PROJECT

Multi-tenant cybersecurity operations centre designed as an open cloud service with machine learning elements

V16 Functional PoC

KPB5 – Implementation of the solution

type – documentation

month of submission – M22

Funded by the European Union
NextGenerationEU

RECOVERY AND RESILIENCE PLAN

GOVERNMENT OFFICE OF THE SLOVAK REPUBLIC

Contents

1 Introduction 4
1.1 Purpose of the document 4
1.2 Starting points and inputs to the PoC 4
1.3 Scope of the PoC 4
1.4 PoC operating environment 5
1.5 Limitations of the PoC 5
2 Components implemented in the INNOSOC PoC 6
2.1 SIEM - ELK Stack 6
2.1.1 ELK Stack deployment model 6
2.1.2 Elasticsearch deployment 6
2.1.3 Logstash deployment 8
2.2 Shuffle 18
2.2.1 Shuffle deployment model 18
2.2.2 Basic integration links of the Shuffle component in the PoC 19
2.3 Wazuh 20
2.3.1 Deployment model of the Wazuh component 20
2.3.2 Environments monitored by Wazuh 20
2.3.3 Deployed and activated Wazuh modules 23
2.3.4 Basic integration links – sending notifications 24
2.4 Greenbone Community edition 29
2.5 OpenCTI 30
2.5.1 Deployment of Shuffle in the OpenStack environment 30
2.5.2 Basic integration links for the OpeCTI component 31
2.6 MISP 32
2.7 GLPI 32
2.8 NetBox 34
2.8.1 Role of NetBox in the PoC 34
2.8.2 Deployment and basic configuration of NetBox 35
2.8.3 Use of NetBox within the PoC 37
2.8.4 Current status and limitations of NetBox in the PoC 41
2.8.5 DFIR IRIS 41
2.8.6 DFIR IRIS deployment model in the OpenStack environment 41
2.8.7 Basic integration links of the DFIR IRIS component in INNOSOC 42
2.9 Kibana (ELK) 43

2.9.1 Dashboards created 43
2.9.2 Technical parameters and verification of functionality 45
2.10 Wiki.js 46
3 Deployment and basic configuration of the PoC 49
3.1 Deployment model 49
3.2 Virtual machines and services used 49
3.3 Basic integration links 49
4 Conclusion 50
4.1 XX 50

1 INTRODUCTION

1.1 Purpose of the document

The purpose of this document is to document the functional PoC within work package KPB5 – Implementation of the solution of the INNOSOC project. The document follows on from the project's previous architectural and implementation outputs and summarizes the state in which the proposed solution is implemented as a functional experimental concept.

The document captures the functional state of the implemented solution as an integrated whole, i.e. as a multi-tenant SOC-type cloud service that is deployed in experimental mode and ready for formal verification. Its main objective is to demonstrate that the proposed solution has been implemented in a functional and operational form within the scope of the PoC, and it constitutes the technical and operational documentation of the experimentally created solution..

1.2 Starting points and inputs to the PoC

The PoC builds on the results of the previous work packages and on the follow-up outputs of the project. The basic framework for implementation was created in KPB4 – Architecture of the solution, the aim of which was to create a coherent technological concept on the basis of the previous analytical and design outputs. Within KPB4, the functional and non-functional requirements, the business architecture, the technological architecture and the final DNR – Detailed design of the solution were defined, which represent a direct input to the KPB5 implementation stage.

The results of the project's previous design and analytical activities must also be regarded as substantive inputs to the implementation of the PoC, in particular the design of the virtualization solution, the design of the SOCaaS service and the design of the integration architecture of the individual components. The project plan lists among the main implementation tasks the configuration of the infrastructure, the installation of software applications, their integration and the testing of the proposed solution. The PoC is the result of building step by step on the project's previous architectural, technological and infrastructure outputs.

1.3 Scope of the PoC

The PoC described here captures the implemented parts of the proposed solution to the extent necessary to verify the basic functionality of the service. The subject of this output is mainly a description of the deployed components, their basic links and their operational arrangement in the project environment. The document focuses on what was actually deployed, configured and verified within KPB5. The PoC does not represent a final production solution or a complete commercial service. It is an experimental concept whose purpose is to demonstrate the technical feasibility of the proposed architecture and the basic operability of the solution as a whole. The scope of the PoC is therefore deliberately limited to the functions that it was possible to implement and verify under laboratory conditions.

1.4 PoC operating environment

The PoC is operated on a private OpenStack cloud environment, which forms the basic infrastructure layer of the solution. The individual components are deployed in virtual machines, with containerization by means of Docker and Docker Compose also used depending on the nature of the specific tool. This model makes it possible to separate the individual functional blocks of the solution, simplify their management and at the same time retain flexibility in experimental deployment. The PoC operating environment has the character of a test and verification environment. Its purpose is not to provide a fully fledged production service, but to create a functional platform for the integration, configuration and verification of the proposed components and their mutual cooperation. The environment is therefore designed to enable repeated deployment, configuration changes and the execution of test scenarios.

1.5 Limitations of the PoC

The PoC is provides an experimental proof of concept. Not all planned components and integration links had been brought to the final level of functionality at the time this output was prepared. In some cases, components were successfully deployed on their own, but their integration was only partially implemented. The document thus captures the actual state of implementation, not the final state of the architecture. The limitations concern mainly the degree of completion of individual integrations, the scope of verified scenarios and the overall level of operational maturity of the solution. However, the PoC provides a sufficient basis for technical verification of the concept and for the subsequent evaluation of the results achieved in a further output of the project.

.

BRAIN:IT

FRI

2 COMPONENTS IMPLEMENTED IN THE INNOSOC POC

2.1 SIEM - ELK Stack

2.1.1 ELK Stack deployment model

Within the INNOSOC PoC solution, ELK is used as a SIEM consisting of the Elasticsearch index database, the Logstash parsing tool, which pre-processes the data format before storage in the database, and Kibana, which serves as the WEB GUI for the entire stack. Kibana displays the collected data and provides user and role management, creation of security rules, alert handling, index management, node management and more.

The entire ELK stack is deployed in LXC Linux containers. A separate container is created for each service. The operating system deployed in the containers is Ubuntu 24.04.2.

All LXC containers are deployed on a single physical server. The server has an Intel(R) Xeon(R) CPU E5-2690 processor, 2 physical processors, 8 cores per processor with a frequency of 2.90GHz. The server has 251GB of RAM. The storage space consists of a RAID of HDD disks with a size of 2.7T and a 1.8T WD Red SSD.

2.1.2 Elasticsearch deployment

The Elasticsearch container has the IP address 192.168.1.178. It runs an Elasticsearch cluster consisting of 7 elasticsearch nodes. The nodes have the roles master, hot, warm and cold. The master node is the main node; the other data nodes connect to it and join the cluster. Hot nodes are the nodes where the most recent data are kept. The data are stored on an SSD so that they can be written within the shortest possible time interval. The data are kept there for 5 days. After this period, they are moved to the warm nodes. There are 2 warm nodes and they use HDDs. The data are stored on the warm nodes for 5 to 14 days. The data are then moved to the cold node and deleted after 21 days.

Node	Disks	Retention period	Allocated RAM
master1	HDD	---	16GB
hot1	SSD	<5 days	31GB
hot2	SSD	<5 days	31GB
hot3	SSD	<5 days	31GB
warm1	HDD	<14 days	31GB
warm2	HDD	<14 days	31GB
cold1	HDD	<21 days	16GB

In the project, emphasis is placed on security. Since version 7, Elasticsearch uses by default the x-pack module, which provides several security functions, including authentication, authorization, encryption of communication and user management.

Communication between the nodes is encrypted using TLS certificates. Each node has its own TLS certificate. The communication between Logstash and Elastich search as well as

between Kibana and Elasticsearch is also encrypted. Each Elasticsearch node has its own configuration file.

The certification authority (CA) used to issue the individual certificates is our own. Elastic provides the option of generating a CA using the *elasticsearch-certutil* tool, which is part of the *Elasticsearch* distribution. This tool makes it possible to create a certification authority certificate as well as to issue further certificates signed by this certification authority.

Elasticsearch in addition uses a so-called keystore. The keystore serves as storage for sensitive data such as credentials, tokens or passwords of system users. In our case, we use the keystore to store the credentials of predefined system accounts. This mechanism increases the security of the system in that sensitive data are not stored directly in configuration files.

In addition to protecting sensitive data by means of the keystore, it is also necessary to secure the network communication of the *elk-elasticsearch* container itself. For this purpose, we used the iptables tool, with which we defined firewall rules. Access to the Secure Shell (SSH) service is permitted from the university range, and access to the *Elasticsearch* application is permitted only from the *Kibana*, *Logstash* and *Arkime* machines, defined on the basis of IP addresses.

As part of availability monitoring, in addition to the already mentioned monitoring of the ELK stack using metricbeat, monitoring using the libreNMS tool and the zabbix tool was introduced.

2.1.2.1 Example of a hot node configuration file

```
cluster.name: elastic-cluster
node.name: hot1
node.roles: [data_hot]
path.data: /mnt/hot-nodes/elasticsearch-hot1
path.logs: /var/log/elasticsearch-hot1
network.host: 192.168.1.178
http.port: 9201
transport.port: 9301
discovery.seed_hosts:   ["192.168.1.178:9300",   "192.168.1.178:9302",
"192.168.1.178:9303",   "192.168.1.178:9304",   "192.168.1.178:9305",
"192.168.1.178:9306"]
cluster.initial_master_nodes: ["master1"]
xpack.security.enabled: true
xpack.security.enrollment.enabled: true
xpack.security.http.ssl:
  enabled: true
  keystore.path: certs/elasticsearch/hot1/http.p12
xpack.security.transport.ssl:
  enabled: true
  verification_mode: certificate
  client_authentication: required
  keystore.path: certs/elastic-certificates.p12
  truststore.path: certs/elastic-certificates.p12
http.host: 0.0.0.0
transport.host: 0.0.0.0
xpack.security.audit.enabled: true
```

```
xpack.security.audit.logfile.events.include:          access_denied,
anonymous_access_denied,  authentication_failed,  connection_denied,
tampered_request, run_as_denied, security_config_change
xpack.security.audit.logfile.emit_node_host_address: true
xpack.security.audit.logfile.emit_node_name: true
xpack.security.audit.logfile.emit_node_host_name: true
```

2.1.3 Logstash deployment

Logstash processes data received from the auditbeat and filebeat agents, which are deployed on the other servers at FRI UNIZA. Communication between the agents and Logstash is also encrypted. The Logstash pipeline is divided into three configuration files. For input, filtering and output of the collected data.

For the input, filtering and output of the collected data:

- input.conf – defines the sources of the received data,
- filters.conf – processes the received data, applies filters to the data, handles data normalization
- output.conf – handles sending of the data.

As with the Elasticsearch tool, we also performed a clean installation for the Logstash tool. On the machine ucs-lxc-1 with IP address 192.168.2.11, an LXC container with the Ubuntu 24.04 LTS operating system was created for the purposes of deploying the Logstash tool. The container is named elk-logstash with the assigned IP address 192.168.1.177.

Logstash is deployed in version 8.17.2. Its configuration defines the size of the main memory, similarly to the Elasticsearch nodes, with 16 GB allocated. This value provides a sufficient reserve also for future expansion of the infrastructure and the connection of further monitored devices.

Communication between Logstash and Elasticsearch is secured by a certificate on the Elasticsearch side. Logstash authenticates to Elasticsearch using a username and password. Since we sought to adhere to the principle of the lowest necessary permissions, the so-called "least privilege", it was necessary to create a user who was assigned a role with permissions. The created role is named logstash_writer and the created user logstash_internal. We assigned the following cluster permissions to this role:

- monitor - monitoring the state of the cluster and nodes
- manage_index_template - management of index templates
- manage_ilm - index life cycle management
- manage_pipeline - management of ingest pipelines

And permissions for individual indices:

- write - writing data to an index
- create - creating new documents
- create_index - creating new indices
- manage - full management of an index
- read - reading data from an index
- index - adding or updating documents

The indices to which the privileges are to be assigned were defined using a so-called index pattern. An index pattern is a template or rule that defines a specific index. So-called

wildcards are used in them, in the form of the character " * ", in combination with text that forms part of the index name. For our role, we defined the following index patterns:

1. gvm-greenbone*
2. elasticsearch-elk-elasticsearch*
3. syslog-fortigate*
4. waf-filebeat*
5. auditbeat*
6. suricata*
7. threatintel-filebeat*

Logstash uses a keystore to store the user password for the Logstash API and the password for the logstash_internal user. Network communication is secured using the iptables tool, with which we defined firewall rules. Access to the SSH service and for receiving data via Logstash is permitted from a specific IPv4 range.

As part of availability monitoring, Logstash is monitored using metricbeat and also using the zabbix and libreNMS tools.

2.1.3.1 Example of the input.conf configuration file for the deployed Logstash

The configuration file sets up data reception using the beat module, which receives data on port 5044. Furthermore, Secure Sockets Layer (SSL)/Transport Layer Security (TLS) encryption is set and the paths to the certificates are defined. The certificates were signed by our own certification authority. Logstash requires clients to present a valid certificate, otherwise the connection will not be established, and it also provides its own TLS certificate with which clients verify the identity of Logstash. This is mutual TLS authentication.

```
input {
        beats {
                port => 5044
                ssl_enabled => true
                ssl_certificate                                          =>
              "/etc/logstash/certs/logstash/logstash.crt"
                ssl_key => "/etc/logstash/certs/logstash/logstash.key"
                ssl_certificate_authorities                              =>
              ["/etc/logstash/certs/ca.crt"]
                ssl_client_authentication => "required"
        }
}
```

