**Functional Test Report for**

**Deployment Testing without DNS**

on 16-07-2025

# 

[Testing Scope 3](#_Toc203390203)

[Test Approach 3](#_Toc203390204)

[Time taken for deployment 4](#_Toc203390205)

[Main features tested: 4](#_Toc203390206)

[Test execution statistics 4](#_Toc203390207)

[Automation test results 4](#_Toc203390208)

# 

# Testing Scope

The scope of testing is to verify fitment to the specification from the perspective of

* Deployability
* Functionality

Verification is performed not only from the end user perspective but also from the System Integrator (SI) point of view. Hence Configurability and Extensibility of the software is also assessed. This ensures readiness of software for use in multiple countries. Since MOSIP is an “API First” product platform, Verification scope required comprehensive automation testing for all the MOSIP APIs. An automation Test Rig is created for the same.

# Test Approach

**Reference docs:**

[GitHub Document](https://github.com/mosip/documentation/blob/1.2.0/docs/setup/deploymentnew/v3-installation/1.2.0.2/on-premises-deployment-without-dns.md)

* Deployability
* Functionality

Environment name: mod.mosip.net

Deployment test completed date: 16th July 2025.

# **Main features tested:**

* MOSIP services deployed successfully
* Basic functionality: Unable to run automation - [Bug](https://mosip.atlassian.net/browse/MOSIP-42341)

# **Time taken for deployment**

|  |  |  |
| --- | --- | --- |
| **Step No** | **Step Description** | **Time Taken without DNS** |
| 1 | Pre-requisites | 2 hours |
| 2 | Wireguard | 2 hours |
| 3 | Observation K8s Cluster setup and configuration ([Bug](https://mosip.atlassian.net/browse/MOSIP-42242)) | 4 hours |
| 4 | Observation K8s Cluster Ingress and Storage class setup | 2 hours |
| 5 | Setting up nginx server for Observation K8s Cluster ([Bug](https://mosip.atlassian.net/browse/MOSIP-42247)) | 3 hours |
| 6 | Observation K8's Cluster Apps Installation | 3 hours |
| 7 | MOSIP K8s Cluster setup ([Bug](https://mosip.atlassian.net/browse/MOSIP-42272)) | 4 hours |
| 8 | MOSIP K8 Cluster Global configmap, Ingress and Storage Class setup ([Bug](https://mosip.atlassian.net/browse/MOSIP-42295)) | 4 hours |
| 9 | Import MOSIP Cluster into Rancher UI(Not Succeed – [Bug](https://mosip.atlassian.net/browse/MOSIP-42295)) | 1 hr |
| 10 | MOSIP K8 cluster Nginx server setup | 1 hour |
| 11 | MOSIP External Dependencies setup ([Bug](https://mosip.atlassian.net/browse/MOSIP-42272)) | 1 hour |
| 12 | MOSIP Modules Deployment | 2 hours |
| 13 | API Testrig & DSLRig(Not Succeed – [Bug](https://mosip.atlassian.net/browse/MOSIP-42341)) | 1 hour |
|  | **Total – Time taken** | **30 hrs (4 days)** |

# **High-Level Observation**

* **Secure Network & Cluster Provisioning**Used WireGuard for encrypted access, allowed only valid peer holders, provisioned using RKE, configured keys, installed runtimes, disabled swap, and customized resolution using CoreDNS.
* **Ingress, Storage & DNS Workarounds**Deployed entry controllers and gateways, routed traffic internally, configured persistent access, patched default behavior, generated self-signed certificates, injected mappings into CoreDNS and local entries, and avoided public dependencies.
* **Managed MOSIP Cluster from Command Line While Observation Was Imported into Rancher**  
  Successfully imported the Observation cluster into Rancher UI, while the MOSIP cluster was fully managed via command-line tools (kubectl, helm, RKE), including deploying services and applying manifests directly.
* **Handled Non-Executable Scripts Using CLI Commands**  
  Faced non-executable script issues during setup, used command-line instructions to update permissions, and enabled manual execution.
* **Used Single PEM File for Access Across All VMs**  
  Accessed all virtual environments using one credential file, avoided multiple authentication setups, and simplified secure login configuration.
* **TestRig and DSLRig Execution Blocked by DNS Dependencies**  
  Skipped execution of testrig and dslrig; Helm charts were not adapted for setups without public DNS resolution, causing failure during dependency loading.
* **Verified Each Step**  
  After executing each step, confirmed results through service status checks (e.g., systemctl status, kubectl get pods), UI visibility (e.g., Rancher dashboard, Ingress endpoints), and file presence (ls, cat). Verified each outcome before moving to the next step.