

Architecture Blueprint

*IAG RHEL 8 AMI and RHEL 8 On-prem Build v0.1*

23 May 2021

Contents

1. Architecture Vision 3

1.1. Business Goals and Drivers 3

1.2. High-level Stakeholder Requirements 3

1.3. Technical Vision 4

1.4. Principles, Policies and Standards 4

2. Conceptual Architecture 5

2.1. Proposed Solution 5

2.1.1. RIP Framework changes for on-prem RHEL 8 Build 6

2.1.2. RAPI Changes for on-prem RHEL 8 build 7

2.1.3. Ansible Tower Architecture 7

2.1.4. RIP & Ansible Tower Integration 9

2.1.5. Satellite & Ansible Tower Integration 9

2.1.6. SCM Integration for RHEL 8 OS Hardening 11

2.1.7. Components of RHEL 8 AMI 12

2.1.8. Components of on-prem RHEL 8 build 12

2.1.9. OASIS AWS Collector 13

2.1.10. Baseline monitoring for RHEL 8 AMI 14

2.1.11. Baseline Monitoring for RHEL 8 on-prem 17

2.1.12. Business Capability Architecture 17

2.1.13. Information Systems Architecture 17

2.1.14. Platform Architecture 17

2.2. Assumptions and Dependencies 17

2.3. Architecture Risks and Issues 17

# Architecture Vision

## Business Goals and Drivers

To meet evolving business demands, IAG is looking for new workloads, from artificial intelligence (AI) to the Internet-of-Things (IoT), to drive competitive advantages in crowded marketplaces. Linux provides the innovative muscle to power these differentiated services, but only Red Hat Enterprise Linux 8 delivers this innovation along with a hardened code base, extensive security updates, award-winning support and a vast ecosystem of tested and validated supporting technologies.

IAG currently has RHEL 5,6 and 7 systems in the on-prem infrastructure data centres. Many of these systems have no support or are on extended support, and RHEL 7 has end of life support till June 2024.

Red Hat Enterprise Linux has always been known as the most stable and secure foundation for applications. However, in the past it was hard to get the most up-to-date languages and frameworks that developers wanted without compromising that stability. Red Hat Enterprise Linux 8 introduces Application Streams - fast-moving languages, frameworks and developer tools are updated frequently in this stream without impacting the core resources that have made Red Hat Enterprise Linux an enterprise benchmark. This melds faster developer innovation with production stability in a single, enterprise-class operating system.

## High-level Stakeholder Requirements

As part of this project a standardized RHEL 8 builds will be created for IAG for both on-premise and AWS Cloud.

These RHEL 8 systems need to be highly secured, hardened and configured as per the **CIS** standards, and is already approved by the respective IAG **CSO** team.

Requirement was provided by the IAG technical specification team based on which we are proceeding with the RHEL 8 build detailed in this document.

This project will be a collaborative effort among the different stakeholders. A brief list of stakeholders and their responsibilities as per the below table.

|  |  |
| --- | --- |
| **Stakeholder Team/Name** | **Roles/Responsibilities** |
| IAG | The requirements/changes were already received in the early phase of the Project. |
| IAG **Foundation** Architecture Board | The requirements/technical design approach document has already been approved by Tech Office. |
| Server Tower (TCS) | Design, develop, test and deliver the solutions. Support and make any future enhancements as per new requirements. |
| DC Exit team (TCS) | Provide requirements for the Cloud AMI  Test/Validate the AMIs provided by server tower team |
| IAG Security Architects/Teams | Provide the security requirements if any.  Validate/test security/hardenings |
| SCM team | Provide the security configurations for RHEL8  Provide Ansible Tower Integration support |

## Technical Vision

The latest generally available version of RHEL 8 is RHEL 8.6. RHEL 8 has end of life supports available till 2029.

As part of DC-exit programmes all the IAG applications will be migrated to AWS Cloud infrastructure, and the RHEL 8 is the preferred OS that can support for a longer period.

The on-prem systems built using RHEL8 can be easily migrated to AWS, and vice versa. This can provide a very good base for the hybrid cloud platform that can be built in future.

## Principles, Policies and Standards

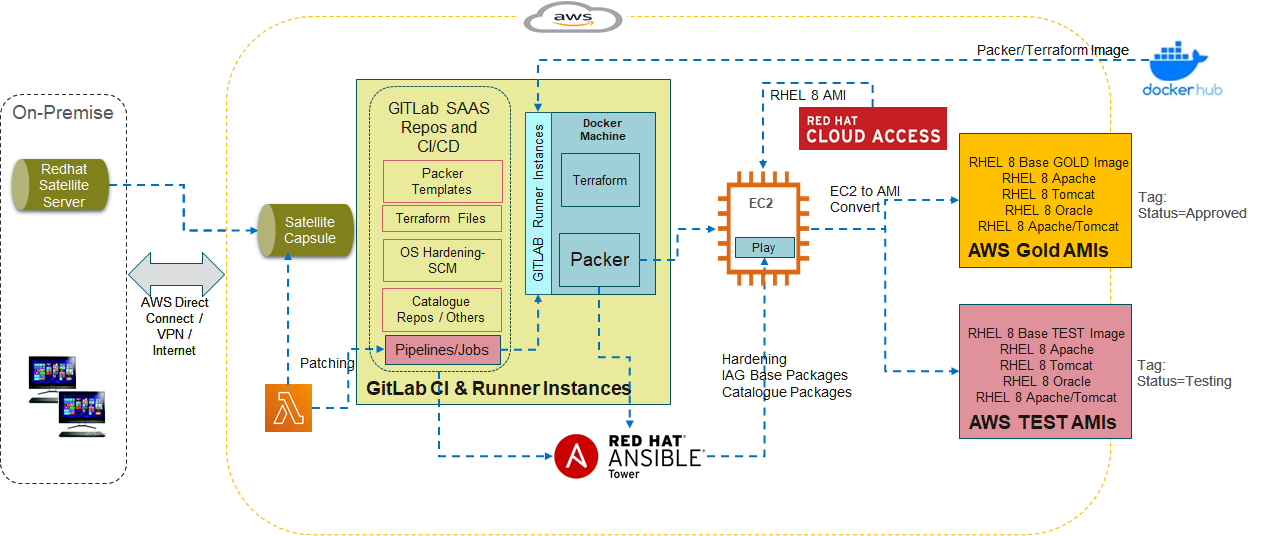
* Any change to the RHEL 8 build process must be approved by IAG Architecture board.
* The RHEL 8 configuration initially agreed would be maintained and updated with proper approvals.
* That the application would be made compatible by the respective application teams.
* The application team would test their applications on a RHEL 8 test environment before migrating their applications to production.

