**TITLE : Real time os security event logger**

INTRODUCTION TO OPERATING SYSTEM :

**1.Introduction**

An Operating System (OS) is system software that serves as a middleman between computer hardware and users. It offers an environment for running applications and handling hardware resources effectively. The main objective of an operating system is to make computing easy for users and utilize the hardware effectively.

**2.History of Operating Systems**

Operating systems have undergone a remarkable transformation over the years. Early computers functioned without an OS, and the users had to communicate with the hardware directly in machine language. During the 1950s and 1960s, batch processing systems came into existence, and several jobs were run one after another. Time-sharing systems, which were introduced in the 1960s, allowed multiple users to communicate with one system at a time. As microcomputers emerged in the 1980s, personal computer operating systems including MS-DOS, Windows, and macOS started dominating the market. Now, operating systems drive everything from smartphones to supercomputers.

**3.Functions of an Operating System**

An operating system performs several critical functions, including:

**3.1 Process Management**

•OS manages the creation, execution, and deletion of processes.  
•It schedules the CPU to manage multiple processes effectively.  
•It provides synchronization and communication among processes.

**3.2 Memory Management**•OS assigns and releases memory space to processes whenever necessary.  
•It employs methods such as paging and segmentation for effective memory usage.  
•Virtual memory enables programs to run even when RAM is not adequate.

**3.3 File System Management**

**•**The OS stores, retrieves, organizes, and manages files on storage devices.

•It offers access control and permissions to provide data security.

•Some common file systems are NTFS, FAT, and ext4.

**3.4 Device Management**

**•**The OS manages hardware devices using drivers.

•It facilitates smooth communication between peripherals such as printers, keyboards, and disk drives and the system.

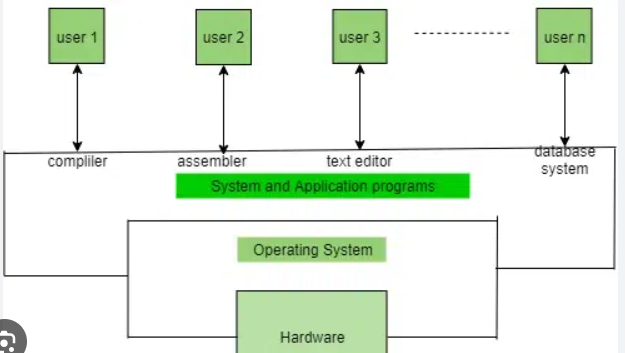
•Device queues and buffering enhance performance.

* 3.5 User Interface: For user interaction, operating systems offer graphical user interfaces (GUI) and command-line interfaces (CLI).   
  • The Linux terminal, macOS GUI, and Windows GUI are a few examples.   
  4. Operating System Types   
  Operating systems can be divided into various groups according to their features and applications.   
  4.1 Batch Operating System: Used in early mainframe computers, this system progressively completes a batch of tasks without requiring user input.   
  For instance, IBM OS/360.   
  4.2 Time-Sharing Operating System: Enables simultaneous usage of system resources by several users.   
  • Uses scheduling methods to guarantee equitable CPU distribution.   
  UNIX is one example.

4.3 Operating System for Distributed Systems   
• Oversees a collection of computers that are linked together via a network.   
• For efficiency, processing is divided among several machines.   
Amoeba and Windows Server are two examples.   
4.4 Operating System in Real Time (RTOS)   
• Minimizes latency in real-time data processing.   
• Applied to industrial automation, robotics, and embedded systems.   
• VxWorks and FreeRTOS are two examples.   
4.5 Network Operating System (NOS): Offers networking features for network computer management.   
• Facilitates network security, file sharing, and remote access.   
Novell NetWare and Windows Server are two examples.

