Aggregating Heterogeneous Computing Resources

August 2023

AUTHOR(S):
Nathaniel James Pacey

SUPERVISOR(S):
Luca Atzori
Krzysztof Michal Mastyna
Joaquim Santos

CERN openlab

PROJECT SPECIFICATION

The project's core objective is to develop a sophisticated web portal that acts as a comprehensive platform for optimizing resource allocation and improving system administration tasks. Key goals include streamlining resource management processes, enhancing resource utilization, monitoring hardware performance, and boosting system reliability.

The envisioned web portal is designed to merge data from multiple sources into a unified interface, granting administrators a real-time overview of the diverse computing nodes within the openlab network. Automation will play a significant role, enabling the portal to gather data from internal lists and deployment tools, potentially utilizing scripts and databases for comprehensive machine information. This consolidated data could be harnessed to generate valuable reports and notifications about user expirations, thereby aiding administrators in effective resource management.

Moreover, the web portal will offer manual functionalities, allowing administrators to manually add or remove non-managed machines, include extra users, and engage directly with specific machine users via broadcast emails. This flexibility contributes to the network's adaptability to changing project needs and resource demands.

The significance of this project stems from its potential to enhance the efficiency and effectiveness of system administrators. By centralizing resource management through the portal, administrators can align resources with project requirements, minimize wastage, optimize workload distribution, and ultimately bolster system stability. These enhancements, in turn, foster a more resilient and dependable computing environment, pivotal for the successful execution of projects within the intricate openlab network.

ABSTRACT

Openlab faces a multifaceted challenge in effectively managing its diverse array of computing resources, which includes on-premise servers, cloud resources for quantum computing, and a variety of CPU and GPU configurations. The unique configurations and management tools required for these resources demand a comprehensive and efficient approach to ensure reliable and optimal performance. This project centers around the primary objectives of extending the capabilities of an existing web portal to substantially enhance resource management for the openlab testbed. The fundamental aspirations include the refinement of resource allocation procedures, optimization of resource utilization, proactive hardware performance monitoring, and the reinforcement of system reliability.

This expansion is realized through the introduction of an array of new features, each strategically tailored to elevate the portal's utility. A new front-end interface will ensure improved user experience and navigation. Enhanced information dissemination, such as real-time load statistics and active user counts, will enrich administrators' insights into resource usage patterns. The new portal will enable administrators to augment the cluster even for machines that cannot use Puppet or Ansible, offering a streamlined process to add, delete, and edit machines and users. All of which will be seamlessly integrated into the online portal, affording administrators greater autonomy over resource management. Automation will hold a crucial role in the portal's operations, driving scheduled updates of machine information and generating expiration reports for machine users automatically. Additionally, the platform will empower administrators to edit machine details, designate nodes as public or private, and automatically generate cumulative reports. This integrated approach streamlines resource management and enhances data analysis for both public and private information. The amalgamated data within the portal not only assists systems administrators in their daily tasks but also serves as a valuable tool for monitoring machine performance, a critical aspect to both openlab's resource management and collaborations with partner companies.

TABLE OF CONTENTS

1. INTRODUCTION

Effectively managing the diverse computing resources in the openlab network is vital for its operations. This challenge is compounded by the variety of computing nodes, including on-premise servers, cloud-based quantum computing resources, and various CPU and GPU configurations. To address these issues and efficiently coordinate resources, track performance, and adapt to new employee influxes, the openlab Systems User Management Portal was developed.

The initial portal lacked comprehensive features required for openlab's intricate computing environment. Consequently, this project's primary objective is to enhance the original portal by introducing features that amplify the Systems Portal's capabilities. These enhancements aim to provide administrators with efficient resource management and intuitive visualization tools tailored to the unique challenges within the openlab network. Achieving this objective involves integrating innovative features into the portal, such as automated routines on schedules and improved manual functions enabled by the portal's enhanced capabilities.

This report details the expansion of the portal's capabilities and functionality. It explains how data is collected from various sources, including Puppet and Ansible configuration files, GitLab repositories, and Puppet and Ansible facts. The report emphasizes the technical aspects of design and development, highlighting the use of Flask web applications and modular blueprints for practicality and flexibility. Additionally, it showcases the portal's ability to automate user expiration notifications, improve data visualization, and empower administrators with intuitive resource management features.

While the project has significantly enhanced the portal's utility, it acknowledges the evolving nature of resource management needs. Future advancements may include adding user accounts with passwords, seamless access to data from password-protected nodes, and creating universally accessible versions of the portal to serve all users within the openlab community. These developments aim to further enhance the openlab Systems User Management Portal, enabling system administrators to efficiently adapt and optimize resources in CERN's dynamic landscape.

BACKGROUND

a. System Administration and Configuration Tools

In the realm of modern computing, the role of system administrators is pivotal. These professionals undertake the critical responsibility of managing and maintaining computing resources to ensure optimal performance and reliability. The intricacies of their tasks span a spectrum of activities, from setting up and configuring machines to monitoring performance and addressing issues as they arise.

In the pursuit of efficient resource management, configuration tools like Puppet and Ansible have become essential tools for administrators. These tools automate the deployment and management of software configurations across diverse systems. Puppet employs a declarative language and utilizes agents to enforce desired system states, while Ansible, a YAML-based, agentless tool, orchestrates remote system configurations using SSH. Both tools optimize performance by standardizing and automating repetitive tasks, which in turn reduces human error and is ideal for large-scale system management [1].

While configuration tools like Puppet and Ansible offer significant advantages, manual configurations are still necessary in specific situations. These manual interventions are typically required for smaller-scale setups or when dealing with non-standard configurations, particularly on specialized resources. However, these manual tasks often consume time, are susceptible to inconsistencies, and may lack scalability when applied in larger environments.

