**Solution Architecture Report - Ansible**

**ERICSSON - Contratista III Program**

**Version 1.1**

Author: Balaratnam Puppala

Henrik Vahlstedt

Owner: Jan Pettersson

Ericsson and Kyndryl Confidential

Document History

Document Location

This is a snapshot of an on-line document. Paper copies are valid only on the day they are printed. Refer to the author if you are in any doubt about the currency of this document.

Revision History

|  |  |
| --- | --- |
| Date of the approval : 2022-03-28 | Date of next revision *Not Applicable* |

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision Number** | **Revision Date** | **Summary of Changes** | **Changes marked** |
| 0.1 | 2021-12-05 | Initial Draft | (N) |
| 0.2 | 2021-12-05 | C3 changes updated | (Y) |
| 0.3 | 2021-12-23 | Network flow diagrams updated | (N) |
| 0.4 | 2022-01-24 | Updated Network ports | (N) |
| 1.0 | 2022-03-28 | Final Approved version |  |
| 1.1 | 2022-05-02 | Updated WSUS setup for windows patching | (Y) |

Approvals

This document requires following approvals.

|  |  |
| --- | --- |
| **Name** | **Title** |
| Jan Pettersson | Kyndryl Chief Architect |
| Henrik Vahlstedt | Kyndryl Account Architect |
| Bhalchandra Waykole | Kyndryl T&T Integration Architect – C3 |
| Various | Kyndryl Internal Technical Design Authority |
|  |  |

Distribution

This document has been distributed to

|  |  |
| --- | --- |
| **Name** | **Title** |
| Various | All approvers of this document (see table immediately above) |
|  |  |

Contents

1. Introduction 5

1.1 Cloud Automation Capabilities 5

1.2 Project Background 7

1.3 Purpose of this document 7

1.4 Scope 7

1.5 Key Contacts 8

2. Requirements 9

2.1 Business requirements 9

2.2 Functional Requirements 9

2.3 Non-functional Requirements 10

3. Current Environment 13

3.1 Current System Context 13

3.2 Current IT 13

3.3 Current Operational Model 14

3.4 Candidate Assets List and Applicable Standards 14

4. Solution Overview 15

4.1 System Context Diagram 15

4.2 Key Users and Interfaces 16

4.3 Architecture Overview Diagram 19

4.4 Architectural Decisions 21

4.5 Conceptual Operational Model 26

4.6 Product and Delivery Maturity 27

5. Operational Design 28

5.1 Specified Operational Model 28

5.2 Application Design 28

5.3 Prerequisites 28

5.4 User Authentication 30

5.5 Network Model 30

5.6 Environments Model 43

5.7 System Management Model 43

5.8 Performance 43

5.9 Support Model 43

5.10 Security Architecture 4.2.1 43

5.11 Managed device Credentials Management 46

5.12 Security and Compliance Controls 47

6. Implementation 49

6.1 Testing and Validation 49

6.2 Deployment 49

6.3 Migration 49

7. Viability Assessment 50

7.1 Assumptions 50

7.2 Dependencies 50

7.3 Risk 51

7.4 Issues 51

8. Bill of Materials 52

9. Appendix 53

9.1 Playbooks 53

# Introduction

The objective of this document is to outline the solution design, to ensure that all stakeholders have the chance to review and comment on the design, and to support formal approvals of the solution design.

The Cloud Automation Community Framework (CACF) with ansible defines a standard, cloud native automation platform and application architecture with the Red Hat flagship products Ansible, Ansible Tower and OpenShift Container Platform. It is supported by a community model for content development in a global content library in Kyndryl GitHub, based on Site Reliability Engineering (SRE) principles. It enables account subject matter experts (SMEs) to reuse and contribute local automation assets from/to the global content library. Together, the CACF platform and the community model comprise the exciting next generation of Kyndryl’s automation strategy.

The Ericsson - Private Cloud Platform in IBM Cloud is managed by Kyndryl Hybrid Services Team (HST), which has its own Ansible and Ansible Tower located in Kyndryl Service Technology Platform (KSTP) formerly called ISPwW (IBM Service platform with Watson) at Amsterdam.

## Cloud Automation Capabilities

These two major Cloud Automation Community Framework (CACF) and KSTP capabilities will change the automation landscape Ericsson - Private Cloud Platform in IBM Cloud managed by Kyndryl.

* **Cloud native application development and deployment:** While Ansible and Ansible Tower could be deployed on traditional virtual machines or cloud-enabled environments, Kyndryl – like many of its clients – is focused on cloud native, container-based implementations going forward. Cloud native applications can leverage the services and elastic infrastructure inherent to containerization and cloud computing. Containers can be rapidly scaled as applications grow, or easily replaced in the event of problems as part of agile DevOps processes. Cloud native deployments allow also for higher availability as containers are brought back up automatically.
* **Many automation functions on ONE Platform:** Going forward, CACF will be the single strategic platform for automation offerings, with multi-tenant AND multi-application deployments. Clients can pick and choose the application services they need on the common platform.
* **Incident- and event-initiated automation:** Integration with Global Systems Management (GSMA) Monitoring and Event Management (M&E) instances in current release, and in future, ticketing systems for automated incident remediation and event management. It is expected that direct incident ticket integration, for incidents to trigger the playbook, will be made available in upcoming CACF releases.
* **Generic Automation**: These are community developed playbooks that can be reused or contributed and are initiated by humans and/or scheduled within CACF. This allows for SMEs to create and run playbooks to reduce any manual activity currently performed.
* **Compliance automation:** This includes security health checking of managed client systems. The scanning results are made available via CACF secure file server and eventually will flow into the compliance reporting tools. Security enforcement.
* **Patch Automation:** This contains the complete lifecycle from patch scanning to patch tracking and patch execution. Patch execution is to be done by the native tools within a client environment, whereas patch scanning and patch tracking are done by CACF.
  + **Patch scanning:** Automated scanning of managed endpoints, with scan results available via CACF secure file server. Patch scanning playbooks have been released for RHEL and Windows. Future CACF release will add additional patch scanning functionality.
  + **Patch execution:** In general, all platform patching activities must be done in native tools, like WSUS for Windows or RedHat Satellite for RHEL. CACF does have basic patch execution playbooks for RHEL in the Continuous Engineering repository.
  + **Patch tracking:** Patch scanning results are uploaded to HST Role Based Continuous Patch (RCP) instances for tracking purposes.
* **Additional use cases:** kyndryl support team may choose to create its own playbooks for new use cases and enable them within their Ansible Tower organization based upon request for Services (RFS) intake approval or Kyndryl Delivery Project Executive (DPE) / Client Partner Executive (CPE) approval. Examples:
  + **Service Request:** (future roadmap/community developed) Integration to an IBM Control Desk (ICD) or ServiceNow QuickStart catalog for automating service requests.
  + **Server Build and Decommission:** (future roadmap/community developed) Automated server provisioning and de-provisioning including post-build activities.
  + **OS Image Management (a.k.a. Bluewash):** Perform any necessary OS customization to the standard OS base image automatically. This may include but not limited to installation of IBM tools agent, configuring necessary OS hardening per IBM CSD / TechSpec and IP address and hostname assignment.
* **Plug-ins / Integrations** are required for key client use cases. For example, CACF requires an integration component, called NEXT, between GSMA M&E and the Ansible Tower instance used within CACF. This (integration) solution for incident- and event-initiated automation is not available “out of the box” with Ansible Tower
* **Community Model for Content Development and Governance:** Recent Kyndryl automation plays have had centralized content development, with automata requirements reviewed through an intake process and then queued for development. New with the CACF framework is a community model approach for content development based on Site Reliability Engineering (SRE) principles. In the spirit of open source, the CACF architecture includes a GitHub platform for hosting both CACF-certified and community-sourced content. A global governance team will oversee content in the master “Continuous Engineering” GitHub project to ensure code stability and quality. Account specific GitHub projects will enable account SMEs to contribute their own automation assets for local use; these can also be shared with the automation community through the Continuous Engineering governance process. More information can be found in the links below:

**Note:** Unless a change to the MSA is specifically referenced in this document (and then implemented via a Change Note to the MSA) the content of this document does not amend or change any of the provisions, obligations, scope et al of the MSA. The MSA will always take precedence.

## Project Background

Ericsson - Private Cloud Platform in IBM Cloud are delivered by the Contratista programme. Kyndryl will modernize and improve the services delivered to Ericsson by introducing Ansible (CACF) as an automation engine. And replacing BiGFix applications for security health checks and patch management. With Ansible, Kyndryl and Ericsson would improve the services and be able to address additional automation capabilities listed below.

The solution will be available to use for the Production, UAT and Dev workloads across all four of Ericsson’s defined IBM Cloud locations as follows:

• Frankfurt (FRA02, FRA04),

• Washington (WDC04, WDC06),

• Sydney (SYD04, SYD05),

• Chennai (CHE01).

## Purpose of this document

The purpose of this update is to capture the design specifications for the Ansible and Patching solution for Ericsson Cloud environment built on the IBM Cloud platform.

Objectives of the document are to:

• Provide guidance to the Solution Design Teams

• Support across all Architecture Board meetings and technical discussions

• Facilitate effective communication between different stakeholders

• Facilitate orientation for new people who join the project or delivery organization

## Scope

The scope of this design is enabling automation with HST Ansible and, to replace HCL BigFix software which is being sunset Ansible will integrate with the Ericsson Private Cloud Infrastructure in all 4 Locations (7 Data centers) in scope of the Purchase Agreement (PA) contract, namely Frankfurt, Chennai, Sydney, and Washington.