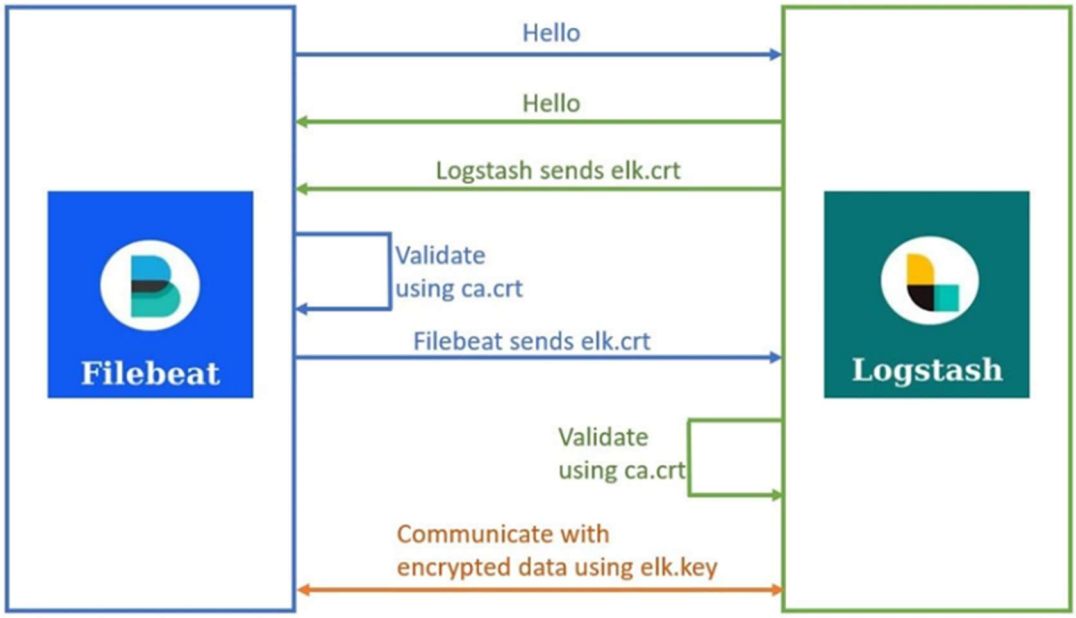


Illustration of mutual TLS

2.1.3.2 Example of the filters.conf configuration file for the deployed Logstash

```
filter { if ("suricata-moloch" in [tags]) or ("suricata-suricata2"
in [tags]) { json { source => "[event][original]" target =>
"suricata_parsed" } date { match => ["[suricata_parsed][timestamp]",
"yyyy-MM-dd'T'HH:mm:ss.SSSSSSZ"] target => "@timestamp" } mutate {
rename => { "[suricata_parsed][src_ip]" => "source.ip"
"[suricata_parsed][src_port]" => "source.port"
"[suricata_parsed][dest_ip]" => "destination.ip"
"[suricata_parsed][dest_port]" => "destination.port"
"[suricata_parsed][proto]" => "network.protocol"
"[suricata_parsed][event_type]" => "event.type"
"[suricata_parsed][flow_id]" => "network.flow_id"
"[suricata_parsed][app_proto]" => "network.application_protocol"
"[suricata_parsed][in_iface]" => "network.interface"
"[suricata_parsed][alert][signature]" => "rule.name"
"[suricata_parsed][alert][signature_id]" => "rule.id"
"[suricata_parsed][alert][category]" => "rule.category"
"[suricata_parsed][alert][severity]" => "rule.severity"
"[suricata_parsed][alert][action]" => "rule.action"
"[suricata_parsed][alert][rev]" => "rule.revision"
"[suricata_parsed][alert][gid]" => "rule.gid"
"[suricata_parsed][flow][pkts_toserver]" =>
"network.packets.to_server" "[suricata_parsed][flow][pkts_toclient]"
=> "network.packets.to_client"
"[suricata_parsed][flow][bytes_toserver]" =>
"network.bytes.to_server" "[suricata_parsed][flow][bytes_toclient]"
=> "network.bytes.to_client" "[suricata_parsed][flow][start]" =>
```

```
"network.flow_start"
"[suricata_parsed][alert][metadata][signature_severity]" =>
"rule.signature_severity"
"[suricata_parsed][alert][metadata][created_at]" =>
"rule.created_at" "[suricata_parsed][alert][metadata][updated_at]"
=> "rule.updated_at" "[suricata_parsed][alert][metadata][cve]" =>
"rule.cve_list" "[suricata_parsed][alert][metadata][deployment]" =>
"rule.deployment"
"[suricata_parsed][alert][metadata][attack_target]" =>
"rule.attack_target"
"[suricata_parsed][alert][metadata][affected_product]" =>
"rule.affected_product" "[suricata_parsed][alert][metadata][tag]" =>
"rule.tag" } convert => { "network.flow_id" => "integer"
"source.port" => "integer" "destination.port" => "integer"
"network.packets.to_server" => "integer" "network.packets.to_client"
=> "integer" "network.bytes.to_server" => "integer"
"network.bytes.to_client" => "integer" "rule.id" => "integer"
"rule.severity" => "integer" "rule.revision" => "integer" "rule.gid"
=> "integer" } } geoip { source => "source.ip" target => "source"
database => "/etc/logstash/geoip/GeoLite2-City.mmdb" } geoip {
source => "destination.ip" target => "destination" database =>
"/etc/logstash/geoip/GeoLite2-City.mmdb" } mutate { remove_field =>
[ "source.ip", "destination.ip"] rename => {
"[suricata_parsed][src_ip]" => "source.ip"
"[suricata_parsed][dest_ip]" => "destination.ip" } } mutate {
remove_field => [ "message", "suricata_parsed"] } }

else if ("auditbeat-studenti" in [tags]) or ("auditbeat-admin" in
[tags]) or ("auditbeat-testing" in [tags]) or ("auditbeat-elk" in
[tags]){ mutate { add_field => { "[@metadata][index]" => "auditbeat-
logs" } add_field => { "event.ingested" => "%{+YYYY-MM-
dd'T'HH:mm:ss.SSSZ}" } rename => {
"[process][parent][process][name]" => "process.parent.name"
"[process][parent][process][executable]" =>
"process.parent.executable" "[process][parent][process][args]" =>
"process.parent.args" "[process][parent][process][pid]" =>
"process.parent.pid" "[process][parent][process][entity_id]" =>
"process.parent.entity_id" "[process][parent][process][title]" =>
"process.parent.title" "[process][parent][process][start_time]" =>
"process.parent.start_time" "[process][parent][process][owner]" =>
"process.parent.owner" "[process][parent][process][group]" =>
"process.parent.group" "[process][parent][process][thread]" =>
"process.parent.thread" } } mutate { replace => {
"process.parent.pid" => "%{[process][parent][pid][0]}" } }

mutate {
  remove_field => ["process.parent.pid"]
}
mutate {
  convert => { "process.parent.pid" => "integer" }
```

```
}
if [process][args] and ( [process][args] != "" ) {

  mutate {
    add_field => { "[process][command_line]" => "%{[process][args]}"
}
  }
  mutate {
    gsub => [
      "[process][command_line]", '\"', '',
      "[process][command_line]", '\[', '',
      "[process][command_line]", '\]', '',
      "[process][command_line]", ',', ' '
    ]
  }
}


} else if "syslog-fortigate" in [log_type] { grok { match => {
"message" => "%{SYSLOGTIMESTAMP:syslog_timestamp}
%{IPORHOST:syslog_host} %{GREEDYDATA:event.original}" } overwrite =>
["event.original"] } kv { source => "event.original" field_split =>
" " value_split => "=" trim_key => """ trim_value => """ } mutate {
rename => { "srcip" => "source.ip" "dstip" => "destination.ip" }
convert => { "srcport" => "integer" "dstport" => "integer"
"sentbyte" => "integer" "rcvdbyte" => "integer" "sentpkt" =>
"integer" "rcvdpkt" => "integer" "duration" => "integer" } add_field
=> { "datetime" => "%{date} %{time}" } } date { match =>
["datetime", "yyyy-MM-dd HH:mm:ss"] target => "@timestamp" timezone
=> "+0100" } mutate { remove_field => ["datetime", "message",
"date"] } if [agent] { mutate { add_field => { "agent_name" =>
"%{[agent][name]}" "agent_id" => "%{[agent][id]}" "agent_type" =>
"%{[agent][type]}" "agent_version" => "%{[agent][version]}"
"agent_ephemeral_id" => "%{[agent][ephemeral_id]}" } } mutate {
remove_field => ["agent"] }

 } } else if "CTF-win" in [tags]{ mutate { add_field => {
"[@metadata][index]" => "ctf-win-logs" } add_field => {
"event.ingested" => "%{+YYYY-MM-dd'T'HH:mm:ss.SSSZ}" } } if
[winlog][provider_name] == "Microsoft-Windows-Sysmon" { mutate {
rename => { "[winlog][event_id]" => "event.sysmon_id"
"[winlog][event_data][Image]" => "process.name"
"[winlog][event_data][CommandLine]" => "process.command_line"
"[winlog][event_data][ParentImage]" => "process.parent.name"
"[winlog][event_data][ParentCommandLine]" =>
"process.parent.command_line" "[winlog][event_data][User]" =>
"user.name" "[winlog][event_data][Hashes]" => "file.hashes"
"[winlog][event_data][TargetFilename]" => "file.target_name"
"[winlog][event_data][CreationUtcTime]" => "file.creation_time"
"[winlog][event_data][ProcessId]" => "process.id"
```

```
"[winlog][event_data][ParentProcessId]" => "process.parent.id" } }
if [file][hashes] { ruby { code => ' hashes =
event.get("[file][hashes]") if hashes hashes.split(",").each do
|hash| key_value = hash.split("=") if key_value.length == 2 end end
end ' } } } if [winlog][provider_name] == "PowerShell" { mutate {
rename => { "[winlog][event_id]" => "event.powershell_id"
"[winlog][event_data][param1]" => "powershell.command"
"[winlog][event_data][param2]" => "powershell.details"
"[winlog][event_data][param3]" => "powershell.output" } } } if
[host][name] { mutate { rename => { "[host][name]" =>
"system.hostname" "[host][os][name]" => "system.os_name"
"[host][os][version]" => "system.os_version" } } } if
[agent][version] { mutate { rename => { "[agent][version]" =>
"logstash.agent_version" } } } mutate { remove_field => [
"[winlog][provider_guid]", "[winlog][record_id]",
"[winlog][opcode]", "[winlog][api]" ]

 } } else if "filebeat-gvm" in [tags] {

if [message] =~ "^\{.*\}$" {
  json {
    source => "message"
  }
} else {
  grok {
    match => { "message" => "%{IPORHOST:client_ip} - -
\[%{HTTPDATE:log_timestamp}\] \"%{WORD:http_method}
%{URIPATHPARAM:request} HTTP/%{NUMBER:http_version}\"
%{NUMBER:response_code} (?:%{NUMBER:bytes_sent}|-)
\"%{DATA:referrer}\" \"%{DATA:user_agent}\"" }
  }
}
date {
  match => [ "log_timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ]
  target => "@timestamp"
}
mutate {
  remove_field => ["log_timestamp"]
}


} else if "filebeat-elasticsearch" in [tags] { json { source =>
"message" target => "audit" skip_on_invalid_json => true } mutate {
add_field => { "user" => "%{[audit][user.name]}" "action" =>
"%{[audit][event.action]}" "timestamp" => "%{[audit][timestamp]}" }
} date { match => ["[audit][@timestamp]", "yyyy-MM-
dd'T'HH:mm:ss,SSSZ"] target => "@timestamp" } } else if "filebeat-
WAF" in [tags] { if [log][file][path] == "/var/log/nginx/access.log"
{ mutate { add_tag => [ "nginx-access" ] } grok { match => {
"message" => '%{IPORHOST:remote_ip} - %{DATA:remote_user}
[%{HTTPDATE:timestamp}] "(?:%{WORD:method}
```

```
%{URIPATHPARAM:request}(?: HTTP/%{NUMBER:http_version})?)"
%{NUMBER:status} (?:%{NUMBER:bytes}|-) "(?:%{URI:referrer}|-)"
"(?:%{DATA:user_agent}|-)"' } remove_field => [ "event.original" ] }
date { match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ] target =>
"@timestamp" } } else if [log][file][path] ==
"/var/log/nginx/error.log" { mutate { add_tag => [ "nginx-error" ] }
grok { match => { "message" => '[%{DAY} %{MONTH} %{MONTHDAY} %{TIME}
%{YEAR}] [%{LOGLEVEL:log_level}]( [%{DATA:module}])?
%{GREEDYDATA:error_message}' } remove_field => [ "event.original" ]
} date { match => [ "timestamp", "dd/MMM/yyyy:HH:mm:ss Z" ] target
=> "@timestamp" } } else if [log][file][path] ==
"/var/log/modsec/kibana.fri.uniza.sk_audit.log" { mutate { add_tag
=> [ "nginx-audit" ] } grok { match => { "message" => "(?m)---
%{DATA:modsec_id}---A--\[%{HTTPDATE:timestamp}\] %{NUMBER:unique_id}
%{IP:src_ip} %{NUMBER:src_port} %{IP:dst_ip} %{NUMBER:dst_port}.*?--
-%{DATA:modsec_id}---B--%{GREEDYDATA:http_request}---
%{DATA:modsec_id}---F--%{GREEDYDATA:http_response}---
%{DATA:modsec_id}---H--%{GREEDYDATA:modsec_alerts}" } tag_on_failure
=> [ "_modsec_grok_fail" ] } date { match => [ "timestamp",
"dd/MMM/yyyy:HH:mm:ss Z" ] target => "@timestamp" } mutate {
remove_field => [ "event.original" ] } } } else if "filebeat-
threatintel" in [tags] { date { match => ["[audit][@timestamp]",
"yyyy-MM-dd'T'HH:mm:ss,SSSZ"] target => "@timestamp" } if
[fileset][name] == "abuseurl" { json { source => "message" target =>
"abuseurl" } mutate { add_field => { "threat_type" =>
"%{[abuseurl][threat]}" "url" => "%{[abuseurl][url]}" "ip" =>
"%{[abuseurl][host]}" "reporter" => "%{[abuseurl][reporter]}"
"malware_tag" => "%{[abuseurl][tags][0]}" "urlhaus_ref" =>
"%{[abuseurl][urlhaus_reference]}" } } } else if [fileset][name] ==
"abusemalware" { json { source => "message" target => "abusemalware"
} mutate { add_field => { "sha256" =>
"%{[abusemalware][sha256_hash]}" "md5" =>
"%{[abusemalware][md5_hash]}" "signature" =>
"%{[abusemalware][signature]}" "vt_result" =>
"%{[abusemalware][virustotal][result]}" "vt_percent" =>
"%{[abusemalware][virustotal][percent]}" "download_url" =>
"%{[abusemalware][urlhaus_download]}" } } } else if [fileset][name]
== "otx" { json { source => "message" target => "otx" } mutate {
add_field => { "indicator" => "%{[otx][indicator]}" "indicator_type"
=> "%{[otx][type]}" "description" => "%{[otx][description]}" } } }

 } else if "arkimecapture-moloch" in [tags] { grok { match => {
"message" => "%{MONTH:log_month} %{MONTHDAY:log_day}
%{TIME:log_time} %{GREEDYDATA:rest_of_message}" } } mutate {
add_field => { "log_timestamp" => "%{log_month} %{log_day}
%{log_time}" } } date { match => [ "log_timestamp", "MMM dd
HH:mm:ss" ] target => "@timestamp" timezone => "Europe/Bratislava" }
grok { match => { "message" => "drop: %{NUMBER:drop_count:int}
(%{NUMBER:drop_percent:float})" } } grok { match => { "message" =>
"recv: %{NUMBER:recv_count:int}" } } } }
```