4.6 Operating System for Mobile Devices   
• Designed for tablets and smartphones; • Enhanced for power economy and touch interactions.   
• For instance, iOS and Android.   
5. Well-known operating systems   
Numerous platforms make extensive use of a number of operating systems:   
• Windows: Created by Microsoft, this operating system is well-known for its intuitive design and extensive use.   
• Linux: An open-source operating system renowned for its adaptability, security, and personalization.   
• macOS: Apple's in-house operating system, renowned for its seamless user interface and compatibility with Apple devices.   
• Unix: A workstation and server operating system with many users.   
• Android: Google's Linux-based mobile operating system.   
• iOS: Performance and security are hallmarks of Apple's mobile operating system.

* 6. Conclusion   
  Operating systems facilitate smooth communication between people and hardware, making them the foundation of contemporary computing. Operating systems are getting increasingly complex as a result of ongoing developments, integrating cloud computing, artificial intelligence, and improved security measures. Both IT workers and computer science students must comprehend the foundations of operating systems.





7. INTRODUCTION TO PROJECT :

**Real-Time OS Security Event Logger**

Creating a Page Replacement Algorithm Simulator that illustrates the effectiveness of different page replacement techniques used in memory management is the aim of this project. The simulator offers an interactive method for evaluating and contrasting various algorithms according to metrics like memory utilization efficiency and page fault rate. The goal of the research is to improve knowledge of operating system memory management strategies and how they affect system performance.

**7.1. Introduction**

A program called a Real-Time OS Security Event Logger is made to track, document, and examine security-related occurrences within an operating system. This logger's main objectives are to offer comprehensive logs for analysis, identify any security risks, and track system activity in real-time. Because operating systems must effectively manage illegal access, privilege changes, and suspicious activity to prevent security breaches, this is essential for preserving system integrity.

.

**7.2 Understanding Real-Time OS Security Event Logging**

### Malicious activity, privilege escalation, and illegal access can all pose security risks to an operating system. In order to identify and stop such breaches, a real-time OS security event logger keeps an eye on system events and records important security-related activity. It enables administrators to react quickly by assisting in the identification of questionable activity, such as unsuccessful login attempts or unauthorized alterations. By examining event trends and warning users of irregularities, the Real-Time OS Security Event Logger offers insights into system security. This improves comprehensive security management and guarantees improved protection of system resources.

### ****7.3 Features of a Real-Time OS Security Event Logger****

• User Interface: A CLI or graphical user interface for tracking security events.   
• Real-time monitoring: Keeps tabs on file alterations, privilege changes, and login attempts.  
• Event logging: Keeps track of security-related actions and provides auditing timestamps.   
  
• Alert System: Notifies users of questionable activity.   
• Performance analysis: identifies trends and evaluates weaknesses in the system.

**7.4. Objectives of the Project**

### • Real-Time Monitoring: Constantly monitor and record operating system security-related events. • Threat Detection: Spot unusual activity, privilege escalations, and illegal access attempts. • Effective Logging: Keep thorough records of security events for analysis and auditing purposes. • User Alerts: Instantaneously alert admins to any security risks. • Data analysis: Offer perceptions on security patterns and weaknesses. • Better Security Management: By facilitating prompt threat response, improve system security.

### ****7.5 Literature Review****

#### ****7.5.1 Background of Real-Time OS Security Event Logging****

#### • For real-time threat detection and system activity monitoring, security event logging is crucial. • To maintain system integrity, an effective logging system aids in detecting malicious activity, privilege changes, and unauthorized access. • Instant threat identification and reaction are made possible by real-time monitoring, which improves security.

#### ****7.5.2 Existing Work and Methods****

• There are several security event logging programs available, such as Windows Event Viewer, Syslog, and SIEM (Security Information and Event Management) programs.   
• Although it takes a lot of processing power, research indicates that real-time monitoring combined with AI-driven anomaly detection enhances threat identification.  
• Linux's auditd and other lightweight logging technologies strike a compromise between system overhead and performance.