This document covers the following scope:

* **Compliance automation:** This includes security health checking of managed client systems. The scanning results are made available via CACF secure file server and eventually will flow into the compliance reporting tools. Security enforcement.
* **Patch Automation:** This contains the complete lifecycle from patch scanning to patch tracking and patch execution. Patch execution is to be done by the native tools within a client environment, whereas patch scanning and patch tracking are done by CACF.

Out of scope are

* Any other activity not specifically covered above is also out of scope.

## Key Contacts

Key contacts for this area are:

Kyndryl Key contacts for this area are:

|  |  |
| --- | --- |
| **Name** | **Job/Project Role** |
| Jan Pettersson | Kyndryl Chief Architect |
| Henrik Vahlstedt | Kyndryl Account Architect |
| Arunprassath Gnanasekaran | Kyndryl Local Delivery Project Executive |
| Bhalchandra Waykole | Kyndryl T&T Architect |
| Various | Kyndryl Internal Technical Design Authority |

Ericsson Key contacts for this area are:

|  |  |
| --- | --- |
| **Name** | **Job/Project Role** |
| Gaurav Gupta | Ericsson Solution Architect |
| Ashish Raj | Ericsson Solution Architect |

### Related Documents

The following documents are related and form a part of the system architecture. For the avoidance of doubt, the related documents form part of the System Architecture document suite for this solution.

| **Document** | **Reference Information (and Location)** |
| --- | --- |
| Ericsson - Private Cloud Design - Solution Architecture Report | 1. <https://ibm.ent.box.com/folder/102514364699> |
| Ericsson - Network Design - Solution Architecture Report | 1. <https://ibm.ent.box.com/folder/87633731366> |
| Ericsson - Service Management - Solution Architecture Report | 1. <https://ibm.ent.box.com/file/603967503211> |
| Citrix XenApp/XenDesktop Solution Design Document | 1. <https://ibm.box.com/s/96snqvyh9dzpkf4wo1t24b0qxd3czedn> |

# Requirements

## Business requirements

The Business Requirements are summarised as follows:

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-BUS-001 | Must | Integrated Service Management environment is required across Private Cloud environment for Ericsson, including platforms, storage, backups, network, hypervisor, automation, security. |
| ERI-ANS-BUS-002 | Must | Provide a cost-effective automation solution to support Cloud enabled and Cloud native solutions to reduce manual labour for maintenance and operations. |
| ERI-ANS-BUS-003 | Must | Provide automation capabilities in the Ericsson - Private Cloud Platform in IBM Cloud to support generic automation (human initiated and scheduled), patching checking, compliance checking. |
| ERI-ANS-BUS-004 | Could | Incident- and event-initiated automation, service request, server build & decommission and other manual IT labour opportunities. |

## Functional Requirements

This section describes the functional requirements for technical components where the behaviour is not defined by a use case model.

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-FR-001 | Must | Ability to trigger automation in the Ericsson - Private Cloud Platform in IBM Cloud. |
| ERI-ANS-FR-002 | Must | Support development of automation workflows and error handling for multiple use cases (patching, compliance, incident remediation, service request, server build and decommission) |
| ERI-ANS-FR-003 | Must | Integration with existing ecosystems (IBM tools such as Netcool, IBM Control Desk, or ServiceNow and other market players) |
| ERI-ANS-FR-004 | Must | Ericsson owns Red Hat Enterprise Linux licenses. This means Ericsson owned Red Hat Satellite servers needs to be used as the native patching tool. |
| ERI-ANS-FR-005 | Must | Ericsson owns the Windows licenses.  IBM’s WSUS servers will to be used as the native patching tool. |
| ERI-ANS-FR-005 | Must | Integration with existing ecosystems (IBM tools such as Netcool, IBM Control Desk, or ServiceNow and other market players) |
|  |  |  |

## Non-functional Requirements

### Regulatory Requirements

No regulatory requirements identified.

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| N/A | N/A | N/A |

### Summary of Volumetric Requirements

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS- NFR-001 | Must | A correct quantity of Ansible proxies will be implemented in order to handle Ericssons server baseline. |

### Qualities

#### Performance

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS- NFR-002 | Must | Provide sufficient performance to meet approved maintenance-window for patching and change |

#### Capacity

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS- NFR-003 | Must | Reduce footprint on client environment by not requiring agents to be installed |
| ERI-ANS- NFR-004 | Must | Automatically scale infrastructure or service as needed |

#### Availability

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS- NFR-005 | Must | High availability (99.7%) since this service is expected to automate daily IT operations and outage of the service might affect overall service quality and SLAs. |

#### Disaster Recovery (RTO and RPO)

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| N/A | N/A | N/A |

#### Security

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS- NFR-007 | Must | Support Ericsson security standard in addition to Kyndryl policies as agreed by Ericsson in the CSD. |
| ERI-ANS- NFR-08 |  | Accounts control access to their Ansible Tower organization and determine which automation playbooks are enabled, and the granular privileges associated with Ansible Tower objects. |

#### Systems Management

.

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-NFR-009 | Must | Support secure multi-tenant deployments. |
| ERI-ANS-NFR-0010 | Must | Complement existing automation solutions and / or provide a capability to replace where applicable with full re-use of existing network connectivity and / or other components |
| ERI-ANS-NFR-011 | Must | Automation capabilities (i.e. Ansible playbooks) should be readily and easily shareable across geographies, or accounts within a geography with no or minimal changes. |
| ERI-ANS-NFR-012 | Must | Solution shall support development of content in a community model as an ‘open-source’ within Kyndryl. |
| ERI-ANS-NFR-013 | Must | Accounts control access to their Ansible Tower organization and determine which automation playbooks are enabled, and the granular privileges associated with Ansible Tower objects |

#### Scalability

.

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-NFR-014 |  | Automatically scale infrastructure or service as needed |

#### Data Integrity

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-NFR-015 | Optional | Data at rest should be encrypted |

#### Reliability

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| N/A | N/A | N/A |

### Constraints

#### Technical Constraints

##### Legacy Integration

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-NFR-016 | Must | Ansible Tower must integrate with M&EaaS Netcool and HST Service Now |

##### Existing Infrastructure

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| ERI-ANS-NFR-017 | Option | Reuse current IEM relay servers as Ansible Proxy servers |

##### Implementation Constraints

No constraints given

| **Requirement ID** | **Priority** | **Summary Statement** |
| --- | --- | --- |
| N/A | N/A | N/A |

# Current Environment

## Current System Context

N/A

## Current IT

The Compliance Support Component of the Security Compliance and Regulatory (C&R) provides oversight for Service Delivery on the adherence to, and the delivery of the required Security Policies, Standards, and Regulations. A defined toolset is utilized to manage compliance issues and ensure schedule attainment for these endpoints and metric attainment of the compliance schedule. The primary tool is HCL BigFix, which is based on agents on each server that monitor patch status and can collect security settings for validation.

Kyndryl will consequently implement BigFix software for:

* System Health Checks
  + At agreed intervals, perform periodic health check scan of the in-scope environment as per Technical Specification parameters and notify Client with list of deviations.
  + perform remediation of identified deviations for in-scope parameters once per year during standard change window.
  + perform the next regular health check based on the health check schedule in case remediation request for previous health check is not approved.
* Patch Management
  + install high, medium, and low patches during established change windows, at agreed intervals
  + apply critical patches based on vendor notification/Kyndryl rating, through the agreed upon change management process.
  + install out of cycle patches via an RFS during Client-provided security change window. If an RFS is not submitted, patches will be applied during the next regular patching schedule.

### BigFix / IEM detailed connections FRA DC

Kyndryl tooling is deployed in M2M (Machine 2 Machine IBM Zone) in the overlay network. The implementation of BigFix (IEM) in Frankfurt is attached for reference. The overview diagram below shows only specific M2M network flows between underlay and overlay for BigFix (IEM).



