Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 9s (2025)

Sentinel Os: Strengthening Operating System Security with Comprehensive Hardening Measures

Dr. Shailaja Uke[1], Dr. Nilesh Uke[2], Archit Kothawade[1], Maitrey Chitale[1], Sharvari Deshmukh[1], Atharva Deshpande[*1]

1- Department of Computer Engineering, Vishwakarma Institute of Technology, Pune, 411037, Maharashtra, India.

2- Principal, Indira College of Engineering and Managment, Pune, 410506, Maharashtra, India

Email- shailaja.uke@vit.edu, nilesh.uke@gmail.com, vishwajit.archit21@vit.edu, maitrey.chitale21@vit.edu, sharwari.deshmukh211@vit.edu, atharva.deshpande212@vit.edu

Article History:

Received: *12-11-2024*

Revised:*24-12-2024*

Accepted:*09-01-2025*

Abstract:
In today's era of escalating cyber threats and pervasive digitization, cybersecurity is essential. India faces a surge in cyberattacks, affecting both government and private sectors. SentinelOS addresses the dynamic threat landscape, remote work challenges, regulatory compliance, supply chain vulnerabilities, and data breach costs. Its deep learning capabilities and robust intrusion detection systems counter sophisticated attacks. SentinelOS ensures business continuity, global awareness, and protection for education, healthcare, and e-commerce sectors. With an intuitive GUI, custom firewall, SNORT integration, and advanced intrusion detection, SentinelOS fortifies organizational cybersecurity, adhering to industry standards and providing scalability and user customization.

Keywords: Snort, Lynis, Ubuntu, Flutter, LUKS.

1. Introduction

In the modern world where the use of digital technologies is on the rise and cyber threats are on the rise as well, it is crucial to have security. India has seen a steady increase in its internet users and has thus witnessed a rising trend of cyber-criminal activities affecting both the government and private sectors. In 2023, Defence Ministry of India decided to move to a more secure Maya Os [1] in order to mitigate from attacks. These attacks result in loss of funds, loss of data and personal information which emphasize the need for strong cybersecurity. To this end, SentinelOS has been designed as an all-encompassing and flexible cybersecurity solution. It is designed to offer high levels of protection against the ever-increasing threats while at the same time being able to integrate with other systems, be easy to use and meet the required industry standards. SentinelOS helps in meeting the most prominent drivers of cybersecurity which include the dynamic and complex threat environment, increased GDPR risk and environment HIPAA. Due it and to avoiding also remote business helps work, disruption in and which addressing sectoral the are supply requirements costly. increasing chain in In compliance risks, education, defenses. Some addition, requirements the healthcare, of the such financial e-commerce, the SentinelOS as impacts and enhanced learning-based is the of government features intrusion developed breaches, to that detection to match are system audit meet the incorporated using done global into the SNORT, by efforts the file Lynis. in project

integrity enhancing include monitoring It cybersecurity a through also custom AIDE, has firewall, and a a a graphical deep regular user system interface making it easy for individuals who are not conversant with technical terms to use it. SentinelOS is highly adaptable and can grow with your business, meaning that organizations can start small and gradually add more security features as they require. Also, it has strong OS hardening options and security settings that can be easily configured depending on the organization's needs.

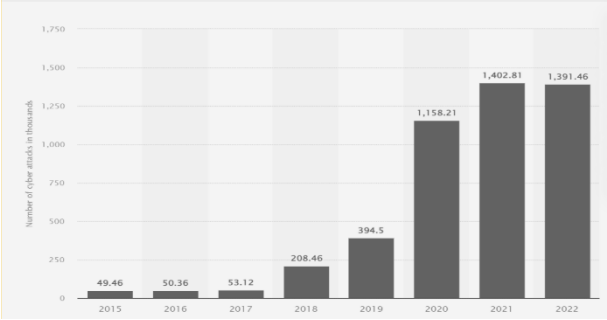


Fig. 1- Cyber Attacks Over the time

2. Literature Survey

Following were the papers that we referred as a part of literature survey.

Paper [2] investigates the radiation reliability of two real-time operating systems (RTOS) designed for dependability: eCos and dOSEK. Both systems were tested under neutron radiation, demonstrating significant reductions in failure rates with their respective fault tolerance mechanisms, such as error detection and recovery techniques. In paper [3] describes a method for creating a hardened Debian Linux virtual machine image for the Azure cloud, adhering to specific security requirements. The research involved analyzing existing standards, developing a tailored hardening standard for Debian, and utilizing tools like Packer and Chef to automate the image creation process, ensuring compliance with security best practices. Paper [4] analyzes the impact of operating systems on the reliability of safety-critical systems. The study focuses on identifying and mitigating soft errors in the FreeRTOS operating system, proposing techniques like redundancy and error correction mechanisms to enhance its robustness and prevent system failures in critical applications.Paper [5] introduces a system hardening architecture that integrates multiple security functions into a single module, creating a layered defense system. This approach aims to increase the difficulty for attackers by requiring them to breach multiple layers of protection before accessing critical data, thus enhancing the overall security posture of the system. Book [6] provides a comprehensive guide to hardening and securing Linux hosts and common applications. It offers practical advice on enhancing security measures, such as firewall configuration and securing network connections, while emphasizing the importance of minimizing risks without significantly impacting user functionality, ensuring a balance between security and usability. Paper [7] proposes an automated approach to correcting misconfigurations in operating systems. By generating capability dependency graphs and utilizing MaxSAT solving, the method aims to identify and address security vulnerabilities while minimizing the impact on system usability, allowing for