CERN openlab operates in a unique environment characterized by a constant influx of new users, resulting in the continuous addition and removal of users from machines. The servers hosted by openlab play multiple critical roles, serving as components of CERN's computing infrastructure for experiments, testing grounds, research environments for collaborative companies, and a vital R&D infrastructure for evaluating cutting-edge technologies. Furthermore, the diverse range of computing resources within openlab introduces complexity, making it clear that a single standardized method is insufficient for resource management tasks. In this dynamic context, the necessity for an advanced web portal that seamlessly consolidates and enhances resource management capabilities becomes evident. Additionally, the reliability of these computing resources is of utmost importance, emphasizing the need for a robust resource management system that ensures uninterrupted operations and stability within this dynamic and demanding environment.

b. Flask Web Applications

In the realm of modern web development, Flask serves as a lightweight yet versatile framework, forming the backbone of openlab's enhanced resource management portal. Developed in Python, Flask provides the essential tools and structure needed to construct dynamic and interactive web solutions. At its core, Flask simplifies the creation of web applications by offering a range of features while remaining adaptable to the unique needs of developers. At the heart of the portal's architecture are Flask's routes and views. Routes define specific URLs to which the application responds, and views are Python functions linked to these routes, guiding the actions taken when users access specific URLs. This modular approach ensures streamlined code organization and facilitates efficient maintenance, making it easier for developers to manage the portal's functionalities [2].

Flask's extensive use of extensions plays a crucial role in enhancing the portal's capabilities. These pre-built solutions, developed by both the Flask team and the broader community, expedite the integration of common web application features such as authentication, database connectivity, and form handling. This extension-driven approach allows developers to seamlessly incorporate intricate functionalities without the need to build them from scratch. Additionally, Flask's templating engine, Jinja2, further enhances the portal's interactivity. By embedding dynamic content within HTML templates, developers can generate and display data-driven information effectively, enhancing the user experience [2].

Harnessing Flask's rich feature set and adaptability, the portal furnishes system administrators within the openlab environment with a resilient and user-friendly platform for efficient resource management, thus guaranteeing the network's agility in response to the ever-changing demands of its computing environment.

2. PREVIOUSLY EXISTING ARCHITECTURE

The openlab Systems User Management Portal is a comprehensive solution designed to address the intricate challenges of managing a diverse range of computing resources within the CERN openlab environment. This section provides a detailed insight into the mechanics and functionalities of the web portal.

a. Web Application Structure

System Architecture and File Structure

The web portal is structured using the Flask framework, divided into multiple directories and modules. The *run.py* script initializes the application, while each module contains essential components such as views, controllers, and background job schedulers. The utilization of templates and static files ensures the creation of dynamic web pages with consistent styling.

Data Visualization

A central objective of the web portal is to present the collected data in a user-friendly and informative manner. Leveraging the Flask framework's dynamic template engine, the portal offers an intuitive graphical user interface (GUI). This interface allows administrators to interact with the data, visualize the status of each node, and access vital information effortlessly.

Blueprint System for Modularity

The architecture of the web portal emphasizes modularity and extensibility. The Blueprint system is employed to organize the application into distinct modules, each responsible for specific functionalities. These modules include *mod_machines* and *mod_auth*, which respectively manage data collection, visualization, notification, and user authentication. The previously existing system can be seen in the master branch of the user-management-portal repository [3].

Authentication and User Management

The *mod_auth* module handles user authentication through a sophisticated claim-based system. The CERN OAuth2 Authorization Service is employed to manage user sessions and access authorization. By validating egroup memberships, the system grants or denies access to the portal's resources. This module ensures secure user management while maintaining a seamless user experience.

Background Jobs for Automation

The portal embraces automation through background jobs powered by the BackgroundScheduler library. These jobs facilitate the automatic execution of critical tasks, including data collection, hardware fact retrieval, and user expiration checks. By automating these processes, the portal minimizes manual interventions, enhances efficiency, and ensures data accuracy.

User Expiration Notification

The portal takes proactive measures to ensure effective user management. A dedicated background job maps each users’ expiration date on each node based on the parsed YAML data. This enables the system to trigger automated email notifications to users regarding impending expirations. By sending notifications at strategic intervals, the portal facilitates timely communication and assists in resource allocation planning.

Broadcast Emails

The Systems Portal facilitates the sending of broadcast emails to specific machine users. This feature enhances communication and coordination by allowing administrators to efficiently convey important information to targeted user groups. Whether for updates, announcements, or notifications, this functionality streamlines the process of reaching out to relevant users.

b. Data Retrieval from Puppet and Ansible

A crucial aspect of the openlab Systems User Management Portal's functionality lies in its ability to retrieve and collate data from Puppet and Ansible. These data sources are the foundation of the portal's accurate representation of the computing resources within the network. This section provides an in-depth exploration of how the portal effectively extracts information from these configuration tools.

Aggregating Dynamic Configuration Data from Puppet and Ansible

Puppet functions as a robust configuration management tool that facilitates the automation and maintenance of system deployment and software configuration. It employs a declarative language and often stores its configurations in YAML format. The portal initiates its data retrieval process by accessing the designated directory housing the Puppet YAML configuration files. These files encapsulate critical details about each network node, encompassing dynamic information such as hostnames, assigned users, and expiration dates. Through periodic scans of this directory, the portal's background job systematically reads and parses the YAML files, methodically extracting the required information.

Similarly, Ansible configuration YAML files yield vital insights into node configurations. The portal's integration process seamlessly combines extracted dynamic data from both Puppet and Ansible into a unified JSON repository through the use of scheduled jobs. By effectively merging dynamic information from both Puppet and Ansible sources, the portal empowers administrators with comprehensive resource insights, facilitating optimal allocation and informed decisions within the dynamic context of CERN openlab.

Static Machine Information and Integration

Complementing its dynamic data offerings, the portal enhances its capabilities with AI-PDB (AI Puppet Database) facts. These facts are sourced from normalized systems information generated by Puppet and encompass a spectrum of hardware attributes. This includes aspects like storage, operating systems, BIOS details, and more. Once parsed, these attributes enhance the dataset, offering administrators a comprehensive overview of each machine's underlying hardware characteristics. Similarly Ansible facts with machine data are also retrieved and put into a JSON file containing the cumulative static data.

Once both the static and dynamic JSON files are populated the portal can take the information for each machine and display it on a cumulative table. To display the information dynamically Jinja loops and conditionals were embedded in the HTML frontend.