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Flow ID | Source Zone | Destination Zone | Protocol & Port (L4 FW) | Overlay/Underlay/Both |
| Ansible\_Tools-group2-flow-1 | IBM FRA-Underlay | IBM FRA-PROD-M2M | UDP-52311 TCP-52311 | Both |
| Ansible\_Tools-group2-flow-2 | IBM FRA-Underlay | IBM FRA-DEV-M2M | UDP-52311 TCP-52311 | Both |
| Ansible\_Tools-group2-flow-3 | IBM FRA-PROD-M2M | IBM FRA-Underlay | UDP-52311 TCP-52311 | Both |
| Ansible\_Tools-group2-flow-4 | IBM FRA-DEV-M2M | IBM FRA-Underlay | UDP-52311 TCP-52311 | Both |

## Current Operational Model

Not applicable

## Candidate Assets List and Applicable Standards

Not applicable

# Solution Overview

## System Context Diagram

The System Context represents the entire Ansible system as a single object and identifies the interfaces between the system and external entities. Ansible is also referred to as Kyndryl Cloud Automation Community Framework (CACF),

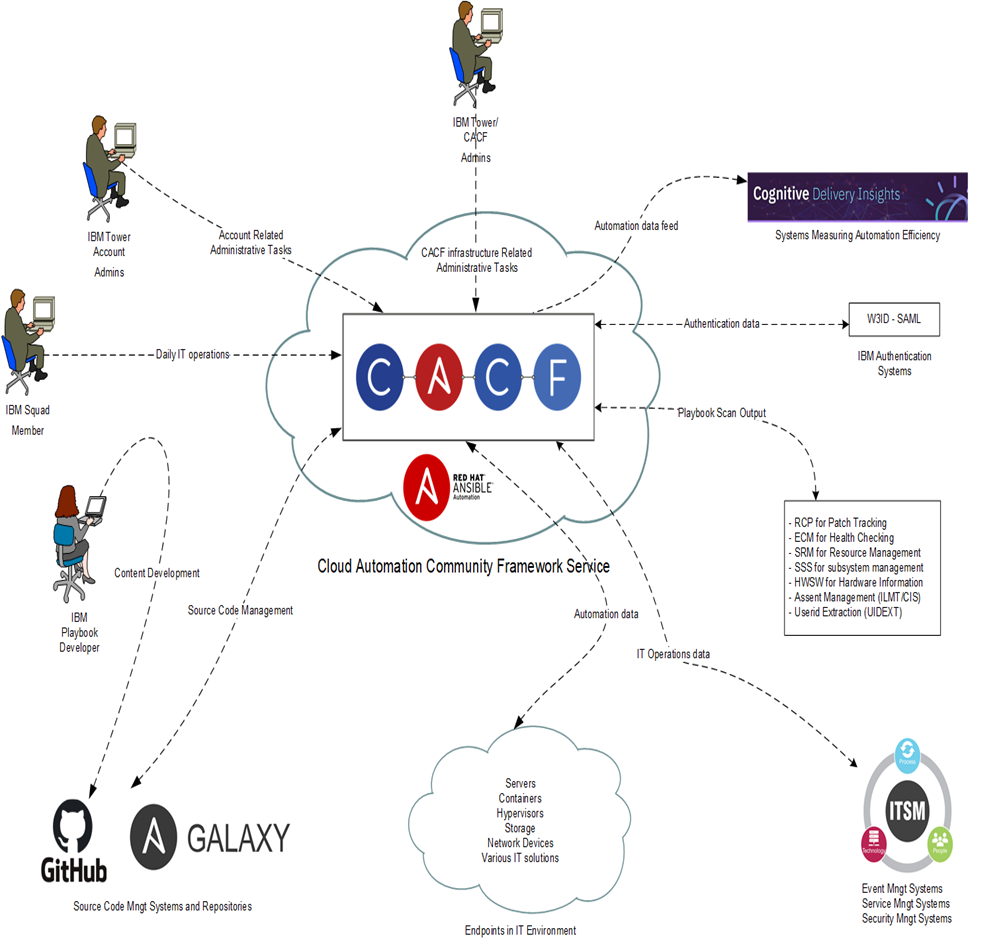


Figure 1: System Context Diagram

## Key Users and Interfaces

* + 1. ***Actors***

Tables 4.2.1.10 to 4.2.1.10 below: Key actors, users and systems.

* + - 1. **Kyndryl Tower / CACF Admins**

|  |  |
| --- | --- |
| **Description** | Kyndryl Tower / CACF Administrators are responsible for ensuring healthy delivery of the framework by performing various administration tasks. |
| **Inputs** | Various |
| **Output** | Various |
| **Type** | Human |
| **Owner** | Kyndryl |

* + - 1. **Kyndryl Tower Account Admins**

|  |  |
| --- | --- |
| **Description** | Kyndryl Tower Account Administrators are responsible to administer the Ansible Tower organization for their account. Add playbooks and provide user administration activities for the account. |
| **Inputs** | Various |
| **Output** | Various |
| **Type** | Human |
| **Owner** | Kyndryl |

* + - 1. **Kyndryl Playbook Developers**

|  |  |
| --- | --- |
| **Description** | Kyndryl Cloud Automation Community Framework Developers are responsible for developing automation contents to address various automation needs. These actors can be from CACF team working on creating standard assets for global usage and/or local account automation developers creating account-specific solutions.  Maintaining and updating existing automation content is also done by these actors. |
| **Inputs** | Automation requests |
| **Output** | Automation content such as Ansible roles/playbooks etc. |
| **Type** | Human |
| **Owner** | Kyndryl |

* + - 1. **Kyndryl Squad Members**

|  |  |
| --- | --- |
| **Description** | This actor represents various roles, including Ansible Tower account admin, account squad members (Windows, Linux, Middleware sysadmins), and Tower account auditors. |
| **Inputs** | Various |
| **Output** | Various |
| **Type** | Human |
| **Owner** | Kyndryl |

* + - 1. **Source Code Management Systems and Repositories**

|  |  |
| --- | --- |
| **Description** | Ansible Playbooks stored and pulled down from Kyndryl GitHub. Developer access and content development through DevOps |
| **Inputs** | Source Code |
| **Output** | Source Code |
| **Type** | System |

* + - 1. **Systems Measuring Automation Efficiency**

|  |  |
| --- | --- |
| **Description** | MCMP AIOps Internal Edition solution will measure automation efficiency and is used by Kyndryl internal support teams. MCMP AIOps Internal Edition is a platform that is connected to a growing number of data source types (e.g., Ansible, Netcool, IPC ticketing system) and data source instances*.* The data from the data sources is regularly pulled by ETLs and stored in the MCMP AIOps Internal Edition Data Lake. MCMP AIOPs Internal Edition will generate reports to be used for measuring automation performance and suggesting improvement areas. |
| **Inputs** | Automation data |
| **Output** | Automation reports |
| **Type** | System |

* + - 1. **Scan Output**

|  |  |
| --- | --- |
| **Description** | Specific playbooks will gather information from the system they run on and bring this data back to the CACF environment. From there it is forwarded to the repositories which require this information. Repository. |
| **Inputs** | Automation data |
| **Output** | Playbook scan results |
| **Type** | System |

* + - 1. **Authentication Systems (Kyndryl)**

|  |  |
| --- | --- |
| **Description** | Kyndryl Single Sign-On utilizing SAML. Kyndryl users will be able to login with their w3 ids. No external user access is available. |
| **Inputs** | Authentication data |
| **Output** | Authentication data |
| **Type** | System |

* + - 1. **IT Service Management Systems**

|  |  |
| --- | --- |
| **Description** | CACF service will be to automate various IT services. Therefore, it requires integrations to the following systems:   * Incident Management – Netcool Event Management Systems * Patch Management – Risk-based Continuous Patching * Security Health Checking – Enterprise Compliance Management * Service Request Automation – IBM Control Desk & ServiceNow * Service Activation & Deactivation – Various Kyndryl systems * Other systems (Asset Mngt, Identity and Access Mngt etc.…)   The systems CACF service need to integrate is also depending on the automation content developed. |
| **Inputs** | Various |
| **Output** | Various |
| **Type** | System |

* + - 1. Endpoints in IT Environment

|  |  |
| --- | --- |
| **Description** | CACF service needs to connect to endpoints (Linux and Windows) in IBM Cloud, though SSH proxies, via SSH or WinRM. |
| **Inputs** | SSH or PowerShell commands |
| **Output** | Command outputs |
| **Type** | System |

## Architecture Overview Diagram

The Ansible (CACF) automation platform is a common, cloud native platform architecture for advanced automation solutions featuring three flagship products from Red Hat – Ansible, Ansible Tower and OpenShift:

1. **Ansible** is an open source automation tool for system configuration, software deployment and orchestration using an agentless architecture which communicates to Linux and Windows endpoints using SSH (Secure Shell) or WinRM (Windows Remote Management) respectively. (For a full list of supported managed endpoints, see this [RedHat Article](https://access.redhat.com/articles/3168091) which does include UNIX variants like AIX). It uses a simple YAML-based syntax for playbook creation. Ansible comes with an extensive library of plugins and modules and has an active developer community continuously contributing new content to the public GitHub.
2. **Ansible Tower** is a web-based user interface on top of Ansible’s command line interface (CLI). Together with Ansible as the automation engine, Ansible Tower offers many features including:

* RBAC – role-based access control via Users, Organizations, Roles, Teams
* Authentication and Multi-tenancy support
* Reporting and Dashboard
* Job and Workflow Templates
* Surveys
* Dynamic Inventory
* Gathering Facts
* Library of existing modules; support for custom module development
* Evidence Collection – integration with ELK, Splunk, Loggly, etc.
* Connection mechanisms (SSH, WinRM/psrp)
* Error handling and retries in playbooks
* Multiple vault options – Tower, Hashicorp, CyberArk, Thycotic
* Batch processing and asynchronous execution
* Ansible Tower zero downtime upgrades

1. **OpenShift Container Platform (OCP)** from Red Hat is a platform-as-a-service built around Docker containers orchestrated and managed by Kubernetes, an open source system for container orchestration that automates application deployment and management. Ansible and Ansible Tower reside on containers within OCP along with the applications.

Together, these Red Hat technologies comprise the software foundation of the CACF automation platform, representing a major shift away from traditional bare metal or virtual machine deployments.

Timeline

Description automatically generated

**Figure 2: RedHat Ansible & Ansible Tower**

### Conceptual Architecture Overview Diagram

The following is a high-level Architecture Overview Diagram) for the service.

Diagram

Description automatically generated

Figure 2: Architecture Overview Diagram

This is a high-level architecture overview diagram for the Ansible service for Ericsson’s IaaS platform in IBM Cloud.