2.1.3.3 Example of the output.conf configuration file for the deployed Logstash

```
output {
  if "auditbeat-studenti" in [tags] {
    elasticsearch {
      hosts => ["https://192.168.1.178:9200"]
      user => "logstash_internal"
      password => "${ES_psswd}"
      ssl_enabled => true
      ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
      ilm_enabled => true
      ilm_rollover_alias => "auditbeat-logs-studenti"
      ilm_pattern => "{now/d}-000001"
      ilm_policy => "auditbeat-logs-ILM"
      manage_template => false
    }
  }
  else if "suricata-moloch" in [tags] {
    elasticsearch {
      hosts => ["https://192.168.1.178:9200"]
      user => "logstash_internal"
      password => "${ES_psswd}"
      ssl_enabled => true
      ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
      ilm_enabled => true
      ilm_rollover_alias => "suricata-moloch"
      ilm_pattern => "{now/d}-000001"
      ilm_policy => "suricata-moloch-ILM"
      manage_template => false
    }
  }
  else if "syslog-fortigate" in [log_type] {
    elasticsearch {
      hosts => ["https://192.168.1.178:9200"]
      user => "logstash_internal"
      password => "${ES_psswd}"
      ssl_enabled => true
      ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
      ilm_enabled => true
      ilm_rollover_alias => "syslog-fortigate"
      ilm_pattern => "{now/d}-000001"
      ilm_policy => "syslog-fortigate-ILM"
      manage_template => false
    }
  }
  else if "auditbeat-admin" in [tags] {
    elasticsearch {
      hosts => ["https://192.168.1.178:9200"]
      user => "logstash_internal"
      password => "${ES_psswd}"
      ssl_enabled => true
```

```
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "auditbeat-logs-admin"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "auditbeat-logs-ILM"
    manage_template => false
  }
}
else if "suricata-suricata2" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "suricata-suricata2"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "suricata-suricata2-ILM"
    manage_template => false
  }
}
else if "auditbeat-testing" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    index => "auditbeat-testing"
    manage_template => false
  }
}
else if "filebeat-gvm" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "gvm-greenbone"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "gvm-greenbone-ILM"
    manage_template => false
  }
}
else if "auditbeat-elk" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
```

```
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "auditbeat-logs-elk"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "auditbeat-logs-ILM"
    manage_template => false
  }
}
else if "filebeat-elasticsearch" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "elasticsearch-elk-elasticsearch"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "elasticsearch-elk-elasticsearch-ILM"
    manage_template => false
  }
}
else if "filebeat-WAF" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "waf-filebeat"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "waf-filebeat-ILM"
    manage_template => false
  }
}
else if "filebeat-threatintel" in [tags] {
  elasticsearch {
    hosts => ["https://192.168.1.178:9200"]
    user => "logstash_internal"
    password => "${ES_psswd}"
    ssl_enabled => true
    ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
    ilm_enabled => true
    ilm_rollover_alias => "threatintel-filebeat"
    ilm_pattern => "{now/d}-000001"
    ilm_policy => "threatintel-filebeat-ILM"
```

```
      manage_template => false
    }
  }
  else if "arkimecapture-moloch" in [tags] {
    elasticsearch {
      hosts => ["https://192.168.1.178:9200"]
      user => "logstash_internal"
      password => "${ES_psswd}"
      ssl_enabled => true
      ssl_certificate_authorities => "/etc/logstash/certs/ca.crt"
      ilm_enabled => true
      ilm_rollover_alias => "arkimecapture-moloch"
      ilm_pattern => "{now/d}-000001"
      ilm_policy => "arkimecapture-moloch-ILM"
      manage_template => false
    }
  }
}
```

2.2 Shuffle

Shuffle Automation is used within the INNOSOC PoC solution for the process of automated execution of processes and transformation of data between various security tools and systems. In our case, this involves the orchestration of data flows, which ensures that data from one system are automatically processed, transformed and redirected to other systems without human intervention.

2.2.1 Shuffle deployment model

The solution runs in Docker containers, with the backend built on OpenSearch, which serves as storage for workflows and Python applications. The web interface is accessible on port 3001.

Technical specification of the Shuffle deployment:

- Environment – OpenStack
- Instance name – shuffe
- IP address – 192.168.4.188
- vCPU – 4
- RAM – 16 GB
- Disk – 80 GB
- OS – Ubuntu 24.04

The Shuffle software is based on docker. This means that, in order to be fully functional, the system on which it will be deployed must have the most up-to-date version of docker. Together with the Shuffle installation there are 2 files, docker-compose and the .env file. Both files serve to configure ports, port forwarding, the backend and the frontend. For Shuffle to be fully functional, it is sufficient to start it using the command sudo docker-compose up –d.

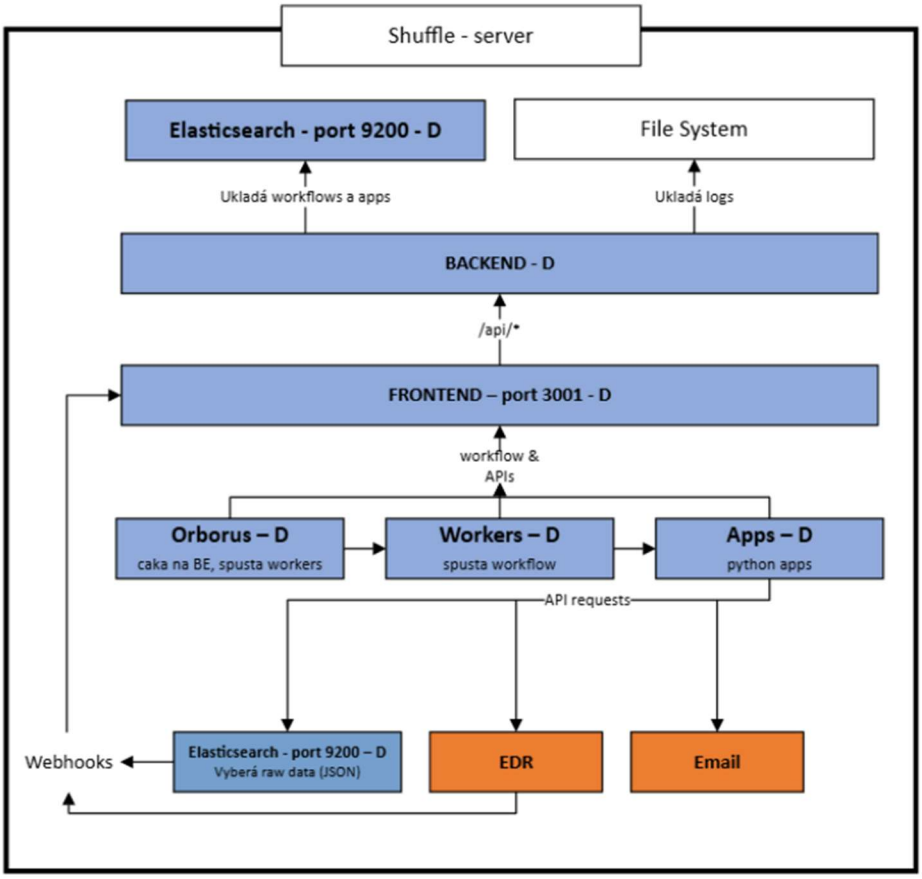


Figure 1 Architecture of the Shuffle components

The figure illustrates the architecture of the Shuffle SOAR platform on the server and the data processing flow. The frontend runs on port 3001 and serves as the web interface through which the user works with workflows and the API. Requests from the frontend are routed to the backend, which provides the application logic and communication with the Elasticsearch database, where workflows and applications are stored, and with the file system, in which logs are stored. The runtime environment contains the Orborus, Workers and Apps components, with Orborus overseeing the backend and launching workers, which execute the individual workflow steps. Apps are scripts, most often written in Python, that communicate with external services. Through API requests and webhooks, Shuffle is connected to external systems such as Elasticsearch, EDR or e-mail services.

2.2.2 Basic integration links of the Shuffle component in the PoC

Services currently connected and tested using Shuffle:

- ELK – we can retrieve data using API calls, search indices and create new indices. However, the IP of the device on which Shuffle runs must be allowed in iptables on the device where ELK runs.
- Discord – we can send notifications in JSON format using a webhook that is created in a group on Discord.
- OpenCTI – we can create new indicators using the API; we create these as JSON and send them using an API call.

- DFIR IRIS – using the API we can create new events, which we can also send in JSON format.

2.3 Wazuh

Wazuh is used within the INNOSOC PoC solution as the central platform for the collection, correlation and analysis of security events, with EDR functionality.

2.3.1 Deployment model of the Wazuh component

The architecture of our solution is based on deploying the universal Wazuh agent on target systems and on three key central components:

- Wazuh-manager (server): Provides data analysis, agent management and threat detection.
- Wazuh-indexer (database): A high-performance full-text search and analytics engine used for indexing and storing security records.
- Wazuh-dashboard: A web user interface designed for data visualization, incident analysis and system configuration

The central components are deployed on a virtual machine (VM) in the XenOrchestra environment. The VM has the following available: 8 virtual CPUs, 8GB of RAM and 250GB of storage.

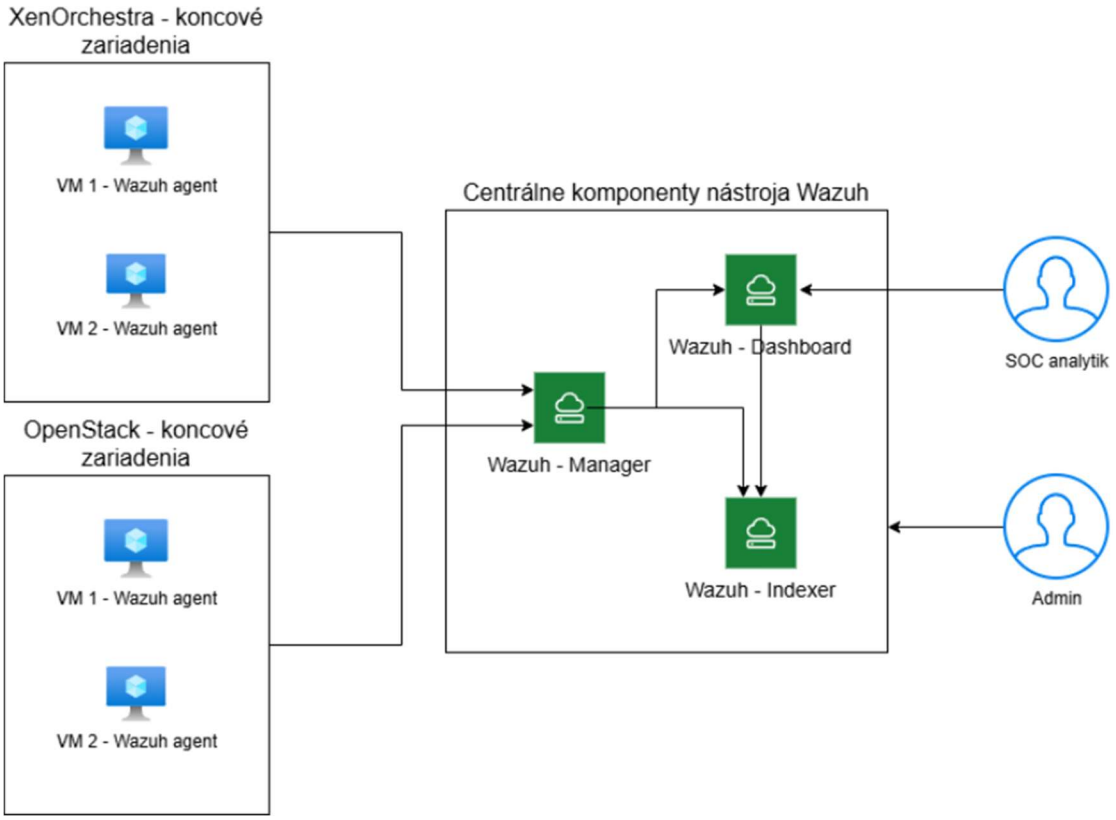
2.3.2 Environments monitored by Wazuh

We monitor the environments of two tenants. The first are the cloud environments XenOrchestra and OpenStack at the Department of Information Networks at the Faculty of Management Science and Informatics. The second is the environment of a selected primary school in Žilina.

2.3.2.1 Environment of the Department of Information Networks, FRI UNIZA

At KIS, the system is configured as an All-in-one installation, with all key components of the Wazuh platform integrated into a single virtual machine (VM). This solution provides centralized collection and analysis of data from the monitored nodes located in the XenOrchestra and OpenStack environments. A graphical representation of the resulting network architecture and component hierarchy is illustrated in the figure below.

Since this is an All-in-one deployment method, all central components have the same IP address: 192.168.1.184. In the case of a Single or multi-node installation, each component needs to have its own IP address (since each component will be on a separate server or VM).



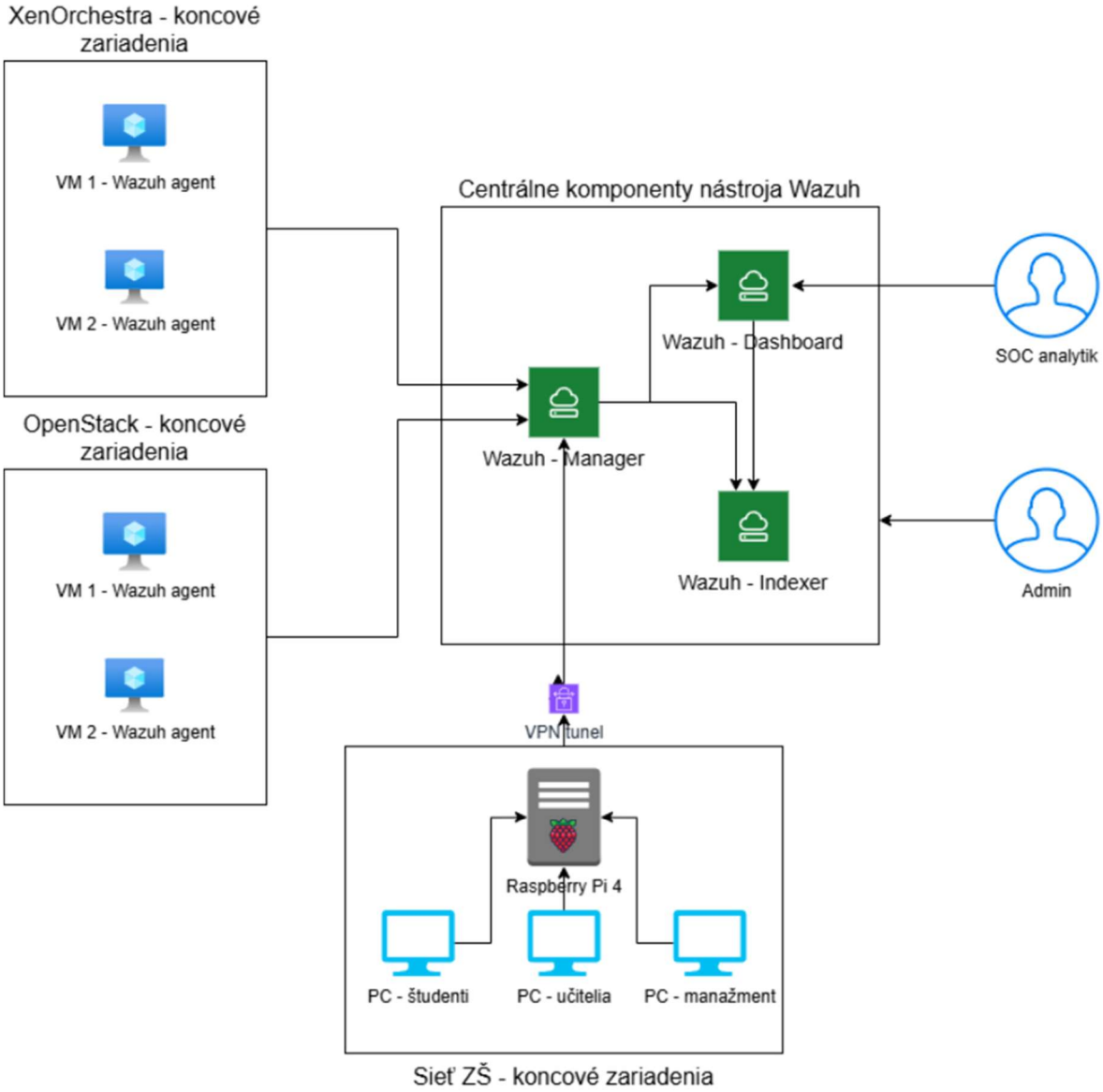
2.3.2.2 *Monitoring of a selected primary school (ZŠ) in Žilina*

The second instance of the monitoring system is implemented in the infrastructure of a selected primary school (ZŠ) in Žilina. The original architecture in this environment consisted of a local server on which the basic components of the Wazuh platform were deployed. This server provided the collection and storage of data from endpoint devices within the school's local network, while remote access to the Wazuh Dashboard interface was provided via the ZeroTier VPN technology.