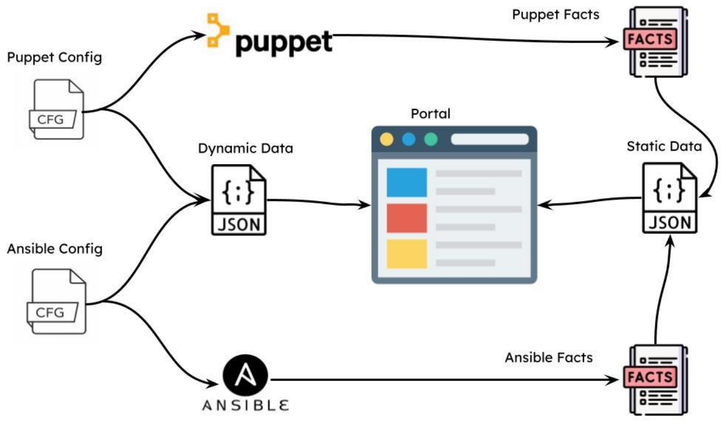


Figure 1. Data extrapolation from nodes with Puppet or Ansible enabled.

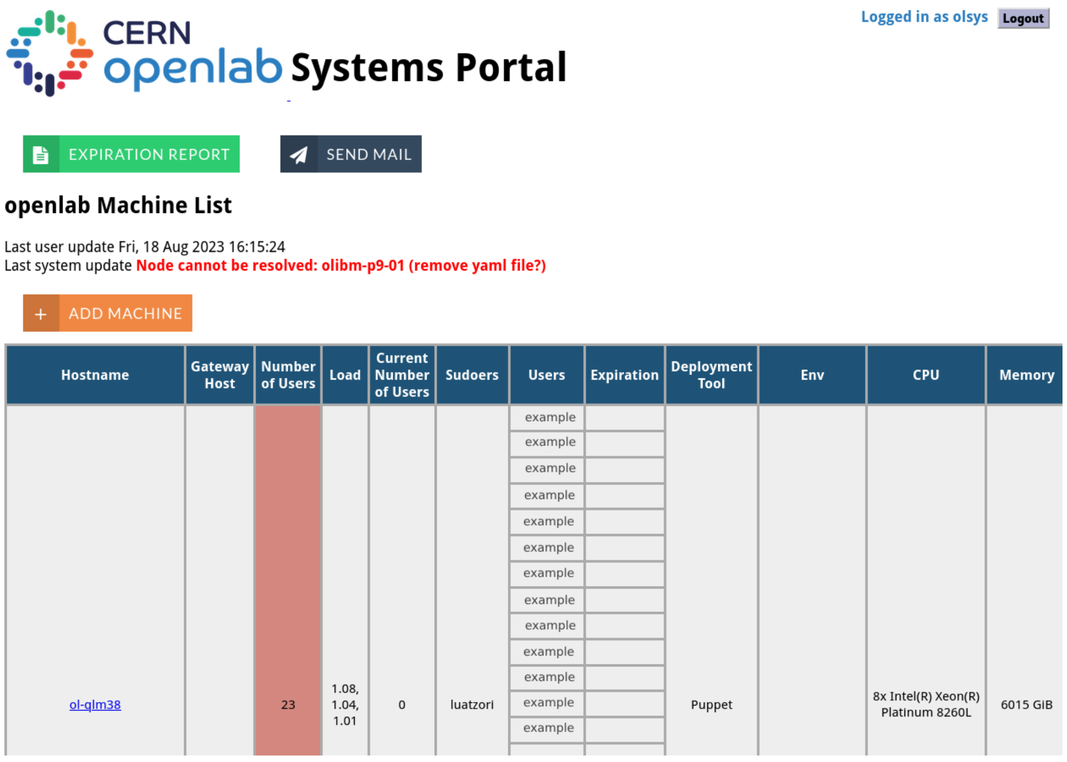
A full diagram displaying the flow of data from both Puppet and Ansible into the portal can be seen above in Figure 1. As mentioned, the dynamic data is populated from configuration files while the Static data is populated from the resulting facts generated by the configuration tool.

3. UPDATED INFRASTRUCTURE AND FUNCTIONALITY

a. Front End Update

Recognizing the importance of a user-friendly interface, the portal underwent significant improvements in its front-end design to align with existing tools and systems at CERN. The aim was to create a more modern experience for system administrators, providing them with familiar features while introducing enhancements for improved usability and accessibility.

The design overhaul incorporated interactive buttons that intuitively guide administrators through various tasks, facilitating quick access to essential functionalities. The addition of images and icons served as visual cues, aiding in the rapid identification of different components and actions within the portal as can be seen below in Figure 2.



openlab Machine List

Last user update Fri, 18 Aug 2023 16:15:24
Last system update Node cannot be resolved: olibm-p9-01 (remove yaml file?)

Hostname	Gateway Host	Number of Users	Load	Current Number of Users	Sudoers	Users	Expiration	Deployment Tool	Env	CPU	Memory
ol-glm38		23	1.08, 1.04, 1.01	0	luatzori	example example example example example example example example example example example example example		Puppet		8x Intel(R) Xeon(R) Platinum 8260L	6015 GiB

Figure 2. Updated Machine List webpage with new colour themes and interactive features.

To enhance usability, a new color palette was adopted, ensuring that the interface maintains visual consistency with other tools commonly used within the CERN environment. Additionally, the number of users background colour is dynamically updated based on the number of users assigned to a machine. This allows systems administrators to quickly identify which machines have limited capacity for additional users. This choice not only fosters a sense of familiarity but also promotes a smoother transition for administrators who frequently work across multiple platforms. For images of each page within the enhanced Systems Portal, please refer to the Appendix a. Frontend Update Images.

b. Additional Manual Controls

The enhanced Systems Portal introduces a set of manual controls that empower administrators with additional functionalities, allowing them to tailor resource management to specific needs. These manual tasks provide a higher degree of flexibility and customization, enhancing the portal's utility.