# Conceptual Architecture

## Proposed Solution

**RHEL 8 AMI Build**

Below is the proposed RHEL 8 AMI build solution. This solution uses standard AMI automated build process using the Gitlab CI/CD pipeline.

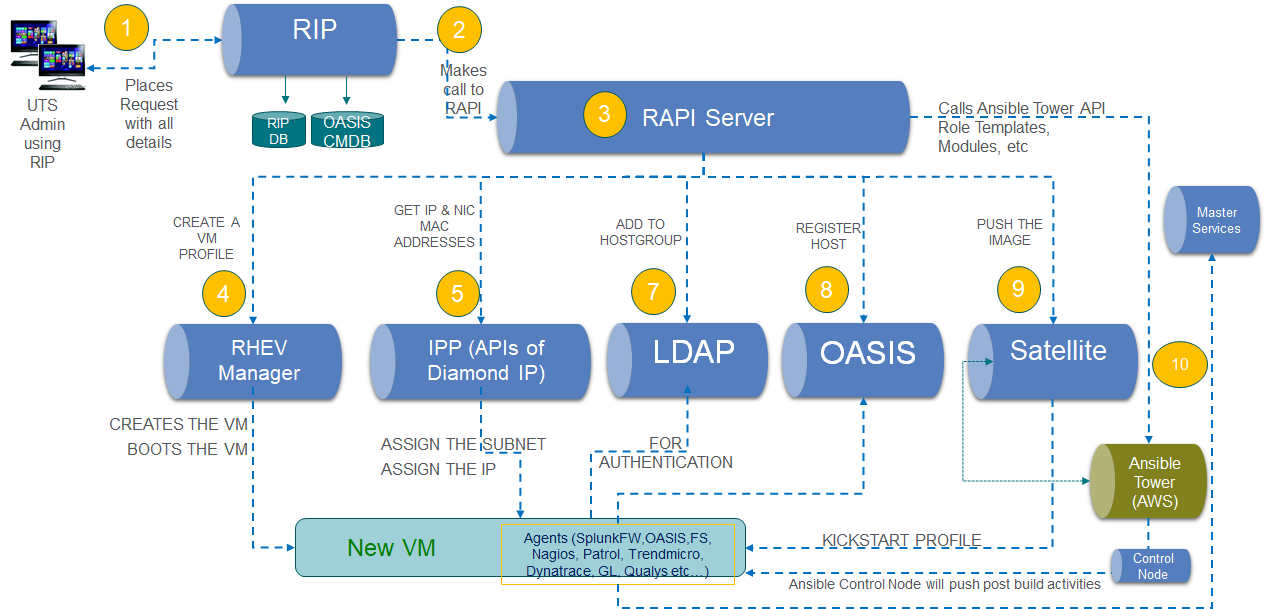


Giltab pipeline - IAG RHEL 8 Gold AMI creation:

* 1. Trigger the Gitlab CI pipeline (https://docs.gitlab.com/ee/ci/pipelines/)
  2. Gitlab Runner will trigger the job (https://docs.gitlab.com/runner/)
  3. The runner as per the pipeline instruction will pull a Packer Image from Docker Hub (<https://hub.docker.com/r/hashicorp/packer>)
  4. Start the new docker machine in an ec2 instance
  5. The Packer template will be pulled from Gitlab repos
  6. Packer will trigger the packer template script
  7. Packer will do below things step by step as part of Packer script
     + - Launch an ec2 instance with Enterprise RHEL 8 Image (access has been given by Redhat)
       - Play the Ansible Roles (part of GITLAB Repo) for Security hardening on the EC2 instance
       - Create a new LVM partition
       - Copy the entire boot partition to this new partition
       - Make it the boot partition
       - Create an AMI and store in the account, Tag Mark the AMI as Status=Testing
  8. Test the AMI using Terraform scripts by launching it and add some automated additional testing, Tag Mark the AMI as Status=Approved, and Share the image with all AWS accounts under IAG Master

**On-prem RHEL 8 Build Provision**

Below is the proposed solution to build and provision RHEL 8 virtual machine, on the on-prem infrastructure.



