ACSIA
XDR PLUS

Automated Cybersecurity Intelligence Application

Architecture Document

Document: ACSIA Architecture & Design V4.01
Release Date: June 2022

Contents

Technical Components

This document describes the design and architecture of the ACSIA XDR Plus security solution and the major components within it.

Base Platform

The ACSIA application engine runs on any of the following major Linux distributions and can be deployed on a physical, virtual, container or cloud platform.

Ubuntu 20.04 and later	CentOS 7
Red Hat 7 and later	Debian 11 and later

ACSIA Engine

The core analytical engine of ACSIA is written in the Java Spring framework that includes Spring Boot, Spring Data & Spring Rest.

ACSIA Frontend

The frontend is a REST API interface that makes calls to the backend using a combination of React.js and the 'Material Design' language.

ACSIA XDR Plus Security

ACSIA comes with the following integrated security components:

- OAuth2 - Secure delegated access
- Multi factor authentication - 2 factor authentication
- TLS for secure communication across clients
- Pwgen - Strong random password generator

ACSIA Open Source Toolstack

There are several opensource tools used by the ACSIA analytical engine including the following:

- Whois
- Postfix
- Python pip
- Binds-utils
- Wget
- Dsnif
- Bc
- Sysstat
- Httpd-tools
- Wazuh
- OSSec
- Elastic
- Kibana
- Logstash
- Lucene
- MySQL
- RabbitMQ
- Docker
- Falco
- ElasticBeats
- Ansible

ACSIA Virtualization Method

ACSIA uses Docker (Linux Containers) which is an operating-system-level virtualization method for running multiple isolated Linux systems (containers) on a control host using a single Linux kernel.

ACSIA Configuration Management System

ACSIA uses Ansible for automating software provisioning, configuration management, and application deployment.

ACSIA Databases

ACSIA uses two separate databases which are segregated for security design reasons as follows:

- MariaDB/MySQL
- MongoDB (Elastic)

ACSIA Message Broker

ACSIA architecture includes RabbitMQ which is the most widely deployed open source message broker.

SIEM Environment

ACSIA uses Elastic Stack tools for its SIEM (Security Information and Event Management) centralized log collector along with the following tools:

- Elastic Search
- Lucene
- Logstash
- Kibana
- ElasticBeats (log shippers)
- OSSEC agents (log shipper)

ACSIA Firewall

ACSIA comes with an embedded host-based firewall. The firewall has innovative features such as killing established TCP connections, banning IP addresses at routing table, therefore blacklisting and whitelisting IP addresses as well as locking individual users (host based).

The “Kill Connection” (host-based) feature is implemented using Berkeley Packet Filter (BPF) that provides a raw interface to data link layers, permitting raw link-layer packets to be sent and received and therefore the ability to stop undesired packets at TCP stack.

Clients can be deployed using Agents or in Agentless mode

ACSIA XDR Plus is a client server architecture and can be deployed using an Agent or in an Agentless mode. The application engine itself has a built-in Elastic stack (Elasticsearch, Logstash and Kibana) and uses Beats provided by Elasticsearch as log shippers.

What is the difference between operating with an agent or an agentless client?

When deploying the agent, the prerequisites in terms of ports and service user requirements are minimal, i.e. there is no service user prerequisite and the ports are consolidated into a single port which is 443 (https). Furthermore, the agent will work autonomously in case the device is not able to reach the ACSIA engine/server.

Another difference is that the agentless deployment requires multiple ports to be opened on the firewalls (both inbound and outbound), as well as the creation of a service user with privileges on connected devices.

Whether deploying ACSIA XDR Plus with an Agent or operating in Agentless mode, the function of the Beats is to send the following logs to the ACSIA application:

- System Logs
- Web Application Logs
- Audit Logs
- Network Traffic
- Kernel/Registry Logs
- Compliance Related Logs

Both Agent and Agentless modes of operation are described below:

Agentless Clients Deployment

The only occurrence where ACSIA XDR Plus interacts with a connected client is when the application wants to ban a malicious IP address or when the end user selects a remediation option provided with the alerts (immediate actions). This is done by ACSIA via its service account (SSH or RDP) so there is no agent required on the client end to perform such action.