Owing to inefficient administration and the need for redundant configuration of two separate managers, the architecture was revised. It was decided to centralize the main components in the XenOrchestra environment at KIS FRI. Data from endpoint devices located at the school are now redirected to and processed centrally at KIS.

The interconnection of the two sites is technically implemented as follows:

- **KIS side:** In addition to the Wazuh manager, the server components of the WireGuard VPN are also operated on the central server.
- **School side:** A Raspberry Pi 4 device acting as the VPN client is deployed in the local network.
- **Network connectivity:** A functional VPN tunnel with the address range $10.0.0.0/24$ is established between the sites.
- **Routing:** A static route is defined on the local router at the school, which routes all traffic destined for the `10.0.0.0/24` network through the Raspberry Pi gateway.



- **Wazuh agents** installed on the end devices at the primary school send the collected data directly to the Wazuh manager in the KIS network via a secure tunnel.

2.3.3 Deployed and activated Wazuh modules

In the initial phase, native Wazuh modules were activated and configured, forming the basic framework for security monitoring and compliance. Selected functions are specified in detail below:

2.3.3.1 *File Integrity Monitoring (FIM)*

This module monitors changes in critical system files and directories. It works on the principle of comparing checksums and file attributes over time.

- **Mechanism:** On initialization, the agent creates a database of hashes of selected files. On every change (creation, modification, deletion), it sends an alert to the manager.
- **Use:** Detection of unauthorized configuration changes or malware injection.

2.3.3.2 *Rootcheck (Rootkit Detection)*

A module focused on detecting hidden threats such as rootkits, Trojan horses or unauthorized modifications of the operating system kernel.

- **Mechanism:** It performs an in-depth check for anomalies in the system, compares the outputs of system calls with a direct reading of the original (so-called raw) data and searches for hidden processes or ports.

2.3.3.3 *Vulnerability Detection*

This module automatically identifies weaknesses in the operating system and installed applications.

- **Mechanism:** The Wazuh agent collects a list of installed packages, which the manager then compares against global vulnerability databases (**CVE** – Common Vulnerabilities and Exposures) in the NVD - National Vulnerability Database.

2.3.3.4 *Policy monitoring & SCA (Security Configuration Assesment)*

It is used to verify whether the system is configured in compliance with security standards (e.g. CIS Benchmarks).

- **Mechanism:** The module runs scripts that check system settings (e.g. password strength, disabled protocols). The result is a compliance score and a list of remediation recommendations.

2.3.3.5 *System Inventory*

It provides a comprehensive overview of the hardware and software properties of the monitored endpoint.

Mechanism: It collects data on network interfaces, running processes, installed packages and OS versions. These data are available in real time via wazuh-dashboard.

2.3.4 Basic integration links – sending notifications

The Wazuh ecosystem offers a wide range of integration options for the automated sending of alerts via external communication channels and third-party tools.

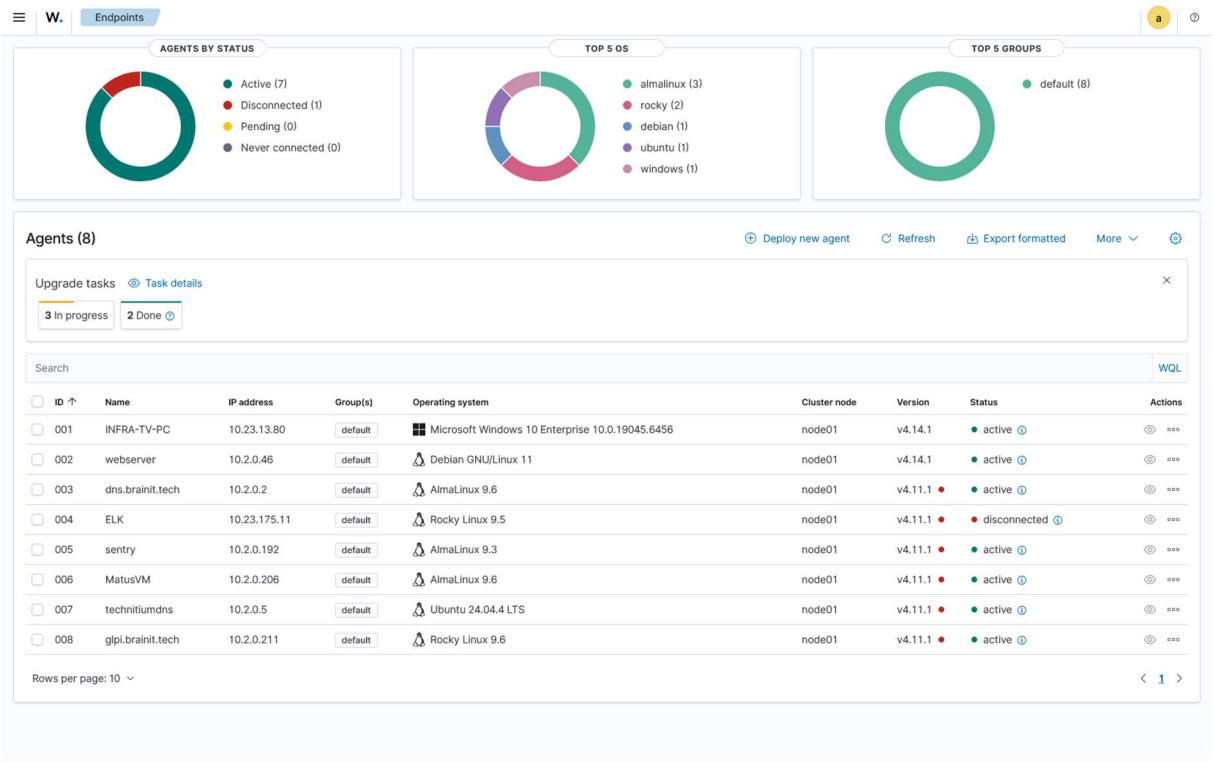
Sending notifications to external platforms (Slack, MS Teams, PagerDuty, etc.) is handled in Wazuh through the Integrations module. The integration is implemented via the Webhook application interface, with the system enabling granular filtering of the events sent based on their severity. This process makes it possible to transform internal alerts into immediate messages for the security team, thereby significantly shortening the response time (MTTR – Mean Time to Respond).

We analysed various tools for sending notifications, such as Slack, PagerDuty and MS Teams. For monitoring purposes, we decided to use MS Teams. The integration is built on the use of an Incoming Webhook (HTTP endpoint). The Wazuh platform sends data in JSON format, which are transformed in Teams into visually structured cards (Adaptive Cards). This format enables a clear display of key metrics such as the severity level, agent identification, rule ID and a detailed description of the event.

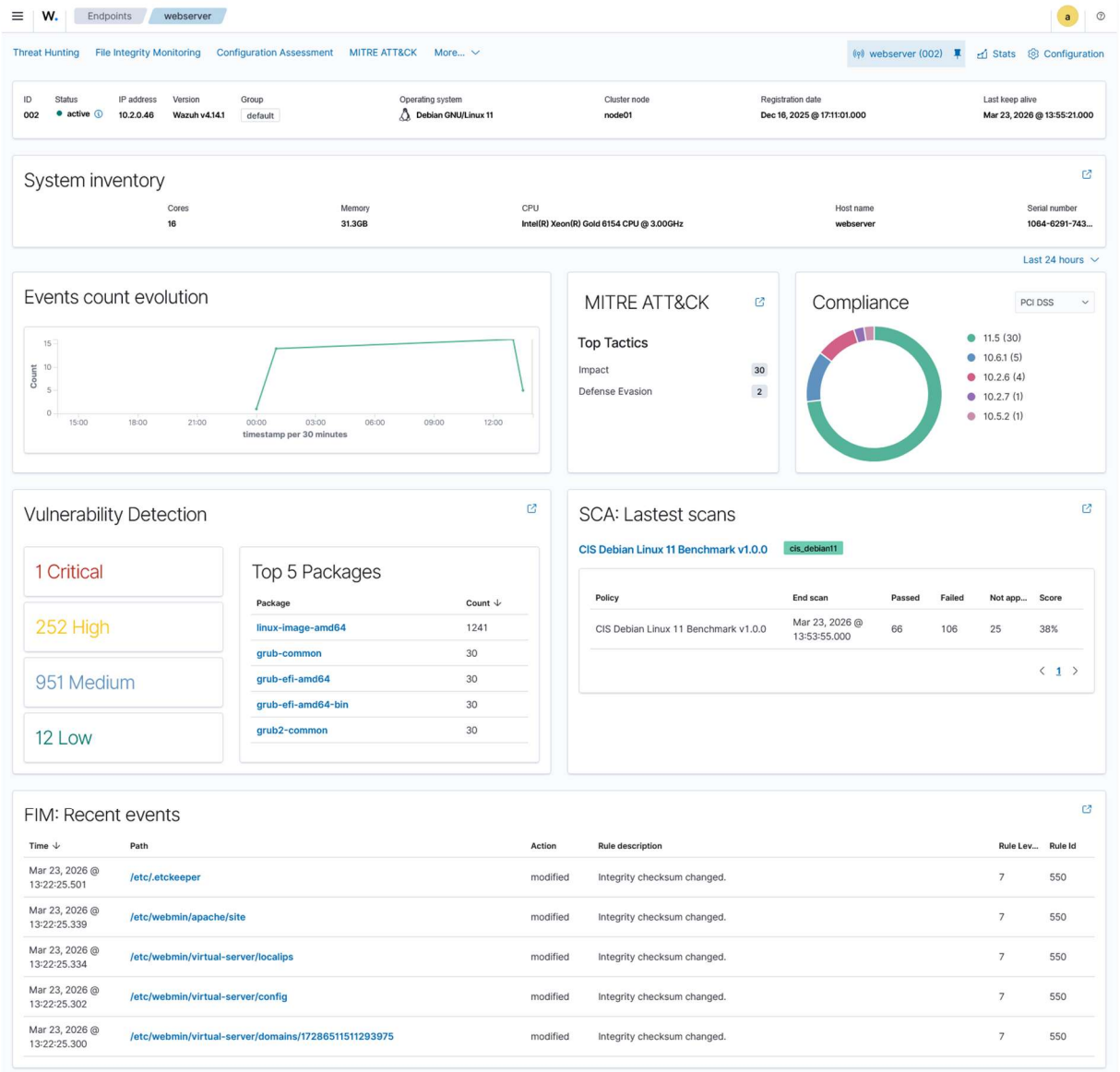
The main reason for this choice was the fact that this platform is fully integrated into the university's existing digital infrastructure. Using the Microsoft 365 ecosystem brings us several advantages:

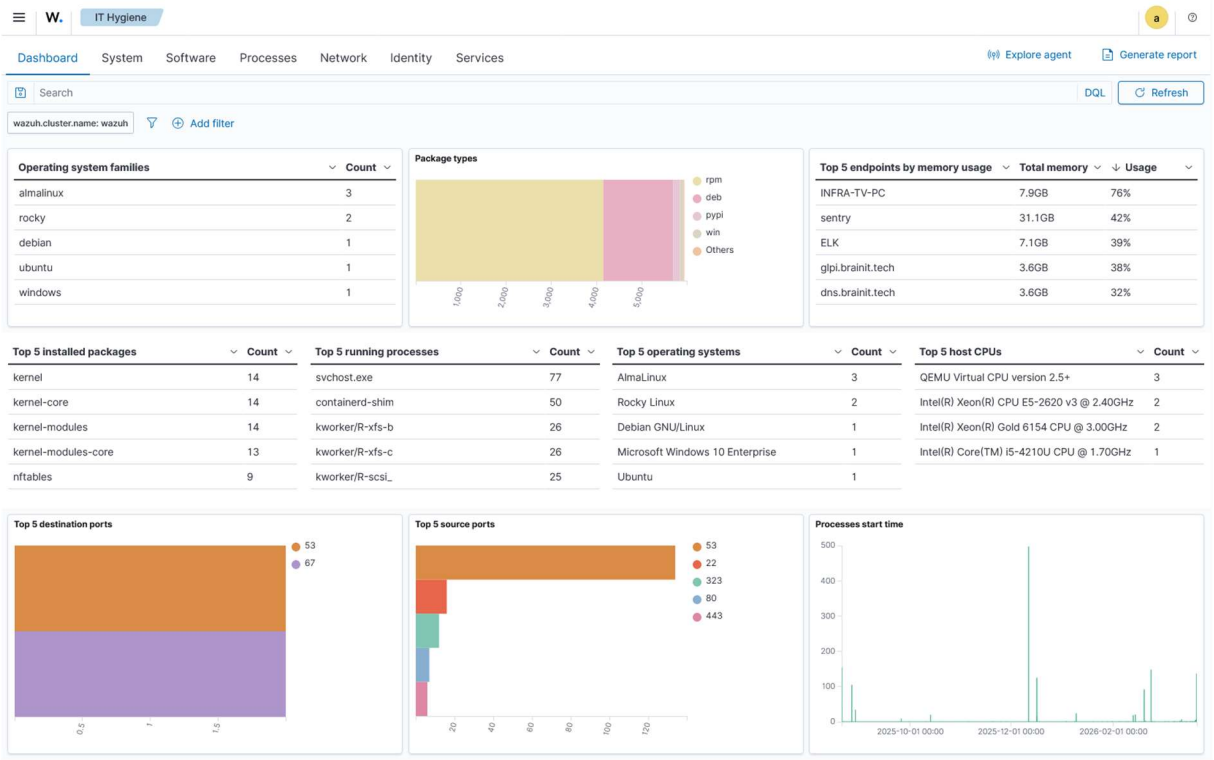
- **Deployment efficiency:** There is no need to build a new communication infrastructure or to manage additional user accounts.
- **Connectivity:** The integration takes place within the university's verified and secured environment.
- **Availability:** It provides immediate access to notifications for all team members without the need to install additional third-party software.