RIP workflow - IAG RHEL 8 VM provisioning:

1. UTS team will have access to the in-house tool RIP to provision RHEL 8 VMs.
2. RIP is UI web application, which calls the RAPI (python flask api server) to provision a VM on the on-prem infrastructure.
3. RAPI internally calls to various tools Satellite, RHEV manager, Diamond IP, Corp LDAP, OASIS, Ansible Tower to create a VM and configure it.

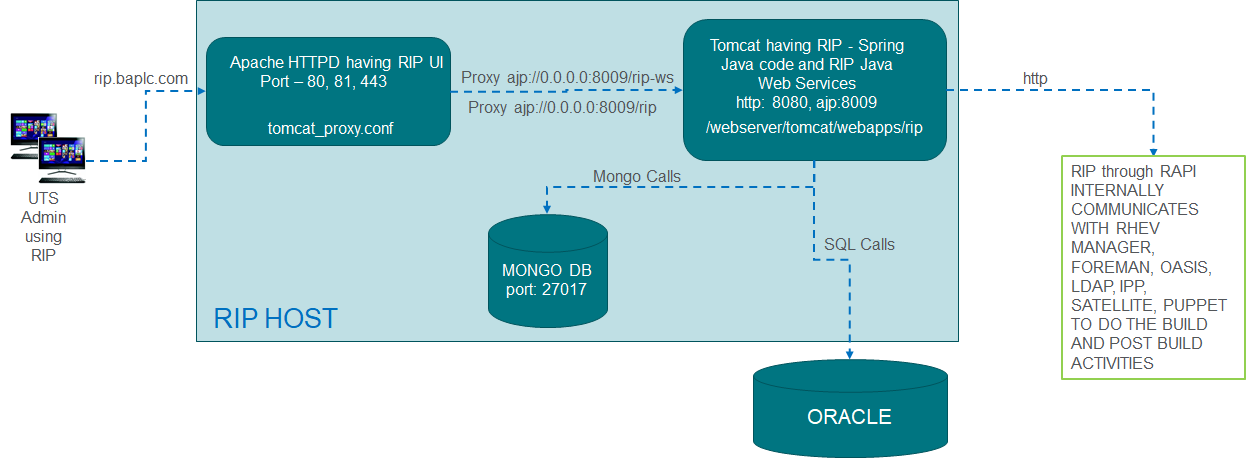
### RIP Framework changes for on-prem RHEL 8 Build

RIP is a 2 tier architecture java based application. It has Apache + Tomcat at the frontend, and a database ( 2 in fact - MongoDB and Oracle ) at the backend.

The end user always talks to the Apache HTTP server. Apache responsible for deciding what traffic it serves itself, and what it passes onto the Tomcat instance. It does this by pattern matching on the URL as defined in the ProxyPass statement.

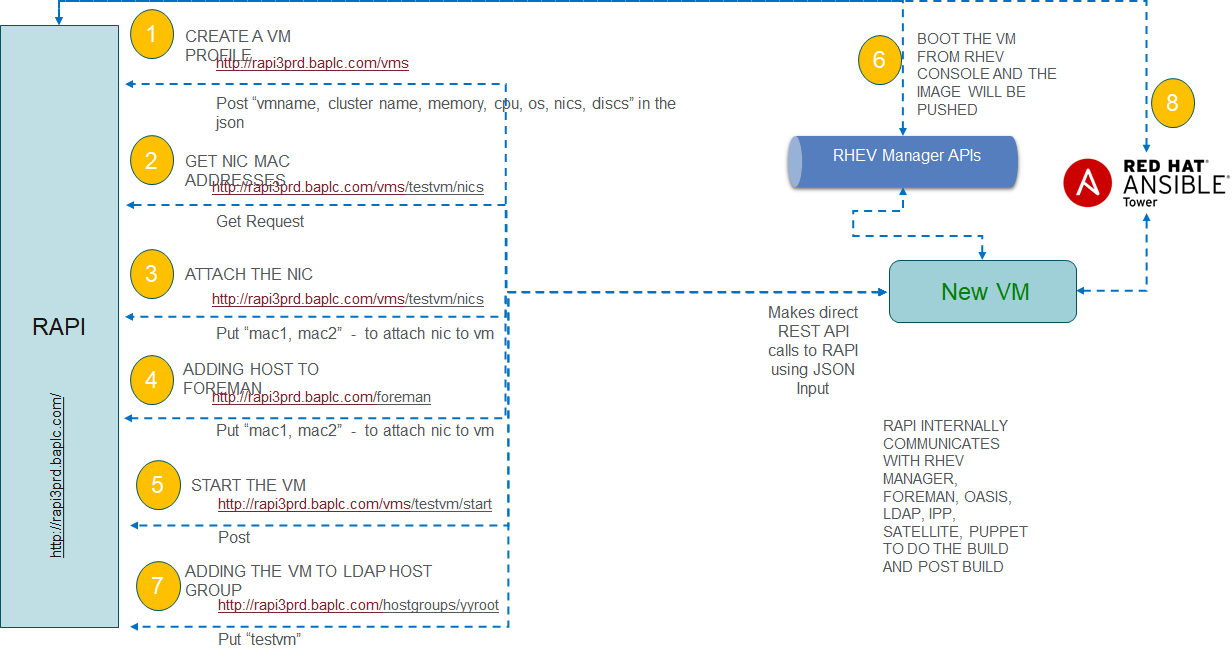
In the case of RIP that static UI Angular code is hosted on Apache. This project, the output of which is a web application that serves all the RIP backend services runs on Tomcat.

