CYSEC

Solution Architecture - Secure AI Inference from the Edge

ARCA Trusted OS

CYSEC

1. Introduction

This document outlines a solution architecture leveraging ARCA Trusted OS for an AI protection use case. This use case considers both the protection of the hosted AI models and the execution of inferences in untrusted edge environments.

The proposed architecture ensures AI models' confidentiality, integrity, and availability by incorporating robust security mechanisms, including virtualization-based isolation, full-disk encryption, Secure Boot, and a read-only root file system. These security features mitigate threats and protect sensitive AI models from unauthorized access or tampering.

This approach has been validated through a proof of concept (PoC) where an open-source Large Language Model (LLM) was securely hosted in a cloud while handling inference requests from the edge. Figure 1 sketches a high-level view of the PoC.

ARCA Trusted OS is designed to enhance security for sensitive applications and data. Specifically, in this PoC, the AI model achieves data-at-rest protection through Full Disk Encryption. Furthermore, the LLM, deployed within a Kubernetes pod, is isolated from other applications. Additionally, security features such as Secure Boot and an immutable root file system minimize the operating system's attack surface, enhancing the AI’s model security.

Additionally, ARCA Trusted OS at the edge safeguards the pre-shared secret required to authenticate requests to the AI model.

For remote management, a WireGuard VPN server is hosted in the cloud, providing a secure communication channel.

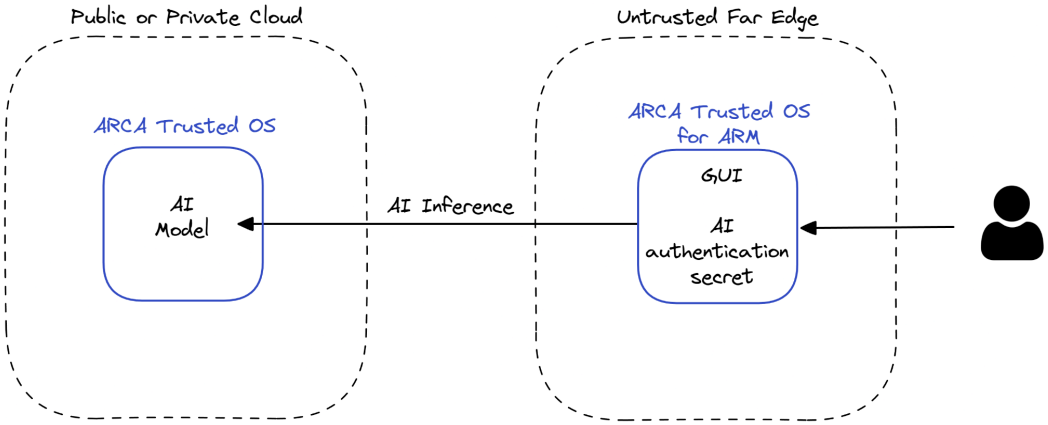

Figure 1. Demonstration general overview

2. Solution Overview

As a demonstration, this architecture has been implemented to support an AI-powered train station assistant. The assistant helps passengers with various inquiries, like platform locations, train schedules, and lost item assistance.

Figure 2 shows passengers interact with an interactive terminal located at the train station (untrusted edge), while AI-generated responses are processed in a private or public cloud. This architecture highlights ARCA Trusted OS's ability to:

- Maintain the confidentiality of AI models hosted in the cloud
- Secure AI inference requests by protecting authentication secrets at the edge
- Enforce strict access control mechanisms in both the edge and cloud environments

