Guideline for securing Red Hat Ansible Automation Platform

Introduction

This document is a security guideline for enterprise deployments of Red Hat Ansible Automation Platform. It contains security configuration and policy recommendations for Ansible Tower using the National Institute of Standards and Technology (NIST) Risk Management Framework (RMF) and supporting security controls from NIST Special Publication 800-53 rev. 5. It provides recommendations that maximize confidentiality, integrity, and availability of Ansible Automation Platform as well as the systems and nodes managed by Ansible.

Some security control families in the RMF are out of scope and fall under the purview of the hosting organization and Ansible users. The control families include: incident response, maintenance, media protection, personnel security, planning, risk assessment, physical and environmental protection, program management, security assessment and authorization, and system and services acquisition.

As for the remaining control families that directly apply to Ansible configurations, Red Hat recommends implementing the steps in this document through a defense-in-depth approach to security. By doing so, Ansible users will be able to build overlapping control mechanisms, providing security in multiple layers and mitigating risk even if one of them fails.

Security in the context of Red Hat Ansible Automation Platform

Ansible is an IT automation tool that can be used to configure systems, deploy software, and orchestrate more advanced IT tasks, such as networking configuration and continuous deployments. Ansible Tower is a component of an annual Red Hat Ansible Automation Platform subscription that is meant to scale and secure your Ansible automation content. This document assumes you are running Ansible Tower as a part of your Ansible Automation Platform subscription.

Ansible Automation Platform, Ansible Tower, and Automation Hub together comprise a general purpose, declarative, automation platform. Once an Ansible playbook is launched (via Ansible Tower, or directly on the command line), the playbook, inventory, and credentials provided to Ansible are considered to be the source of truth. If policies are desired around external verification of specific playbook content, job definition, or inventory contents, these processes must be undertaken before the automation is launched (whether via the Ansible Tower web user interface [UI] or application programming interface [API]). See “Security Best Practices” in the Ansible Tower user’s guide for more details.
Using source control, branching, and mandatory code review as a part of a continuous integration/continuous delivery (CI/CD) pipeline is a best practice for Ansible automation. There are many tools that help create process flow using source control in this manner.

Distributing Ansible content throughout an organization is an important step along the automation adoption journey. A private automation hub can be deployed to host Ansible Collections on-premise and synchronize community or Red Hat Certified content.

**Security control families and guidance**

**Access controls**

Granting broad access to more users than necessary to perform productive work can create significant risk. By applying well-known security principles such as least privilege, defense-in-depth, and role-based access controls (RBAC) within Ansible, like any other system, you will be able to limit necessary functions and provide access only to those who require it.

**Minimize administrative accounts**

Minimizing the access to accounts belonging to Ansible’s managed nodes (e.g., server operating systems, storage appliances, network devices) is crucial for maintaining a secure organization considering a system administrator or root user can access, edit, and disrupt the host operating system and any application running on the host. It is highly recommended to keep the number of people and accounts with root access as low as possible. Do not give out ‘sudo’ to root or ‘awx’ (the Tower user running on the Linux with Tower installed) to users who are not trusted. In an Ansible Tower context, any ‘system administrator’ or ‘superuser’ account can edit, change, and update any inventory or automation definition in Ansible Tower. Restrict this to the minimum set of users possible for low-level Ansible Tower configuration and disaster recovery only.

Users and groups in a private Ansible Automation Hub are isolated from those in Ansible Tower but should be governed using the same best practices.

**Remove access to credentials from users**

If an automation credential is only stored in Tower, it can be further secured. Services such as OpenSSH can be configured to allow credentials on connections only from specific addresses. It is highly recommended to use different credentials for automation from those used by system administrators to log into a server. Although direct access should be limited where possible, it is sometimes used for disaster-recovery or other ad-hoc management purposes, allowing for easier auditing.

**Enforce separation of duties**

**Roles and collections**

Different aspects of automation may need to access a system at different levels. For example, you may have low-level system automation that applies patches and performs security baseline checking, while a higher-level piece of automation deploys applications. In an Ansible context, roles allow users to separate tasks into a logical structure. When building roles, you should consider separation of duties from a security perspective. By using different keys or credentials for each aspect of automation or role, the effect of any one key vulnerability is minimized, while also allowing for easy baseline auditing.

See “Roles” and “Using Collections” in the Ansible Tower User Guide for more details.
Source code repository
When setting up your Ansible Tower instance, you should create or move all playbooks into a managed source control repository. It is best practice to create a code review process to require all playbook changes to production-branch playbooks to have multiple reviewers sign off on the changes. Ensure that all authorized users have read access to the repository so that existing playbooks can be reused in new work and access to production playbooks can be controlled.

Separation of environments
Automation development and testing (dev/test) environments should contain a set of dedicated hardware on separate networks with separate environmental credentials that replicate the production environment. These dev/test environments are ideal for research, experimentation, development, and testing. Do not allow development Ansible Tower clusters to interact with any servers outside of these development network isolated locations. Sensitive virtual local area networks (VLANs) should be identified, secured, and isolated by being managed by separate Ansible Tower instances.