RIP UI changes and Web services changes will be made to Provision RHEL 8 VMs and extracting the progress and logs and show it in the reporting.



### RAPI Changes for on-prem RHEL 8 build

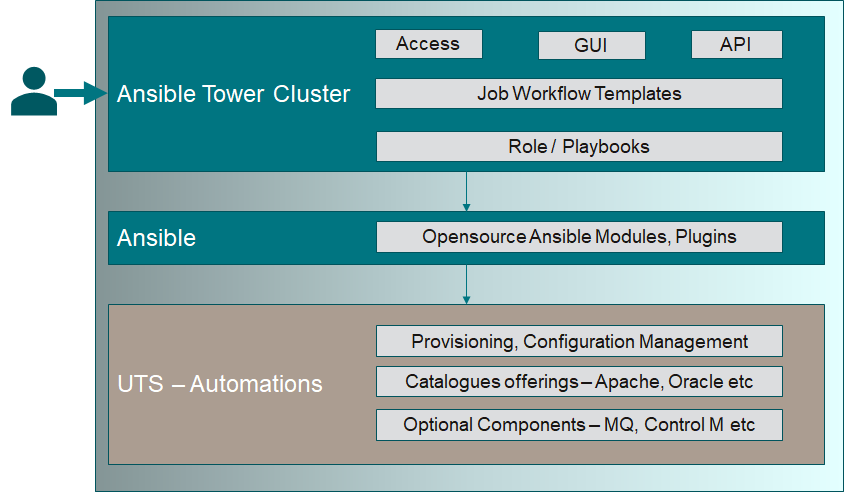
RAPI is a set of Python Flask based rest APIs which connects to the different IAG infrastructure tools to provision a VM. Additional APIs will be added to provision RHEL 8 systems and integrations with Ansible Tower.



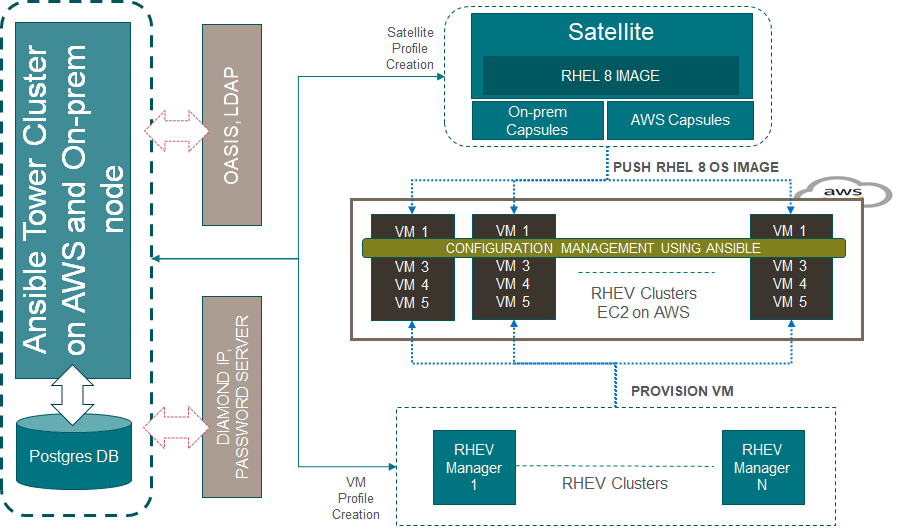
### Ansible Tower Architecture

Some of the Ansible features are:

* Ansible is easy to code, manage.
* Agentless
* Graphical user interface dashboard
* Vendor support and industry standard.
* Multi playbook workflow using Templates
* Rest API, Web Services
* Future extendable to Windows platform, Network and other IT Automation



* We have a 3 node AWS based Ansible Tower cluster.
* There is one more node in on-prem as well.
* All configuration code would be maintained in Gitlab.
* Ansible Tower Job templates would be in sync with Gitlab code.
* By making an Ansible Tower API call, we can push the latest configurations to AWS AMI and servers build by RIP on-premise.
* RIP/RAPI will be integrated with Ansible tower.
* RHEL 8 AMI build process will be integrated with Ansible tower, so that AMI configurations can be pushed by Ansible Tower templates.
* SCM configurations for RHEL 8 hardening will be another set of Job templates maintained by SCM team will be triggered as part of the RHEL 8 AMI and on-prem build process.



### RIP & Ansible Tower Integration

Ansible Tower APIs are exposed at the below URL, RIP will be integrated to Ansible Tower using the below API.

https://tower.elc.internationalairlinesgroup.com/api/v2

### Satellite & Ansible Tower Integration

The primary roles of Satellite are:

- Content Management and Patching

- Life cycle management

- Provisioning mechanisms

- Source of Truth for all systems, store and allow hierarchical management of configuration parameters/variables.

- Manage server side content (yum repos etc.)

- Provide low-level tooling for provisioning (e.g. PXE backend where appropriate, various types of boot images, etc).

Whereas the primary roles of Ansible Tower are:

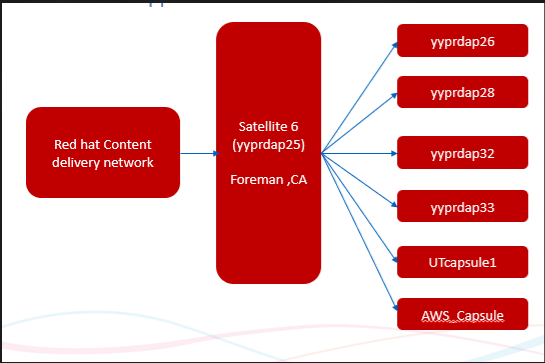
- Know everything Satellite knows about each system (using a Dynamic Inventory).

