The Bombay Salesian Society's
Don Bosco Institute of Technology
Department of Information Technology

DeployX

Remote Command Execution System

Team members:

Parth Shikhare
Nischay Chavan
Chetan Chaudhari

Guide:

Prof. Vaishali K

Outline of Presentation

1. Introduction
2. Problem Statement
3. Scope of the Project
4. Current Scenario
5. Need for the Proposed System
6. Review of Literature
7. System architecture /Design
8. Technology Used
9. References
10. Conclusion along with GitHub Repository Link

Introduction

Introduction

- **Problem**: Tedious and complex software management across multiple systems (labs, training centers). Existing tools are often too technical or costly.
- **Solution**: DeployX – a centralized, lightweight client-server solution for automated software deployment.
- **Functionality**: Installs/uninstalls applications on multiple clients from one admin PC.
- **Key Feature**: LAN-only operation (no internet needed), ideal for secure networks.
- **Benefits**: User-friendly GUI, real-time tracking, reduced technical barriers, faster installations, improved consistency, strong administrative control.

Problem Statement

Problem Statement

- **Manual Overload**: System administrators are burdened with time-consuming, repetitive, and error-prone manual software installations and updates.
- **Inconsistent Systems**: This leads to fragmented software versions, causing compatibility issues and disrupting critical workflows.
- **Lack of Centralized Control**: Without a unified system, monitoring, tracking, and maintaining deployments is inefficient and difficult.
- **Impact on Productivity**: Ultimately, these challenges reduce IT management quality and hinder overall organizational productivity.

Scope of the project

Scope of the Project

- **Centralized Command Execution**: Remotely run shell commands from a web UI on multiple client systems.
- **Lightweight Client-Server**: Uses small Python agent communicating via WebSockets (HTTP Polling as a fallback) with a Node.js controller.
- **Real-time Feedback**: Get instant command output from each machine.
- **Flexible Deployment:** Operates over LAN or Internet (LAN preferred); Agents support Windows, Linux, & macOS clients, ideal for secure internal networks.
- **For IT Admins**: Simplifies bulk updates, installations, and diagnostics.

Current Scenario

Current Scenario

- **Manual Bottlenecks**: Most setups rely on slow, error-prone manual software installations and updates on each machine.
- **Enterprise Tool Limitations**: Popular solutions (Ansible, Puppet, Chef) are often expensive, require licensing, and lack intuitive graphical interfaces.
- **Scripting Gaps**: Custom scripting lacks centralized control, live feedback, and often struggles with cross-platform compatibility and high technical demands.
- **DevOps Overkill**: Tools like Jenkins and GitLab target cloud-native, microservices, and DevOps pipelines – not streamlined on-premise system management.
- **RMM Disconnect**: RMM platforms are typically built for hybrid corporate networks with larger budgets, lacking focus on offline or lab-specific environments.

