WIPRO NGA Program – LSP Batch

Capstone Project Presentation – 04 May 2024

Project Title Here – LINUX SYSTEM METRICS DEVICE DRIVER PROJECT

Presented by – SRIHARI DORNADULA

LINUX SYSTEM METRICS DEVICE DRIVER PROJECT

Introduction to Linux System Metrics

- Linux is an open-source operating system that supports a wide range of hardware.
- System metrics provide valuable insights into performance, resource utilization, and system health.
- This project aims to develop a device driver that collects and reports various system metrics.

Objectives of the Project

- The main objective is to create a Linux device driver for metrics collection. collection.
- This driver will gather data on CPU, memory, and I/O usage.
- The project aims to enhance monitoring capabilities in Linux environments.

Importance of System Metrics

- System metrics are essential for diagnosing performance issues.
- They help in identifying resource bottlenecks and optimizing workloads.
- Metrics also play a critical role in system planning and capacity management.

Overview of Device Drivers

- A device driver acts as a bridge between the hardware and the operating system.
- Linux supports a variety of device drivers, including character, block, and network drivers.
- The device driver for this project will be a character driver to interface with user space.

Metrics to be Collected

- Key metrics include CPU usage, memory consumption, and disk I/O statistics.
- Additional metrics may include network traffic and system load averages.
- The driver will periodically collect and store these metrics for analysis.

Design Considerations

- The design must ensure minimal overhead on system performance.
- It should also provide accurate and timely data collection.
- Scalability and compatibility with various kernel versions are essential.

Development Environment

- The project will utilize a Linux development environment, including GCC and Make.
- Kernel headers and development libraries will be necessary for building the driver.
- Testing will be conducted on various distributions to ensure compatibility.

Driver Implementation

- The implementation will follow the Linux kernel module framework.
- Functions will be defined for module initialization and cleanup.
- The driver will use system calls to access hardware metrics and user space.

Data Collection Mechanism

- Data will be collected through kernel timers and interrupts.
- The driver will implement polling mechanisms for periodic data updates.
- Collected data will be stored in a buffer for user-space access.

User Space Interaction

- A user-space utility will be developed to interface with the driver.
- This utility will read and display the collected metrics.
- It will also allow users to configure data collection intervals.

Testing and Validation

- Rigorous testing will be performed to ensure driver reliability and accuracy.
- Unit tests will validate individual functions within the driver.
- Integration tests will confirm the interaction between the driver and user-space tools.

Challenges and Solutions

- Common challenges include managing kernel memory and ensuring data integrity.
- Solutions may involve using appropriate locking mechanisms to prevent data races.
- Error handling will also be implemented to gracefully manage failures.

Future Enhancements

- Future work could include expanding the range of metrics collected.
- Enhancements may also focus on improving the user interface for the utility.
- Support for real-time data streaming could be explored for advanced monitoring.

Conclusion

- The Linux System Metrics Device Driver Project aims to enhance system monitoring.
- By providing valuable insights into system performance, it benefits system administrators.
- Continuous improvement and community feedback will drive future development.