All Kyndryl backend servers like CACF Ansible Tower, Event Management components like Netcool Impact and Ticketing Systems are in M&EaaS Tools POD in Kyndryl Services Tools Platform (KSTP) at Amsterdam.

There are three types of communications between the IaaS platform in IBM Cloud and the M&EaaS Tools POD; 1) Automation Flow, 2) Event Flow, 3) and Binary Distribution Services (BDS) Flow. All communications will be done through the IPSec VPN network over the Internet. The detail of these network communications is discussed in “Network Connectivity” section in this document.

## Architectural Decisions

This section summarizes the architectural decisions that have been made regarding the automation platform.

#### Platform Architectural Decisions – AD-ANS-001: Platform Hosting

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Platform | **Topic** | Platform |
| **Architectural Decision** | Platform Hosting | **ID** | AD-ANS-001 |
| **Issue or Problem Statement** | There are multiple alternatives between private dedicated ands shared service instances for Ansible/CACF in Kyndryl. | | |
| **Assumptions** | No security, legal and/or contractual restrictions to use shared HST services. | | |
| **Motivation** |  | | |
| **Alternatives** | HST has its own shared instance available in CACF Tools POD in KSTP(ISPwW) at Amsterdam . Private dedicated instance. | | |
| **Decision** | HST has its own shared instance available in CACF Tools POD in KSTP (ISPwW) at Amsterdam. | | |
| **Justification** | HST shared instance is preferred due to efficiency of cost and delivery resources There are no security, legal and/or contractual restrictions to board it. | | |
| **Implications** |  | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### Security Architectural Decisions – AD-ANS-002: Endpoint Credentials Management

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Security | **Topic** | Security |
| **Architectural Decision** | Endpoint Credentials Management | **ID** | AD-ANS-002 |
| **Issue or Problem Statement** | How will credentials be managed in Ansible Tower? Credentials are utilized by Tower for authentication when launching Jobs against machines, querying inventory sources, and importing content from GitHub. Multiple options exist for credentials management in Ansible Tower. | | |
| **Assumptions** | * Ansible Tower grants users and teams the ability to use credentials without exposing the credential to the user. * Multiple credential types can be supported on the same Tower instance. | | |
| **Motivation** | Credentials must be securely stored and encrypted. | | |
| **Alternatives** | * Multiple credential types are supported by Ansible Tower: <https://docs.ansible.com/ansible-tower/latest/html/userguide/credentials.html> * The credential types associated with external vaults like CyberArk and HashiCorp Vault are part of the credential plugins capability. This will require Ansible Tower to directly have connectivity with the respective vault. | | |
| **Decision** | Use Ansible internal vault (default). Endpoint credentials will be stored within the Ansible Tower solution itself. | | |
| **Justification** | It is considered that privilege escalation to Windows Administrator and Linux root by sudores is acceptable and compliant with Ericsson’s security standard for non-human interactive function IDs and all the activities are traceable. If this is the case, use of Ansible internal vault is the easiest secure implementation. | | |
| **Implications** | The Ansible Tower machine credentials feature is like how Ansible engine handles credentials on the command line interface.  *Machine credentials enable Tower to invoke Ansible on hosts under your management. You can specify the SSH username, optionally provide a password, an SSH key, a key password, or even have Tower prompt the user for their password at deployment time. They define SSH and user-level privilege escalation access for playbooks and are used when submitting jobs to run playbooks on a remote host.*  A process will need to be defined for Ansible Tower credentials management. | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### Infrastructure Related Architectural Decisions – AD-ANS-003: SSH Proxy Placement

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Account Infrastructure Related | **Topic** | Infrastructure |
| **Architectural Decision** | SSH Proxy Placement | **ID** | AD-ANS-003 |
| **Issue or Problem Statement** | Where should SSH Proxy servers be hosted physically and logically? | | |
| **Assumptions** | Firewall ports can be opened in the customer firewall (NSX Edge FW) to allow communication between SSH Proxies and endpoints on the customer network | | |
| **Motivation** | Minimize required SSH Proxies and minimize the need for additional network configurations | | |
| **Alternatives** | 1. Customer premises – Overlay network 2. Kyndryl premises – Underlay network | | |
| **Decision** | SSH Proxy servers will be hosted at customer premises | | |
| **Justification** | Ericsson’s security principles require that Kyndryl tooling servers are placed in Ericsson’s network (overlay), There is a dedicated security zone, machine 2 machine for tooling purposes. The IaaS platform has existing relay servers for BigFix. These servers can be repurposed as Ansible proxies. | | |
| **Implications** | At this moment there is no known limitation in number of endpoints which can be served by an SSH proxy server. The amount of proxy server’s requirement is more directly by the network layout of the client.. | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### AD-ANS-004 SSH Proxy Availability

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Account Infrastructure Related | **Topic** | Infrastructure |
| **Architectural Decision** | SSH Proxy Availability | **ID** | AD-ANS-004 |
| **Issue or Problem Statement** | Do we need High Availability for SSH Proxy? | | |
| **Assumptions** |  | | |
| **Motivation** | SSH Proxy servers need HA config to decrease the offline duration in case of an outage. | | |
| **Alternatives** | 1. Only Standalone SSH Proxy 2. Have Primary and Secondary SSH Proxies | | |
| **Decision** | Have Primary and Secondary SSH Proxies (within each network zone) to reach the managed endpoints. | | |
| **Justification** | Ericsson runs mission-critical workloads in IBM Cloud. The existing relay servers for BigFix are paired for HA and can be repurposed Ansible. | | |
| **Implications** |  | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### AD—ANS-005: Native Patching Tools

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Account Infrastructure Related | **Topic** | Infrastructure |
| **Architectural Decision** | Native Patching tools | **ID** | AD—ANS-005 |
| **Issue or Problem Statement** | Which native patching tools should be used; Ericsson owned, or Kyndryl owned? | | |
| **Assumptions** | Client allows to connect to their native patching tools (Red Hat Satellite, Microsoft WSUS) | | |
| **Motivation** | The client needs to be license compliant. | | |
| **Alternatives** | 1. Use Ericsson owned native patching tools 2. Use Kyndryl provided native patching tools 3. Use a combination, Ericsson owned patching tool for Linux servers. And Kyndryl provided native patching tool for Microsoft Windows. | | |
| **Decision** | Use a combination of Ericsson and IBM owned native patching tools | | |
| **Justification** | OS license of the endpoints are owned by Ericsson.  RedHat Enterprise Linux the servers in the overlay will be patched via Ericsson Satellite/Capsule servers. This as the Satellite servers are using Ericsson subscriptions.  Microsoft Windows Servers and desktops will use Ericsson’s WSUS for license activation. Patching in the overlay will be done via Kyndryl provided native patching tool, WSUS. | | |
| **Implications** |  | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### AD-ANS-006: WSUS Master and child server placement for Windows patching

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | Account Infrastructure Related | **Topic** | Infrastructure |
| **Architectural Decision** | WSUS Master and child server placement for Windows servers desktop patching | **ID** | AD—ANS-006 |
| **Issue or Problem Statement** | Location to place the WSUS up-stream and down-stream servers to download the Microsoft patches from vendor to distribute to the managed endpoints for patching | | |
| **Assumptions** | Client will not allow to download the patches at overlay. Which required open to internet connectivity from WSUS server. | | |
| **Motivation** | The client needs to be license compliant. | | |
| **Alternatives** | 1. Place the WSUS master at underlay *(i.e. it is being deployed as part of RFS)* and WSUS child server at each data center (FRA,SYD,CHE01 & WDC). The window patches will be downloaded at underlay WSUS master server and patches will be pushed to Overlay WSUS child servers to distribute on target endpoints. 2. Place the WSUS master at overlay and enable the internet connection to Microsoft sites to download the packages | | |
| **Decision** | Option 1: Place the WSUS master at underlay *(i.e. it is being deployed as part of RFS)* and WSUS child server at each data center (FRA,SYD,CHE01 & WDC). The window patches will be downloaded at underlay WSUS master server and patches will be pushed to Overlay WSUS child servers to distribute on target endpoints. | | |
| **Justification** | OS license of the endpoints are owned by Ericsson.  Microsoft Windows Servers and desktops will use Ericsson’s WSUS for license activation. Patching in the overlay will be done via Kyndryl provided native patching tool, WSUS.  WSUS master server at underlay will have the internet connection to download the patches from vendor site. | | |
| **Implications** |  | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

#### AD-ANS-007: WSUS Master placement at Underlay

|  |  |  |  |
| --- | --- | --- | --- |
| **Subject Area** | WSUS master and child at underlay and overlay | **Topic** | Infrastructure |
| **Architectural Decision** | Placement of WSUS master and child servers | **ID** | AD-ANS-007 |
| **Issue or Problem Statement** | Location to place the WSUS up-stream and down-stream servers to download the Microsoft patches from vendor to distribute to the managed endpoints for patching | | |
| **Assumptions** | Client will not allow to download the patches at overlay. Which required open to internet connectivity from WSUS server. | | |
| **Motivation** | The client needs to be license compliant. | | |
| **Alternatives** | 1. Place the WSUS masters one per DC at underlay *( i.e., One WSUS master at each DC* FRA,SYD,CHE01 & WDC). The window patches will be downloaded at underlay WSUS master servers and patches will be pushed to Overlay WSUS child servers to distribute on target endpoints (windows servers and desktops). 2. Place the one WSUS master at FRA DC at Underlay and push the patches to all WSUS child’s at each DC (FRA,SYD,CHE01&WDC) and enable the internet connection to Microsoft sites to download the packages | | |
| **Decision** | Option 2: Place the one WSUS master at FRA DC at Underlay and push the patches to all WSUS child’s at each DC (FRA.SYD.CHE01&WDC) and enable the internet connection to Microsoft sites to download the packages | | |
| **Justification** | * Microsoft Windows Servers and desktops will use Ericsson’s KMS for license activation. Patching in the overlay will be done via Kyndryl provided native patching tool, WSUS. * WSUS master server at underlay will have the internet connection to download the patches from vendor site. | | |
| **Implications** | * WSUS master doesn’t have HA (High availability) capability. If any reason WSUS Master get failed. The last known working backup will be restored. * WSUS Child should be deployed at overlay and should communicate to master to enable the services | | |
| **Derived requirements** |  | | |
| **Related Decisions** |  | | |

## Conceptual Operational Model

Not applicable.