ACSIA uses elastic stack shippers and OSSEC/Wazuh agents to receive logs and traffic from the clients. The shippers are called beats and include:

- winlogbeat (windows event logs)
- falco (kernel traffic logs)
- filebeat (main sysylogs and webapp logs)
- packetbeat (network traffic)
- auditbeat (audit logs)
- ossec (security and audit related logs)

The following ports between the ACSIA server and clients must be opened: Ports 22 TCP (for Linux clients) and 5986 TCP (for Windows Clients) between the ACSIA server and clients must be opened for use by Ansible playbooks to deploy beats (shippers) on clients as well as executing remediation actions such as blocking offensive IP addresses, locking users etc… The following ports between the clients and ACSIA server must be opened 1514 UDP, 1515 TCP and 5044 TCP.

ACSIA requires a service user to be created on each client (can be centralized using LDAP or AD where available) to push Ansible playbooks.

Client Agent

A client agent is available starting from v4 of ACSIA XDR Plus. With the agent mode deployment, the aforementioned Beats are bundled into a single package for Windows, Linux and MAC OS operating systems. The bundles can be downloaded from ACSIA UI (see the official guide for details) and will auto-install once the downloaded executable is run. Unlike the agentless mode, the ACSIA agent consolidates all communication ports into unified port 443 (HTTPS) and 444 (TCP/UDP). The agent doesn’t require a service user for ACSIA.

Log Shipping

Beats provided by Elastic search are used as the log shipper to transfer the various log files to the Security Information Event Manager in ACSIA. The data volumes involved are miniscule (being measured in kilobits) so there is no performance overhead on the client or network when deploying ACSIA.

All client traffic is encrypted using Transport Layer Security (TLS V1.2 or later) using client server certificates.

Architectural Design

High Level Architectural Block View

The ACSIA XDR Plus product incorporates Endpoint Detection and Response (EDR) capabilities with Intrusion Prevention (IPS) and Intrusion Detection Systems (IDS) into a single platform, all of which are supported by our real-time Security Information and Event Management (SIEM) system.

We have then enhanced this Extended Detection and Response (XDR) by including a predictive Threat Intelligence feed that bans bad IP Addresses, anonymous exit nodes, and sources of malware from accessing the network. Eliminating these sources of threats from being able to even view your network infrastructure. This is why we refer to the product as ACSIA XDR Plus.

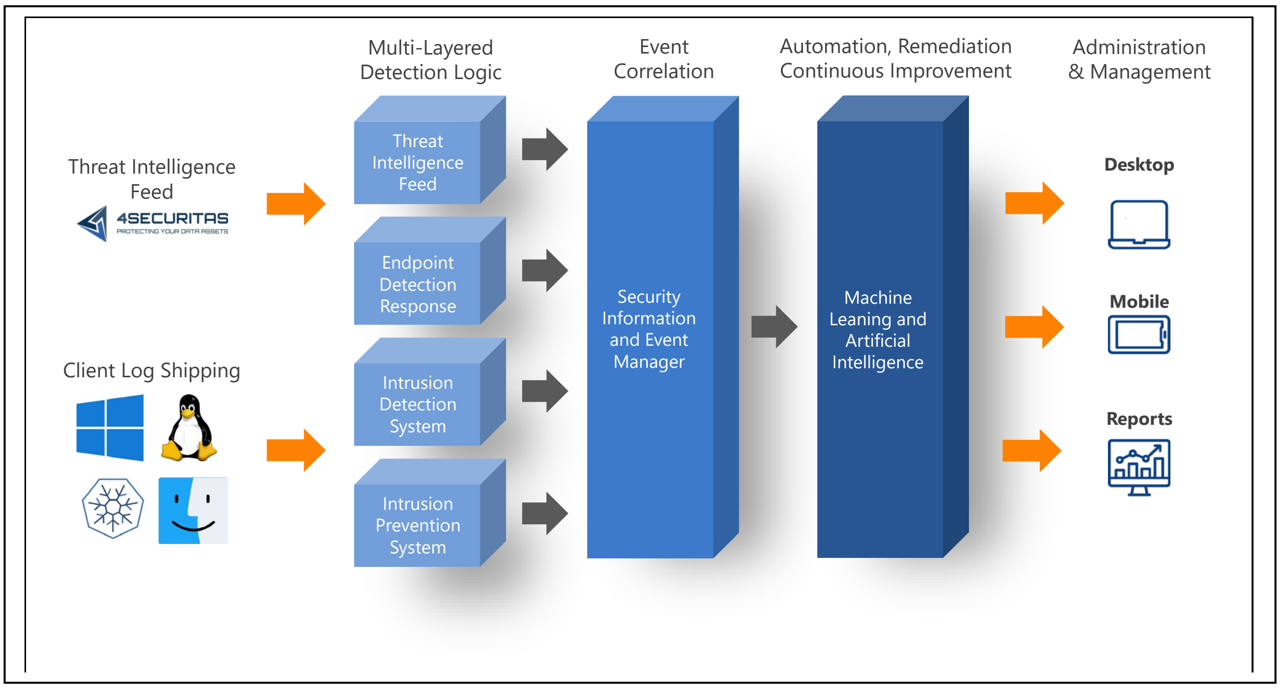
Multi-Layered Detection Logic
Event Correlation
Automation, Remediation Continuous Improvement
Administration & Management
Threat Intelligence Feed
4SECURITAS
PROTECTING YOUR DATA ASSETS
Client Log Shipping
Threat Intelligence Feed
Endpoint Detection Response
Intrusion Detection System
Intrusion Prevention System
Security Information and Event Manager
Machine Leaning and Artificial Intelligence
Desktop
Mobile
Reports

