Distributed OS Manager: Control Your Network from Anywhere

Lakshmi Siva Kanth Reddy Kondamadugula
A04327619
Dpt of Computer Science
Texas A & M University
Corpus Christi, USA

Jobit Jose
A04328225
Dpt of Computer Science
Texas A & M University
Corpus Christi, USA

Chandrahaas Reddy Guntaka
A04328583
Dpt of Computer Science
Texas A & M University
Corpus Christi, USA

Abstract: An application designed to provide users with remote management capabilities for their distributed operating systems. This application allows users to monitor, configure, and control various aspects of their distributed network infrastructure from any location, using a convenient graphical interface. Key features of this application include centralized management of distributed nodes and resources, and remote execution of administrative tasks such as process management, file system operations, and network configuration. The application modernizes web and mobile technologies to ensure seamless access and responsiveness across different devices and platforms. With Distributed OS Manager, users can stay always connected to their distributed network infrastructure, enabling them to respond promptly to issues, optimize resource utilization, and ensure the smooth operation of their distributed operating systems from anywhere.

INTRODUCTION

Distributed System is a field of computer science that defined as computer systems whose inter-communicating components are located on different networked computers. The nodes in a distributed system communicate and coordinate the actions by passing messages to each other to achieve a common goal. The hope is that together, the system can maximize resources and information while preventing failures, as if one system fails, it won't affect the availability of the service. Three significant challenges of distributed systems are maintaining concurrency of components, overcoming the lack of a global clock, and managing the independent failure of components. The main advantages of distributed systems are

1. Better Performance: By using the resources of numerous computers to tackle the workload, distributed systems can perform at higher levels than centralized systems.
2. Cost Effectivity: Although distributed systems consist of high implementation costs, they are relatively cost-effective in the long run. Compared to a mainframe computer, where a single system is composed of several processors, the distributed system is made up of several computers together. This type of infrastructure is far more cost-effective than a mainframe system.
3. Efficiency: Distributed systems are made to be efficient in every aspect since they possess multiple computers.

Each of these computers could work independently to solve problems. This is not only considered to be efficient, but also it significantly saves time for the user.

4. Scalability: Distributed systems are made to be scalable. When there is increase in workload, user can add more workstations. There is no necessary to upgrade a single system. Moreover, no restrictions are placed on the number of machines. It means that these machines will be able to handle the high demand workloads very easily.
5. Reliability: Distributed systems are more reliable than single system in terms of failures. Even in the case of a single node malfunctioning, it does not pose problems to the remaining servers. Other nodes can continue to function fine.
6. Fault Tolerance: The ability to continue operation even if one or more node fail is mentioned as fault tolerance, and distributed systems should build to be fault-tolerant.
7. Security: Data breaches and illegal access can be stopped by including security systems in distributed systems.

There are some disadvantages for distributed systems and that are listed below.

1. Compatibility: In distributed system, compatibility across multiple nodes and software systems will be problem since it may employ various hardware, software, and protocols.
2. Startup Cost: Compared to single system, the implementation cost of a distributed system is significantly high. The infrastructure used in a distributed system makes it more expensive. In addition to that, the constant transmission of information and processing overhead again increases the cost.
3. Overheads: Overheating is common problem faced by distributed systems. It happens when all the stations try to operate at a time. Even though it brings desired results, eventually there will be an increase in computing time. This ultimately impacts the system response time.
4. Network Dependency: Distributed system is prone to network errors which result in communication breakdown. The information may fail to deliver or not in a correct sequence. And also, troubleshooting errors is a difficult task since the data is distributed across various nodes.

PROBLEM STATEMENT

In today's increasingly interconnected world, managing the distributed systems efficiently is a significant challenge. Since organizations rely more on distributed systems to for their operations, there is an urgent need of a web-based remote-control system that controls the nodes within those systems. This system must address several key challenges such as Complexity of Distributed Systems, User-Friendly Interface, Real-Time Interaction, Security, Efficient Data Handling and Versatility.

OBJECTIVE

Develop a Web-Based Remote-Control System: Design and implement a web application that enables users to control and manage nodes in a distributed system. The project leverages distributed system principles to demonstrate seamless real-time interaction, focusing on user-friendly interfaces, secure communication, and efficient data handling. This system aims to

streamline operations and maintenance tasks that require managing the nodes, suitable for various applications.