more efficient and effective security policy management. Paper [8] explores the use of secure enclaves to enhance the security of web servers. By leveraging technologies like Intel SGX, the study demonstrates how to protect sensitive data, such as private keys, from both the operating system and the hypervisor, providing a more secure environment for critical web server operations. Paper [9] examines the implementation of system hardening on Windows systems using industry-standard Center for Internet Security (CIS) controls. The case study demonstrates the effectiveness of these controls in improving security posture and provides a practical approach to applying and assessing the impact of these measures, enabling organizations to maintain a high level of security across their Windows infrastructure. Paper [10] challenges the notion that kernel hardening significantly impacts performance. It introduces the SPLIT KERNEL design, allowing selective runtime enablement of hardening mechanisms. This approach minimizes performance overhead in normal operation while providing enhanced security when needed. The study demonstrates the practicality and effectiveness of this design through real-world workload evaluations. Paper [11] investigates vulnerabilities in hardened Windows 10 systems, particularly those arising from insufficient security configurability and misconfiguration. The paper highlights the importance of proper security measures to protect sensitive data, emphasizing the need for continuous assessment and improvement of security configurations. Paper [12] proposes a kernel hardening function to enhance the stability of the Linux kernel. By modifying the panic() function within the kernel code, the proposed mechanism aims to recover from incorrect address values within the kernel stack frame, thereby preventing system crashes. Experimental results demonstrate the effectiveness of this approach in restoring normal system operation after induced panics. Paper [13] analyzes attacks leveraging Operating System Fingerprinting within a university campus environment. The paper emphasizes the importance of modifying TCP/IP features to minimize the information revealed about the host's identity and thwart known fingerprinting tools. It suggests future developments, such as the design of TCP Wrapper, to enhance security against these types of attacks. Paper [14] demonstrates the feasibility of automating the process of hardening a Linux operating system using open-source tools like Anaconda and OpenSCAP. The study successfully deployed a securely configured CentOS system, providing a valuable reference for setting up a fully hardened virtualization environment. While some manual intervention was required, the study highlights the effectiveness of automated tools in achieving a high level of security. Paper [15] investigates and emphasizes the fundamental security configurations necessary to strengthen the security stance of a default Linux installation. The paper highlights the vulnerability of default Linux installations to various threats and provides industry best practices for hardening Linux servers, including securing common services like SSH and Apache, and utilizing intrusion detection tools like OSSEC. Paper [16] presents a design for a kernel hardening module aimed at mitigating recoverable errors within the Linux operating system. The proposed module integrates into the ASSERT macro to identify and handle recoverable errors, preventing system halts and improving system stability. Paper [17] introduces the Amaranth project, which aims to develop a secure UEFI firmware for virtual machines. The project focuses on implementing security hardening techniques, including an operating system integrity checking mechanism and firmware size

reduction, to address contemporary security issues related to UEFI firmware. In paper [18], the authors have created a deep learning model to analyse moving frames. The algorithm produces high accuracy. Paper [19], proposes a novel real-time video-based hand gesture recognition model that uses video normalization, a modified spatiotemporal feature extraction approach, and soft computing algorithms for recognition. Paper [20], develops a computer vision-based system for hand gesture recognition to aid communication for individuals with hearing impairments, comparing image classification algorithms and implementing a real-time video recognition model for common signs.

3. Methods

In this section, we will talk about tools and technologies used. Below is the proposed system architecture:

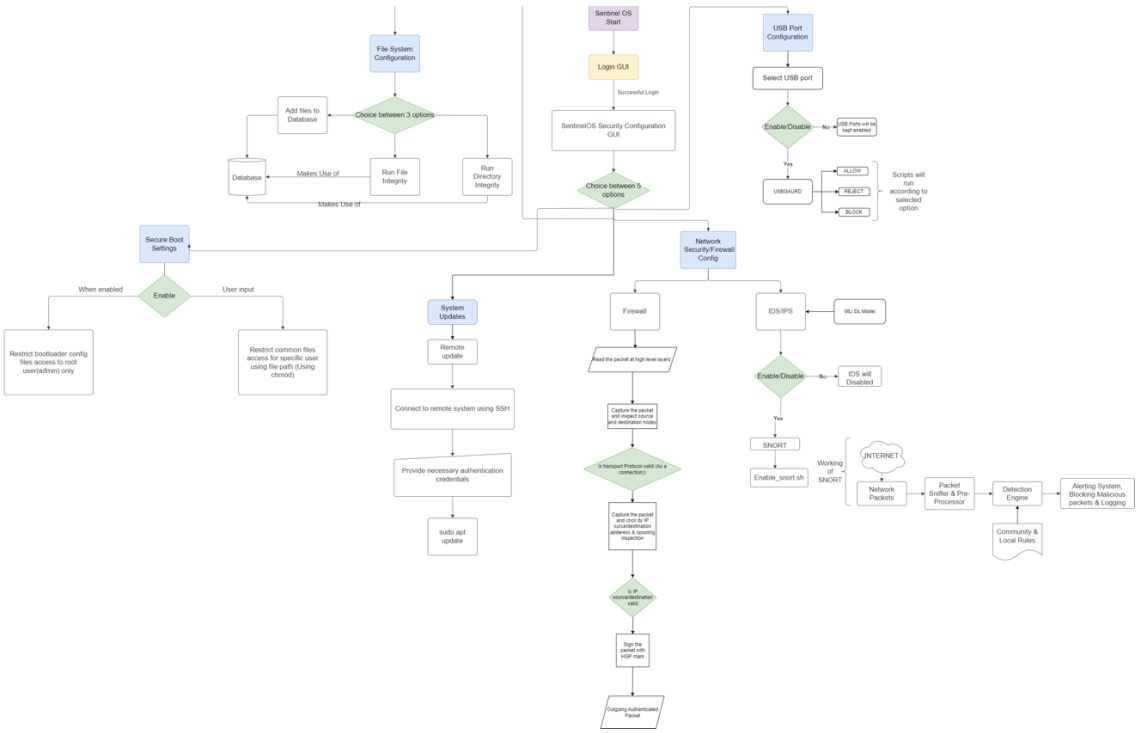


Fig.2 – System Architecture

3. 1 Snort

Snort is an open-source network intrusion detection and prevention system (IDS/IPS) that provides real-time traffic analysis and packet logging capabilities. The system can be installed on Ubuntu through a series of steps, including package repository updates, dependency installation, and configuration of rule sets. Snort operates in promiscuous mode, allowing it to intercept and analyze all network packets regardless of their destination address. Its core functionality includes packet sniffing, signature-based detection, anomaly-based detection, logging and reporting, and real-time alerting capabilities.

Snort utilizes a rule-based system for detecting and responding to network threats. These rules follow a specific syntax that defines conditions for detection and response, including header

fields, options, and actions. For example, rules can be created to detect ICMP Echo Requests, monitor HTTP traffic, or block SSH brute force attempts. When compared to other IDS/IPS solutions like Suricata and Bro (Zeek), Snort stands out for its widespread adoption, though Suricata offers better performance through multithreading capabilities.

To improve the effectiveness of Intrusion Detection and Prevention Systems (IDS/IPS), we implemented Artificial Neural Networks (ANNs). We used the NSL-KDD dataset, which contains 41 features for each network connection and categorizes traffic into five types: Normal, Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R), and Probing. The ANN model has an input layer with 41 neurons, representing the dataset features. It also includes one or more hidden layers with varying numbers of neurons, using the ReLU activation function. The output layer has five neurons, using softmax activation for classification. Before training, we preprocessed the data using normalization and one-hot encoding. We trained the model using the backpropagation algorithm and evaluated its performance using metrics such as accuracy, precision, recall, and F1-score. The overall metrics are 97.5%, 96.7%, 95.9%, and 96.3% respectively.