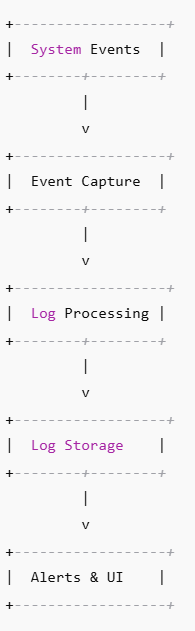
### ****7.6 System Design and Implementation****

#### ****7.6.1 System Requirements****

**• Hardware: o 64-bit processor, at least 4GB of RAM.  
Enough room to store security records and event logs.   
  
• Software: Python, C++, or Java are examples of programming languages.   
o Logging Frameworks: Windows Event Viewer, Syslog, or Linux auditd.   
o Database: Security logs are stored in SQLite/MySQL.   
o Analysis & Visualization: Matplotlib and Pandas are used to display logs graphically.   
For an interactive interface, use the Tkinter/PyQt GUI libraries.   
• Operating System: o Cross-platform compatibility with Windows, Linux, and macOS.**

**7.6.2 Architecture of the Simulator**

**Flowchart of the Simulator Architecture:**



### ****7.6.3 Implementation of Real-Time OS Security Event Logger****

#### ****6.3.1 Event Logging using FIFO (First-In-First-Out)****

When the log storage reaches its limit, the oldest logged event is deleted first. This is straightforward yet ineffective because it ignores the frequency or severity of the incident.  
  
  
• Log Storage Diagram:   
[First Event] [Event 2] [Event 3]   
Arrival of Event 4 -> Removal of the Oldest (Event 1) -> [Event 2] [Event 3] [Event 4]

**6.3.2 Least Recently Used (LRU)**

Gets rid of the log entry that hasn't been viewed in a long time.   
 More effective than FIFO, although log access timestamps must be tracked.

* **Diagram:**

Access History:

[Login] [File Access] [Process Start]

New Event (Network Activity) -> Remove least recently accessed -> [File Access] [Process Start] [Network Activity]

**6.3.3 Optimal Logging (Predictive Log Management)**

* Removes the log entry that will be needed the farthest in the future.
* Most efficient but requires prior knowledge of security event patterns.
* **Diagram:**

Future Reference:

[Failed Login] [File Access] [Network Alert]

New Event arrives -> Remove log that will be least relevant in future

**6.3.4 Least Frequently Used (LFU) Logging**

* Removes the log entry that occurs the least number of times.
* Works well for repetitive security threats but may retain outdated logs.
* **Diagram:**

Log Frequency:

[Login (1x)] [File Access (2x)] [Malware Alert (3x)]

New Event Arrives -> Remove least frequent log (Login)

### 6.3.5 ****Clock (Second Chance) Algorithm for Logging****

* An improvement over FIFO that prioritizes logs with higher importance.
* Uses a circular queue to efficiently manage logs.
* If an event has a **"high priority"** flag, it gets a second chance before removal.
* **Diagram:**

Log Storage with Priority:

[Event1] (flag=1) [Event2] (flag=0) [Event3] (flag=1)

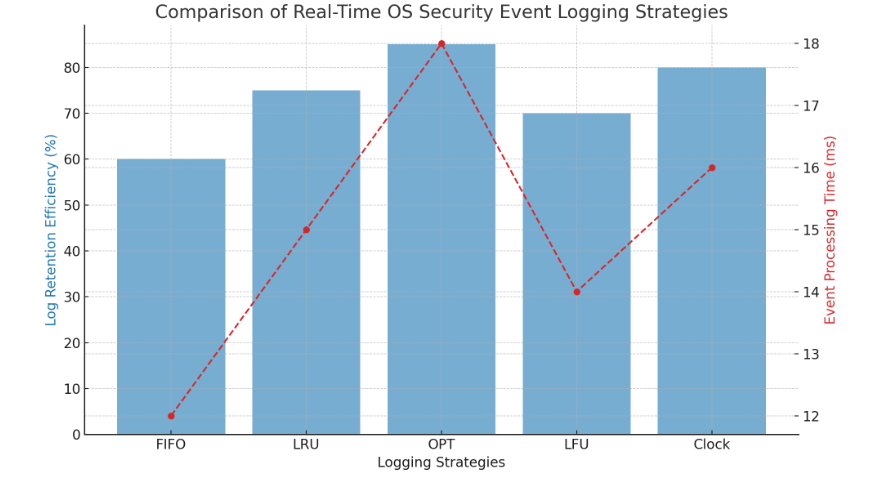
Clock hand moves -> Removes [Event2] (flag=0)