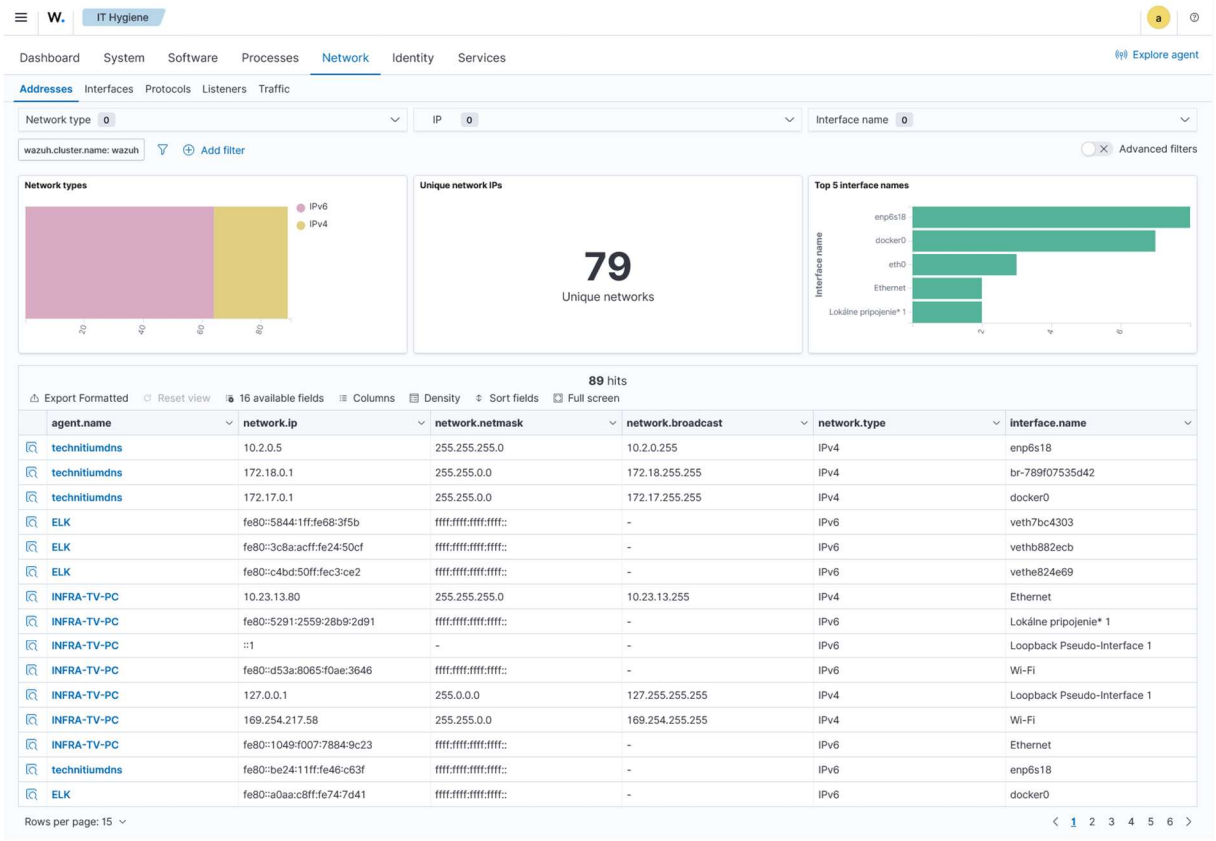
Example from the Wazuh dashboard:

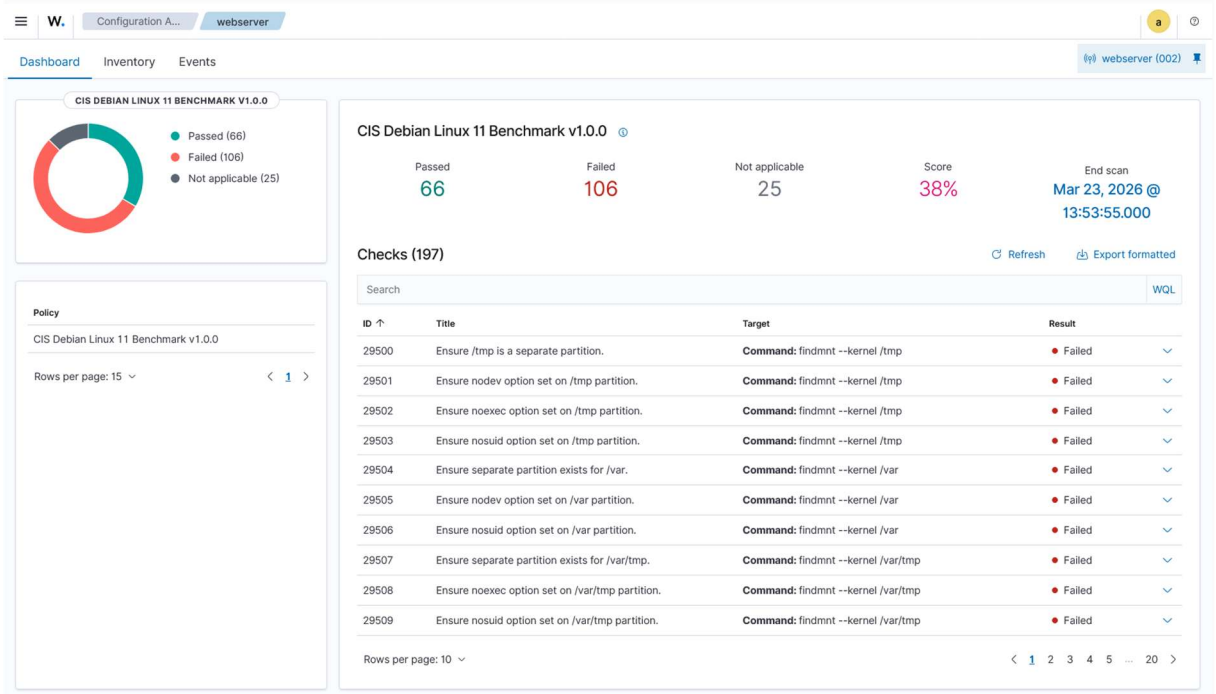


Funded by the European Union
NextGenerationEU
RECOVERY AND RESILIENCE PLAN
GOVERNMENT OFFICE OF THE SLOVAK REPUBLIC


Endpoints
webserver
Threat Hunting
File Integrity Monitoring
Configuration Assessment
MITRE ATT&CK
More...
webserver (002)
Stats
Configuration
ID 002
Status active
IP address 10.2.0.46
Version Wazuh v4.14.1
Group default
Operating system Debian GNU/Linux 11
Cluster node node01
Registration date Dec 16, 2025 @ 17:11:01.000
Last keep alive Mar 23, 2026 @ 13:55:21.000
System inventory
Cores 16
Memory 31.3GB
CPU Intel(R) Xeon(R) Gold 6154 CPU @ 3.00GHz
Host name webserver
Serial number 1064-6291-743...
Last 24 hours
Events count evolution
Count
timestamp per 30 minutes
MITRE ATT&CK
Top Tactics
Impact 30
Defense Evasion 2
Compliance
PCI DSS
11.5 (30)
10.6.1 (5)
10.2.6 (4)
10.2.7 (1)
10.5.2 (1)
Vulnerability Detection
1 Critical
252 High
951 Medium
12 Low
Top 5 Packages
Package
Count
linux-image-amd64 1241
grub-common 30
grub-efi-amd64 30
grub-efi-amd64-bin 30
grub2-common 30
SCA: Lastest scans
CIS Debian Linux 11 Benchmark v1.0.0
cis_debian11
Policy
End scan
Passed
Failed
Not app...
Score
CIS Debian Linux 11 Benchmark v1.0.0
Mar 23, 2026 @ 13:53:55.000
66
106
25
38%
FIM: Recent events
Time
Path
Action
Rule description
Rule Lev...
Rule Id
Mar 23, 2026 @ 13:22:25.501 /etc/.etckeeper modified Integrity checksum changed. 7 550
Mar 23, 2026 @ 13:22:25.339 /etc/webmin/apache/site modified Integrity checksum changed. 7 550
Mar 23, 2026 @ 13:22:25.334 /etc/webmin/virtual-server/localips modified Integrity checksum changed. 7 550
Mar 23, 2026 @ 13:22:25.302 /etc/webmin/virtual-server/config modified Integrity checksum changed. 7 550
Mar 23, 2026 @ 13:22:25.300 /etc/webmin/virtual-server/domains/17286511511293975 modified Integrity checksum changed. 7 550

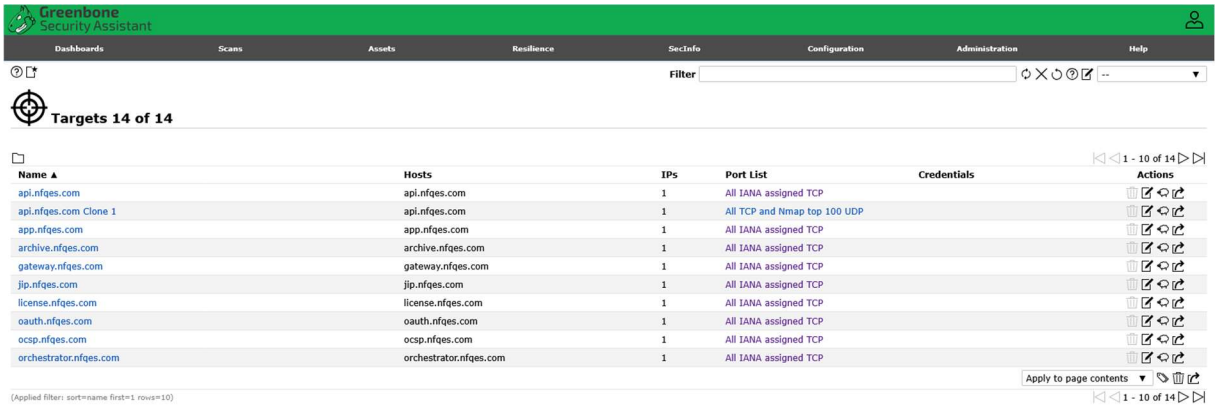

IT Hygiene
Dashboard
System
Software
Processes
Network
Identity
Services
Explore agent
Generate report
Search
DQL
Refresh
wazuh.cluster.name: wazuh
Add filter
Operating system families
Count
almalinux 3
rocky 2
debian 1
ubuntu 1
windows 1
Package types
rpm
deb
pypi
win
Others
1,000
2,000
3,000
4,000
5,000
Top 5 endpoints by memory usage
Total memory
Usage
INFRA-TV-PC 7.9GB 76%
sentry 31.1GB 42%
ELK 7.1GB 39%
glpi.brainit.tech 3.6GB 38%
dns.brainit.tech 3.6GB 32%
Top 5 installed packages
Count
kernel 14
kernel-core 14
kernel-modules 14
kernel-modules-core 13
nftables 9
Top 5 running processes
Count
svchost.exe 77
containerd-shim 50
kworker/R-xfs-b 26
kworker/R-xfs-c 26
kworker/R-scsi_ 25
Top 5 operating systems
Count
AlmaLinux 3
Rocky Linux 2
Debian GNU/Linux 1
Microsoft Windows 10 Enterprise 1
Ubuntu 1
Top 5 host CPUs
Count
QEMU Virtual CPU version 2.5+ 3
Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz 2
Intel(R) Xeon(R) Gold 6154 CPU @ 3.00GHz 2
Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz 1
Top 5 destination ports
53
67
0.5
1
1.5
Top 5 source ports
53
22
323
80
443
20
40
60
80
100
120
Processes start time
500
400
300
200
100
0
2025-10-01 00:00
2025-12-01 00:00
2026-02-01 00:00



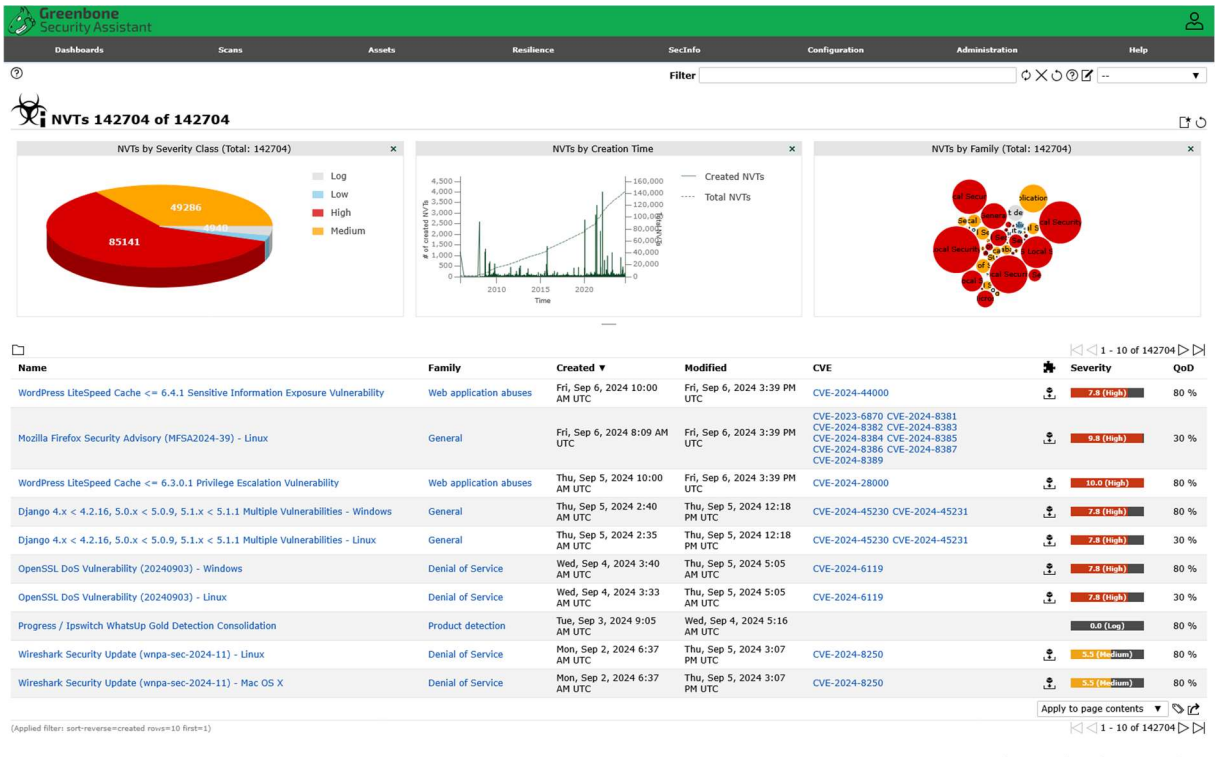


Funded by the European Union
NextGenerationEU
RECOVERY AND RESILIENCE PLAN
GOVERNMENT OFFICE OF THE SLOVAK REPUBLIC