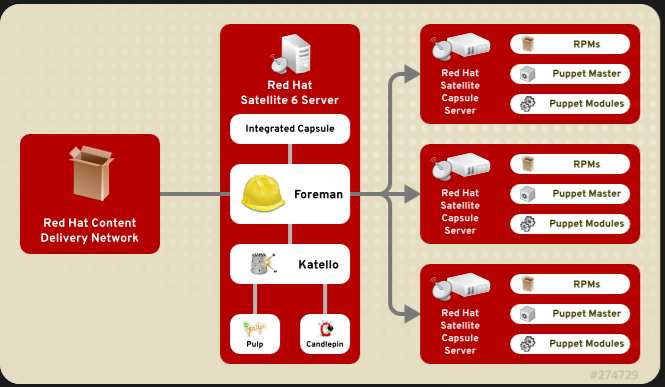
- Securely store and employ the credentials used to interact with external systems.

- Provide a configuration management system using the parameter/variable values from Satellite.

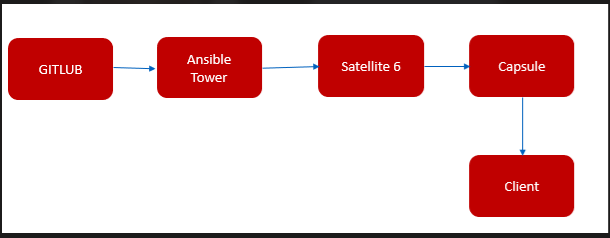
- Orchestrate user-defined workflows. For example, a workflow can enhance the Satellite’s provisioning capability by performing actions on other external systems before or after the actual deployment (create a VM, update a CMDB, update DNS, configure firewall rules etc.), or provision several different systems together as part of deploying a related product stack (for example, an OpenShift cluster).

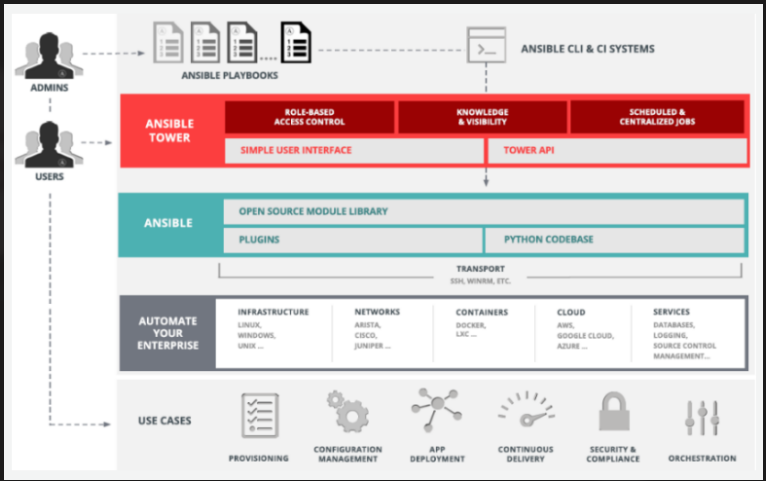
IAG Satellite Platforms-



Satellite Architecture-

Using Ansible Tower to automate the task-



Ansible Tower complete Architecture-

### SCM Integration for RHEL 8 OS Hardening

RHEL 8 hardening standards and controls are defined, developed and maintained by SCM team. The Ansible roles/code base for these hardening configurations is available in the below GiltLab group repositories. These Gitlab repositories are integrated with Ansible Tower as Ansible Job templates.

<https://gitlab.com/iairgroup/restricted/cyber-scm-fim/hardening>

Once the RHEL 8 Server is provisioned on-premise or during the AWS AMI creation process, an Ansible Tower call back will be triggered so that corresponding RHEL 8 Job/Workflow will be triggered for RHEL 8 server hardening.

<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/pdf/security_hardening/red_hat_enterprise_linux-8-security_hardening-en-us.pdf>

SharePoint link of SCM project:

<https://iairgroup.sharepoint.com/sites/FIMProjectImplementation/Shared%20Documents/Forms/>

### Components of RHEL 8 AMI

Following components/agent software will be part of RHEL 8 AMI configurations:

|  |  |  |  |
| --- | --- | --- | --- |
| Scope | Tool | Vendor | Remarks/ Contacts /Owner |
| Monitoring | Cloudwatch Agent/ Splunk/AWS Cloudtrail |  |  |
| Alert | BES |  |  |
| Backup | AWS Backup/Veritas Netback up |  |  |
| EDR | Falcon-sensor | Crowdstrike |  |
| PCI Data Compliance |  |  |  |
| Patching/Lifecycle Management | Satellite/Capsule | Redhat |  |
| Configuration Management | Ansible Tower | Redhat |  |
| CMDB | ServiceNow |  |  |
| Vulnerability Assessment | AWS Inspector or Qualys |  |  |
| Password Management | Password server |  |  |
| Discovery Services | AWS Systems Manager/ AWS Licence Manager |  |  |
| User Authentication / Authorization | AWS SSM / BA CD |  |  |
| IP Management | AWS Route 53 |  |  |
| Filesystem | EBS/EFS |  |  |
| Workflow Automation | Terraform/Ansible |  |  |
| Virtualization | AWS |  |  |
| Cluster | NA |  |  |

