whether it is applications or underlying infrastructure.

OpsAsCode is an approach that lets you do this, using automation that codifies the actions you want to take accurately and consistently and is stored in a repository as a single source of truth so you can call upon it with confidence anytime you need to take action. Event-Driven Ansible can call upon these single sources of truth and documentation you have about your operation contained in playbooks or rulebooks to act when a specific type of condition exists in your environment.



Test drive Event-Driven Ansible

Event-Driven Ansible is advanced automation included in Ansible Automation Platform. Now an even more powerful platform, it can help you improve the speed and pace of IT service delivery, improve efficiency, and increase resilience.

This helps you ensure a consistent and accurate response, even if your skilled people are unavailable, because your operational demands are all codified in rulebooks and playbooks. Event-Driven Ansible also helps teams focus on higher-value tasks and innovation, which can result in a more productive workforce, better customer experiences or more revenue.

There is more than one way to read about, discuss, and try Event-Driven Ansible. Use the links below to get started.

Interactive labs

Take these self-paced, interactive labs to get familiar with Event-Driven Ansible so you can more easily apply it to your needs.

[Sign up for a self-paced lab](#)

In-depth webinar

Register for this free webinar to watch a detailed Event-Driven Ansible demonstration and learn how to get started.

[Learn more](#)

Read more

Read more about Ansible Automation Platform and Event-Driven Ansible.

[Visit the website](#)

Start your trial

Did you know Event-Driven Ansible comes with the latest version of Ansible Automation Platform?

[Start trial](#)

Engage consulting services to jump-start event-driven automation

Red Hat Consulting and also Red Hat partner services can help you get started more quickly, assist with culture change, and realize benefits from automation. Contact your Red Hat partner of choice or learn about [Red Hat Consulting services](#).