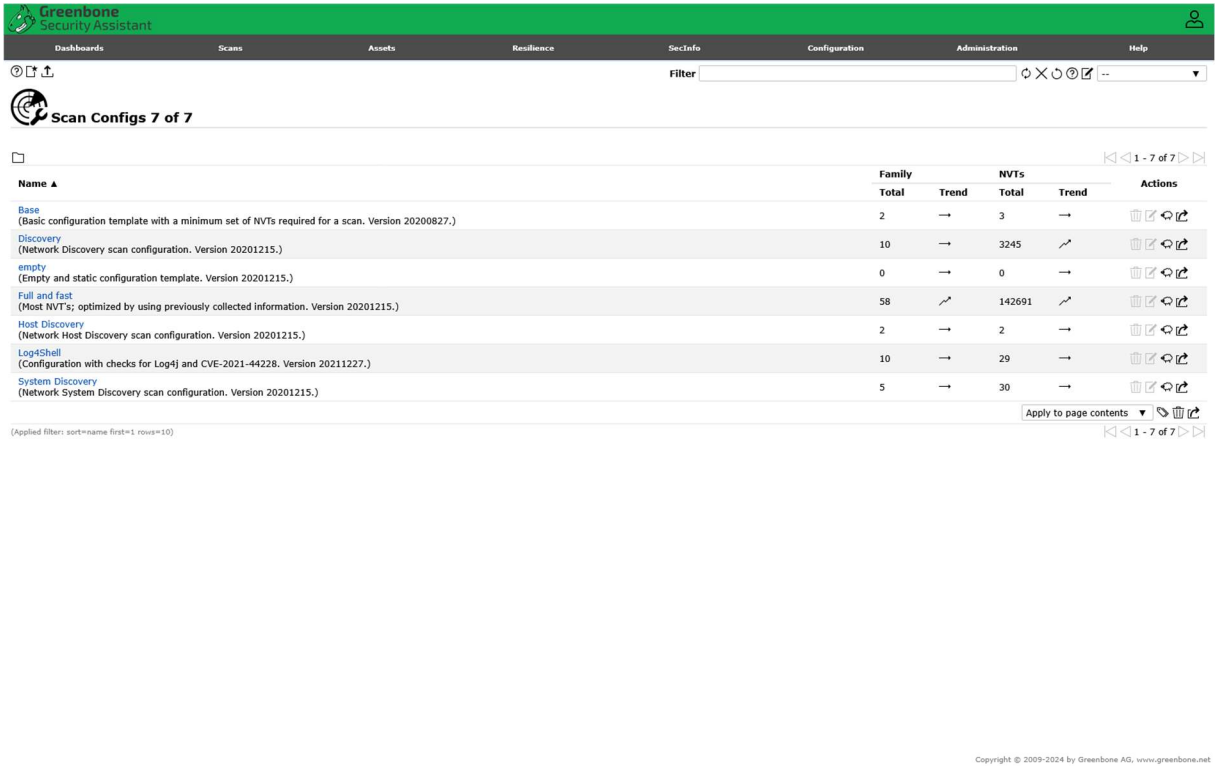
2.4 Greenbone Community edition



Copyright © 2009-2024 by Greenbone AG, www.greenbone.net



Copyright © 2009-2024 by Greenbone AG, www.greenbone.net



2.5 OpenCTI

OpenCTI is used within the INNOSOC PoC solution as the central threat repository, where indicators of compromise (IoC), TTP techniques, campaigns and actors are collected. The platform makes it possible to analyse these data and subsequently distribute them to other security tools without the need for manual intervention.

OpenCTI (Open Cyber Threat Intelligence Platform) is an open-source platform designed for managing, analysing and sharing cyber threat information (CTI). It makes it possible to aggregate data from various sources, correlate them and transform them into a unified model based on the STIX standard.

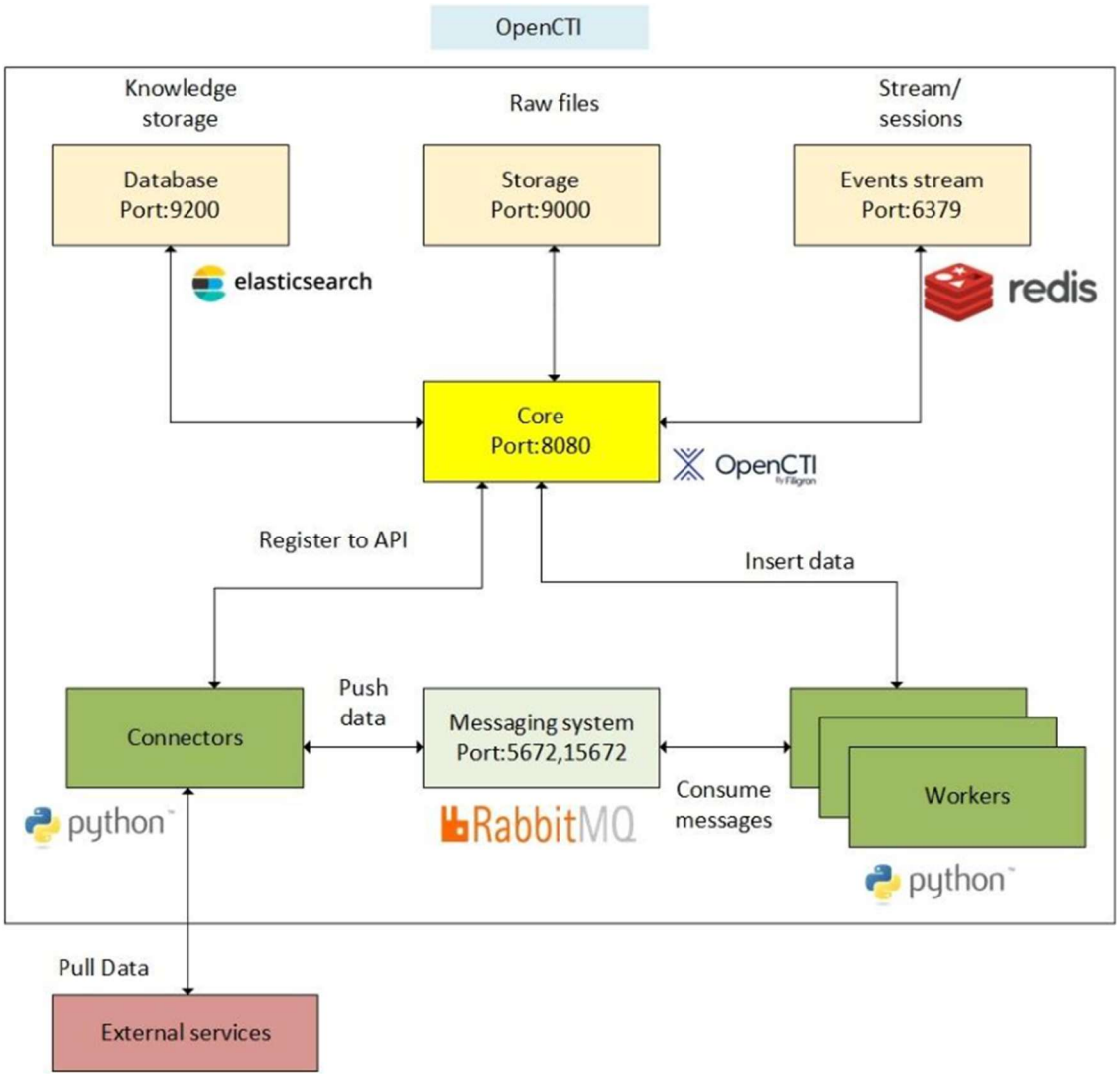
2.5.1 Deployment of Shuffle in the OpenStack environment

Technical specification of the Shuffle deployment:

- Environment – OpenStack
- Instance name – OpenCTI_s
- IP address – 192.168.4.121
- vCPU – 4
- RAM – 16 GB
- Disk – 80 GB
- OS – Ubuntu 22.04

The diagram in the figure illustrates the architecture of the OpenCTI platform on the server and the data processing flow between the individual components. The central element

is OpenCTI Core running on port 8080, which provides the application logic and communication between services. Core communicates with the Elasticsearch database on port 9200, where all CTI data are stored, as well as with the file storage on port 9000, which is used to store raw data and attachments. Redis on port 6379 is used to manage events and the cache.



The runtime environment contains the RabbitMQ messaging system (ports 5672 and 15672), which provides asynchronous communication between components. Connectors retrieve data from external sources, register via the API with Core and then send the data to RabbitMQ. Workers process these messages, perform data transformation and enrichment, and then store them back in OpenCTI via Core.

2.5.2 Basic integration links for the OpeCTI component

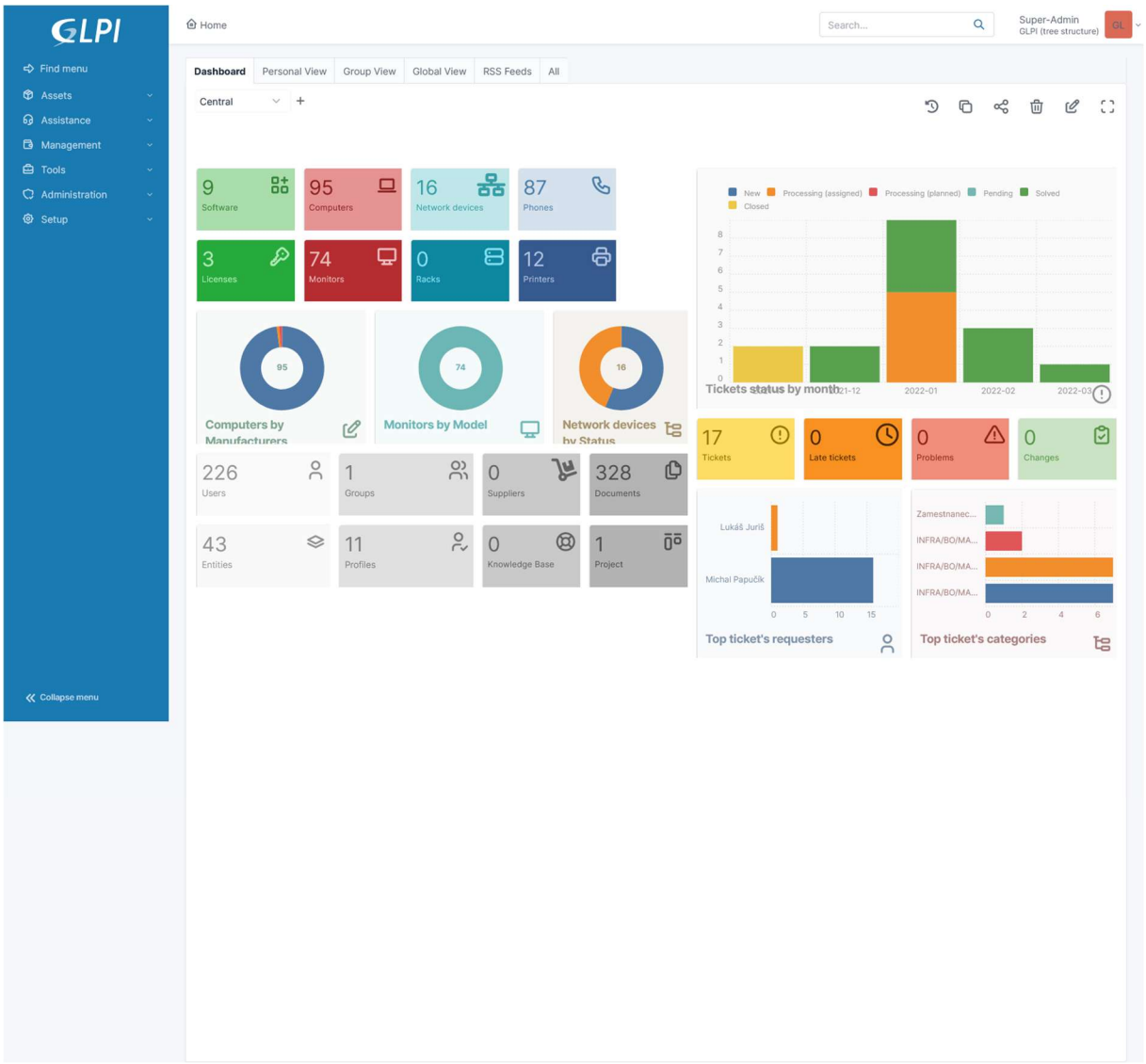
Services currently connected and tested using OpenCTI:

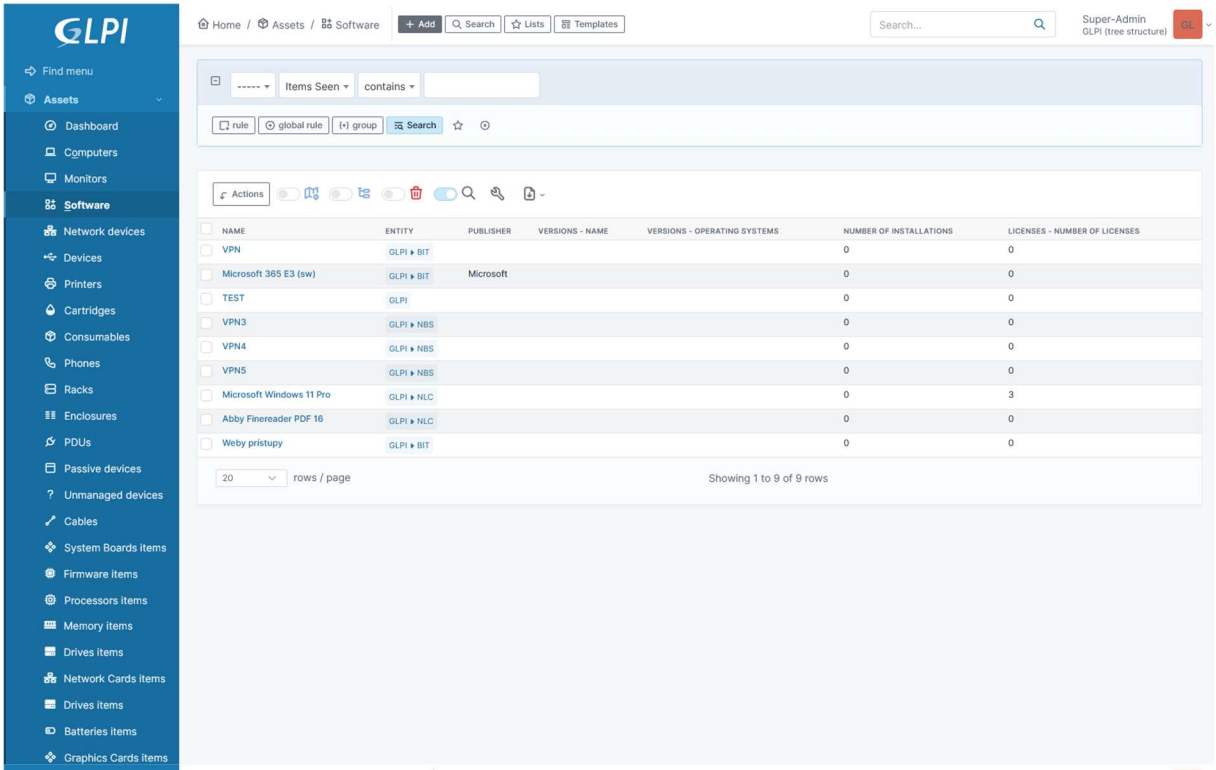
- **Abuse.ch (ThreatFox, URLhaus, Malwarebazzar)** – automatic import of IoCs
- **MITRE ATT&CK** – mapping of techniques and tactics
- **SOAR (Shuffle)** – automation of incident response