Fig 1: Architectural Block View

Integrating the Predictive, Proactive, EDR, IDS and IPS functions into a single Application Engine facilitates the capture and correlation of multiple event types for superior analytical detection, automated remediation and continuous improvement through Artificial Intelligence and Machine Learning.

ACSIA XDR Plus Internal Workflow Diagram

The workflow diagram (fig 2) depicts ACSIA's architectural workflow and integration points with open source components described earlier in this document.
It also shows the ACSIA server workflow where data from clients are received by 'logstash' and stored in an unstructured form in 'elasticsearch' database. The data stream is queued and managed by the message broker 'RabbitMQ' which serves the analyzer containing the ACSIA core engine.

The analyzed data is enriched and visualized in the ACSIA Web UI (such as alerts, messaging, notifications etc…) and stored in a MySQL database.

Kibana formats all the elasticsearch data and presents it to the end user.

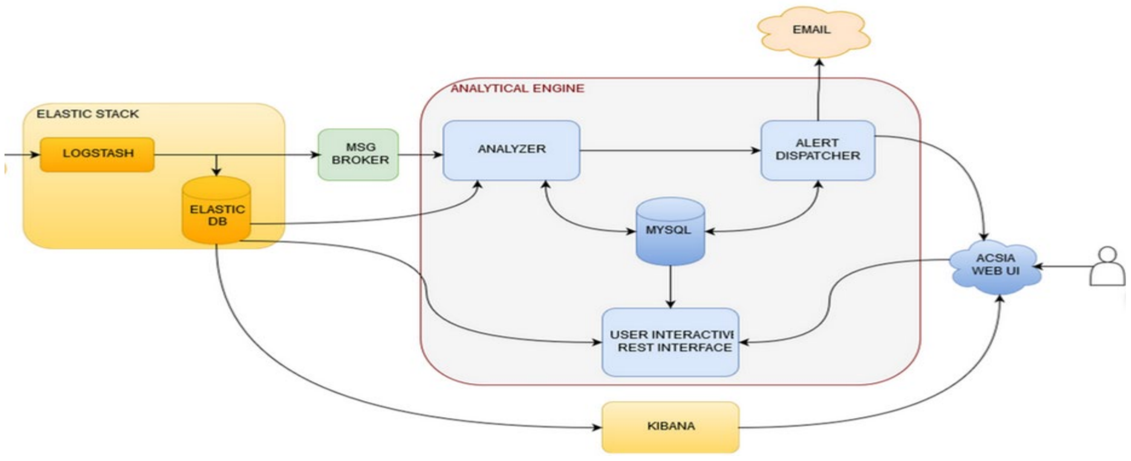

Fig 2: Internal Workflow Diagram

ACSIA XDR Plus Agentless Client Data Flow Diagram

The following flow chart (fig 3) shows how the Agentless Clients interact with ACSIA by shipping their logs in real time. All open source components used by ACSIA are running within docker containers with the sole exception of the ACSIA analytical engine.