Machine Management

Within the openlab environment, certain machines lack the Puppet or Ansible configuration management tools, creating a distinct challenge for systems administrators. To address this, administrators now have the ability to manually add or remove non-managed machines from the network. This feature provides a customizable approach to resource management, allowing for tailored adaptations to infrastructure needs while maintaining an accurate and current representation of the network within the portal.

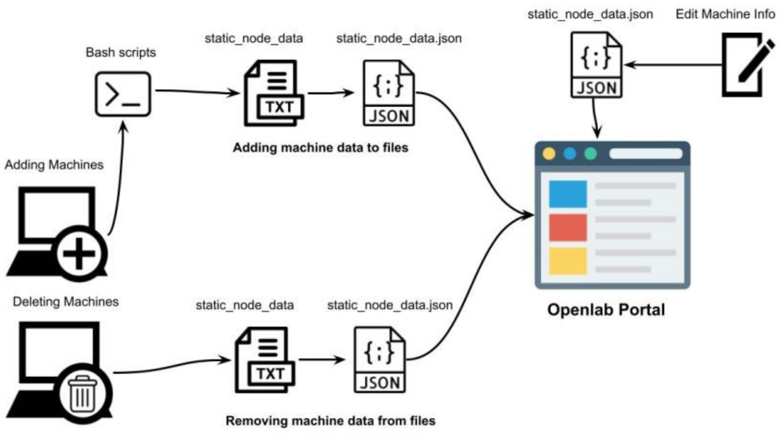


Figure 3. Updated machine management systems.

As shown above in Figure 3, the portal utilizes bash scripts (*node_data.sh* and *ssh_commands.sh*) to connect to the added machine via ssh then extract both dynamic and static hardware information using a variety of commands before populating the text file. The commands run on the added machine and can be found in the *ssh_commands.sh* which is called from the *node_data.sh*, both files full code can be found in the *mod_machine* module on the most updated branch of the portal repository *node_create* [4]. The text file is parsed using Python after the shell command is finished and populates a JSON file (*static_node_data.json*). The data from the JSON file is then used to populate the user portal with the new machine information by calling the scheduled job, *static_node_add*. Note that this function updates all machine information not just the added machine.

If a node necessitates access through a gateway node from the portal, you can specify the gateway node on the webpage while adding a machine, as illustrated in Figure 10 of Appendix A. The script effectively utilizes both the designated gateway node and machine name to establish ssh connections to both nodes. Subsequently, it initiates the execution of commands and the extraction of information, ensuring a streamlined process for data retrieval.

Deleting machines simply uses Python to remove the machine information from both the text and JSON file before updating the portal. It is important to note that while the text files are located locally in the *mod_machines* directory the JSON files are stored on the shared *eos* directory.

As well, the ability to edit certain information (remote, rack, usage, notes) was added to the portal and allows systems administrators to add information about the node that is not accessible to the portal.

User Management

The manual controls integrated into the portal encompass user management functionality, empowering administrators to add or remove additional users as necessary, along with updating user expirations. Given the dynamic nature of openlab, where employees regularly join and depart, and diverse projects demand varying resource allocations, user compositions on each machine experience frequent fluctuations. As a result, the capability to oversee user management through the portal offers a substantial acceleration of systems administrators' tasks.

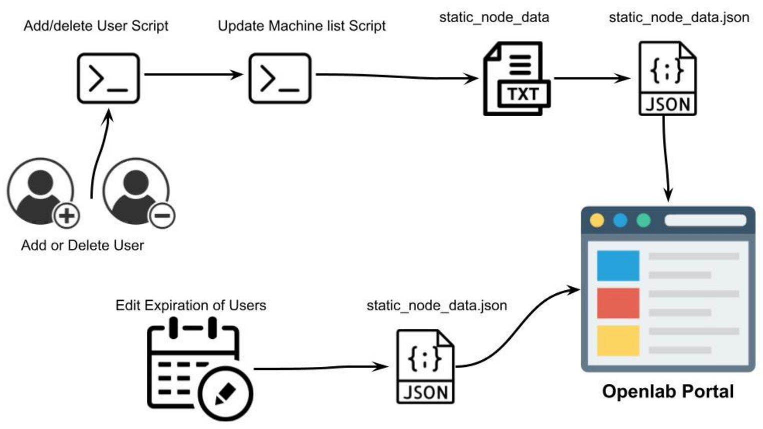


Figure 4. Updated user management systems.

The process of managing users can be seen above in Figure 4. When adding or deleting a user the portal uses a bash script to connect to the node and then run the add/delete command; then the portal updates all the added machines by running the schedule job to update all added machines where the new machine information is written to the text file and then to the JSON file before being added to the portal.

Additionally, by selecting the username within the machine list table, the portal enables the update of user expiration dates, seamlessly recorded within the JSON file. This updated information then synchronizes with the expiration report scheduled job, leading to automated and timely email notifications based on user expiry dates per machine.

c. Additional Automated Jobs

The integration of automated scheduled jobs is fundamental to the seamless functioning of the Systems Portal. To accommodate the extended functionalities of the portal, several supplementary automated tasks have been implemented. These encompass updating additional nodes, generating expiration reports, sending notifications to added users, and producing cumulative reports. These automated jobs collectively contribute to the enhanced efficiency and comprehensive utility of the portal.

Scheduled Updates

The Portal incorporates a variety of background jobs operating on predefined schedules, tailored to the fluctuation frequency of the data. For instance, the static hardware data, which experiences infrequent changes for each machine, undergoes updates every 30 minutes. Conversely, the dynamic data, which is more volatile, receives updates every 15 minutes. With the inclusion of nodes lacking configuration tools

(Puppet or Ansible), periodic checks for machine information becomes essential. Consequently, in alignment with the other nodes, the information for added machines is refreshed every 30 minutes. Additionally, dynamic information for nodes managed by Puppet such as current user counts and load averages is not available through facts and is therefore updated every 30 minutes using a job scheduler with the static node update method.

Similar to the addition of nodes, the updating process employs bash scripts to extract information by reading hostnames and executing shell commands. The extracted data is then overwritten into text files, subsequently used to update the corresponding JSON files. It is crucial to ensure the timely update of dynamic machine data, while retaining permanent information, such as user validity periods and node locations, in their original state.