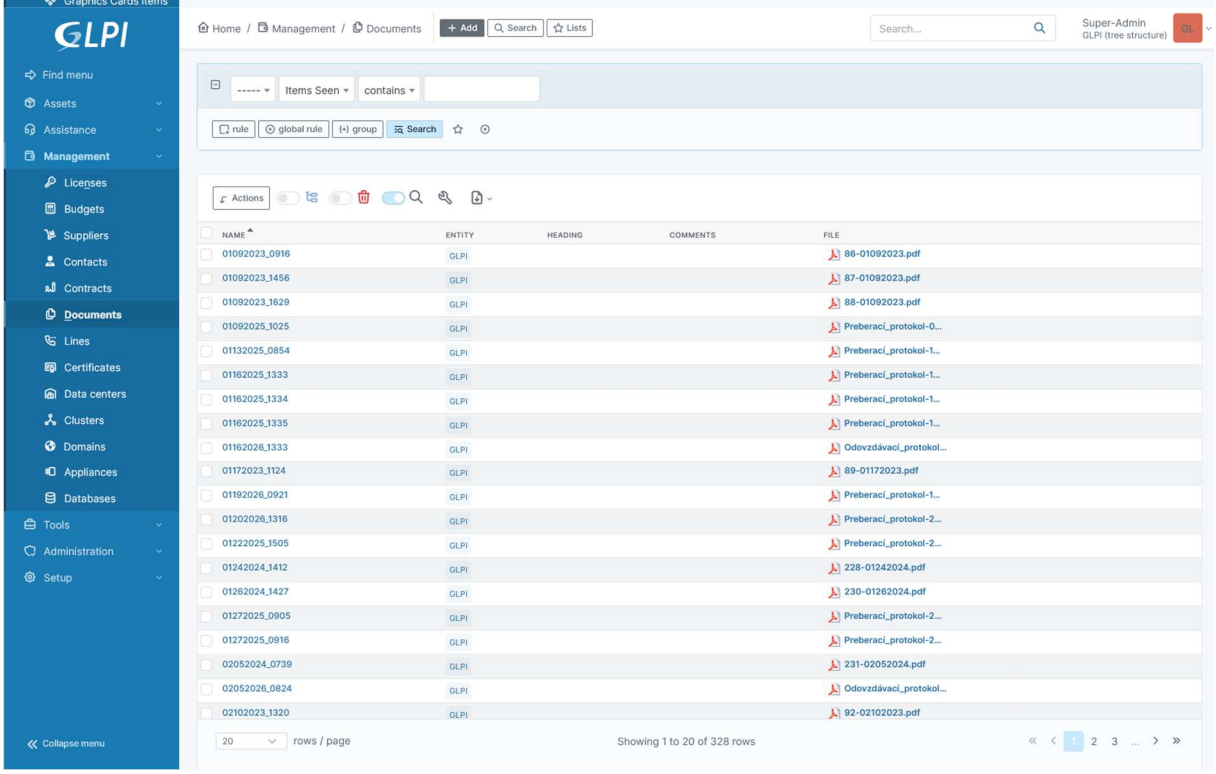
2.6 MISP

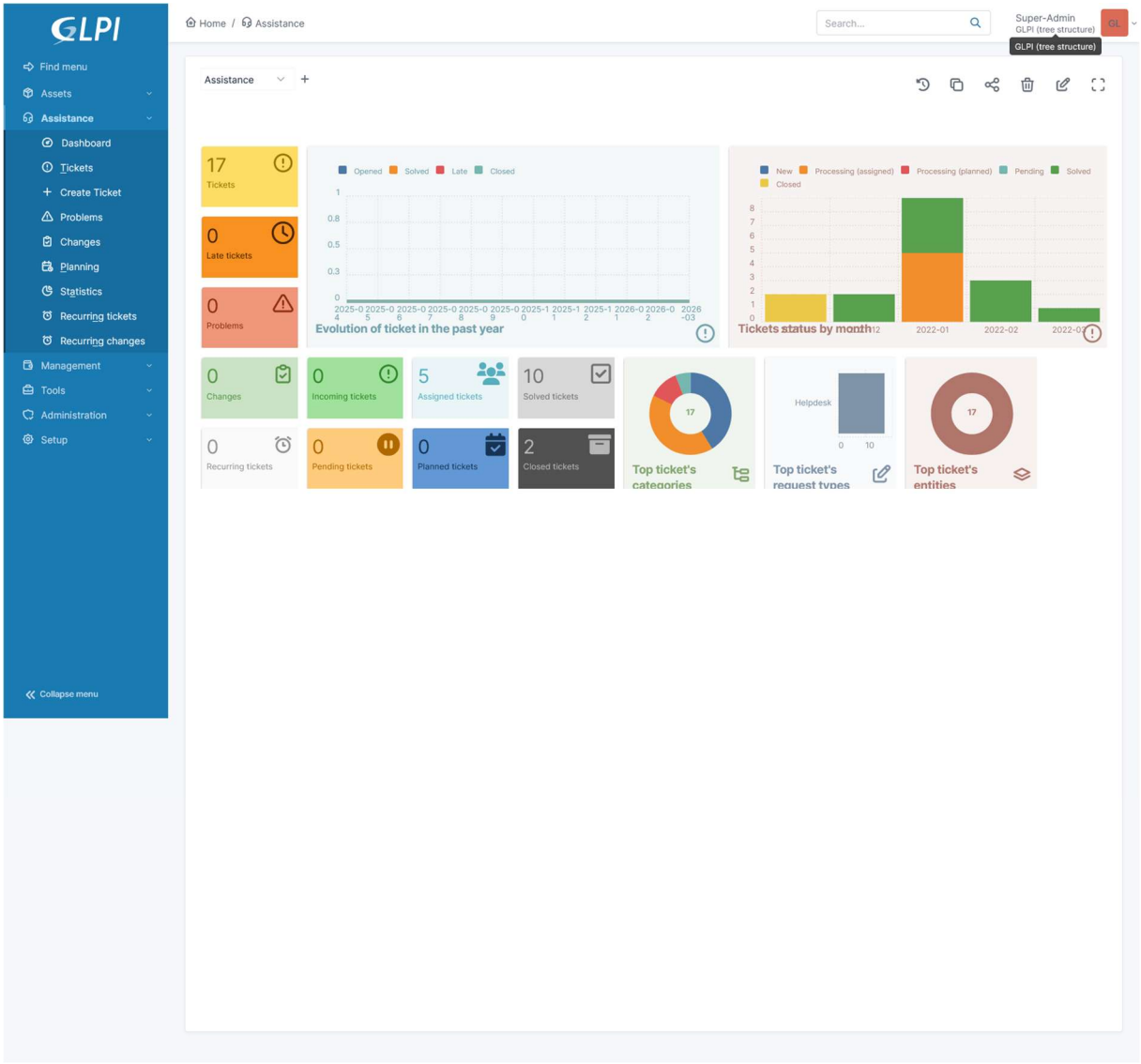
There will not be any, we have nothing; the CTI component will be covered only by the OpenCTI platform.

2.7 GLPI




GLPI
Home / Assets / Software
+ Add
Search
Lists
Templates
Search...
Super-Admin
GLPI (tree structure)
Find menu
Assets
Dashboard
Computers
Monitors
Software
Network devices
Devices
Printers
Cartridges
Consumables
Phones
Racks
Enclosures
PDUs
Passive devices
Unmanaged devices
Cables
System Boards items
Firmware items
Processors items
Memory items
Drives items
Network Cards items
Drives items
Batteries items
Graphics Cards items
Items Seen
contains
rule
global rule
group
Search
Actions
NAME
ENTITY
PUBLISHER
VERSIONS - NAME
VERSIONS - OPERATING SYSTEMS
NUMBER OF INSTALLATIONS
LICENSES - NUMBER OF LICENSES
VPN
Microsoft 365 E3 (sw)
TEST
VPN3
VPN4
VPN5
Microsoft Windows 11 Pro
Abby Finereader PDF 16
Weby prístupy
20
rows / page
Showing 1 to 9 of 9 rows


GLPI
Home / Management / Documents
+ Add
Search
Lists
Search...
Super-Admin
GLPI (tree structure)
Find menu
Assets
Assistance
Management
Licenses
Budgets
Suppliers
Contacts
Contracts
Documents
Lines
Certificates
Data centers
Clusters
Domains
Appliances
Databases
Tools
Administration
Setup
Collapse menu
Items Seen
contains
rule
global rule
group
Search
Actions
NAME
ENTITY
HEADING
COMMENTS
FILE
01092023_0916
01092023_1456
01092023_1629
01092025_1025
01132025_0854
01162025_1333
01162025_1334
01162025_1335
01162026_1333
01172023_1124
01192026_0921
01202026_1316
01222025_1505
01242024_1412
01262024_1427
01272025_0905
01272025_0916
02052024_0739
02052026_0824
02102023_1320
86-01092023.pdf
87-01092023.pdf
88-01092023.pdf
89-01172023.pdf
228-01242024.pdf
230-01262024.pdf
231-02052024.pdf
92-02102023.pdf
20
rows / page
Showing 1 to 20 of 328 rows



2.8 NetBox

2.8.1 Role of NetBox in the PoC

NetBox is used within the INNOSOC PoC solution as a supporting tool for recording and managing infrastructure context. In the proposed SOC solution, it does not act as a detection or response component, but serves as a source of information on assets, addressing, network segments and selected relationships between infrastructure elements. Its importance lies mainly in the fact that it makes it easier to understand which device, system or part of the network a particular event relates to. Within the PoC, NetBox is understood as a context layer that complements the technical data obtained from the other components of the solution. When analysing security events, it can thus be used to look up basic information about devices, IP ranges, sites or the logical placement of infrastructure elements. Such an approach increases

the clarity of the solution and creates the prerequisite for a more accurate assessment of security situations.

NetBox in the SOC offers the context of the environment in which security logs and alerts originate. In this sense, it represents a ***complementary*** but important component of the solution, which supports orientation within the infrastructure and lays the groundwork for further integration use in extended scenarios.

2.8.2 Deployment and basic configuration of NetBox

Within the PoC, NetBox was deployed in the OpenStack environment as a standalone service intended for recording and managing infrastructure data. From an implementation perspective, it was containerized, using the application and supporting components required for it to run, in particular the database layer and cache/queue services. This deployment approach corresponds to the nature of an experimental deployment and at the same time allows easier administration, updating and repeatable deployment.

In the PoC, the basic configuration of NetBox focused on making the web interface accessible and on preparing the data model for recording selected infrastructure objects. In terms of configuration, the system was prepared to work with a PostgreSQL database and with a separate Redis layer for tasks and cache. Within the PoC, basic objects were entered into the system, in particular sites, devices, address spaces and the related infrastructure context.