METHODOLOGY

The methodology section tells about the research methods, tools, technologies, and processes employed to design, implement, and evaluate the network automation project. The structured the approach followed in this project ensures a systematic, organized, and effective execution of the network automation initiatives to achieve the desired objectives, goals, and outcomes.We are designing and implementing a web application that enables users to control and manage nodes in a distributed system. The project leverages distributed system principles to demonstrate seamless real-time interaction, focusing on user-friendly interfaces, secure communication, and efficient data handling. This system aims to streamline operations and maintenance tasks that require managing the nodes, suitable for various applications.
We are doing this to do Node Management, Resource Management, and Network Configuration Management.
Tools and Techniques used to achieve these:

1. Flutter (lang)
Flutter is an open-source mobile app development framework, which was created by Google. It allows the developers mainly to build locally compiled applications for mobile, web, and desktop from a single codebase.
Flutter uses the Dart programming language and provides the good set of libraries and tools for building fast, beautiful, and reliable applications. Its key features include:

1. Cross-platform development
2. Fast development
3. Beautiful UI
4. Native performance
5. Reactive framework
6. Extensive libraries

2.Firebase (database and hosting)

Firebase is a type of cloud-based platform gives various services for building web and mobile applications. Two of its core services are:

1. Firebase Realtime Database (RTDB)
2. Firebase Hosting

3.Android Studio (IDE)

Android Studio is the official Integrated Development Environment (IDE) for Android app development. It's kind of toolset that helps developers to create, debug, and test Android apps. Here are some key aspects of the Android Studio IDE:

1. User Interface
2. Code Editing
3. Project Management
4. Debugging and Testing
5. Design and Layout
6. Version Control
7. Plugins and Extensions
8. Code Analysis and Lint
9. Real-time Collaboration
10. Android SDK and NDK Support

These are the methodologies we used for this project.

IMPLEMENTATION AND RESULT

Figure (1) represents the schematic overview of Distributed Operating System (OS) Manager web application, illustrating its primary components and their interconnections. Below is a detailed description of each component within the system.

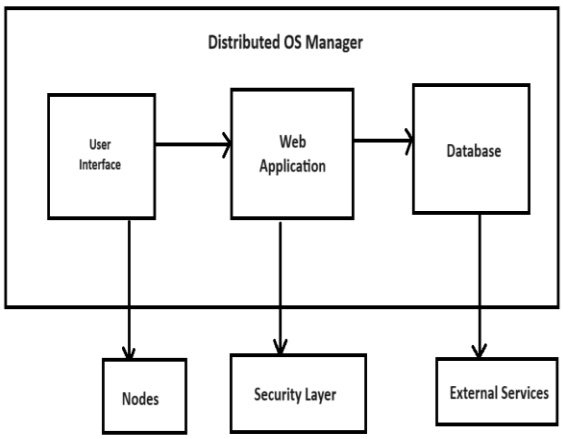

Figure (1): Distributed Operating System Manager

Distributed OS Manager: This central entity helps in the operation and also management of our distributed OS environment. It is responsible for the arranging various tasks and allocating resources correctly across multiple nodes in a system.

Components

User Interface

Purpose: Provides primary interaction point for the users.

Function: Facilitates input of commands, visualization of system, and displays outputs or system responses to user inputs.

Web Application

Purpose: Act as middleware and operational core of system.

Function: Handles processing of the HTTP requests, mediates interactions between the user interface and backend components, and enforces business logic.

Database

Purpose: Serves as the storage mechanism for the distributed systems.

Function: Stores and manages data such as user information, system configuration and operational log details.

Nodes

Purpose: Represents the individual systems within the given distributed system.

Function: Execute assigned tasks and process as part of the distributed operating system functionality.

Security Layer

Purpose: Ensure the integrity and security of system.

Function: Implements security protocol including authentication and encryption to protect system against unauthorized access.

External Services

Purpose: Encloses additional functionalities not as part of the core system.

Function: Interacts with cloud services, third party APIs and external services to extend the ability of the Distributed OS Manager.

Flowchart

The Figure (2) illustrates the workflow for a management interface within a Distributed Operating System, outlining the steps a user would follow to perform various management tasks. The process begins with the user logging in, which is then followed by an authentication step to verify their credentials. Once authenticated, the user gains access to the central dashboard.

From the central dashboard, the user has the option to choose from several management tasks, categorized broadly into node management, resource management, network configuration management, and alerts and notifications settings.