Need of proposed system

Need of proposed solution

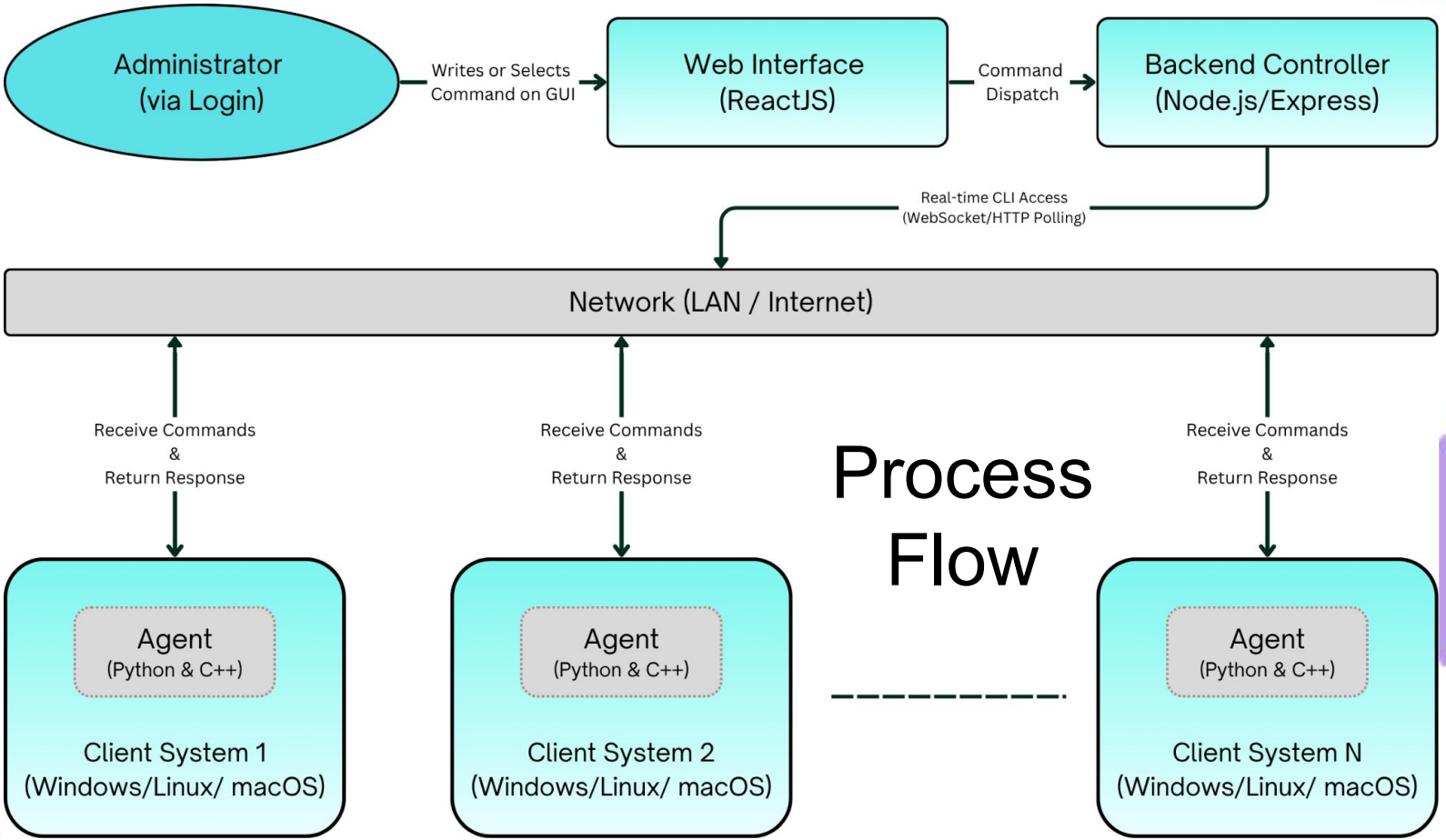
- **Automated Efficiency**: Eliminate tedious manual tasks, drastically reducing time and human error in software deployment and updates.
- **Accessible Central Control**: Provide an intuitive GUI-based centralized system that doesn't require deep technical expertise or expensive licenses.
- **Real-time Visibility & Control**: Offer live deployment status and output logs from each client for immediate feedback and robust error tracking.
- **Targeted & Secure Operation**: Deliver a purpose-built solution for LAN/Intranet environments, ensuring security for offline and sensitive networks.
- **Conflict Prevention**: Automatically scan existing applications to avoid deployment conflicts, ensuring system stability.
- **Cross-Platform Flexibility**: Support diverse environments with seamless management across Windows, Linux, and macOS clients.
- **Cost-Effective Alternative**: Offer a practical, budget-friendly solution specifically tailored for educational institutions, computer labs, and office environments, unlike expensive enterprise RMMs or complex DevOps tools.