### Components of on-prem RHEL 8 build

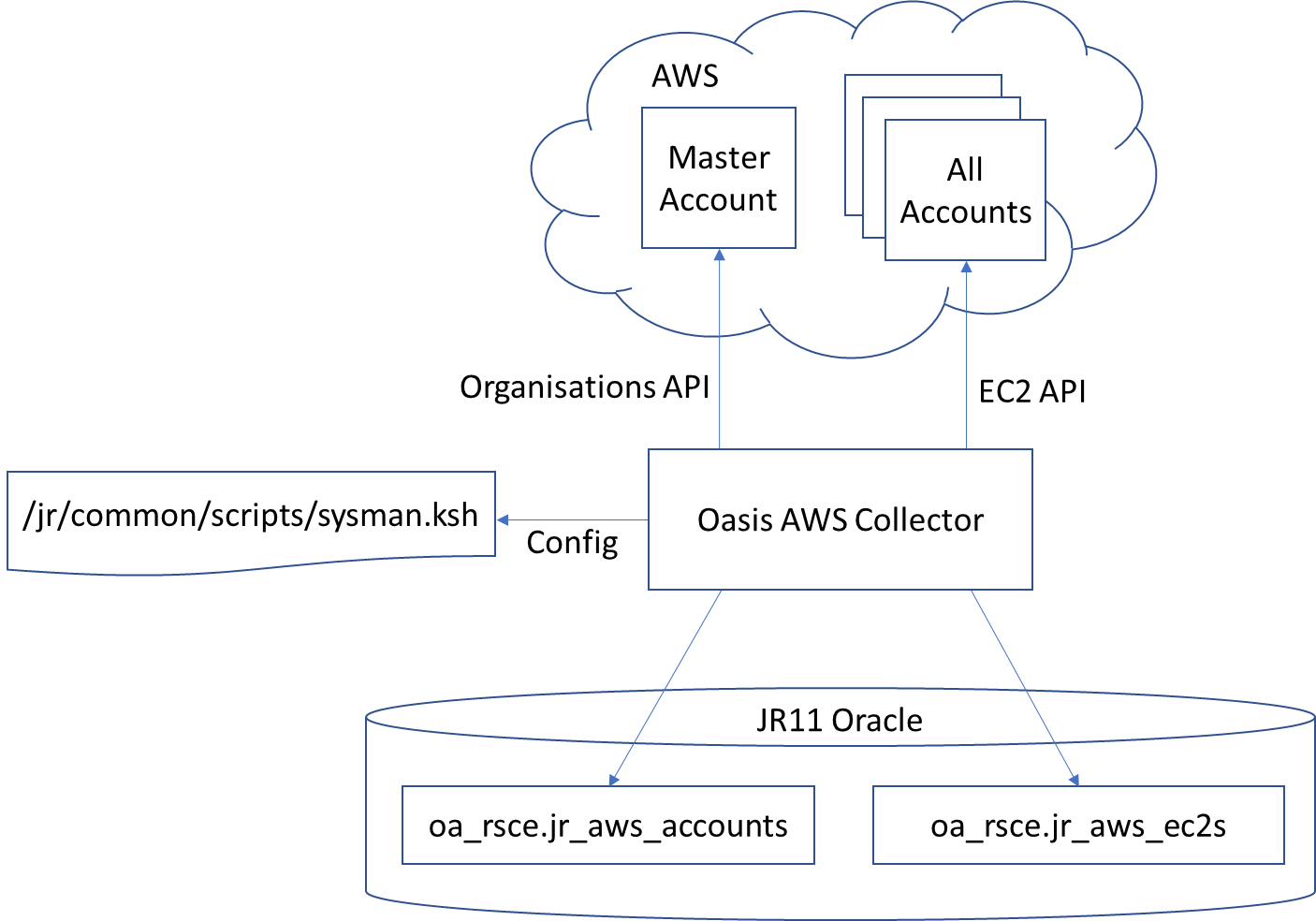
Following components/agent software will be part of RHEL 8 on-prem builds:

|  |  |  |  |
| --- | --- | --- | --- |
| Scope | Tool | Vendor | Remarks/ Contacts /Owner |
| Monitoring | Nagios/ Splunk |  |  |
| Alert | BES |  |  |
| Backup | Veritas Netback up |  |  |
| EDR | Falcon-sensor | Crowdstrike |  |
| PCI Data Compliance | Grounlabs/Trendmicro |  |  |
| Patching/Lifecycle Management | Satellite/Capsule | Redhat |  |
| Configuration Management | Ansible Tower | Redhat |  |
| CMDB | SNOW/ OASIS |  |  |
| Vulnerability Assessment | Qualys |  |  |
| Password Management | Password server |  |  |
| Discovery Services | Service Now |  |  |
| User Authentication / Authorization | LDAP/BASU |  |  |
| IP Management | Diamond IP |  |  |
| Filesystem | LVM, EXT3/4, XFS, NFS |  |  |
| Workflow Automation | RIP |  |  |
| Virtualization | RHV |  |  |
| Cluster | RHCS |  |  |

### OASIS AWS Collector

This component is used to harvest Account and EC2 information out of AWS and synchronise it into the Oasis database.

It consists of a standalone executable Java archive (.jar file)



This component writes data into OASIS tables directly jr\_aws\_accounts and jr\_aws\_ec2s, both under the oa\_rsce user.