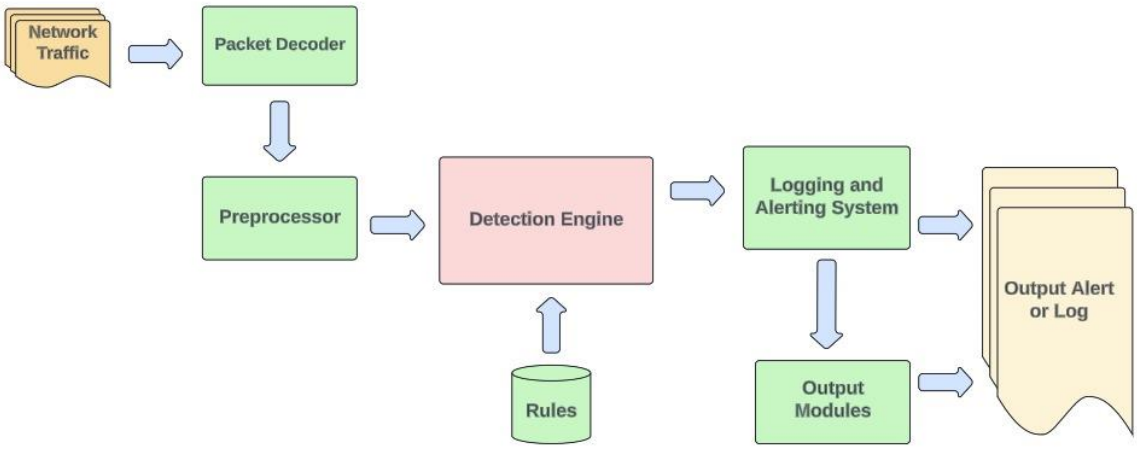


Fig. 3- Working of Snort

3.2 AIDE

AIDE (Advanced Intrusion Detection Environment) is a very capable host-based intrusion detection system for monitoring file and directory changes on UNIX systems. It functions by creating cryptographic hashes (those of rmd160, md5, tiger, crc32, sha1, sha256, sha512) of file attributes and storing them into a database, which is then to be compared in periodic scans. Installation is quite simple by installing a package and initializing the database with some very simple commands. Where it shines is in detecting unauthorized changes as a result of initiating scans of systems and comparing versus baseline configurations; it has therefore become an essential tool for monitoring system integrity.

3.3 LUKS

Linux Unified Key Setup (LUKS) is the standard for Linux full disk encryption. It works at the block level using symmetric cryptography with a master-key system to protect data. It formats partitions with LUKS encryption, generates key slots, and operates on encrypted volumes using the cryptsetup utility. LUKS is appealing because it is an open standard, offers excellent key management features with support for various key slots, and does not significantly reduce performance while still providing strong security using efficient encryption ciphers such as AES.

3.4 Lynis

Lynis is a complete security auditing tool for Unix-based systems, with advanced system scanning capabilities to detect security vulnerabilities, configuration problems, and compliance issues. The software assesses a variety of system facets, including configuration files, installed packages, user accounts, and network settings, comparing them against security best practices. It is lightweight, boasts a comprehensive report function, and presents options for customized testing, ideal for systems administrators who must constantly maintain a rigorous security stance.

3.5 Firewall

An overview of two major firewall administration tools, UFW (Uncomplicated Firewall) and iptables. UFW is designed to work as a front-end to the iptables rules that allow users to execute commands easily and manage firewall rules with application profiles. Iptables has a chain of rules allowing more refined control over packet filtering and network address translation. Although both tools allow management of network traffic, they suffice different reported needs—UFW suited towards users who want something simple, and iptables aimed at advanced control and detailed traffic manipulation by experienced administrators.

3.6 Summary of Flutter, Dart, and FFI Integration

3.6.1 Overview

Flutter, Google's open-source UI development kit, paired with the Dart programming language, provides a comprehensive framework for cross-platform application development. The installation process involves downloading the Flutter SDK, which includes Dart, setting up environment variables, and validating the installation using the flutter doctor command. This integrated development environment enables developers to create applications that can target multiple platforms while maintaining a single codebase, significantly reducing development overhead and time-to-market for applications.

3.6.2 Core Components and Their Integration

The framework's architecture revolves around three primary components working in harmony. Flutter provides the UI framework with its reactive approach and rich widget ecosystem, allowing developers to create sophisticated user interfaces through both pre-built and custom widgets. The Hot Reload feature significantly enhances the development workflow by enabling real-time code changes visualization. Dart, as the underlying programming language, brings strong typing, asynchronous programming capabilities, and efficient compilation through both

JIT (Just-In-Time) for development and AOT (Ahead-Of-Time) for production builds. The FFI (Foreign Function Interface) serves as a crucial bridge between Dart code and native C/C++ implementations, enabling seamless integration with platform-specific APIs and existing native libraries.

3.6.3 Shell Script Management and Language Comparison

In the context of shell script management, the Dart/Flutter ecosystem offers several advantages over traditional solutions like Python. The strongly typed nature of Dart provides enhanced code safety and maintainability through compile-time error detection. The language's built-in asynchronous programming support, utilizing async/await syntax, enables efficient handling of I/O-bound operations and compute-intensive tasks. The Flutter Plugin API allows direct access to platform-specific functionality, while the FFI capability makes it easy to interface with existing C/C++ code bases, which could be beneficial for systems programming and performance-critical operations.

3.6.4 FFI Implementation and Benefits

The implementation of FFI in Dart is one of the most sophisticated methods of integrating native code into Dart programs. The entire process involves defining C functions, creating Dart FFI bindings, and loading native libraries at runtime. Such an approach brings several benefits to using the FFI framework over traditional systems:

- **Performance Optimizations**: Direct C function calls minimize overhead and improve execution speed compared to other interpreted solutions.
- **Platform Independence**: Behavior is platform-independent due to the FFI implementation, although one can cite specific optimizations where appropriate.
- **Code Integration**: The fact that there is no need to rewrite the entire application allows for a gradual transition while maximizing the resource.
- **Development Flexibility**: Combining high-level Dart features with low-level C access offers developers the best of both worlds in terms of development efficiency and system-level control.

4. Results

Through deep learning-powered Snort, SentinelOS drastically improved cybersecurity management and introduced an advanced mechanism for intrusion detection. AIDE was deployed for file integrity monitoring, and Lynis provided a detailed audit of the entire system. Therefore, the user-friendly GUI developed with Dart/Flutter was able to target users at different technical levels. Since testing has been done to confirm the scalability capacity of toe-the-line treatment in normal-play, the performance of the tool has not been degraded just as a result of loading. All the above create a very strong and versatile security tool in SentinelOS.