These regular updates empower systems administrators to swiftly identify any modifications made to machine configurations. They also provide real-time insights into dynamic factors, such as machine load and the current number of users. This dynamic information offers administrators valuable insights into node usage patterns, enabling them to make informed decisions regarding resource management and allocation.

Expiration Reports and Mailing

The upgraded portal boasts a streamlined process for managing user expiration, along with an efficient system for broadcast mailing. Through scheduled background jobs, the portal generates comprehensive expiration reports that highlight users with impending access expiration on their designated machines. What sets this process apart is its seamless integration with user management – each time an additional user is added, their validity period is meticulously checked and incorporated into the expiration report. This ensures that administrators have an up-to-date view of user access timelines. The portal also automates periodic emails to users based on their expiration date, enabling timely notifications and user engagement.

Moreover, the portal's functionality extends to broadcasting emails to specific machine users, enabling effective communication of updates, maintenance, or other relevant information. By seamlessly combining expiration management and broadcast mailing, the portal not only streamlines administrative tasks but also enhances communication and user experience within the system.

Cumulative Reports and Integration

Since the portal incorporates several data sources and displays them cumulatively it is import to have the ability to generate file representing the accumulation of this data. Therefore, the addition of cumulative JSON and csv reports were introduced.

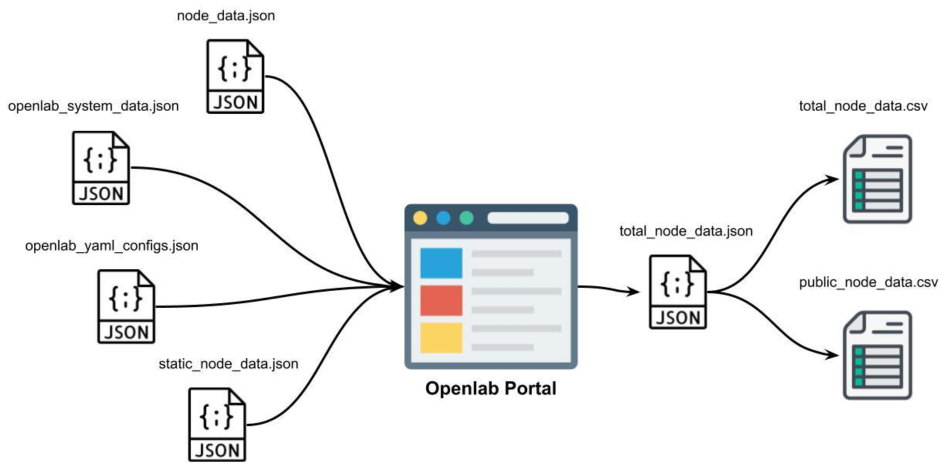


Figure 5. Amalgamation of all data sources and production of cumulative reports.

As shown in Figure 5, the static, dynamic and additional information (*openlab_system_data.json*, *openlab_yaml_configs.json* and *node_data* respectively) from the Puppet and Ansible nodes are integrated into the portal along with the data from the added nodes (*static_node_data.json*). This data is then amalgamated into one JSON file (*total_node_data.json*) before populating the *total_node_data.csv*.

Additionally, systems administrators can use the portal to specify which machines data should be added to a public csv file through the use of a checkbox on the edit host page (as can be seen in Appendix a. Figure 9) that gets written to the total JSON file. The public csv is populated by checking which nodes have been specified as public in the total JSON and then populating the public nodes data into the public csv file.

4. FUTURE WORK

While the enhanced Systems Portal has significantly advanced resource management capabilities within the openlab network, there remain avenues for further development and refinement. The following areas represent potential directions for future work.

Enhanced User Management

The portal could evolve to allow the automated generation of user accounts complete with passwords, accompanied by an automated email requesting users to change their default passwords upon first login. This streamlined process not only enhances security but also simplifies the onboarding experience for new users.

Access to Password-Protected Nodes

Future iterations of the portal could explore the inclusion of mechanisms to retrieve data from password-protected nodes within the network. This enhancement would enable administrators to access and manage a broader range of resources, expanding the portal's utility and coverage.

Public Version of the Portal

Creating a public version of the portal, where sensitive information is removed, and security vulnerabilities are addressed, could broaden the portal's reach and usefulness. Such a version could serve as a valuable resource for users outside of CERN, facilitating collaboration and information dissemination while adhering to security and privacy standards.

As well, a version of the portal that allowed any openlab user to login and only view the computing information that their user has access to, would allow openlab members to better understand their own computing resources.

These potential enhancements underscore the portal's ongoing evolution as a dynamic resource management tool. By addressing these aspects in future iterations, the portal can continue to adapt and cater to the changing needs of administrators and users within the openlab network.

5. CONCLUSION

In conclusion, this project aimed to improve resource management within the openlab network by developing a practical web portal. The portal's objectives included enhancing system administration tasks, optimizing resource allocation, and improving hardware performance monitoring.

By consolidating data from various sources into a single interface, the portal offers administrators real-time insights into the network's computing nodes. The revamped front-end design prioritized user-friendliness, incorporating interactive elements and visual cues for a smoother experience. Manual controls were added to facilitate tasks like managing machines and users, providing more flexibility. Automation played a key role in the portal's functionality. Scheduled background jobs handle tasks like updating machine data, generating expiration reports, and sending notifications to users. These automated processes streamline administrative work and ensure accurate data.

Looking forward, potential improvements include automating user account generation and password management, accessing password-protected nodes, and creating a public version of the portal. These possibilities align with the portal's adaptability to the evolving needs of the openlab network. In essence, this project contributes practical solutions for efficient resource management, aligning with openlab's commitment to innovation and effective collaboration.

6. Works Cited

[1] Red Hat, “Ansible vs. Puppet: What you need to know,” Red Hat, 28 November 2022. [Online]. Available: https://www.redhat.com/en/topics/automation/ansible-vs-puppet. [Accessed 10 August 2023].