See “Role-based access controls” in the Ansible Tower User Guide for more details.

Audit and Accountability
For any administrative access, auditing and watching for actions is critical. For the system overall (Ansible Automation Platform, Ansible Tower, and Red Hat Enterprise Linux’), this capability is provided by the built-in audit support and built-in logging support. For Ansible Tower, this capability is provided by the built-in activity stream support that logs all changes within Ansible Tower, as well as by the automation logs.

Most organizations have an aggregated logging and auditing tool suite or platform. As a result, it is highly recommended to view logs in the log aggregator rather than reviewing them on the local system. It is also recommended to use the security information and event management (SIEM) system as a passive logger, and implement normalized activity level alerts in a logging aggregator, which can be helpful for stopping potentially harmful job execution. Establish baseline log expectations so that deviations can create appropriate alerts for review by information security specialists and system auditors.

Awareness and training
Create a personnel training policy within the organization so that users must take Red Hat training courses such as “Ansible Essentials” and “Advanced Automation: Ansible Best Practices” before being granted access rights as an Ansible Tower User.

Configuration management
It is highly recommended that administrators manage Ansible inventories in a controlled configuration management database (CMDB). Ansible can then use the CMDB and create inventories dynamically based on the contents, approvals, and permissions of the CMDB. Any role modification should be handled with a multiapprover business automation platform.
Overview

Contingency planning

Clustering is an approach to redundancy that allows for sharing load between hosts. Each instance should be able to act as an entry point for UI and API access. This should enable Ansible Tower administrators to use load balancers in front of as many instances as they wish and maintain good data visibility, especially in the case of host failure. See “Clustering” in the Ansible Tower User Guide for more information.

The ability to backup and restore Ansible Tower has been integrated into the Ansible Tower setup playbook. Please see “Backing up and restoring tower clusters” in the Ansible Tower User Guide for more information.

Identification and authentication

External account stores and identity providers

Maintaining a full set of users in Ansible Tower can be a time-consuming and error-prone task in a large organization. Ansible Tower supports connecting to external account sources via LDAP, SAML 2.0, cloud-based identity providers, and certain OAuth2 providers. This capability eliminates a source of error when working with permissions. Please see “Token-based authentication” and setting up other authentication types in the Ansible Tower User Guide for more information.

Credential management

Implementing a privileged access or credential management solution to protect credentials from compromise is a highly recommended best practice for many enterprises. Organizations should audit the use of and provide additional programmatic control over access and privilege escalation. See “Secret handling” in the Ansible Tower User Guide for more information.

Multifactor authentication

Because authentication to Ansible Tower can be enforced through an external system, it is possible and recommended to utilize two- or multifactor authentication (2FA/MFA) as an extra layer of security in enterprise environments. If implementing MFA, it is recommended to incorporate unique authentication such as an enterprise identity provider and a credential management system.

System and communications protection

The connections to internal resources such as Memcached and Redis are over a local unix socket, restricted to the awx service user. The connection to the PostgreSQL database is handled via password authentication over transmission control protocol (TCP), either via localhost or remotely (external database). This connection can use PostgreSQL’s built in support for secure sockets layer or transport layer security (SSL/TLS), as natively configured by the installer support.

Ansible Tower and Ansible Automation Hub are accessed externally via standard HTTP/HTTPS on standard ports, provided by nginx. All connections from Ansible Tower to managed nodes are done via standard secure mechanisms as specified such as SSH, WinRM, or SSL/TLS. See “Connection security” in the Ansible Tower User Guide for more information.
System and information integrity

The use of a business automation toolset and workflows to perform the execution of jobs via the Ansible Tower API is common as organizations mature their automation capabilities. As a result, many organizations do not need to log into Ansible Tower directly, but rather build services that dynamically authenticate and call the API as needed. A thorough implementation of this would include the execution of jobs as part of larger workflows rather than execution independent of those workflows. When doing so, organizations should build rules that perform analysis on whether the jobs have changed, how widespread are their effects, and who the approving entities should be, based on the servers affected.

Conclusion

When securing Ansible Tower in an enterprise environment, the principle of defense-in-depth or overlapping security applies. By incorporating overlapping security controls and mechanisms mentioned in this document and by continuously improving based on updates to Ansible recommendations and NIST’s RMF, organizations adopting Ansible Tower can effectively maximize risk mitigation and information security.

About Red Hat

Red Hat is the world’s leading provider of enterprise open source software solutions, using a community-powered approach to deliver reliable and high performing Linux, hybrid cloud, container, and Kubernetes technologies. Red Hat helps customers integrate new and existing IT applications, develop cloud-native applications, standardize on our industry leading operating system, and automate, secure, and manage complex environments. Award winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500. As a strategic partner to cloud providers, system integrators, application vendors, customers, and open source communities, Red Hat can help organizations prepare for the digital future.