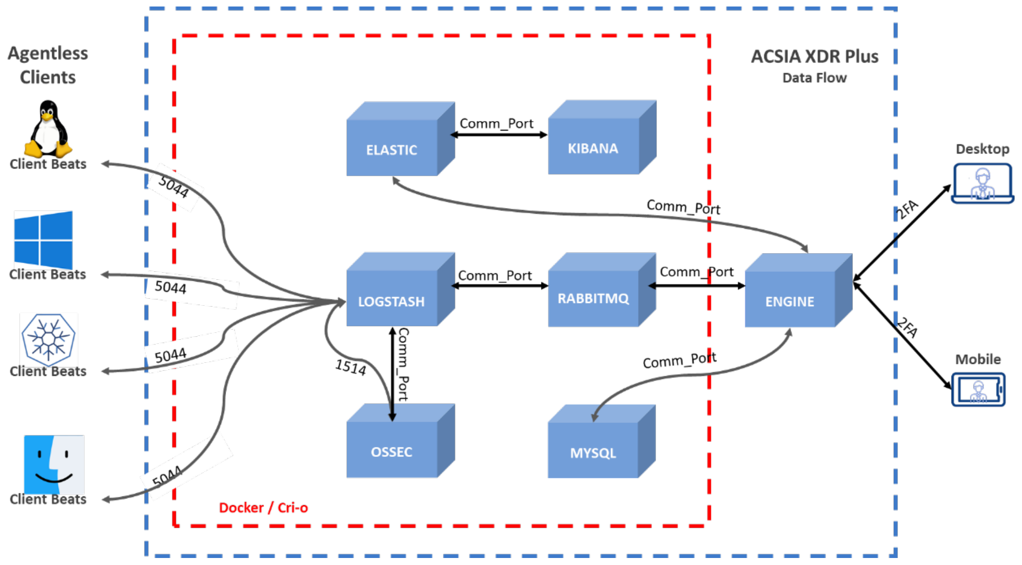

Fig 3: Agentless Client Data flow Diagram

The figure above shows the ACSIA stack showing Elastic Kibana, Logstash, OSSEC, RabbitMQ, and MySQL deployed as Docker container images, while the core Engine and UI are installed on the VM.

ACSIA XDR Plus Client Agent Data Flow Diagram

The following flow chart (fig 4) shows how the agent deployment interacts with ACSIA by shipping their logs in real time through a consolidated secure port such as 443. All open source components used by ACSIA are running within docker containers with the sole exception of the ACSIA analytical engine.

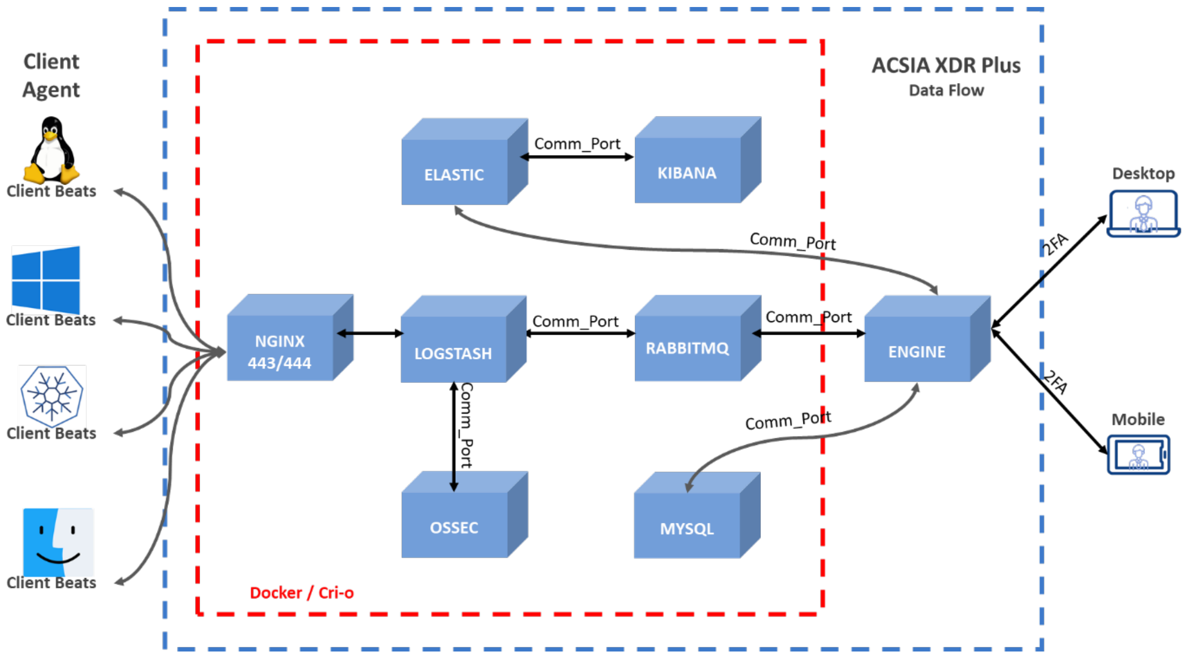

Fig 4: Client Agent Data flow Diagram

More information on ACSIA can be found using the following link: ACSIA documentation
Contact us: email sales@4securitas.com, www.4securitas.com