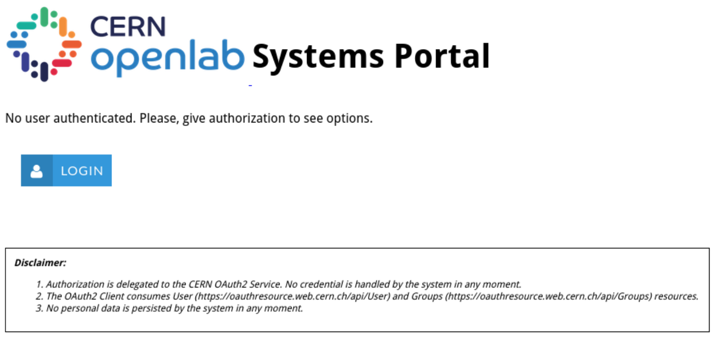
[2] Flask, “Flask Documentation,” Pallets, 2010. [Online]. Available: https://flask.palletsprojects.com/en/2.3.x/. [Accessed 2 July 2023].

[3] CERN, “openlab systems User Management Portal,” CERN, 2023. [Online]. Available: https://gitlab.cern.ch/openlab-systems/user-management-portal/-/tree/master?ref_type=heads. [Accessed June 2023].

[4] CERN, “Updated openlab systems User Management Portal,” CERN, 2023. [Online]. Available: https://gitlab.cern.ch/openlab-systems/user-management-portal/-/tree/node_create?ref_type=heads. [Accessed 2023].

7. APPENDIX

a. Frontend Update Images

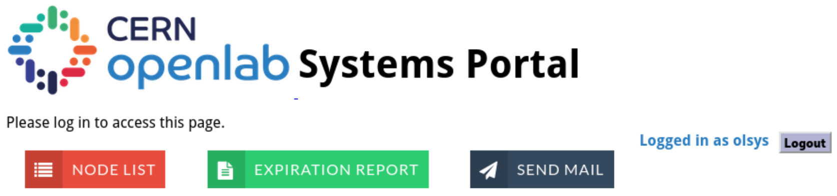


No user authenticated. Please, give authorization to see options.

Disclaimer:

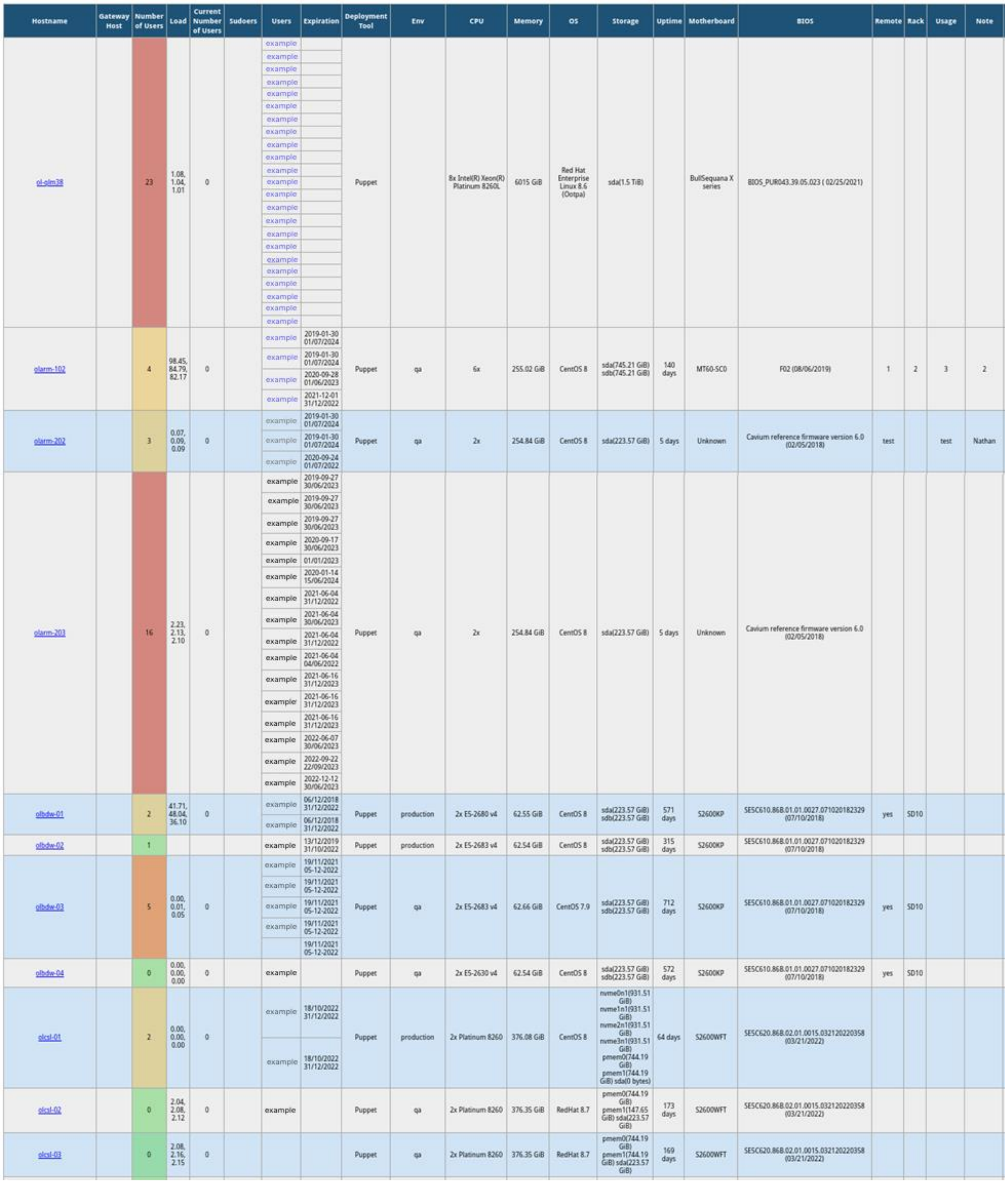
1. *Authorization is delegated to the CERN OAuth2 Service. No credential is handled by the system in any moment.*
2. *The OAuth2 Client consumes User (https://oauthresource.web.cern.ch/api/User) and Groups (https://oauthresource.web.cern.ch/api/Groups) resources.*
3. *No personal data is persisted by the system in any moment.*

Figure 6. Updated System Portal login page.