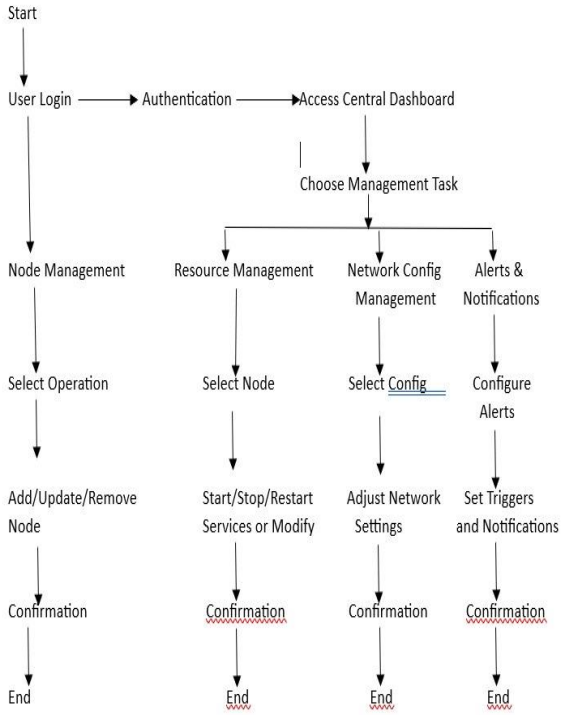

Figure (2): Flowchart

In User Login, the credentials of the host is being checked and it is the first line of defence against unauthorized access. A username and password field is provided for successful login.

In Node Management, the host can select an operation related to the node within the system. These operations include adding a new node, updating an existing node configuration, or removing a node from the system. After completion of operation, a confirmation is received, signifying the end of this workflow sequence.

Within Resource Management, the host select a node to manage the resources. The host can view the services, and processes used by the nodes.

For Network Configuration Management, the host selects a network configuration to adjust. This also involve modifying network parameters or settings to optimize performance or connectivity. Following the adjustments, a confirmation is provided, ending the network configuration workflow.

FUTURE RECOMMENDATIONS

As the performance of Distributed OS Manager improves, it is always necessary to consider future implementations and optimizations that can further boost the functionality, performance, and usability of application. This section provides the few recommendations aiming into highlighting important areas to improve the performance. Starting from improving security measures to optimizing performance and fostering community engagement, provided recommendations aims to launch the Distributed OS Manager towards greater effectiveness, scalability, and user satisfaction.

1. Enhanced Security Measures

To strengthen the security of application, it is very important to implement advanced authentication ideas and best encryption protocols. Adding the features such as two-factor authentication 2FA adds an extra security, reducing the risk of unauthorized access.

2. Integration with Monitoring and Alerting Systems

Enhancing the application with integration capabilities for monitoring and alerting systems enables users to keep an eye on how well their distributed systems are doing in real-time. By enabling alerts, users can easily respond to issues like system failures or reduction in performance, minimizing downtime and optimizing system reliability.

3. Enhanced User Experience (UX) Design

Continuously refining the user experience UX design considering the user feedback and is necessary. User friendly interfaces, clear navigation paths, and fluid layouts should be prioritized to ensure a seamless and user experience across different devices and platforms.

4. Long-Term Maintenance and Support
Establishing an in depth plan for long term maintenance is always mandatory for ensuring the success of the web application. Regular updates, bug fixes, and security patches should be provided to address evolving user needs and mitigate emerging threats. Along with the same, documentation, tutorials, and support resources should be available to users to assist them.

Result

Link: https://aos-project-4e82f.web.app

To illustrate the effectiveness and userfriendly design of Distributed OS Manager, the following pages gives a series of screenshots of the results and various functionalities which are experienced by the user while doing the operations.