## Product and Delivery Maturity

Ansible is part of the Red Hat software portfolio for software provisioning, configuration management, and application-deployment tool enabling infrastructure as code. Kyndryl are globally replacing IBM BigFix with Ansible and have incorporated Ansible as a core tool in their automation.

# Operational Design

## Specified Operational Model

Following Kyndryl and Ericsson standard operational model.

## Application Design

N/A

## Prerequisites

The following information outline the pre-requisites for an Ansible-based solution to perform automation on the managed endpoints.

### Prerequisites for Managed Endpoints

#### Linux

* + Python must be installed.
    - For version 2, 2.6 or higher.
    - For version 3, 3.5 or higher.
  + Port 22 (SSH access) must be open from SSH Proxies.
  + Must be able to reach SSH Proxies (BDS flow) on port 8081 (HTTPS)
  + SUDO must be configured and enabled for the automation users.
  + Recommended (not mandatory) to have a separate 5 GB filesystem for /var/opt/ansible.
    - Add 10 GB when optional CDS service is used.
  + **Credentials:**
    - Create “playbook” group on each endpoint
    - Create one privileged ID and add to “playbook” group on each endpoint
    - Install public SSH key on each endpoint
    - For GCM Scans, credentials used need to be able to become ‘root’.

#### Windows

* + Windows 2012 or higher versions are supported.
    - Windows Server 2008 is supported by Ansible Tower but is out of Windows support and should be replaced.
  + Servers must have PowerShell 3.x or higher.
  + PowerShell memory must be configured at least 512 MB.
  + PSRemote must be enabled.
  + Servers must have either port 5985 and 5986 (HTTPS) opened from SSH Proxies.
  + Must be able to reach SSH Proxies (BDS flow) on port 8081 (HTTPS)
  + Systems must have .NET 4.x or higher.
  + **Credentials:**
    - This ID is deployed through Active Directory (GPO)
    - This ID is add to local “Administrator” group
    - For GCM Scans, credentials used need to be able to become ‘system’.

*Note: Additional detail on supported platform and versions can be found here:* [*https://access.redhat.com/articles/3168091*](https://access.redhat.com/articles/3168091)

### Prerequisites for Proxy servers

SSH proxies will be used to bridge between the Ansible platform and the managed endpoints, so the following additional requirements apply.

* RHEL is recommended to be used as SSH Proxy (Version 7 or higher is preferred).   
  *Note: Windows is not supported to be used as SSH Proxy.*
* SSH connectivity between Ansible tower and SSH proxy servers.
* Minimum 2 CPUs and 4 GB of memory
  + Add 2 CPUs and 4 GB memory for Health Check, Patch Check automation
* Minimum 50 GB of free disk space.
  + Add 25 GB disk space for Health Check, Patch Check automation and UIDEXT for BDS flow – content replication.
  + Add 250 GB disk space for operational (software distribution) service.
* BDS flow – content replication
  + SSH Proxies must be able to reach Master Binary Repository (BDS flow) on port 8081 (HTTPS).
  + Must have OpenJDK 1.8+, rsync, libsemanage-python and libselinux-python installed:
    - sudo yum install java-1.8.0-openjdk rsync libsemanage-python libselinux-python
  + Playbook to install and configure Binary Repository software (Nexus) is available.
* **Credentials:**
  + One (1) ID, **ID**, on each SSH Proxy (non-privileged) will be used.

## User Authentication

* **New credentials for Ansible automation will be created**
  + One (1) local ID for each Ansible Tower Proxy server (Primary and Backup)
    - **ID**
  + One (1) local ID for each Client Proxy server (Primary and Backup) - Applicable for only type B customer
    - **ID**
  + One (1) System Administrator like ID for each Linux endpoint (***SSH key*** authentication, plus SUDO template)
    - **< 3digit of account id> <ipat1>**
  + One (1) ID, Active Directory (preferred) or Local in the local administrator group, for each Windows endpoint server (***password-based*** authentication)
    - **system\_< 3digit of account id> <ipat1>**
* **UID parameters:**
  + Password length: minimum 15 characters
  + Password expiration: maximum 90 days /As per customer Policy
  + Interactive login validation
  + IP Whitelisting on target system: only Jumphost / proxy IP address

## Network Model

HST Ansible Tower instances are deployed in Kyndryl Tools POD in KSTP (ISPwW) in the Amsterdam. This zone is fully owned and managed by Kyndryl HST and not accessible from client networks directly.

The integration to the services hosted on HST Tools POD from the client located in IBM Cloud is established via Ipsec tunnel, enabling connectivity between client network and services hosted on Kyndryl Tools POD. This is the new network between KSTP (ISPwW) to Underlay and overlay environments.

The sections describe the two (2) types of data flows, Ansible and Patch management.

### Ansible

Ansible Tower to Proxy connectivity use case are initiation of health- and security runbooks. The existing IEM (BigFix) relay servers in the machine to machine zone in the overlay will be used as Ansible Proxy servers.

Diagram

Description automatically generated

**Figure 3: Ansible connectivity diagram**

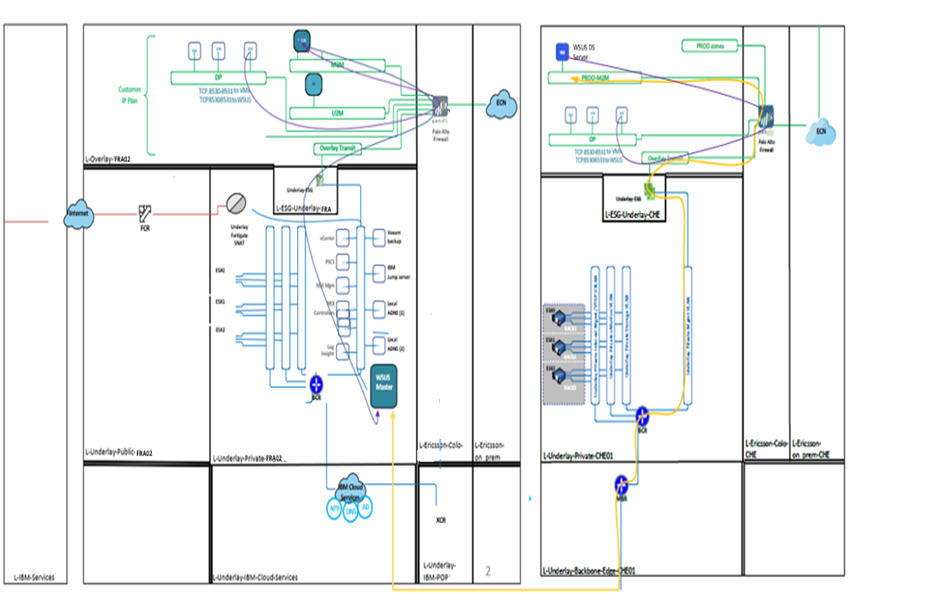
The new IPsec VPN tunnel will be used to connect between Kyndryl Tools POD (KSTP) and the IBM Cloud. For, Kyndryl shared FortiGate to terminate the IPsec VPN tunnel in Kyndryl Tools POD (KSTP). For this type of connectivity, Ansible Tower will reach out directly to the DRES Jump host located in the same zone, Kyndryl Tools POD. Next hop will be SSH Proxy, which is the existing MCMS proxy server located in the overlay network, machine 2 machine zone in IBM Cloud.

Specific playbooks/solutions build on Ansible require large binary files and packages to be transported to the endpoints in the client network. To avoid transporting this package over the network for every execution of the Ansible playbook, a Binary Distribution Service (BDS) is provided to replicate this information only when it is updated.

In the KSTP location a Master Binary Repository is made available from where the Proxies in the overlay, machine 2 machine zone, retrieve the binary files and packages from. The Ansible Proxies contain the Client Binary Repository where the endpoints will retrieve the binary files and packages from.

### Patching – Servers and Desktop

Kyndryl will patch the operating systems via the vendor native tools such as Microsoft Windows Update Services and RHEL Satellite servers. Existing Kyndryl’s Repo servers in M2M will be used; one for Prod / UAT and one for DEV and used for patching of OS and databases in the overlay.



