# SOLUTION DESIGN AND IMPLEMENTATION

## 1. Description

In this part we will focus on the implementation of our solution in a test environment and then perform some fundamental tests to demonstrate the effectiveness of our solution.

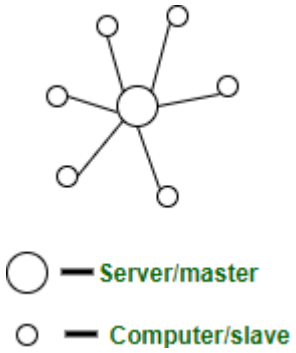
## 2. Selected implementation method

The decision made was to install Wazuh + ELK Stack from scratch on a virtual machine of one of our servers with VMWare technology, for the following reasons:

1. We depend on the current infrastructure of our environment based on VMWare Virtualization technology.
2. We will have high availability with two active virtual clusters.
3. The material used ()
4. pre-compiled images are useful for getting a general idea of the system and getting started quickly, but to get to know it thoroughly and master it, it's better to do a clean install from scratch.
5. Our Linux servers are redhat Enterprise linux server (rhel8) which will serve as an agent, Ubuntu desktop on which we will install our SIEM Wazuh and a kali linux machine which will allow us to carry out attacks towards our agents.
6. We will have 2 Windows systems, one virtual and one physical.

## 3. Choice of network architecture

As we saw in the section, Wazuh can be installed with a centralized architecture (sharing the same Wazuh server and Elastic Stack) or distributed (separating the Wazuh server from the elastic stack). In our case, given our environment, which is a test environment to demonstrate the effectiveness of our solution, we opted for a centralized network architecture. This is an architecture relating to the client/server architecture where one or more client nodes are directly connected to a central server. This is the most common type of system used in many organizations where a client sends a request to a corporate server and receives a response.




**Picture 1**: Centralized visualization of the system

## 3.1. Characteristics of centralized architecture

Presence of a global clock: As the whole system is composed of a central node (a server/a master) and many client nodes (a computer/a slave), all the client nodes synchronize with theclock global (the central node clock)

A single central unit: A single central unit that serves/coordinates all the other nodes in the system.

Component-dependent failure: Failure of the central node leads to failure of the entire system.This makes sense because when the server is down, no other entity is there to send/receive responses/requests

### 3.2. Component of the centralized architecture

The components of the centralized architecture are:

Node (Computer, Mobile, etc.)

Server

Communication link (Cables, WI-FI, etc…)

## 4. Limits of centralized architecture

Unable to scale vertically after a certain limit – After a certain limit, even if you increase thehardware and software capabilities of the server node, the performance will not increase noticeably, leading to cost benefit ratio < 1

Bottlenecks can appear when traffic increases, because the server can only have a finite number of open ports that can listen for connections from client nodes. So when high traffic occurs like a buy sale, the server can basically experience a denial of service attack

## 5. Benefits of centralized architecture

- Easy to physically secure. It is easy to secure and maintain server and client nodes due to their location
- Smooth and elegant personal experience - A customer has a dedicated system that they use (e.g. a personal computer) and the business has a similar system that can be modified to meet custom needs
- Dedicated resources (memory, CPU cores, etc.)
- More cost effective for small systems up to a certain limit. As central systems require less funds to set up, they have an advantage when small systems need to be built
- Fast updates are possible - Only one machine to update.
- Easy detachment of a node from the system. Simply remove the client node connection from the server and voila! Detached knot

## 6. Disadvantages of centralized architecture

Highly dependent on network connectivity - The system may fail if the nodes loseconnectivity because there is only one central node.

Highly dependent on network connectivity – System may fail if nodes lose connectivity because there is no central node

No progressive system degradation – abrupt failure of the entire system

Less possibility of saving data. If the server node goes down and there is no backup, you immediately lose the data

Difficult server maintenance – There is only one server node and for availability reasons, it is inefficient and unprofessional to take the server down for maintenance. Thus, updates must be performed on the fly (hot update), which is difficult and the system may break down.

## 7. Visual architecture retained for the test environment


**Picture 2**: Network architecture of the test environment

## 8. Description of the architecture

As we can see in the previous chapter we opted in our case for an All-in-one architecture (allin one) more precisely centralized because of the resources that we have at our disposal but for the production environments (in company for example) it is advisable to deploy the Wazuh server and Elasticsearch on different hosts. The Wazuh architecture is agent-based, running on monitored endpoints, which transfers security data to a central server. Additionally agentless devices (such as firewalls, switches, routers, access points etc.) are supported and can actively submit log data via Syslog, ssh, or helps their own API. The diagram above represents a Wazuh deployment architecture. It shows the components of thesolution. In our case we opted for a hybrid environment, that is to say composed of physical and virtual machine to be more concrete during the tests. However, we will therefore have aLinux machine running on Ubuntu which will serve as a server on which our SIEM Wazuh, Windows 10 and Redhat will be installed which will serve as agents that the SIEM will monitor and defend against attacks and a linux machine running on Kali Linux which will allow us to carry out attacks on our agents. We will also use a switch (switch) preferably CISCO to interconnect our machines in a local network and finally a Fortigate firewall whichwill also be monitored via the Syslog protocol by our SIEM. In our case we opted for a hybridenvironment, that is to say composed of physical and virtual machine to be more concret iiikjijijjiijij during the