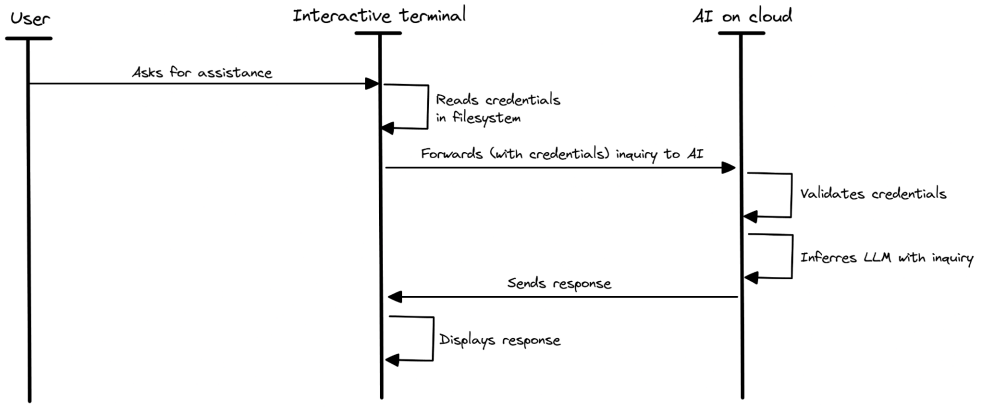

Figure 2. Sequence diagram

The architecture is divided into two primary segments: the cloud and the edge.

- **Cloud**: The AI model runs within a containerized environment. Kubernetes orchestrates the containers to ensure availability and scalability. An AI proxy validates incoming requests by verifying a pre-shared key in the request header.
- **Edge**: A graphical user interface (GUI) is provided as a container (running on Podman). Each user request triggers an HTTPS request to the LLM in the cloud, passing through the AI proxy. The request contains a pre-shared key stored on the device's file system. ARCA Trusted OS protects this key through full-disk encryption combined with a secure boot.

3. Architecture Design

Figure 3 shows the global design of the PoC. As explained previously, one part of the design is implemented in a cloud, whereas the other part runs on an arm-based board at the edge. Both parts will be described in two separate sections of this chapter.

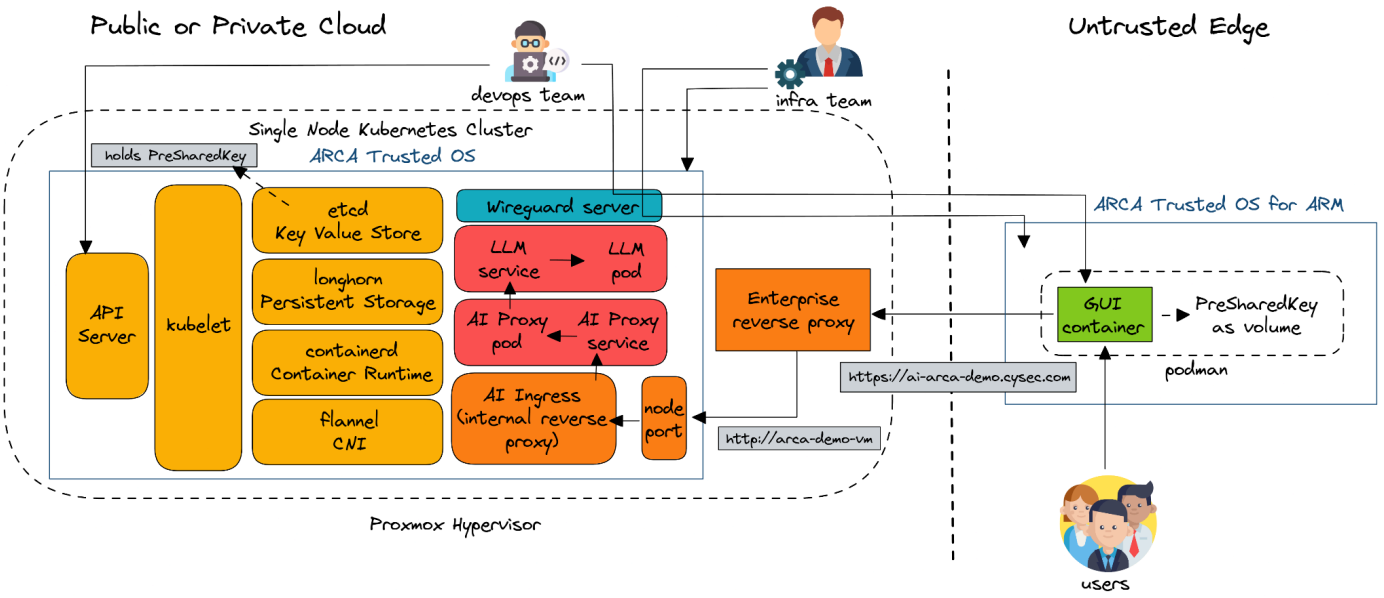

Figure 3. Detailed implementation

Untrusted Edge

The edge device runs ARCA Trusted OS for ARM on a Raspberry Pi 4B, benefiting from security features such as Secure Boot (leveraging TPM or OTP), full-disk encryption, and a read-only root file system.