**Figure 5: Patching connectivity diagram**

As depicted in above diagram,

* WSUS Upstream (Master) server will be placed in IBM underlay in FRA and will use IBM Squid proxy to download patches from Microsoft site using Port 443 and download files using Port 80.
* Centralized WSUS Upstream (Master) server will be placed in underlay in FRA having connectivity to all WSUS downstream (Child) dedicated server in M2M zone in all DC’s (i.e., in FRA , CHE, SYD and WDC).
* WSUS downstream will be placed in M2M zone in overlay in all DCs ( FRA, CHE, SYD and WDC) and will connect to WSUS Master server in FRA via Palo Alto through underlay ESG.
* Connectivity from WSUS upstream server to downstream server and WSUS upstream server to desktop client & servers is via Palo Alto.
* Network flow is bidirectional between WSUS upstream & downstream server and WSUS downstream server to clients.
* WSUS setup will be implemented in WDC-PROD, WDC-PROD, SYD-PROD & FRA-PROD & DEV.
* Additional WSUS downstream servers will be required for DEV and PROD/UAT environments.
* We need help from AD team in E// to enable Group Policy for the clients
* Overlay downstream server to underlay Master server communication is through the SNAT in the respective location underlay NSX Edge.
* Managed End Points – Windows servers & Desktops.

#### Patching of RedHat Enterprise Linux Server

* RHEL Servers in the overlay will be patched via Ericsson Satellite/Capsule servers.
* Satellite servers are using Ericsson subscriptions.

#### Patching of Windows Servers Server

* Windows Servers in the overlay will be patched via Kyndryl WSUS in overlay M2M which will receive the patches from the dedicated upstream WSUS server in the underlay that will receive patches from Microsoft.
* Ericsson’s WSUS will be used for license activation.

#### Patching of Desktops

Desktop operating systems have a dedicated patching infrastructure. Hence Kyndryl will re-use the existing WSUS master placed in underlay. Which act as upstream server. The required patches will be pushed to overlay WSUS same downstream servers to patch the target windows desktop endpoints.

### Patching – Citrix

#### Citrix Infrastructure

Citrix infrastructure servers will follow the standard patch process as for infrastructure and application servers using Ansible and Microsoft WSUS.

#### Citrix XenApp/XenDesktop

##### PVS Provisioned XenApp/XenDesktop

PVS provisioned images are identical, hence no need to perform patch management on the respective XenApp/XenDesktop image.

All installations and patching are done in one place and then streamed to the target, XenApp/XenDesktop, when they boot-up. Build consistency is assured because all your servers are using a shared copy of the disk image, a master image.

Provisioning uses a master image within your environment to manage virtual machines, enabling you to manage and update target devices through one master image. This is fully integrated and administrated in Citrix Studio and vCenter

When running provisioning you create a master image, which will be used for provisioning several identical server images.

Advantages of using provisioning

* All software and patches are applied to a master image. Provisioning ensures all images are identical every time we decide to apply a new image version, making it fast and secure to de-ploy updates.

Manual/Semi-automated patching will be performed on master images provisioned with the Citrix services PVS. All provisioned servers and desktops will be updated next time they restart. No Kyndryl tools will be installed on master images as agreed in the reference architecture.

All provisioned servers are reset to master image at every server restart.

##### Manual Provisioned XenApp/XenDesktop

Manually deployed referred to as (vanilla VDA) XenApp/XenDesktop (VDA) servers will have a full tool stack like an application server. These servers will follow the standard patch process as for infrastructure and application servers.

For additional information regarding Citrix and patching of the same, see Citrix XenApp/XenDesktop Solution Design Document [4].

### IBM Support Connections

No change depending on Ansible, see the ‘Ericsson - Network Design - Solution Architecture Report’

### Data Centre LAN Design

No change depending on Ansible, see the ‘Ericsson - Network Design - Solution Architecture Report’

### Firewall Rules

The table below shows the connections between application components across servers. The PaloAlto firewalls in Ericsson’s colocation should be configured according to this table.

#### Firewall Rules – Ansible and Server-Patching

##### Frankfurt Development

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Source details | Destination details | Port | Protocol | Notes | Zone |
| 1- FRA Dev DC | Kyndryl Linux jump ( NAT IP) | IEM Linux /Ansible FRA Dev over lay Jump Customer | Port 22 | TCP- Unidirectional | Linux proxy connectivity | Kyndryl Underlay - FRA Overlay |
| 2- FRA Dev DC | WSUS master server at underlay(NAT IP) | FRA DEV WSUS child server at overlay  (windows) | Port 8530/8531 | TCP - Bidirectional | WSUS Master server at FRA underlay to ER WSUS child at overlay | Kyndryl Underlay- FRA Dev Overlay |
| 3- FRA Dev DC | FRA Dev WSUS child server at overlay  (windows) | All Windows FRA Dev End points | Port 8530/8531 | TCP - Bidirectional | WSUS servers for Windows server patching | FRA Dev- M2M-FRA Dev |
| 4- FRA Dev DC | IEM Linux /Ansible over lay Jump Customer Jump | FRA Dev Windows End points | Port 5985/5986 | TCP -Unidirectional | ER ansible proxy to Windows EP | FRA Dev-M2M-FRA Dev |
| 5- FRA Dev DC | IEM Linux /Ansible over lay Jump Customer Jump | FRA Dev Linux Endpoints | Port 22 | TCP- Unidirectional | ER ansible proxy to Linux EP | FRA Dev -M2M-FRA Dev |
| 6- FRA Dev DC | Linux Endpoints | FRA Dev satellite Server | Port 443/3128 | TCP- Unidirectional | ER Linux server to ER Linux Satellite | Ericsson Satellite Server |
| 7- FRA Dev DC | Linux Endpoints | IEM Linux /Ansible over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl FRA Dev-M2M |
| 8- FRA Dev DC | Windows Endpoints | IEM Linux /Ansible over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl FRA Dev-M2M |
| 9 - FRA Dev DC | IEM Linux /Ansible over lay Jump Customer Jump | Kyndryl Linux jump ( NAT IP) | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl Underlay -M2M - Kyndryl FRA Dev |

Table: Firewall configuration – Frankfurt, Development

##### Frankfurt Production

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Source details | Destination details | Port | Protocol | Notes | Zone |
| 1-FRA PROD DC | Kyndryl Linux jump ( NAT IP) | IEM Linux /Ansible FRA Prod over lay Jump Customer | Port 22 | TCP- Unidirectional | Linux proxy connectivity | Kyndryl Underlay - FRA Prod Overlay |
| 2-FRA PROD DC | WSUS master server at underlay( NAT IP) | FRA Prod WSUS child server at overlay  (windows) | Port 8530/8531 | TCP - Bidirectional | WSUS Master server at FRA underlay to ER WSUS child at overlay | Kyndryl Underlay- FRA Prod Overlay |
| 3-FRA PROD DC | FRA Prod WSUS child server at overlay  (windows) | All Windows FRA Prod End points | Port 8530/8531 | TCP - Bidirectional | WSUS servers for FRA Prod Windows server patching | FRA Prod - M2M-FRA Prod |
| 4-FRA PROD DC | IEM Linux /Ansible overlay Jump - Customer | All Windows FRA Prod End points | Port 5985/5986 | TCP -Unidirectional | ER ansible proxy to Windows FRA Prod EP's | FRA Prod-M2M-FRA Prod |
| 5-FRA PROD DC | IEM Linux /Ansible overlay Jump- Customer | All Linux FRA Prod Endpoints | Port 22 | TCP- Unidirectional | ER ansible proxy to FRA Prod Linux EP's | FRA Prod -M2M-FRA Prod |
| 6-FRA PROD DC | Linux Endpoints | FRA Prod Satellite Server | Port 443/3128 | TCP- Unidirectional | ER Linux server to ER Linux Satellite | Ericsson Satellite Server |
| 7-FRA PROD DC | Linux Endpoints | IEM Linux /Ansible FRA Prod over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl FRA PROD-M2M |
| 8-FRA PROD DC | Windows Endpoints | IEM Linux /Ansible FRA Prod over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl FRA Prod-M2M |
| 9-FRA PROD DC | IEM Linux /Ansible over lay Jump- Customer | Kyndryl Linux jump ( NAT IP) | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl Underlay -M2M - Kyndryl FRA Prod |

Table: Firewall configuration – Frankfurt, Production

##### **Chennai**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Source details | Destination details | Port | Protocol | Notes | Zone |
| 1-CHE DC | Kyndryl Linux jump ( NAT IP) | IEM Linux /Ansible CHE over lay Jump Customer | Port 22 | TCP- Unidirectional | Linux proxy connectivity | Kyndryl Underlay - Che Overlay |
| 2-CHE DC | WSUS master server at underlay( NAT IP) | CHE WSUS child server at overlay  (windows) | Port 8530/8531 | TCP - Bidirectional | WSUS Master server at FRA underlay to ER WSUS child at overlay | Kyndryl Underlay- CHE Overlay |
| 3-CHE DC | CHE WSUS child server at overlay  (windows) | All Windows CHE End points | Port 8530/8531 | TCP – Bidirectional | WSUS servers for CHE Windows server patching | CHE - M2M- CHE |
| 4-CHE DC | IEM Linux /Ansible over lay Jump- Customer Jump | All Windows CHE End points | Port 5985/5986 | TCP -Unidirectional | ER ansible proxy to Windows CHE EP’s | CHE -M2M- CHE |
| 5-CHE DC | IEM Linux /Ansible over lay Jump- Customer Jump | All Linux CHE Endpoints | Port 22 | TCP- Unidirectional | ER ansible proxy to CHE Linux EP’s | CHE -M2M-CHE |
| 6-CHE DC | Linux Endpoints | CHE satellite Server | Port 443/3128 | TCP- Unidirectional | ER Linux server to ER Linux Satellite | Ericsson Satellite Server |
| 7-CHE DC | Linux Endpoints | IEM Linux /Ansible CHE over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | CHE-M2M |
| 8-CHE DC | Windows Endpoints | IEM Linux /Ansible CHE Overlay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | CHE-M2M |
| 9-CHE DC | IEM Linux /Ansible over lay Jump- Customer Jump | KyndrylLinux jump ( NAT IP) | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | Kyndryl Underlay -M2M - CHE |