As the names imply, jr\_aws\_accounts holds details on AWS accounts while jr\_aws\_ec2s holds details on every EC2 instance within each account.

### Baseline monitoring for RHEL 8 AMI

Till any other new mechanism is evolved, we can follow the existing method.

IAG Tech Service Ops staff use Interlink Software’s BES product to provide a single-pane-of-glass view of service health. This section details how IAG have integrated AWS events with the existing BES service in order to give IAG Tech Service Ops staff a view of AWS-based service health.

Interlink BES integration with AWS is achieved using AWS SQS (Simple Queue Service) queues.

The **iag-tech-management-prd** AWS account (**706552515435**) has been configured with the following two SQS queues:

* **SQS\_BES\_UAT** (used by BES UAT)
* **SQS\_BES\_PRD** (used by BES PRD)

The production and UAT instances of BES have been configured to pull messages from these queues.

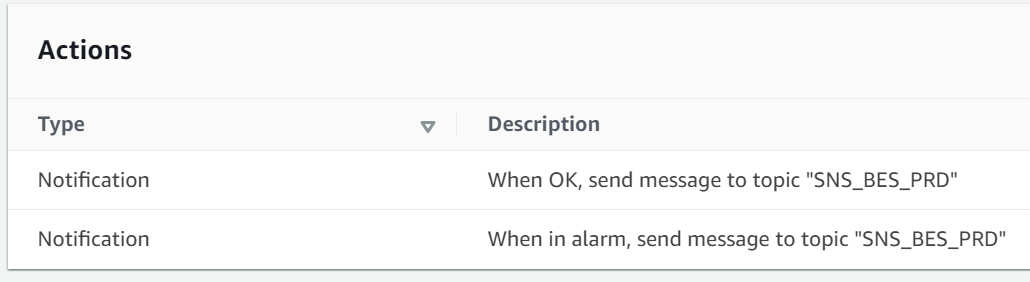
The **iag-tech-management-prd** AWS account contains two AWS SNS (Simple Notification Service) topics. The SQS queues are subscribed to these topics. Any messages sent to these topics will end up on the subscribed queues:

* **SNS\_BES\_UAT** (SQS\_BES\_UAT is subscribed to this topic)
* **SNS\_BES\_PRD** (SQS\_BES\_PRD Is subscribed to this topic)

Diagram

Description automatically generated

CloudWatch alarms can be configured to send notifications to an SNS topic:



The following diagram shows the complete end-to-end flow of a custom CloudWatch agent alarm into BES:

CloudWatch Agent config -> CloudWatch Alarm -> SNS topic -> SQS queue -> BES

Diagram

Description automatically generated

For giving access to CloudWatch of new AWS accounts, SNS policy (under AWS Account 706552515435) need to be updated as below:

You must allow the desired resources in the AWS account to perform the **sns:Publish** action on the SNS topic(s). If the ETM team are happy for the messages to be received in the production BES server, then you can update the access policies of both SNS topics.

If the monitoring is new, and if ETM wish to test it in UAT before exposing PRD to these messages, only give access to the SNS\_BES\_UAT topic. Liaise with ETM if you are not sure.

Here is a sample SNS access policy that allows CloudWatch alarms in the splunk-cluster-prd account (542780778927) to publish on the SNS\_BES\_PRD topic:

**{**

**"Sid": "SNS\_Publish\_CloudWatch\_542780778927",**

**"Effect": "Allow",**

**"Principal": {**

**"AWS": "\*"**

**},**

**"Action": "sns:Publish",**

**"Resource": "arn:aws:sns:eu-west-1:706552515435:SNS\_BES\_PRD",**

**"Condition": {**

**"ArnLike": {**

**"AWS:SourceArn": "arn:aws:cloudwatch:\*:542780778927:alarm:\*"**

**}**

**}**

**}**

### Baseline Monitoring for RHEL 8 on-prem

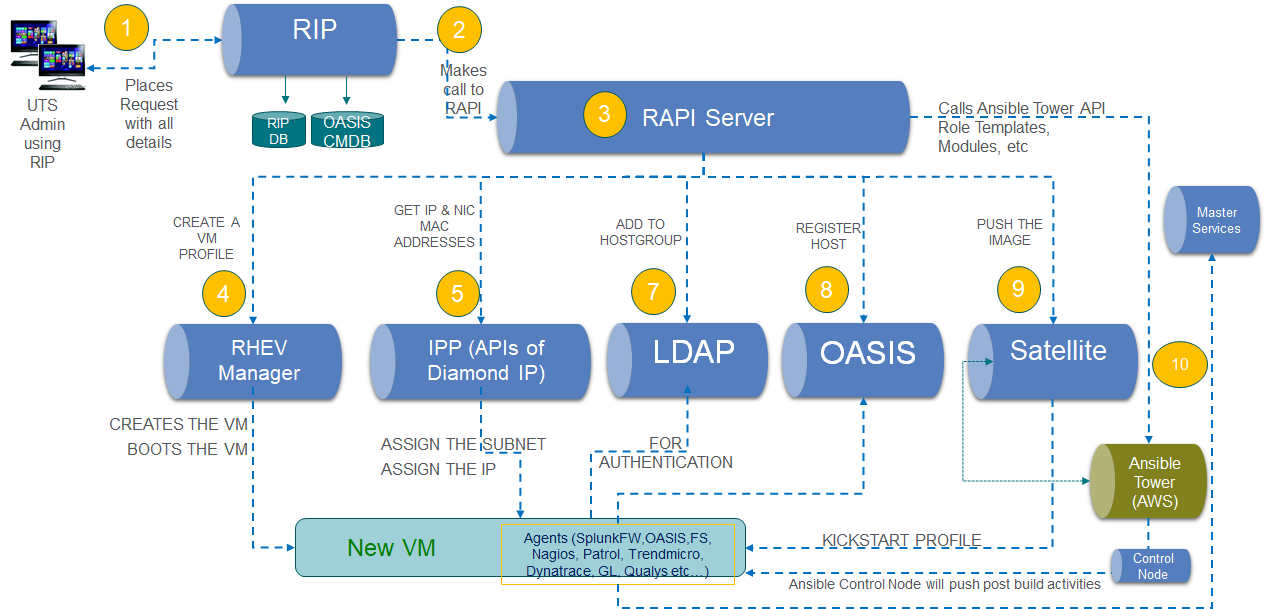
* On-prem RHEL8 Baseline monitoring details will be added. It will continue to use the Baseline monitoring mechanisms/parameters of on-prem RHEL7.