The device hosts only the user interface application and an associated pre-shared key for authentication. The GUI container (managed by Podman) mounts the pre-shared key as a volume, ensuring secure storage thanks to the by-default full-disk encryption.

The system generates an HTTPS request with embedded authentication credentials for every user action, which is securely transmitted to the cloud.

Cloud

The cloud environment, deployed on either a private or public infrastructure, is based on a Proxmox virtualized environment. A virtual machine (VM) running ARCA Trusted OS for x86 serves as the AI model hosting environment, featuring the same security mechanisms as the edge device. Secure Boot is enabled via a virtual TPM provided by the hypervisor.

Kubernetes (installed using kubeadm) serves as the container orchestrator, with essential components such as:

- CNI: Flannel (for simple networking requirements)
- CSI: Longhorn (for storage management)
- Ingress Controller: NGINX

The end-to-end data flow is as follows:

1. The enterprise reverse proxy terminates HTTPS connections for requests directed to https://ai-arca-demo.cysec.com and forwards them to arca-demo-vm.
2. The VM exposes a NodePort for an internal reverse proxy, redirecting requests to the AI proxy service.
3. The AI proxy service verifies credentials stored in etcd before forwarding valid requests to the LLM service.
4. The LLM service, powered by Ollama, processes the request and generates a response.

All the components, from the CNI to the Ollama, have been deployed with their respective official Helm charts.

Management

A WireGuard VPN server is deployed as a Kubernetes container, allowing remote management of edge devices. It facilitates OS updates and application maintenance. Further details on this implementation can be found here.

4. Deployment Considerations

The deployment of this solution is managed by two primary teams: **DevOps** and **Infrastructure**.

- **DevOps Engineers** manage Kubernetes clusters and access the GUI remotely through WireGuard.
- **Infrastructure Engineers** maintain both the cloud and edge environments, ensuring operational stability.

ARCA Trusted OS, designed as a read-only distribution, limits installed packages for security. It supports containerized deployments using containerd and includes multiple versions of kubelet to facilitate smooth cluster upgrades. The OS embeds one kubelet for each minor version supported by the OS. At the time of this document writing, the latest available kubelet version and the one used for this PoC is v1.31.1.

Regarding networking and storage:

- **CNI**: Flannel is used due to its simplicity; alternatives like Calico or Cilium can be considered based on security requirements.
- **CSI**: Longhorn is used, but other storage solutions can be integrated as needed.

Security is enforced through a pre-shared key (PSK), which, while not the most advanced authentication method, demonstrates the architecture's capability to securely store and validate sensitive credentials. The AI proxy verifies authentication using the custom X-Pre-Shared-Key HTTP header.

The ingress controller of choice is NGINX, though alternatives like HAProxy can also be used.

5. Conclusion

This AI architecture demonstrates a robust and secure approach to hosting AI models in the cloud while executing inference requests at untrusted edge locations. By leveraging ARCA Trusted OS, the solution ensures:

- Confidentiality: AI models are securely stored and executed with strict access control mechanisms.
- Integrity: Secure Boot and a read-only root file system prevent unauthorized modifications.
- Availability: Kubernetes ensures high availability and efficient resource management.

The use of a pre-shared key for authentication highlights ARCA Trusted OS's ability to protect sensitive material at both the cloud and edge. This ability protects the AI model and its usage in cases of theft of or tampering with the edge device or the VM image. While PSK-based authentication may not be ideal for production environments, it serves as an effective demonstration of the security model.

By combining modern containerization strategies with robust security mechanisms, this architecture provides a scalable and secure solution for AI inference at the edge. Future iterations can explore additional authentication methods, such as mutual TLS or hardware-based security tokens, to further enhance security.

If you want to reproduce this demonstration, please follow the instructions provided in our public technical documentation.