Table: Firewall configuration – Chennai

##### **Sydney**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Source details | Destination details | Port | Protocol | Notes | Zone |
| 1- SYD DC | Kyndryl Linux jump ( NAT IP) | IEM Linux /Ansible SYD overlay Jump Customer | Port 22 | TCP- Unidirectional | Linux proxy connectivity | IBM Underlay - SYD Overlay |
| 2- SYD DC | WSUS master server at underlay( NAT IP) | SYD WSUS child server at overlay  (windows) | Port 8530/8531 | TCP - Bidirectional | WSUS Master server at FRA underlay to ER WSUS child at overlay | IBM Underlay- SYD Overlay |
| 3- SYD DC | SYD WSUS child server at overlay  (windows) | All Windows SYD End points | Port 8530/8531 | TCP - Bidirectional | WSUS servers for SYD Windows server patching | SYD - M2M- SYD |
| 4- SYD DC | IEM Linux /Ansible over lay Jump- Customer | All Windows SYD End points | Port 5985/5986 | TCP -Unidirectional | ER ansible proxy to Windows SYD EP's | SYD -M2M- SYD |
| 5- SYD DC | IEM Linux /Ansible over lay Jump- Customer | All Linux SYD Endpoints | Port 22 | TCP- Unidirectional | ER ansible proxy to SYD Linux EP's | SYD -M2M- SYD |
| 6- SYD DC | Linux Endpoints | SYD satellite Server | Port 443/3128 | TCP- Unidirectional | ER Linux server to ER Linux Satellite | Ericsson Satellite Server |
| 7- SYD DC | Linux Endpoints | IEM Linux /Ansible SYD over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | SYD-M2M |
| 8- SYD DC | Windows Endpoints | IEM Linux /Ansible SYD Overlay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | SYD M2M |
| 9- SYD DC | IEM Linux /Ansible over lay Jump | Kyndryl Linux jump ( NAT IP) | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | IBM Underlay -M2M - SYD |

Table: Firewall configuration – Sydney

##### **Washington**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.No | Source details | Destination details | Port | Protocol | Notes | Zone |
| 1- WDC DC | Kyndryl Linux jump ( NAT IP) | IEM Linux /Ansible WDC over lay Jump Customer | Port 22 | TCP- Unidirectional | Linux proxy connectivity | IBM Underlay - WDC Overlay |
| 2- WDC DC | WSUS master server at underlay( NAT IP) | WDC WSUS child server at overlay  (windows) | Port 8530/8531 | TCP - Bidirectional | WSUS Master server at FRA underlay to ER WSUS child at overlay | IBM Underlay- WDC Overlay |
| 3- WDC DC | WDC WSUS child server at overlay  (windows) | All Windows FRA Prod Endpoints | Port 8530/8531 | TCP - Bidirectional | WSUS servers for WDC Windows server patching | WDC - M2M- WDC |
| 4- WDC DC | IEM Linux /Ansible over lay Jump | All Windows WDC Endpoints | Port 5985/5986 | TCP -Unidirectional | ER ansible proxy to Windows WDC EP's | WDC -M2M- WDC |
| 5- WDC DC | IEM Linux /Ansible over lay Jump | All Linux WDC Endpoints | Port 22 | TCP- Unidirectional | ER ansible proxy to WDC Linux EP's | WDC -M2M-WDC |
| 6- WDC DC | Linux Endpoints | WDC satellite Server | Port 443/3128 | TCP- Unidirectional | ER Linux server to ER Linux Satellite | Ericsson Satellite Server |
| 7- WDC DC | Linux Endpoints | IEM Linux /Ansible WDC over lay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | WDC-M2M |
| 8- WDC DC | Windows Endpoints | IEM Linux /Ansible WDC Overlay Jump | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | WDC-M2M |
| 9- WDC DC | IEM Linux /Ansible over lay Jump | Kyndryl Linux jump ( NAT IP) | Port 8081 | TCP -Unidirectional | For BDS (Binary repository) | IBM Underlay -M2M - WDC |

Table: Firewall configuration – Washington

#### Firewall Rules - Desktop-Patching

##### Frankfurt

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Flow id** | **Description** | **Flow Details** | **Source Zone** | **Destination Zone** | **Application (L7 FW)** | **Protocol & Port (L4 FW)** | **Port Opening Required at** |
| WSUS-Flow-1 | WSUS Underlay – WSUS Overlay | WSUS Child IP NATed in underlay ESG to reach from WSUS Master to WSUS Child IP in M2M | IBM Underlay FRA | FRA Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-2 | WSUS Overlay – WSUS Underlay | WSUS-M2M Child IP NATed in Underlay ESG to reach WSUS Mater IP in underlay | FRA Prod/UAT M2M | IBM Underlay FRA | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-3 | WSUS Underlay – WSUS Overlay | WSUS Child IP NATed in underlay ESG to reach from WSUS Master to WSUS Child IP in M2M | IBM Underlay FRA | FRA DEV M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-4 | WSUS Overlay – WSUS Underlay | WSUS-M2M Child IP NATed in Underlay ESG to reach WSUS Mater IP in underlay | FRA DEV M2M | IBM Underlay FRA | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-5 | Managed VMs to WSUS – M2M | Managed VMs to WSUS-M2M PROD | [Managed\_to\_VMs\_Subnet] FRA-PROD/UAT Zone | FRA Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-6 | IBM Overlay WSUS-M2M to Managed VMs | IBM Overlay WSUS-M2M PROD to Managed VMs | FRA Prod/UAT M2M | [Managed\_to\_VMs\_Subnet] FRA-PROD/UAT Zone | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-7 | Managed VMs to WSUS – M2M | Managed VMs to WSUS-M2M DEV | [Managed\_to\_VMs\_Subnet] – FRA DEV Zone | FRA DEV M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-8 | IBM Overlay WSUS-M2M to Managed VMs | IBM Overlay WSUS-M2M DEV to Managed VMs | FRA DEV M2M | [Managed\_to\_VMs\_Subnet] – FRA DEV Zone | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-9 | IBM Underlay WSUS Mater in FRA - to SQUID Proxy | IBM underlay WSUS Master – SQUID Proxy in FRA | IBM Underlay FRA | FRA SQUID Proxy | HTTP, HTTPS | TCP 80, 443 | Vyatta |

##### Chennai

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Flow id** | **Description** | **Flow Details** | **Source Zone** | **Destination Zone** | **Application (L7 FW)** | **Protocol & Port (L4 FW)** | **Port Opening required at** |
| WSUS-Flow-1 | WSUS Underlay in FRA – WSUS Overlay in CHE | WSUS Child IP NATed in underlay ESG in CHE to reach from WSUS Master to WSUS Child IP in M2M | IBM Underlay FRA | CHE Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-2 | WSUS Overlay in CHE – WSUS Underlay in FRA | WSUS-M2M Child IP NATed in Underlay ESG to reach WSUS Mater IP in underlay | CHE Prod/UAT M2M | IBM Underlay FRA | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-3 | Managed VMs to WSUS – M2M | Managed VMs to WSUS-M2M | [Managed\_to\_VMs\_Subnet] CHE PROD Zone | CHE Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |

##### Washington

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Flow id** | **Description** | **Flow Details** | **Source Zone** | **Destination Zone** | | **Application (L7 FW)** | **Protocol & Port (L4 FW)** | **Port Opening required at** |
| WSUS-Flow-1 | WSUS Underlay in FRA – WSUS Overlay in WDC | WSUS Child IP NATed in underlay ESG in WDC to reach from WSUS Master to WSUS Child IP in M2M | IBM Underlay FRA | WDC Prod/UAT  M2M | WSUS | | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-2 | WSUS Overlay in WDC – WSUS Underlay in FRA | WSUS-M2M Child IP NATed in Underlay ESG to reach WSUS Mater IP in underlay | WDC Prod/UAT M2M | IBM Underlay FRA | WSUS | | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-3 | Managed VMs to WSUS – M2M | Managed VMs to WSUS-M2M | [Managed\_to\_VMs\_Subnet] WDC PROD Zone | WDC Prod/UAT M2M | WSUS | | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-4 | IBM Overlay WSUS-M2M to Managed VMs | IBM Overlay WSUS-M2M to Managed VMs | WDC Prod/UAT M2M | [Managed\_to\_VMs\_Subnet] WDC -PROD Zone | WSUS | | TCP 8530-8531 | Palo Alto |

##### Sydney

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Flow id** | **Description** | **Flow Details** | **Source Zone** | **Destination Zone** | **Application (L7 FW)** | **Protocol & Port (L4 FW)** | **Port Opening required at** |
| WSUS-Flow-1 | WSUS Underlay in FRA – WSUS Overlay in SYD | WSUS Child IP NATed in underlay ESG in SYD to reach from WSUS Master to WSUS Child IP in M2M | IBM Underlay FRA | SYD Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-2 | WSUS Overlay in SYD – WSUS Underlay in FRA | WSUS-M2M Child IP NATed in Underlay ESG to reach WSUS Mater IP in underlay | SYD Prod/UAT M2M | IBM Underlay FRA | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-3 | Managed VMs to WSUS – M2M | Managed VMs to WSUS-M2M | [Managed\_to\_VMs\_Subnet] SYD PROD Zone | SYD Prod/UAT M2M | WSUS | TCP 8530-8531 | Palo Alto |
| WSUS-Flow-4 | IBM Overlay WSUS-M2M to Managed VMs | IBM Overlay WSUS-M2M to Managed VMs | SYD Prod/UAT M2M | [Managed\_to\_VMs\_Subnet] SYD PROD Zone | WSUS | TCP 8530-8531 | Palo Alto |

## Environments Model

Ansible is deployed in all environments in all four locations (FRA, CHE, WDC & SYD).