5. Discussion

Implementation of SentinelOS offers the following key benefits: increased security, user convenience, scope for customizing, and abiding compliance regimes. The tools and interface of the system can be easily used by both technical and non-technical users in controlling cyber

security settings. The other advantages include these features being adjustable, offering scale, and being privacy and compliance friendly for organizations irrespective of their size, catering to a wide variety of industry use-cases.

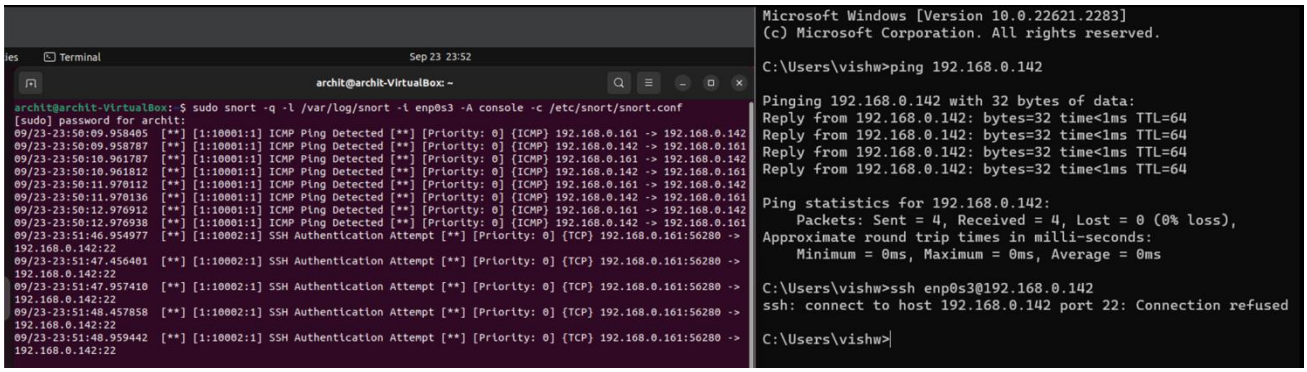


Fig. 4- Working of Snort as an IPS.

```
The following additional packages will be installed:
  vim-common vim-runtime vim-tiny
Suggested packages:
  ctags vim-doc vim-scripts indent
The following NEW packages will be installed:
  vim vim-runtime
The following packages will be upgraded:
  vim-common vim-tiny
2 upgraded, 2 newly installed, 0 to remove and 214 not upgraded.
Need to get 8,570 kB/9,361 kB of archives.
After this operation, 37.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim-runtime all 2:8.2.3995-1ubuntu2.15 [6,835 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim amd64 2:8.2.3995-1ubuntu2.15 [1,735 kB]
Fetched 8,570 kB in 3s (2,886 kB/s)
(Reading database ... 200008 files and directories currently installed.)
Preparing to unpack .../vim-tiny_2%3a8.2.3995-1ubuntu2.15_amd64.deb ...
Unpacking vim-tiny (2:8.2.3995-1ubuntu2.15) over (2:8.2.3995-1ubuntu2.10) ...
Preparing to unpack .../vim-common_2%3a8.2.3995-1ubuntu2.15_all.deb ...
Unpacking vim-common (2:8.2.3995-1ubuntu2.15) over (2:8.2.3995-1ubuntu2.10) ...
Selecting previously unselected package vim-runtime.
Preparing to unpack .../vim-runtime_2%3a8.2.3995-1ubuntu2.15_all.deb ...
Adding 'diversion of /usr/share/vim/vim82/doc/help.txt to /usr/share/vim/vim82/d
oc/help.txt.vim-tiny by vim-runtime'
Adding 'diversion of /usr/share/vim/vim82/doc/tags to /usr/share/vim/vim82/doc/t
ags.vim-tiny by vim-runtime'
Unpacking vim-runtime (2:8.2.3995-1ubuntu2.15) ...
Selecting previously unselected package vim.
Preparing to unpack .../vim_2%3a8.2.3995-1ubuntu2.15_amd64.deb ...
Unpacking vim (2:8.2.3995-1ubuntu2.15) ...
Setting up vim-common (2:8.2.3995-1ubuntu2.15) ...
Setting up vim-runtime (2:8.2.3995-1ubuntu2.15) ...
Setting up vim (2:8.2.3995-1ubuntu2.15) ...
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vim (vim) in a
uto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vimdiff (vimdi
ff) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rvim (rvim) in
 auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rview (rview)
in auto mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vi (vi) in aut
o mode
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/view (view) in
 auto mode
```