**7.7. Simulation Results and Analysis**

 **Input:** User provides security event logs and storage capacity.

 **Processing:** The system applies logging strategies to decide which logs to retain or discard.

* **Output:** Log retention efficiency, alert accuracy, and storage optimization.
* **Graphical Representation:**
* Comparison of Security Event Log Retention Efficiency and Processing Time



**7.8. Challenges Faced & Solutions**

**7.8.1 Challenges**

* **High Log Volume:** Managing a large number of security events in real time without data loss.
* **Efficient Storage:** Ensuring critical logs are retained while avoiding unnecessary data accumulation.
* **Real-Time Processing:** Detecting and analyzing security threats without causing system slowdowns.
* **Visualization Complexity:** Displaying log trends and security breaches effectively.

**7.8.2 Solutions**

 **Improved Data Handling: For speedy access, effective data structures like queues and hash maps were used.**• Storage Management: To strike a balance between storage and accessibility, log rotation and compression techniques were put into practice.   
• Parallel Processing: To process security events in real time without system lag, multithreading was used.   
• Graphical Representation: Lightweight libraries were used to visualize security alerts in real time.

**7.9. Future Scope**

The term AI-Driven Threat Detection refers to the application of machine learning models to predict and prevent security breaches.   
• Cloud Integration: Enhancing the monitoring of security events for cloud-based and distributed operating systems.   
Automated incident response is the process of developing self-healing systems to counter threats in real time.   
  
• Improved Data Visualization: To improve security incident analysis, 3D graphical representations are used.

**7.10. Applications of the Security Event Logger**

Cybersecurity monitoring: Assists security analysts in tracking and addressing OS threats in real time.   
• Compliance auditing: By keeping a thorough record of system events, this method makes sure businesses adhere to security regulations.  
• Forensics & Investigation: Aids in locating system weaknesses and illegal access.   
  
• Operating System Development: By examining current security concerns, it helps create safe OS architectures.

**7.11.NEED FOR THIS :**

To solve important issues with system security and event monitoring, the Real-Time OS Security Event Logger was created. This is why it is crucial:   
1. Real-Time Threat Detection: Conventional logging techniques increase system risk by failing to identify security breaches immediately.  
2. Effective Log Management: Without adequate optimization, high security event volumes may exceed processing and storage capacity.   
  
3. Absence of Centralized Monitoring: For improved analysis, security incidents involving several system components require a single logging system.   
4. Compliance & Auditing Requirements: In order to comply with security and regulatory requirements, organizations must maintain accurate event logs.   
5. Automated Response & Forensics: Assists security experts in examining previous events, identifying trends in attacks, and enhancing security protocols going forward.   
This guarantees effective threat management, real-time monitoring, and improved system security.

**7.12. Conclusion**

Because it offers threat identification, effective log management, and real-time monitoring, the Real-Time OS Security Event Logger is essential to contemporary system security. By guaranteeing that security lapses, illegal access, and irregularities are quickly identified and recorded for additional examination, it improves system integrity.   
Through the integration of centralized log management, automated response mechanisms, and compliance tracking, this solution assists enterprises in meeting regulatory requirements, enhancing forensic analysis, and fortifying cybersecurity. Its capabilities will be further enhanced by future developments like cloud-based log management and AI-driven threat prediction, making it a crucial tool for protecting real-time operating systems.  
In order to safeguard digital infrastructures and guarantee the dependability and security of contemporary computing environments, this research highlights the significance of proactive security event tracking.