## System Management Model

Standard managed service model follows SLAs.

## Performance

The Ansible environment is provided as a service and HST will add resources when needed in the backend environment.

In the client environment, the account will provide additional Ansible proxy servers based on Ericsson’s baselines. The expected performance impact be the respective runbook executed on Ericsson’s endpoint will be equal or less than the current BigFix implementation.

## Support Model

Following Kyndryl and Ericsson standard support model.

## Security Architecture 4.2.1

This section provides an overview into how CACF will handle user authentication and authorization, and the compliance controls in place.

### Authentication to Ansible Tower

Account users will access Ansible Tower’s UI to perform tasks related to automation enablement (i.e., create job templates, run or schedule playbooks). They will not have access to the backend infrastructure.

Authentication is configured using Kyndryl w3ID or account LDAP. Account users will be provided with a URL for the Ansible Tower UI and prompted for their credentials as shown below:

Graphical user interface, text, application, email

Description automatically generated

**Figure 22: Red Hat Ansible Tower login prompt**

### Authorization via Ansible Tower RBAC

Ansible Tower has a built in Role-based Access Control (RBAC) model which allows an account to have very fine-grained control over who can manage and execute automation jobs.

Each account will be represented as an “organization” within Ansible Tower.

Diagram

Description automatically generated

**Figure 23: Red Hat Ansible Tower organization structure**

Each account will assign a “Tower Account Administrator” who acts as the Ansible Tower organization admin. This person will have full root-level control over all Tower objects (inventories, teams, projects, jobs, credentials) within that Ansible Tower organization only. They can view, update or execute jobs in the Ansible Tower UI.

The team, which supports the CACF instance, owns the “Tower System Administrator” role, which is akin to a superuser, and have root-level privileges across all Ansible Tower organizations in the instance

The diagram below illustrates the responsibilities of the “Tower System Administrator” role versus the “Tower Account Administrator” role:

Diagram

Description automatically generated

**Figure 24: “Tower System Administrator” and “Tower Account Administrator” responsibilities.**

## Managed device Credentials Management

Machine credentials (i.e., functional ID, SSH private key) for accessing the managed devices are stored and encrypted in the Ansible Tower internal vault. Access to the credentials is controlled within an account’s organization by the Ansible Tower RBAC.

Credentials are encrypted before they are saved to the database. Once saved, they cannot be retrieved in clear text on the Ansible Tower user interface.

The credentials are decrypted as needed, at automation run time. Graphical user interface, application

Description automatically generated

**Figure 25: Managed device Credentials Management**

### Credentials Encryption on Ansible Tower

For CACF users who authenticate via LDAP, Ansible Tower does not store any password or secrets.

For the encryption of secrets used for automation or are a result of automation – e.g., machine credentials with passwords, keys, authentication tokens – Ansible Tower uses AES in CBC mode with a 256-bit key for encryption, PKCS7 padding, and HMAC using SHA256. The encryption/decryption process derives the AES-256-bit encryption key from the SECRET\_KEY and environment variables.

### Using External Vaults for Managed Device Credentials

Not used in the Ericsson solution.

CACF team developed a role that will help you integrate with either a Vault system that's not supported by a default deployment of Ansible Tower, or any Vault system when you don't have direct connectivity from the Tower to it (e.g.: A vault system that's hosted in customer premises and cannot be reached from outside their network). It uses the Client Jumphost to connect to any legacy, 3rd party or unsupported system and it has multiple options to leverage the retrieved user/password from the Vault.

The role works as a black box system and requires a script that must be executed from your Client Jumphost which will reach the Vault. While sample scripts are provided here, you need to write your own and ensure that it can connect to the Vault. The script needs to finish by printing the information in a specific format to standard output, with exit status 0.

## Security and Compliance Controls

The CACF solution is ITSS and GDPR compliant. Global CACF has also completed the following (findings are Kyndryl Confidential):

* Security penetration testing with the GTS Security PMO and X-Force Red
* Internal security reviews (part of the GTS TI&A release process)
* Threat Modelling
* GDPR Assessment

### Connectivity Overview

No change. Following the standard model.

### Access Controls

#### Identification

No change. Following the standard model.

#### Authentication

No change. Following the standard model.

#### Authorisation

No change. Following the standard model.

#### Integrity

No change. Following the standard model.

#### Confidentiality

No change. Following the standard model.

### Non-repudiation

No change. Following the standard model.

### Auditing & Accountability

No change. Following the standard model.

#### Logging

No change. Following the standard model.

#### Monitoring & Alerting

No change. Following the standard model.

#### Reporting

No change. Following the standard model.

### Security Management

#### Security Standards and Design Compliance

No change. Following the standard model.

#### Security Process and Procedures

No change. Following the standard model.

# Implementation

## Testing and Validation

Kyndryl and Ericsson will be executed the following tests by scanning a set of endpoints with Ansible playbooks.

1. Subsystem scan – To scan installed software.
2. Patch scan – To report on the patch status of an end point.
3. Inventory Scan – To scan / discover what servers are in the environment, by Hostname.
4. MEF's scan – To scan user IDs on the system, part of the IAM 7 ID validation process.
5. Health check scans – To scan if the the security settings agreed in the Customer Security Document (CSD) is applied.
6. Patching – Patched a Linux server through Ansible using Ericsson’s Satellite server.
7. Patching – Windows, patched Windows servers through Ansible using Kyndryl’s WSUS server.

## Deployment

The deployments steps to implement Ansible in the environment are:

* Enable base connectivity to KSTP Ansible services through new IPsec VPN
* Use existing BigFix proxy servers, deploy Ansible proxy
* Establish required firewall rules
* Create the service accounts
* Enable playbooks

## Migration

Ansible is a new implementation hence no migration needed.

# Viability Assessment

The following section describes any architecture / design assumptions, dependencies, risks and issues that have surfaced through the design process.

In that context, it should be noted that unless implemented via a Change Note to the MSA this document is not capable of creating a contractual or financial obligation for Ericsson and any risk, dependency, assumption referenced herein cannot be relied upon by the Service Provider unless and until it is converted to a Ericsson Responsibility under Schedule 20 via the Change Control Process.

## Assumptions

| **Assumption** | **Impact if Assumption proves false** |
| --- | --- |
| Access to all Kyndryl managed endpoints is allowed. | Manual operation/work |
| Required software are installed on all Kyndryl managed endpoints. | Manual operation/work |
| Ericsson approves Kyndryl standard playbooks. | Manual operation/work |
| Network to all Kyndryl managed endpoints | Manual operation/work |
| All managed endpoints have in supported OS/Software versions applicable for Ansible service. | Manual operation/work |

## Dependencies

Describe the dependencies to other existing or planned systems and to projects running simultaneously.

| **Dependencies** | **Impact if Dependency proves false** |
| --- | --- |
| Service account creation | Delay in deployment |
| Firewall approval and implementation | Delay in deployment |
| Approval of the solution by Security and Centre of Excellence. | Delay in deployment |
| All WSUS master and child servers should have the same level of Operating system version. | Delay in deployment |
| The Bi-directional data flow from WSUS Master at FRA to Childs at FRA,CHE,SYD & WDC should allow for pushing the windows patches with port 8530 & 8531. | Delay in deployment |
| The WSUS child group policies should be configured to managed endpoints over the AD(active directory) groups to sync the patches. | Delay in deployment |
| All the endpoints should communicate with 8530 & 8531 port with WSUS child. | Delay in deployment |
| If we are enabling the https (8531) flow from Underlay to overlay. The SSL certificates to be implemented at WSUS master, WSUS child and managed endpoints. | Delay in deployment |
| The certificate configuration across the endpoints should be push with available automation. | Delay in deployment |

## Risk

No technical risks identified.

| **Risk** | **Probability** | **Level of Impact** | **Impact** | **Mitigation** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

## Issues

No issues identified.

# Bill of Materials

No applicable.

# Appendix

## Playbooks

The following data will be collected as part of runbook execution for security health checking and patching.

1. Version number

2. Source name

3. Customer ID

4. Customer name

5. Country

6. Host name

7. IP address

8. OS name

9. OS version

10. Scan Type

11. Application category

12. Application name

13. Instance name

14. Date of the health check

15. Policy name

16. Policy version

17. Policy item

18. Health Check Result

19. Non-compliance severity

20. Check name

21. Check description

22. Agreed to settings

23. Reason of the non-compliance

24. Submitter name

25. Performed by

26. External reference document

27. External sequence number