tests. However, we will therefore have a Linux machine running on Ubuntu whichwill serve as a server on which our SIEM Wazuh, Windows 10 and Redhat will be installed which will serve as agents that the SIEM will monitor and defend against attacks and a linuxmachine running on Kali Linux which will allow us to carry out attacks on our agents. We will also use a switch (switch) preferably CISCO to interconnect our machines in a local network and finally a Fortigate firewall which will also be monitored via the Syslog protocolby our SIEM. In our case we opted for a hybrid environment, that is to say composed of physical and virtual machine to be more concrete during the tests. However, we will thereforehave a Linux machine running on Ubuntu which will serve as a server on which our SIEM Wazuh, Windows 10 and Redhat will be installed which will serve as agents that the SIEM will monitor and defend against attacks and a linux machine running on Kali Linux which will allow us to carry out attacks on our agents. We will also use a switch (switch) preferablyCISCO to interconnect our machines in a local network and finally a Fortigate firewall whichwill also be monitored via the Syslog protocol by our SIEM. However, we will therefore havea Linux machine running on Ubuntu which will serve as a server on which our SIEM Wazuh,Windows 10 and Redhat will be installed which will serve as agents that the SIEM will monitor and defend against attacks and a linux machine running on Kali Linux which will allow us to carry out attacks on our agents. We will also use a switch (switch) preferably CISCO to interconnect our machines in a local network and finally a Fortigate firewall whichwill also be monitored via the Syslog protocol by our SIEM. However, we will therefore havea Linux machine running on Ubuntu which will serve as a server on which our SIEM Wazuh,Windows 10 and Redhat will be installed which will serve as agents that the SIEM will monitor and defend against attacks and a linux machine running on Kali Linux which will allow us to carry out attacks on our agents. We will also use a switch (switch) preferably CISCO to interconnect our machines in a local network and finally a Fortigate firewall whichwill also be monitored via the Syslog protocol by our SIEM.

## 9. Wazuh Agents (Windows 10(Virtual & Physical), RHEL8)

Wazuh agents continuously send events to the Wazuh server for analysis and threat detection. In order to start dispatching them, agents establish a connection with the server service for agent connection, which listens on port 1514 (configurable). The Wazuh server then decodes and verifies the events received using the analysis engine. Events that trigger a rule are supplemented with alert data such as rule ID and rule name. Events can be queued in one or both of the following files, depending on whether a rule is triggered or not.

**/var/ossec/logs/archives/archives.json**: The file contains all the events whether they triggered a rule or not.

**/var/ossec/logs/alerts/alerts.json**: The file only contains events that triggered a rule with a sufficiently high priority (the threshold is configurable).

Wazuh's message protocol uses AES encryption by default, with 128bit per block and 256bit keys (Blowfish encryption is also optional)

## 10. Wazuh Server - Elastic Stack Communication (Ubuntu (virtual))

The Wazuh server will use Filebeat (a module supporting data collection, analysis and visualization) to send alerts and events data to the Elasticsearch server, using TLS encryption. Filebeat reads output data from the Wazuh server and sends it to Elasticsearch (by default listening on port 9200/tcp). Once the data is indexed by Elasticsearch, Kibana is used to extract and visualize the information. Wazuh's web UI works inside Kibana, as a plugin. It queries the Wazuh RESTfull API (by default listening to port 55000/TCP on the Wazuh server) to display information about the configuration and status of the Wazuh server and agents. It can also modify, via API calls, agents or server configuration settings when desired. This communication is encrypted with TLS and authenticated with a username and password

## 11. Port Required

For the communication of the Wazuh components, several services are used. We will try to list below the list of default ports used by its services. Users can change these port numbers if needed

Table 1: Ports and Services used by Wazuh components

| Component | Software | Port | Protocol | Objective |
|---|---|---|---|---|
| Wazuh server | Administrator Wazuh | 1514 | TCP (Default) | Agent Login Service |
| | | 1514 | UDP | Agent Login Service |
| | | 1515 | TCP | Agent Registration Service |

| | | | | |
|---|---|---|---|---|
| | | 1516 | TCP | wazuh cluster daemon |
| | | 514 | UDP (Default) | Wazuh system log collector (disable by default) |
| | | 514 | TCP | Wazuh system log collector (disable by default) |
| | API Wazuh | 55000 | TCP | Wazuh RESTful API |
| Elastic-Stack | elasticsearch | 9200 | TCP | Elasticsearch RESTful API |
| | | 9300 – 9400 | TCP | Elasticsearch cluster communication |
| | Kibana | 443 | TCP | Kibana web interface |