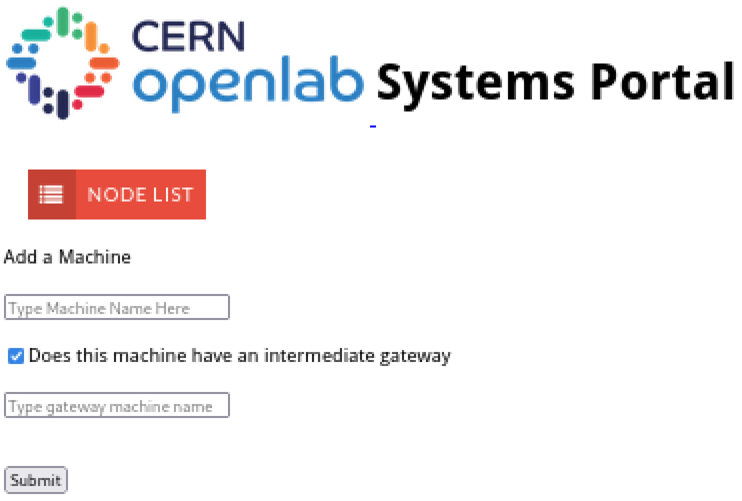
Please log in to access this page.

Figure 7. Home page of the updated system portal.



Hostname	Gateway Host	Number of Users	Load	Current Number of Users	Sudoers	Users	Expiration	Deployment Tool	Env	CPU	Memory	OS	Storage	Uptime	Motherboard	BIOS	Remote	Rack	Usage	Note
ol-glm38		23	1.08, 1.04, 1.01	0		example example		Puppet		8x Intel(R) Xeon(R) Platinum 8260L	6015 GiB	Red Hat Enterprise Linux 8.6 (Ootpa)	sda(1.5 TiB)		BullSequana X series	BIOS_PUR043.39.05.023 (02/25/2021)				
olarm-102		4	98.45, 84.79, 82.17	0		example example example example	2019-01-30 01/07/2024 2019-01-30 01/07/2024 2020-09-28 01/06/2023 2021-12-01 31/12/2022	Puppet	qa	6x	255.02 GiB	CentOS 8	sda(745.21 GiB) sdb(745.21 GiB)	140 days	MT60-SC0	F02 (08/06/2019)	1	2	3	2
olarm-202		3	0.07, 0.09, 0.09	0		example example example	2019-01-30 01/07/2024 2019-01-30 01/07/2024 2020-09-24 01/07/2022	Puppet	qa	2x	254.84 GiB	CentOS 8	sda(223.57 GiB)	5 days	Unknown	Cavium reference firmware version 6.0 (02/05/2018)	test		test	Nathan
olarm-203		16	2.23, 2.13, 2.10	0		example example example example example example example example example example example example example example example example	2019-09-27 30/06/2023 2019-09-27 30/06/2023 2019-09-27 30/06/2023 2020-09-17 30/06/2023 01/01/2023 2020-01-14 15/06/2024 2021-06-04 31/12/2022 2021-06-04 30/06/2023 2021-06-04 31/12/2022 2021-06-04 04/06/2022 2021-06-16 31/12/2023 2021-06-16 31/12/2023 2021-06-16 31/12/2023 2022-06-07 30/06/2023 2022-09-22 22/09/2023 2022-12-12 30/06/2023	Puppet	qa	2x	254.84 GiB	CentOS 8	sda(223.57 GiB)	5 days	Unknown	Cavium reference firmware version 6.0 (02/05/2018)				
olbdw-01		2	41.71, 48.04, 36.10	0		example example	06/12/2018 31/12/2022 06/12/2018 31/12/2022	Puppet	production	2x E5-2680 v4	62.55 GiB	CentOS 8	sda(223.57 GiB) sdb(223.57 GiB)	571 days	S2600KP	SE5C610.86B.01.01.0027.071020182329 (07/10/2018)	yes	SD10		
olbdw-02		1				example	13/12/2019 31/10/2022	Puppet	production	2x E5-2683 v4	62.54 GiB	CentOS 8	sda(223.57 GiB) sdb(223.57 GiB)	315 days	S2600KP	SE5C610.86B.01.01.0027.071020182329 (07/10/2018)				
olbdw-03		5	0.00, 0.01, 0.05	0		example example example example	19/11/2021 05-12-2022 19/11/2021 05-12-2022 19/11/2021 05-12-2022 19/11/2021 05-12-2022 19/11/2021 05-12-2022	Puppet	qa	2x E5-2683 v4	62.66 GiB	CentOS 7.9	sda(223.57 GiB) sdb(223.57 GiB)	712 days	S2600KP	SE5C610.86B.01.01.0027.071020182329 (07/10/2018)	yes	SD10		
olbdw-04		0	0.00, 0.00, 0.00	0		example		Puppet	qa	2x E5-2630 v4	62.54 GiB	CentOS 8	sda(223.57 GiB) sdb(223.57 GiB)	572 days	S2600KP	SE5C610.86B.01.01.0027.071020182329 (07/10/2018)	yes	SD10		
olcsl-01		2	0.00, 0.00, 0.00	0		example example	18/10/2022 31/12/2022 18/10/2022 31/12/2022	Puppet	production	2x Platinum 8260	376.08 GiB	CentOS 8	nvme0n1(931.51 GiB) nvme1n1(931.51 GiB) nvme2n1(931.51 GiB) nvme3n1(931.51 GiB) pmem0(744.19 GiB) pmem1(744.19 GiB) sda(0 bytes)	64 days	S2600WFT	SE5C620.86B.02.01.0015.032120220358 (03/21/2022)				
olcsl-02		0	2.04, 2.08, 2.12	0		example		Puppet	qa	2x Platinum 8260	376.35 GiB	RedHat 8.7	pmem0(744.19 GiB) pmem1(147.65 GiB) sda(223.57 GiB)	173 days	S2600WFT	SE5C620.86B.02.01.0015.032120220358 (03/21/2022)				
olcsl-03		0	2.08, 2.16, 2.15	0				Puppet	qa	2x Platinum 8260	376.35 GiB	RedHat 8.7	pmem0(744.19 GiB) pmem1(744.19 GiB) sda(223.57 GiB)	169 days	S2600WFT	SE5C620.86B.02.01.0015.032120220358 (03/21/2022)				

Figure 8. Updated systems portal, machine list with all the additional hardware fields included.