Review of Literature

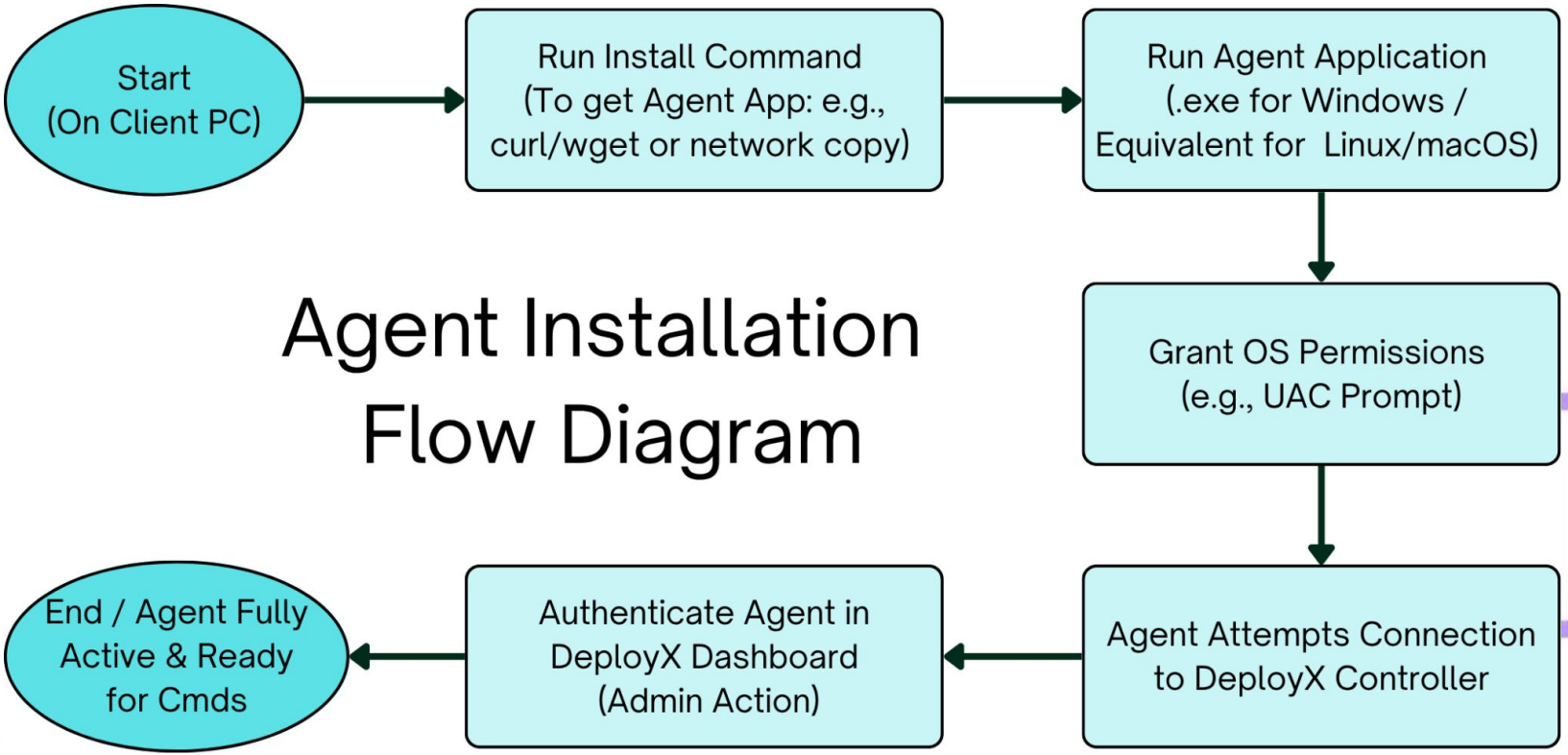
Review of Literature

- **Ansible / Puppet / Chef:** Popular DevOps tools for deployment automation, but require complex configuration and scripting and it is not suitable for non-technical users or offline LAN labs.
- **PDQ Deploy:** A GUI-based deployment tool, efficient for Windows environments. However, it is paid, Windows-only, and not ideal for multiple OS or budget-restricted setups.
- **Microsoft SCCM / Intune:** Enterprise-grade solutions with cloud integration, suitable for domain-managed systems. Too heavy and complex for small-scale, local networks.
- **PowerShell & Bash Scripts:** Used for custom deployments but require technical expertise, lack central control, and provide no feedback interface.
- **DeployX Solution:** Fills this gap by offering a lightweight, GUI-driven, real-time deployment system designed for educational and institutional needs.

System Flow Diagram


Administrator
(via Login)
Writes or Selects
Command on GUI
Web Interface
(ReactJS)
Command
Dispatch
Backend Controller
(Node.js/Express)
Real-time CLI Access
(WebSocket/HTTP Polling)
Network (LAN / Internet)
Receive Commands
&
Return Response
Receive Commands
&
Return Response
Receive Commands
&
Return Response
Process
Flow
Agent
(Python & C++)
Client System 1
(Windows/Linux/ macOS)
Agent
(Python & C++)
Client System 2
(Windows/Linux/ macOS)
Agent
(Python & C++)
Client System N
(Windows/Linux/ macOS)

Agent Installation Flow Diagram



Technology Used

Technology used

- **Frontend:**React.js, Tailwind CSS
- **Backend :** Node.js, Express.js
- **Agent:** Python/C++
- **Database:** MongoDB
- **Python libraries**:
  - **Subprocess:** Native, efficient for running commands and installers silently.
  - **APScheduler:** Robust, supports cron, intervals, date-based jobs, persistent scheduling.
  - **Zeroconf:** Simple, cross-platform LAN discovery without needing IP scans.
  - **Socket/Socket.io:** Lightweight TCP communication; scalable with event-based Socket.IO.
  - **Psutil:** The most efficient and cross-platform system monitoring library
  - **winreg/wmic:** Auto scan installed applications in device
  - **Loguru:** Simple formatting of logs and history.
  - **Shutil & tarfile:** Used for creating checkpoints and rollback.
  - **Plotly:** Interactive, clean reports.

References

References

- Documentation on Python Sockets: https://docs.python.org/3/library/socket.html
- PDQ Deploy Official Docs: https://www.pdq.com
- Ansible Documentation: https://docs.ansible.com
- React Documentation: https://react.dev/learn
- NodeJs Documentation: https://nodejs.org/docs/latest/api/
- ExpressJs Documentation: https://expressjs.com/
- MongoDB Documentation: https://www.mongodb.com/docs/

Conclusion

Conclusion

- **Solves Core Problems**: Eliminates manual inefficiencies, inconsistencies, and lack of control in IT deployments.
- **Key Advantage**: Offers an accessible, centralized, and secure solution tailored for LAN/Intranet environments.
- **Delivers Impact**: Drastically reduces time, improves consistency, and provides real-time oversight for IT administrators.
- **Future-Ready**: Clear roadmap for advanced features like authentication, dashboards, and broadcast capabilities.
- **Your Solution For**: Streamlined, secure, and efficient software management.

Thank You!

GitHub Repository Link:-

https://github.com/Nischay-loq/DeployX