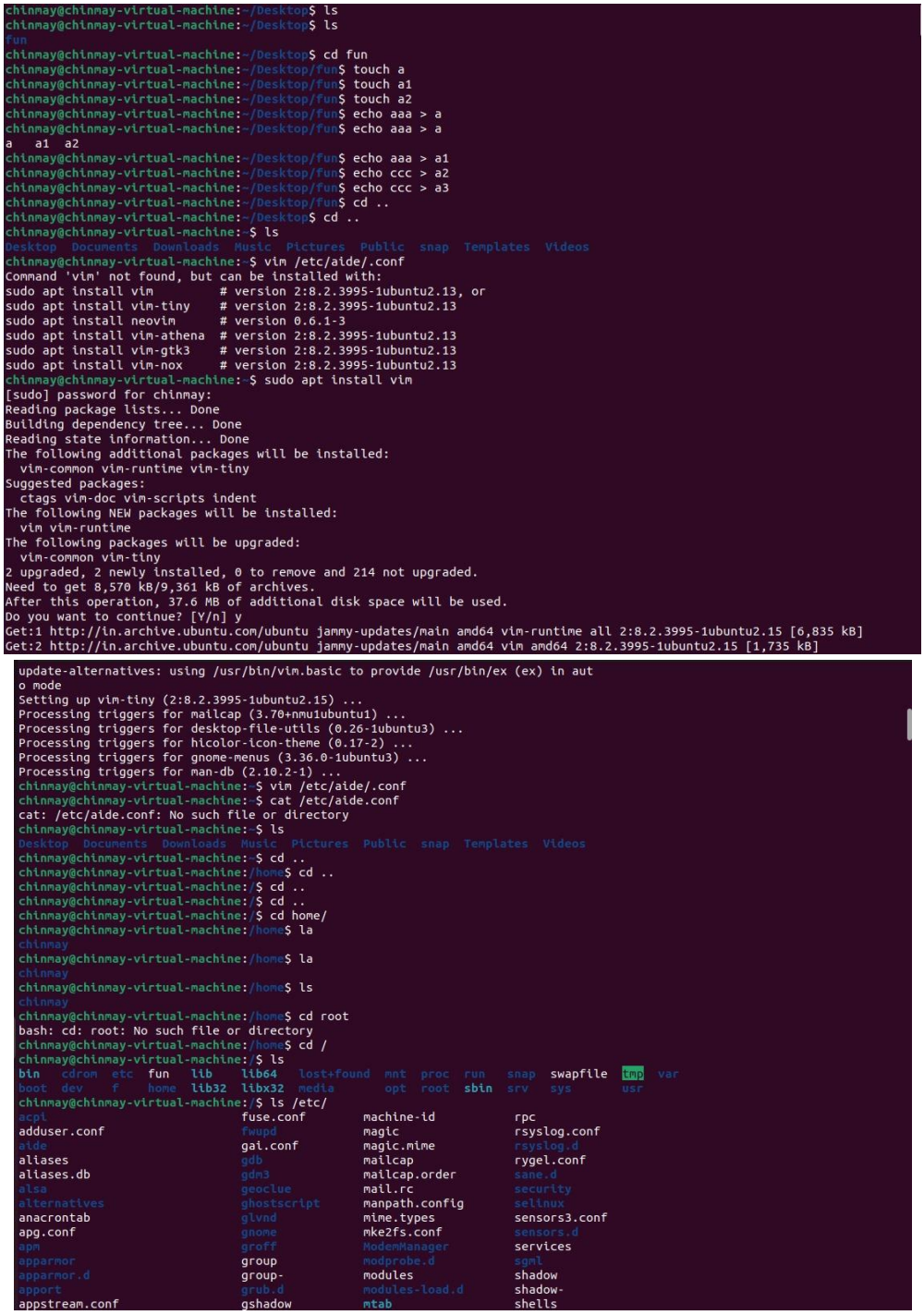
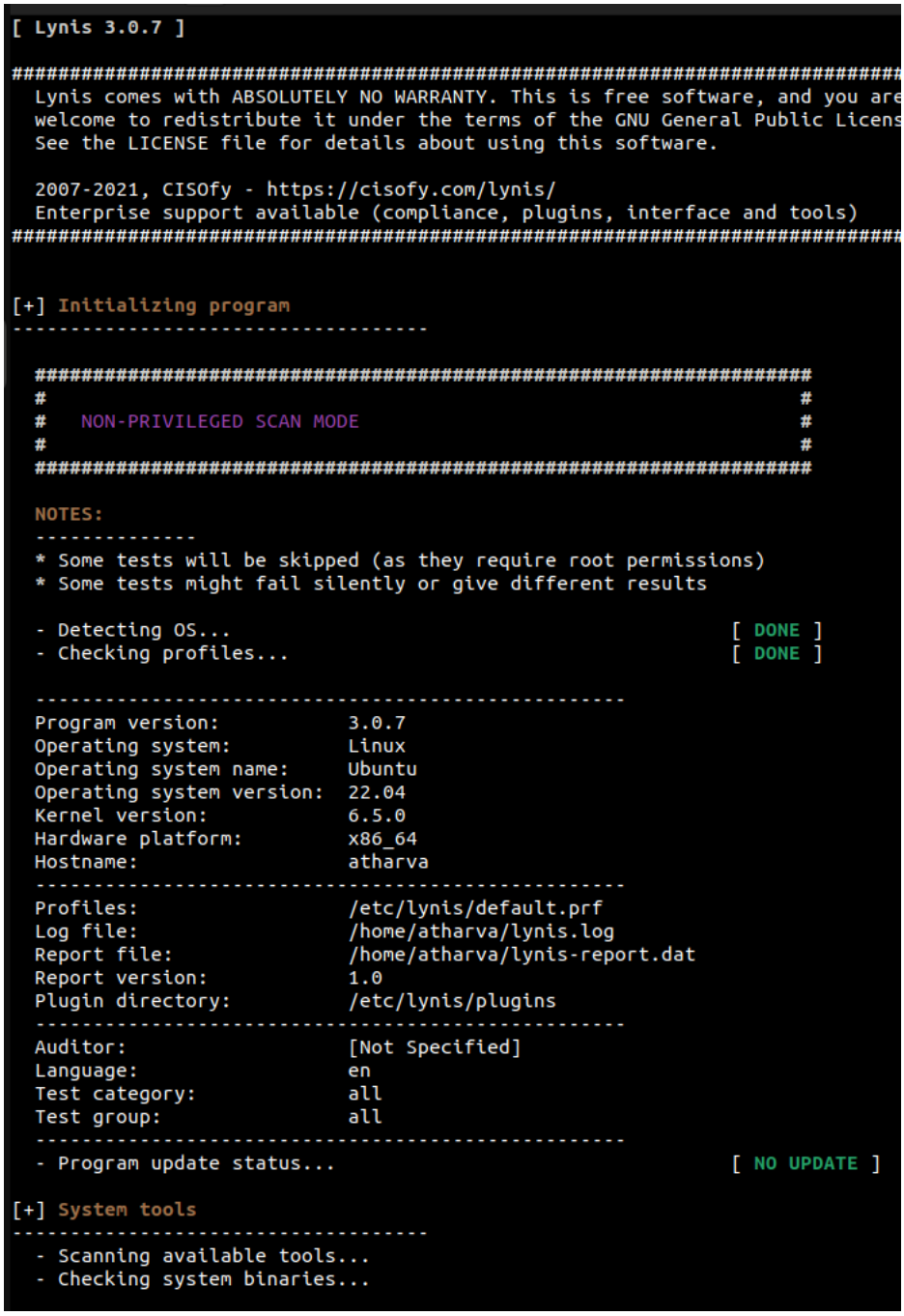


Fig. 5- Working of AIDE


[Lynis 3.0.7]
Lynis comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
welcome to redistribute it under the terms of the GNU General Public Licens
See the LICENSE file for details about using this software.
2007-2021, CISOfy - https://cisofy.com/lynis/
Enterprise support available (compliance, plugins, interface and tools)
[+] Initializing program
NON-PRIVILEGED SCAN MODE
NOTES:
* Some tests will be skipped (as they require root permissions)
* Some tests might fail silently or give different results
- Detecting OS... [DONE]
- Checking profiles... [DONE]
Program version: 3.0.7
Operating system: Linux
Operating system name: Ubuntu
Operating system version: 22.04
Kernel version: 6.5.0
Hardware platform: x86_64
Hostname: atharva
Profiles: /etc/lynis/default.prf
Log file: /home/atharva/lynis.log
Report file: /home/atharva/lynis-report.dat
Report version: 1.0
Plugin directory: /etc/lynis/plugins
Auditor: [Not Specified]
Language: en
Test category: all
Test group: all
- Program update status... [NO UPDATE]
[+] System tools
- Scanning available tools...
- Checking system binaries...