### Business Capability Architecture

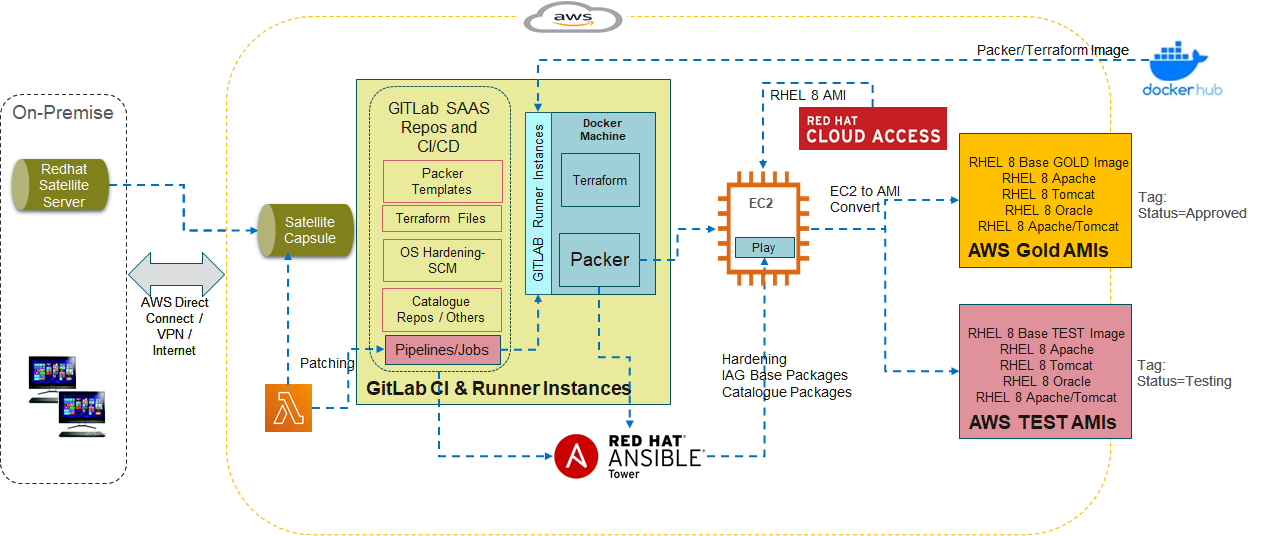
NA

### Information Systems Architecture

On-Prem Architecture:

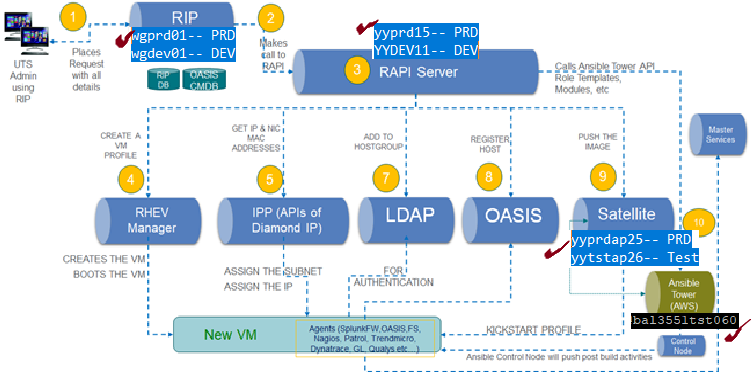


Cloud Architecture:

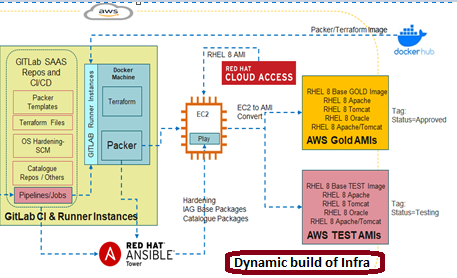


### Platform Architecture

On-Prem

****

Cloud



## Assumptions and Dependencies

* Dependency on Satellite Integration with Ansible Tower.
* Dependency on GitLab Integration with Ansible Tower.
* Dependency on RAPI Integration with Ansible Tower.
* Dependency on SCM team to provide hardening scripts at the right time which will be collaborated with other respective scripts for their deployment.

## Architecture Risks and Issues

* Using existing complex architecture of RIP and RAPI (in house tool).