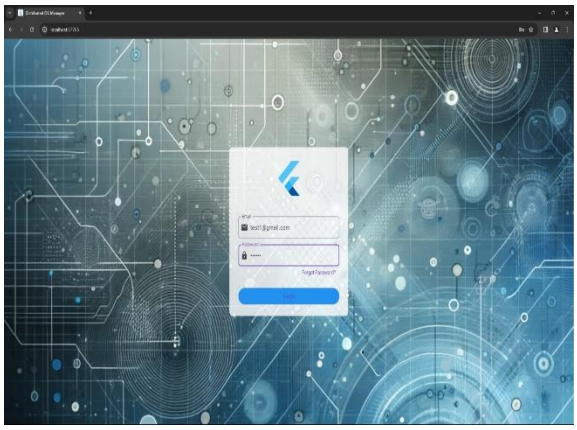

Figure 3: Login Page

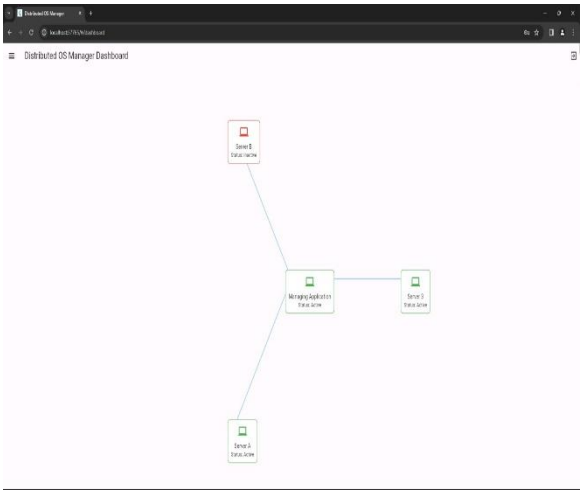

Figure 4: Dashboard

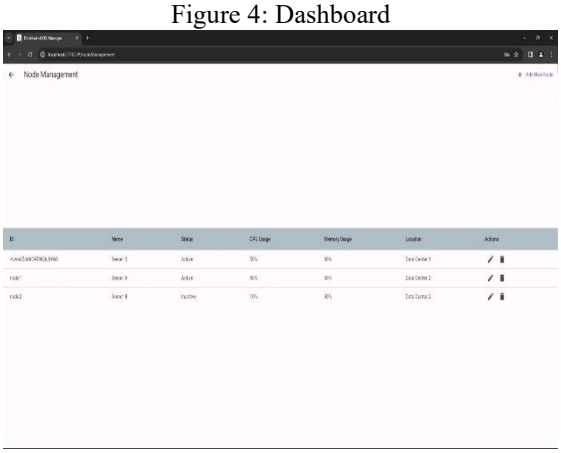

Figure 5: Node Management

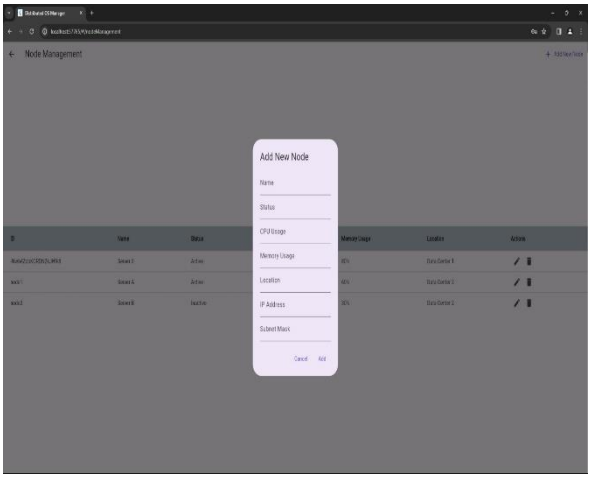

Figure 6: Node Management 2

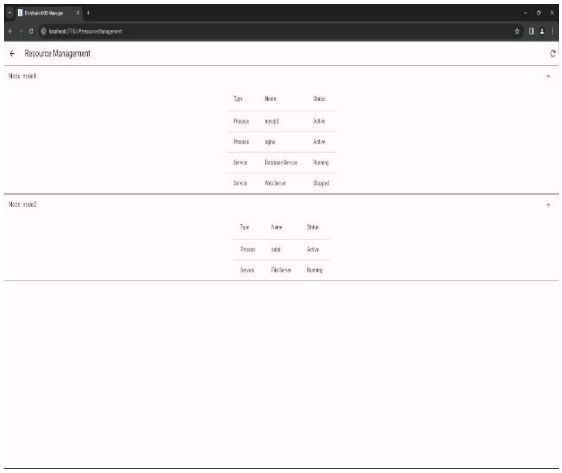

Figure 7: Resource Management

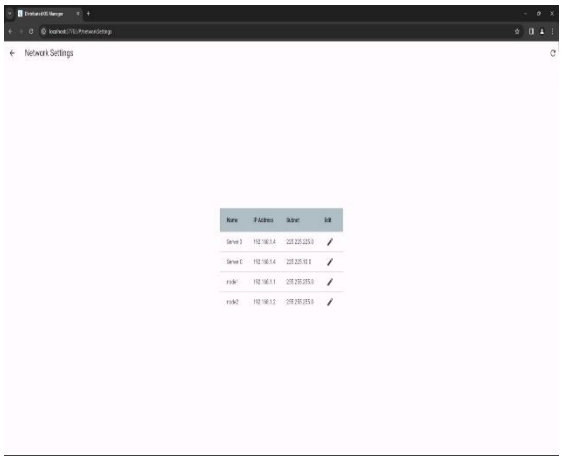

Figure 8: Network Settings

CONCLUSION

The project was successfully designed and implemented the management interface facilitates control over nodes, resources, and network configurations in a distributed operating system. The login system was created to ensure a secure user login, setting a strong security for the system. Following authentication, the dashboard provided users with access to various management functions, allowing different operation management.

Throughout the project, challenges such as ensuring real-time data synchronization across nodes and maintaining high security for data transactions were came across with productive solutions, which included the use of security layer and real-time monitoring tools. The project met its objectives but also provided an architecture that can adapt to a increased loads and future expansions in network size.