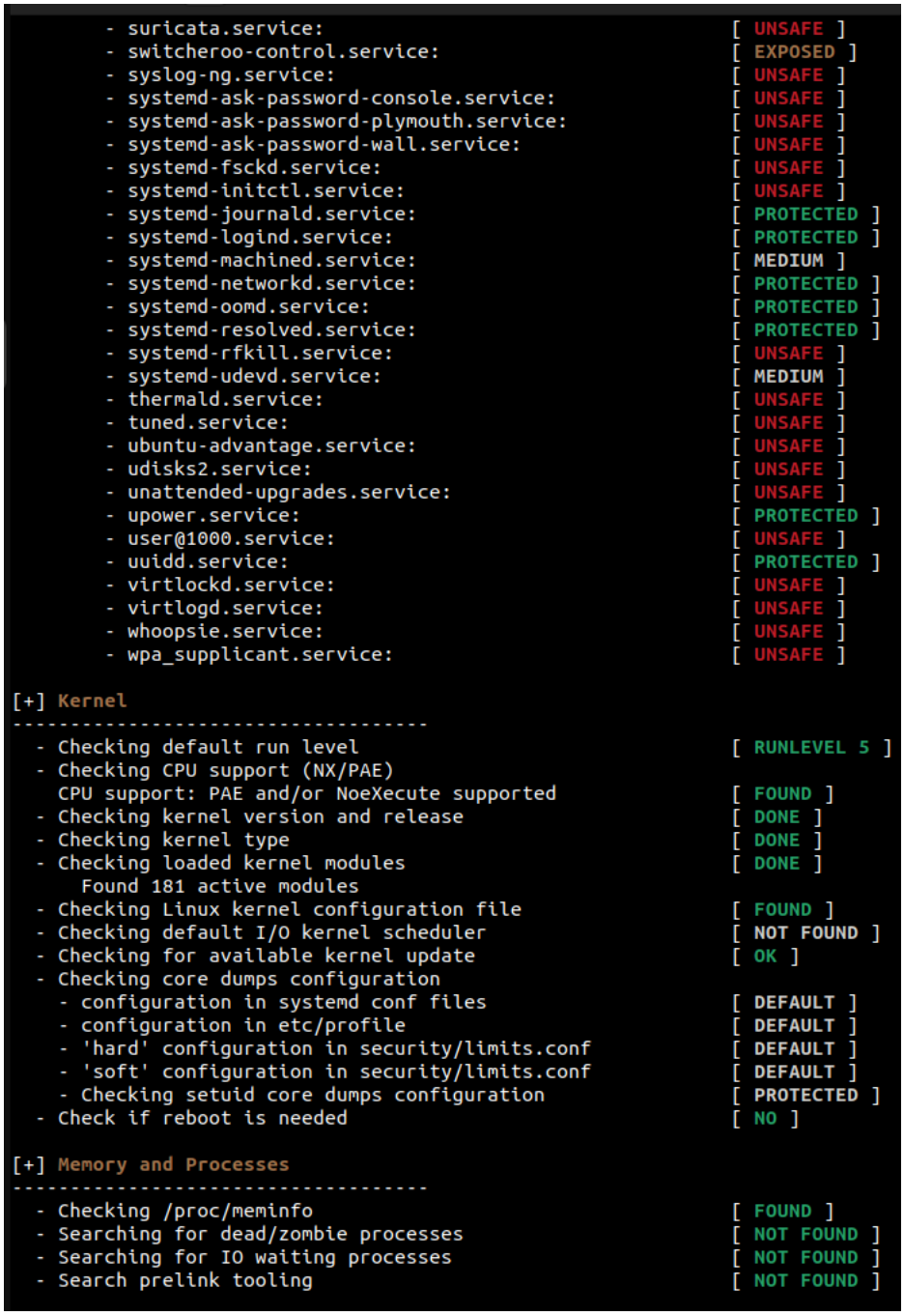
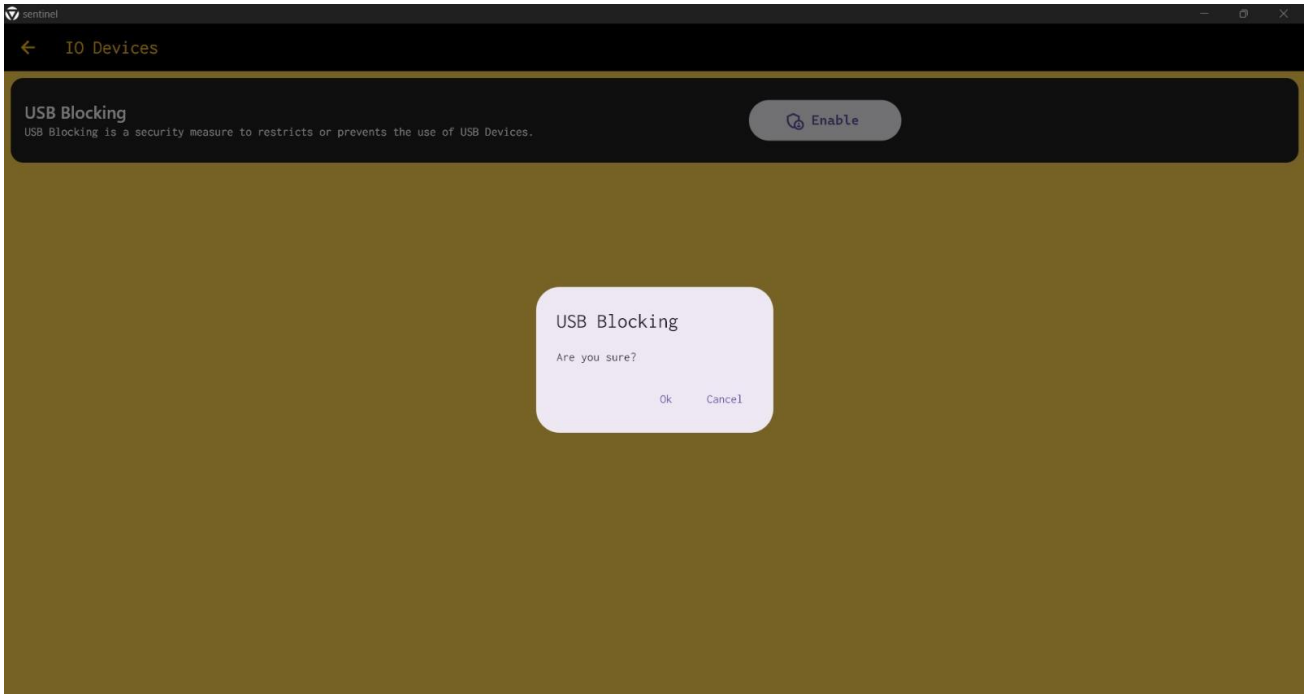