CERN openlab Systems Portal

NODE LIST

Add a Machine

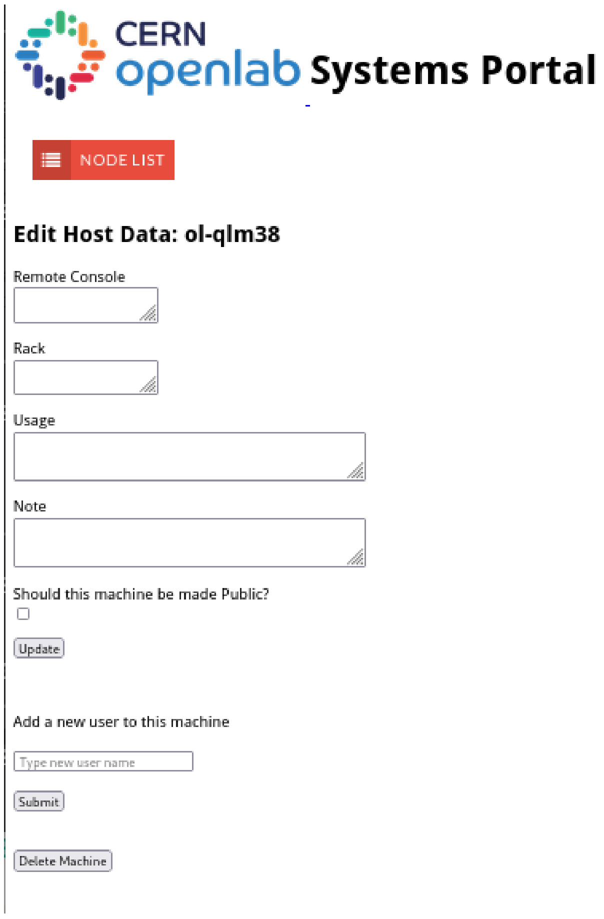
Type Machine Name Here

[✓] Does this machine have an intermediate gateway

Type gateway machine name

Submit

Figure 10. Add an additional machine webpage with the gateway field selected.


CERN openlab Systems Portal

NODE LIST

Edit Host Data: ol-qlm38

Remote Console

Rack

Usage

Note

Should this machine be made Public?

☐

Update

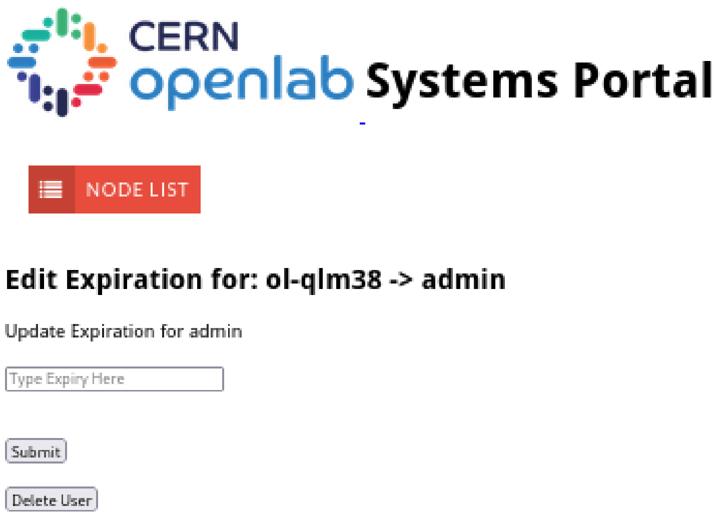
Add a new user to this machine

Type new user name

Submit

Delete Machine

Figure 9. Updated edit host page for ol-qlm38.


Systems Portal

NODE LIST

Edit Expiration for: ol-qlm38 -> admin

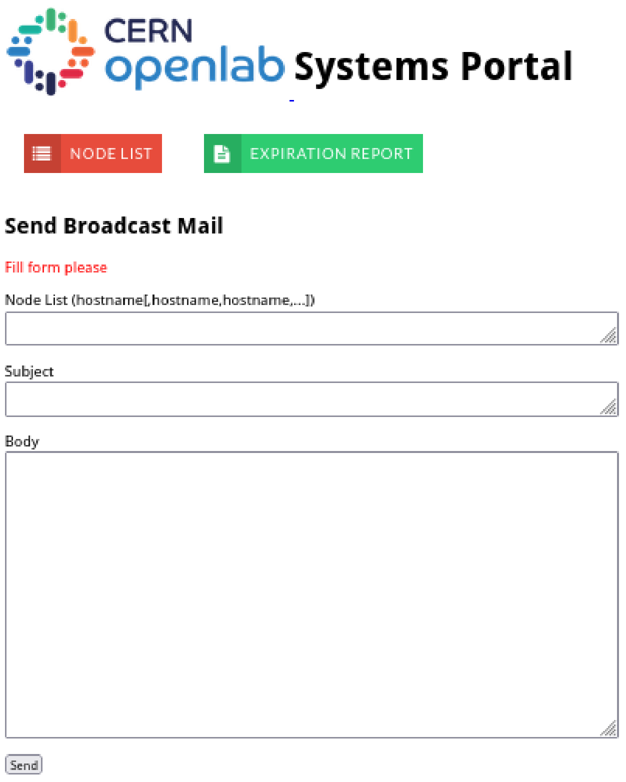
Update Expiration for admin

Type Expiry Here

Submit

Delete User

Figure 12. Edit user webpage for the machine ol-qlm38 and user admin.


Systems Portal

NODE LIST

EXPIRATION REPORT

Send Broadcast Mail

Fill form please

Node List (hostname[,hostname,hostname,...])

Subject

Body

Send

Figure 11. Updated send broadcast mail page.