Fig. 6- Working of Lynis Software as it tells which Services are Unsafe, OK


sentinel
IO Devices
USB Blocking
USB Blocking is a security measure to restricts or prevents the use of USB Devices.
Enable
USB Blocking
Are you sure?
Ok
Cancel


sentinel
Archit Kothawade
vishwajit.archit21@vit.edu
FEATURES
Home
Firewall
IDS/IPS
Disk Encryption
Secure Bootloader
I/O Devices
Home Page
SENTINEL OS

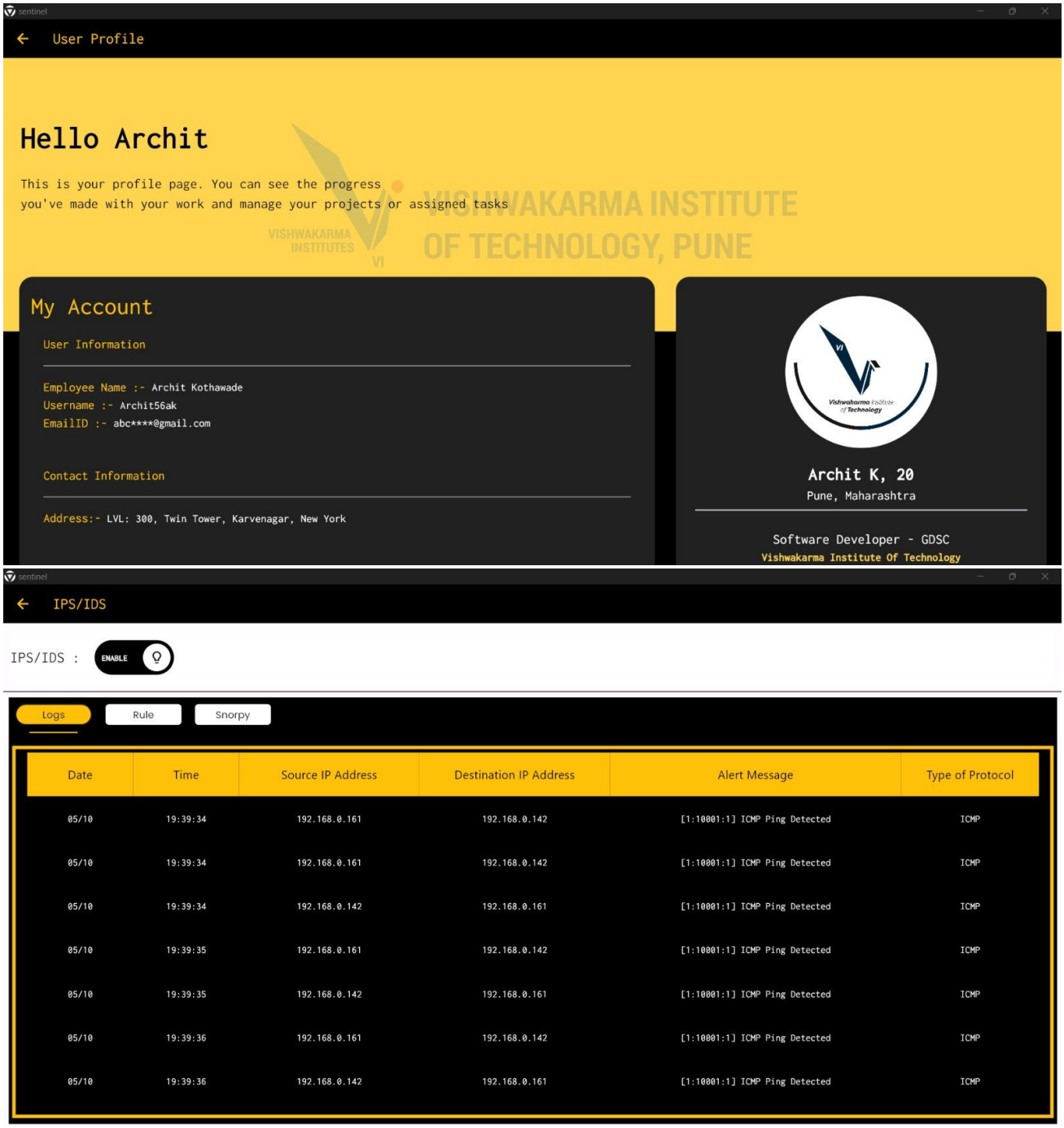


Fig. 7- Different Aspects of GUI

6. Future Scope

SentinelOS is quite promising for the purposes of advancement, including the integration of advanced models for machine learning such as Auto Encoders, Deep Belief Networks and more for improved risk detection, automated risk analysis, and cloud-based scalability to port technologies like Docker and Kubernetes. Other plans include including more customizable dashboards, enhancing accessibility, integrating with SIEM systems and third-party APIs, and

ensuring compliance with regulations such as GDPR and HIPAA. In these developments, SentinelOS will hope to position itself as a holistic, future-ready cybersecurity solution.

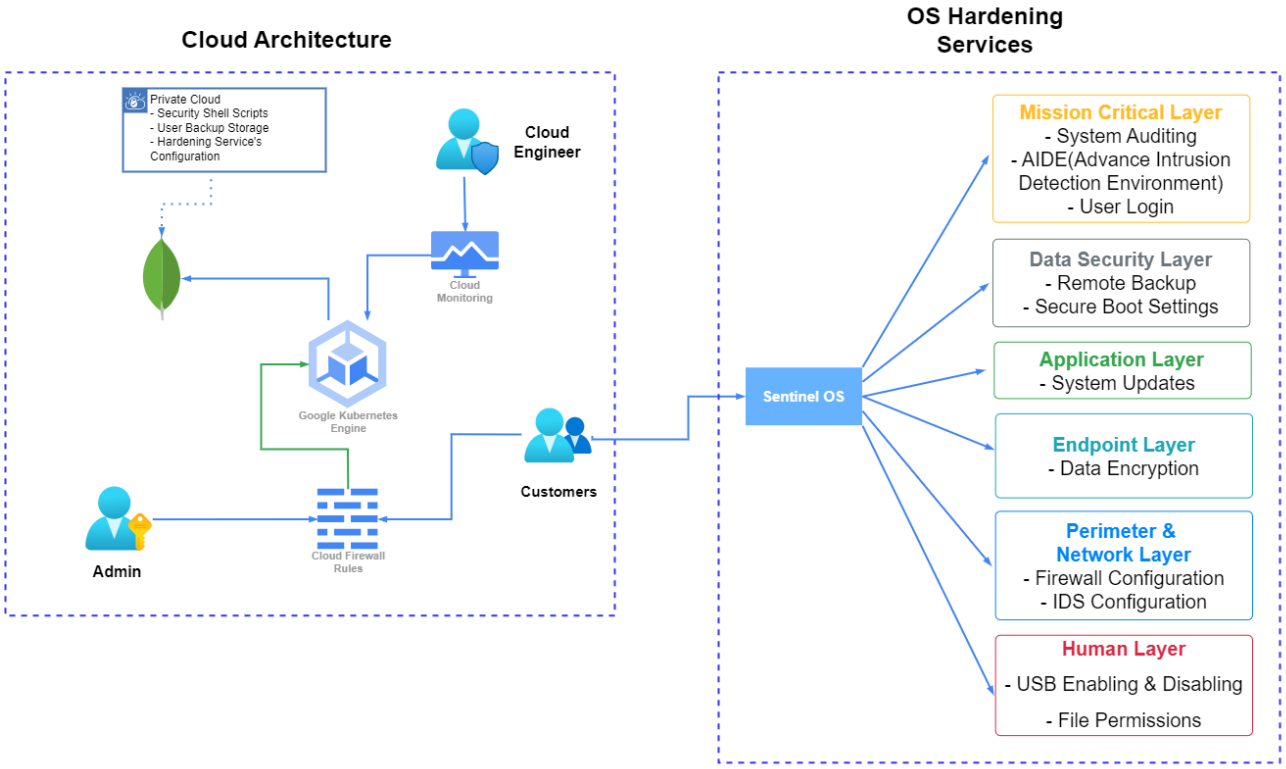


Fig. 8- Proposed Cloud Architecture

7. Conclusion

SentinelOS offers a comprehensive and scalable cybersecurity framework with features such as a deep learning-powered intrusion detection system, file integrity monitoring, regular system auditing, and a user-friendly GUI. It provides organizations with robust tools for managing cybersecurity, ensuring compliance, and addressing diverse industry requirements. Future enhancements in machine learning integration, cloud capabilities, and user experience, along with ongoing compliance updates, will further solidify SentinelOS as an essential tool for protecting digital assets in an evolving threat landscape.

Refrences

[1] D. Peri, “Defence Ministry to switch to locally built OS Maya amid threats,” 2023. [Online]. Available: https://www.thehindu.com/news/national/defence-ministry-to-replace-microsoft-os-with-maya/article67172875.ece.

[2] T. Santini et al., “Effectiveness of software-based hardening for radiation-induced soft errors in real-time operating systems,” in *Architecture of Computing Systems-ARCS 2017:*

30th International Conference, Vienna, Austria, April 3–6, 2017, Proceedings, 2017, pp. 3–15.

[3] M. Jogi, "Establishing, Implementing and Auditing Linux Operating System Hardening Standard for Security Compliance," University of Tartu, Tartu, 2017.

[4] A. Bosio, S. Di Carlo, M. Rebaudengo, and A. Savino, "Toward the hardening of real-time operating systems," in *2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2022, pp. 1–6.

[5] A. E. Ibor and J. N. Obidinnu, "System hardening architecture for safer access to critical business data," *Nigerian Journal of Technology*, vol. 34, no. 4, pp. 788–792, 2015.

[6] J. Turnbull, "Hardening the Basics," in *Hardening Linux*, pp. 1–77, 2005.

[7] Z. Han, L. Cheng, Y. Zhang, and D. Feng, "Operating System Security Policy Hardening via Capability Dependency Graphs," in *Information Security Practice and Experience: 11th International Conference, ISPEC 2015, Beijing, China, May 5-8, 2015, Proceedings*, 2015, pp. 3–17.

[8] A. Brandão, J. S. Resende, and R. Martins, "Hardening cryptographic operations through the use of secure enclaves," *Computers & Security*, vol. 108, p. 102327, 2021.

[9] R. Sasidharan, "A case study to implement windows system hardening using CIS controls," *International Journal of Computer Trends and Technology*, vol. 70, no. 7, pp. 1–7, 2022.

[10] A. Kurmus and R. Zippel, "A Tale of Two Kernels: Towards Ending Kernel Hardening Wars with Split Kernel," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, Scottsdale, Arizona, USA, 2014, pp. 1366–1377.

[11] M. Sreerag, M. Sethumadhavan, and P. P. Amritha, "Identifying and Mitigating Vulnerabilities of Hardened Windows Operating System," in *Information and Communication Technology for Competitive Strategies (ICTCS 2020) ICT: Applications and Social Interfaces*, 2022, pp. 623–632.

[12] S.-J. Jang, "Kernel Hardening by Recovering Kernel Stack Frame in Linux Operating System," *International Journal of Advancements in Computing Technology*, vol. 1, no. 1, 2009.

[13] R. Kaur and M. Singh, "Hardening OS identity by customised masking techniques," in *Proceedings of 2009 Indo US Workshop and Conference on Cyber Security, Cyber Crime and Cyber Forensics*, 2009.

[14] K. Henttunen, "Automated hardening and testing CentOS linux 7: security profiling with the USGCB baseline," 2018.

[15] A. Nepal, "Linux server & hardening security," 2014. [Online]. Available: https://doi.org/10.13140/2.1.5079.2329.

[16] S.-J. Jang, "Design of the Kernel Hardening Function in the Linux Network Module," *IJCSNS*, vol. 6, no. 8B, p. 135, 2006.

[17] M. Krichanov and V. Cheptsov, “UEFI virtual machine firmware hardening through snapshots and attack surface reduction,” in *2021 Ivannikov Ispras Open Conference (ISPRAS)*, 2021, pp. 30–36.

[18] N. Uke and S. Uke, “Proximity approach for object detection in video,” *International Journal of Control and Automation*, vol. 13, no. 2, pp. 868-876, 2020.

[19] S. N. Uke and A. Zade, “Optimal video processing and soft computing algorithms for human hand gesture recognition from real-time video,” *Multimedia Tools and Applications*, vol. 83, no. 17, pp. 50425–50447, Nov. 2023, doi: 10.1007/s11042-023-17608-8.

[20] Atharva Balasaheb Chivate, P. D. Chopade, Shreeshail Chitpur, O. N. Chavan, and Shailaja Uke, “Comparative Analysis of Multiple ML Models and Real-Time Translation,” pp. 400–406, Oct. 2023, doi: https://doi.org/10.1109/icccmla58983.2023.10346894.