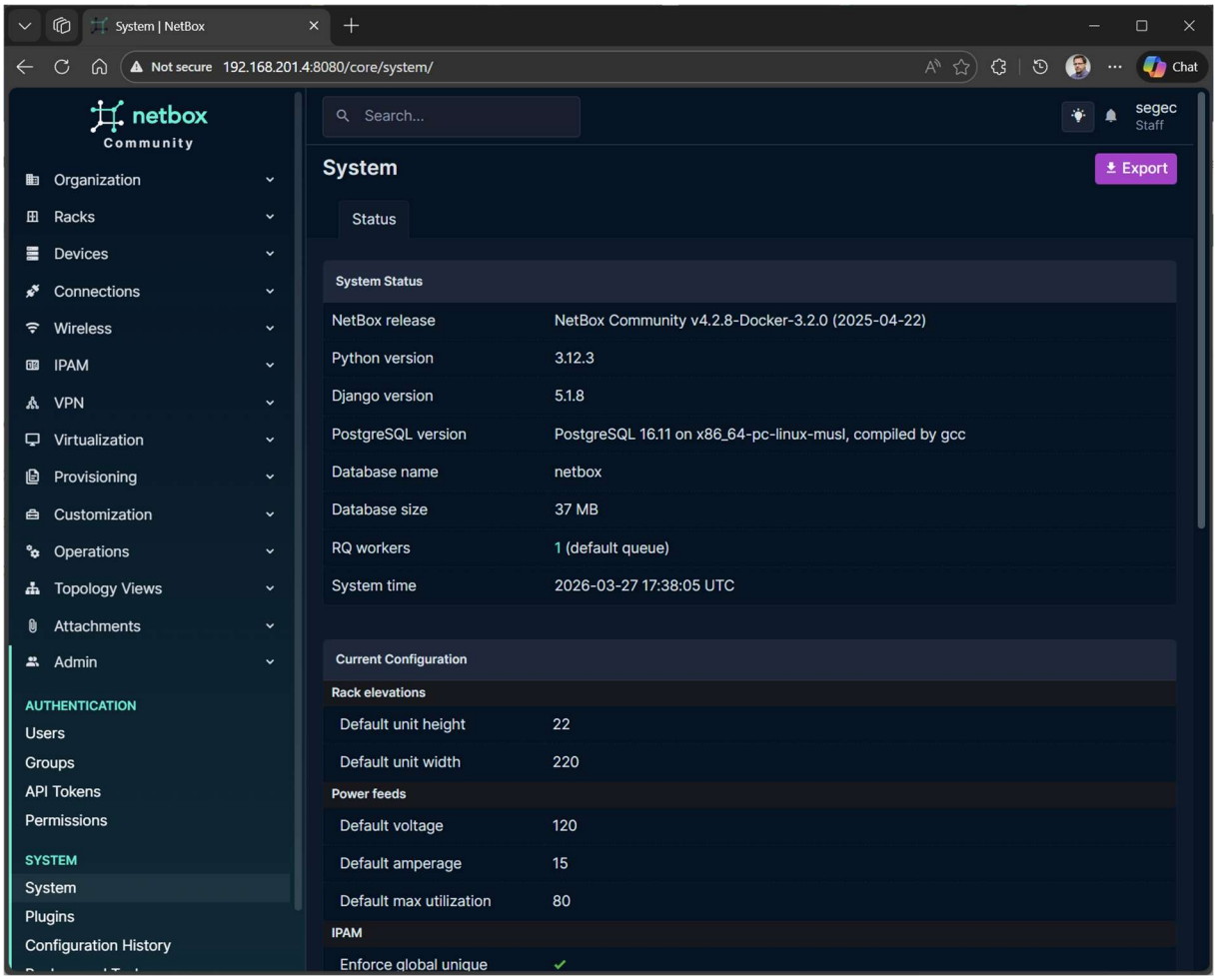


Figure 1 System information

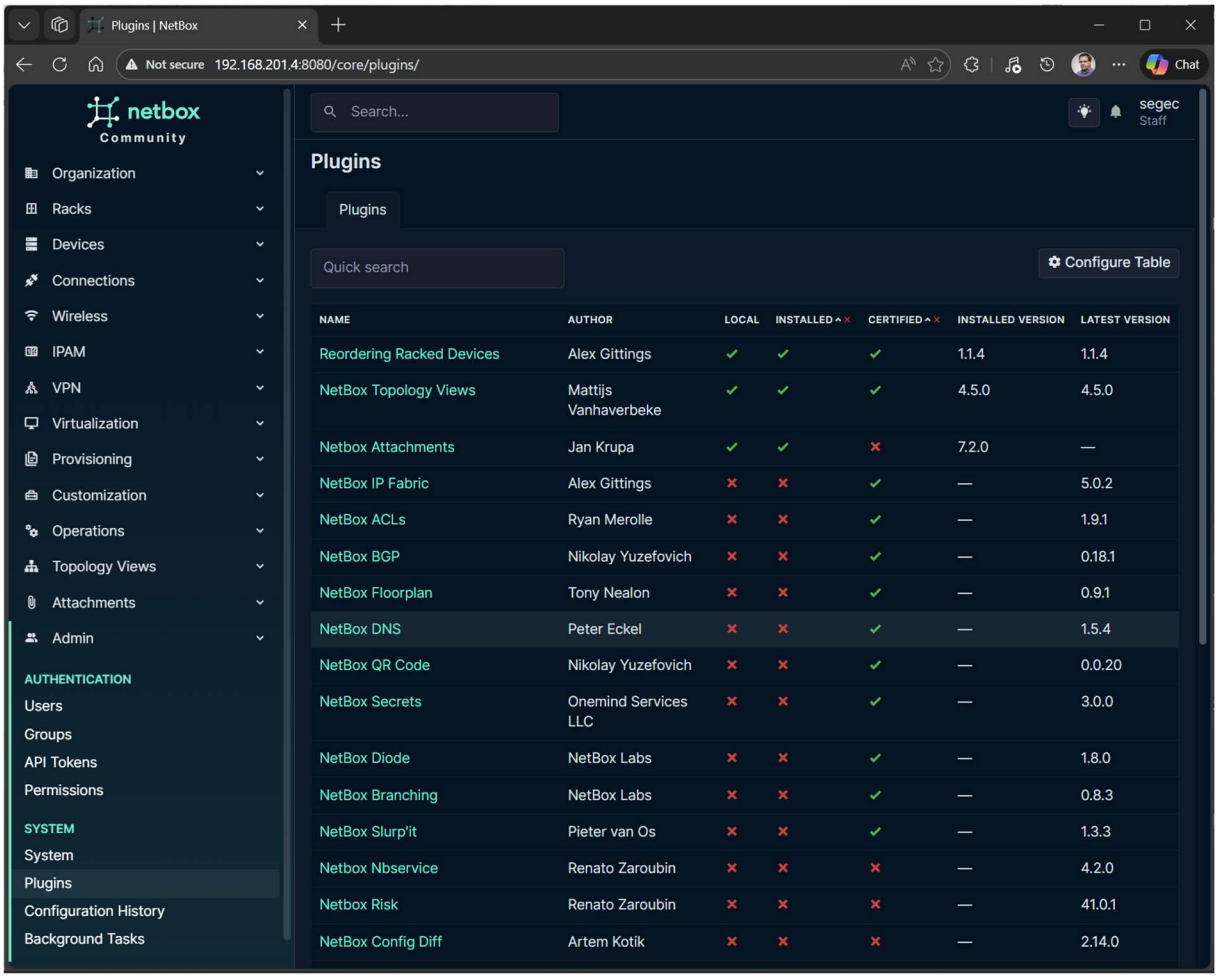


Figure 2 Plugins

The aim was not to create a complete production CMDB or a detailed model of the entire network, but to prepare a representative data set on which the use of NetBox in the proposed solution can be demonstrated.

2.8.3 Use of NetBox within the PoC

Within the PoC, NetBox was used as a supporting source of infrastructure context for the other parts of the solution. Its main contribution lay in the fact that it made it possible to look up basic information about devices, IP addressing, sites and network segments to which events or findings from other tools could be assigned.

Practical use of NetBox at this stage was oriented mainly towards record-keeping and manual or partially assisted context lookup. NetBox thus did not act as a detection or response tool, but complemented the solution with the data needed for a better understanding of the technical environment. To this extent, its deployment confirmed that it is a suitable complement to the PoC solution and a usable basis for further extension of integration scenarios.

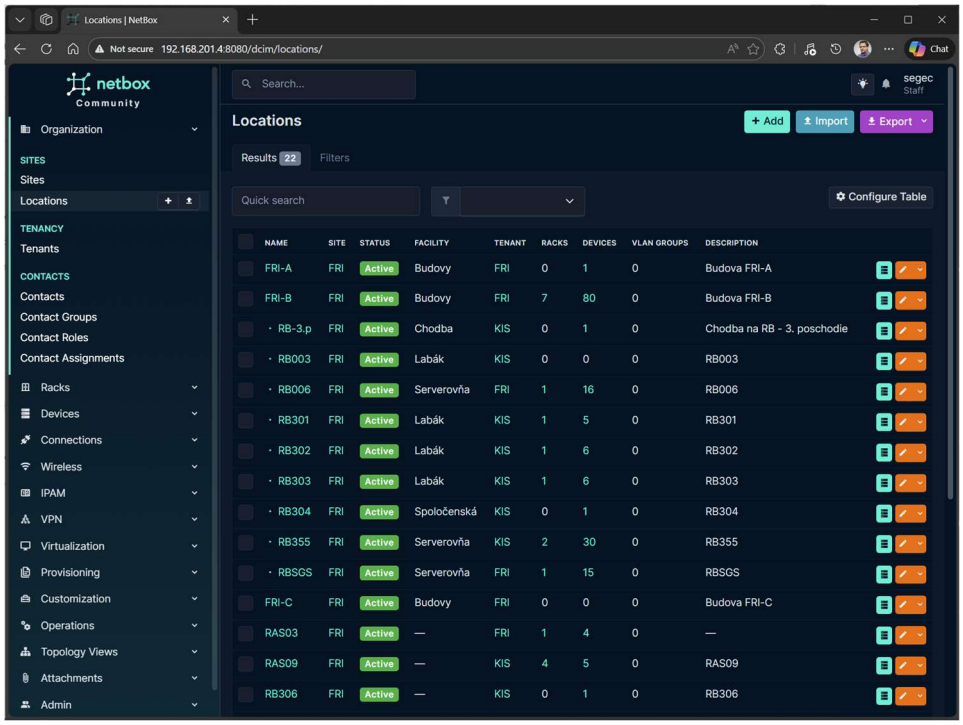


Figure 3 Modelling the organization

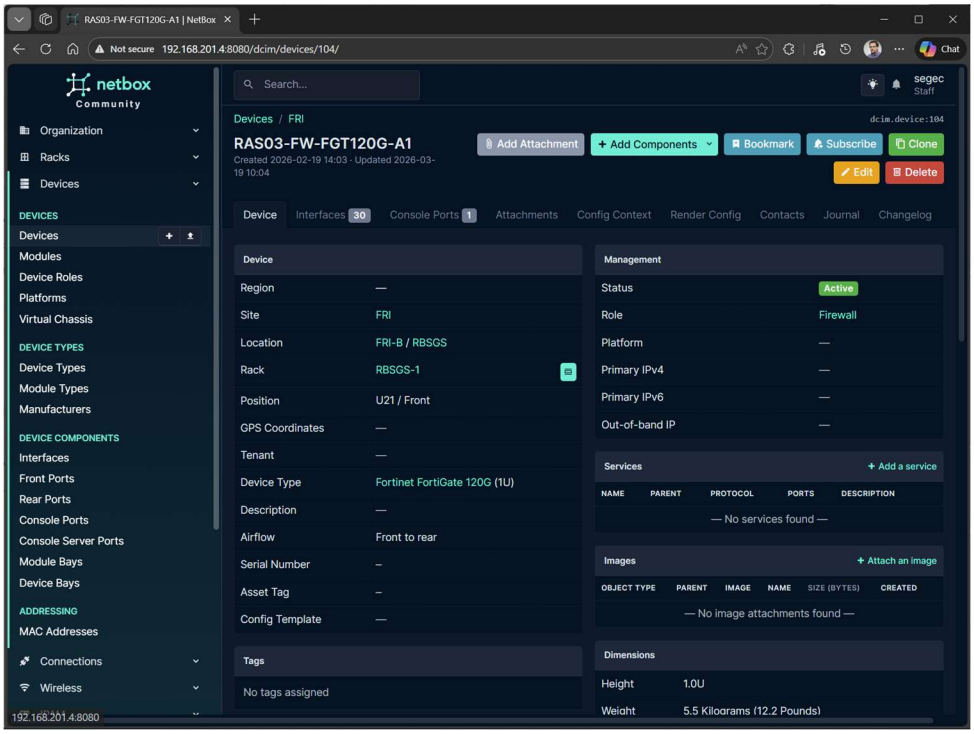


Figure 4 Modelling a device – Firewall

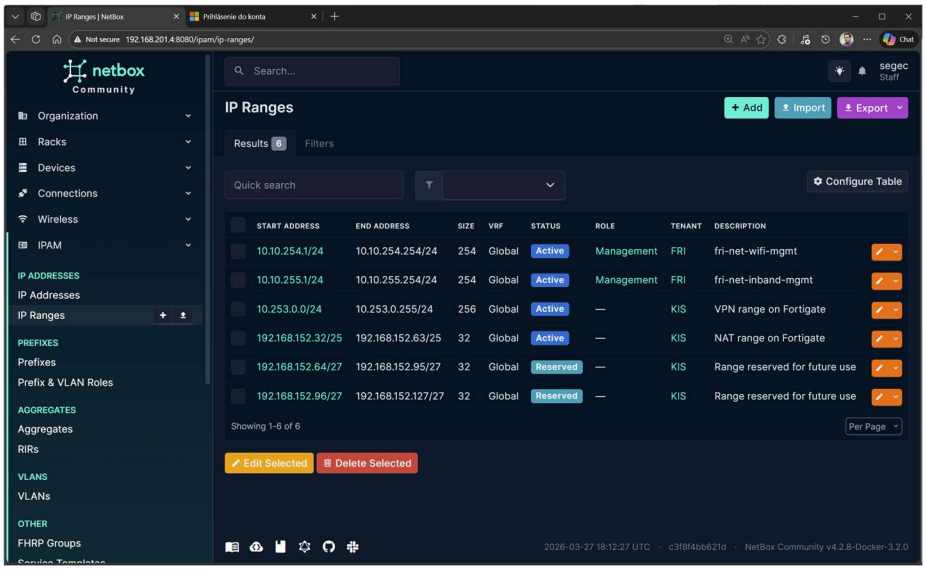
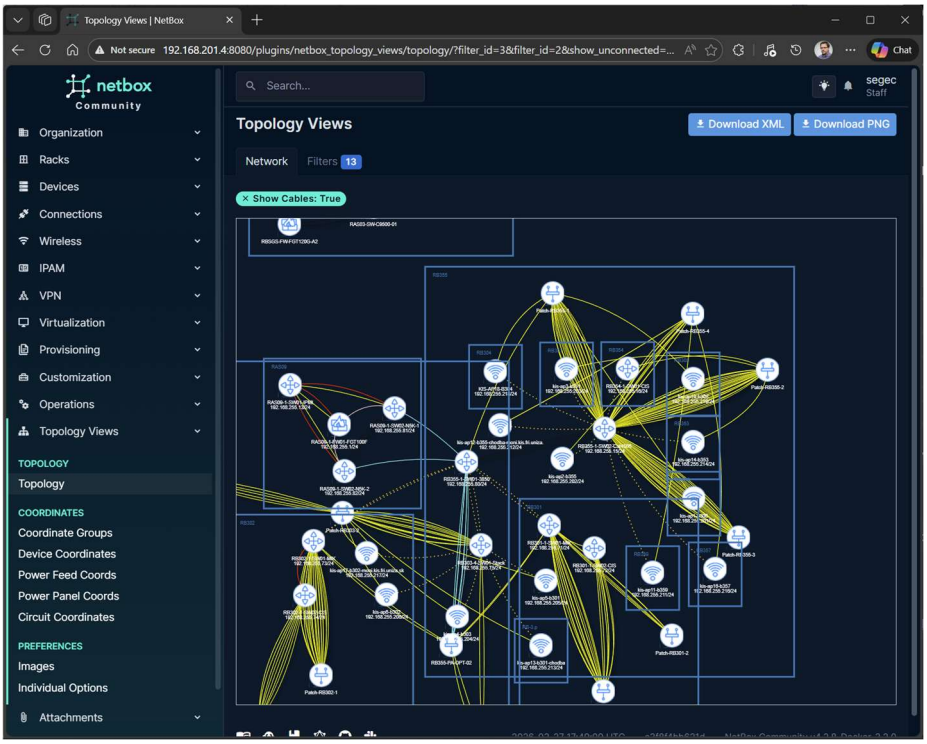


Figure 5 Address management



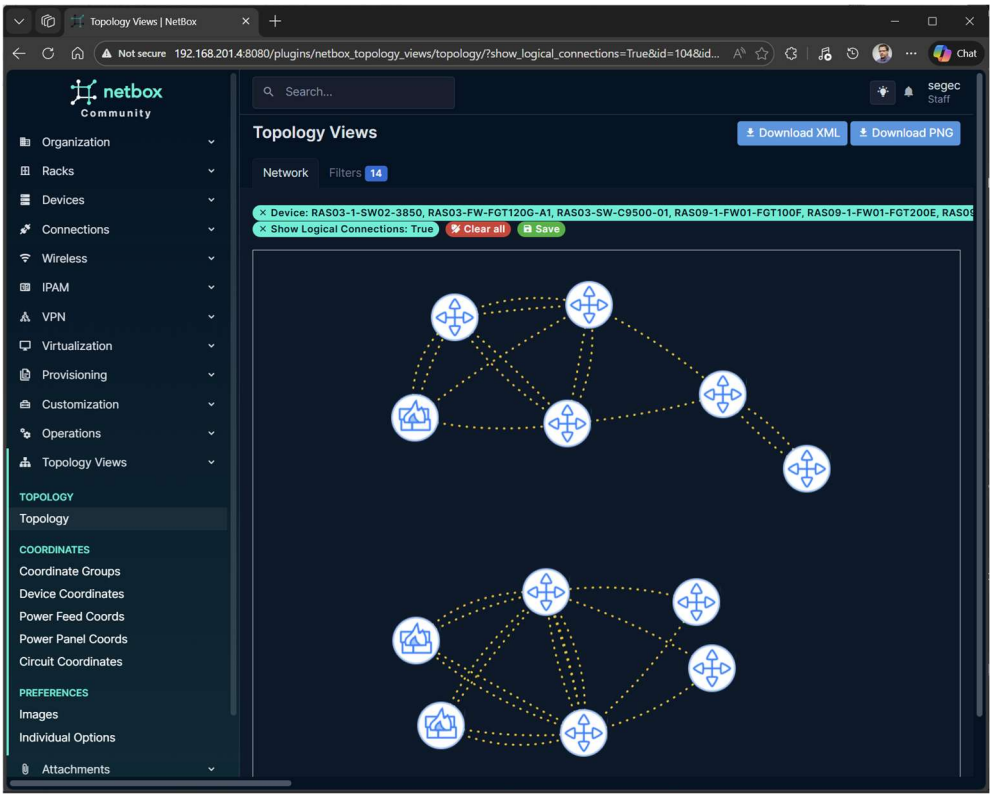


Figure 6 Topology generation

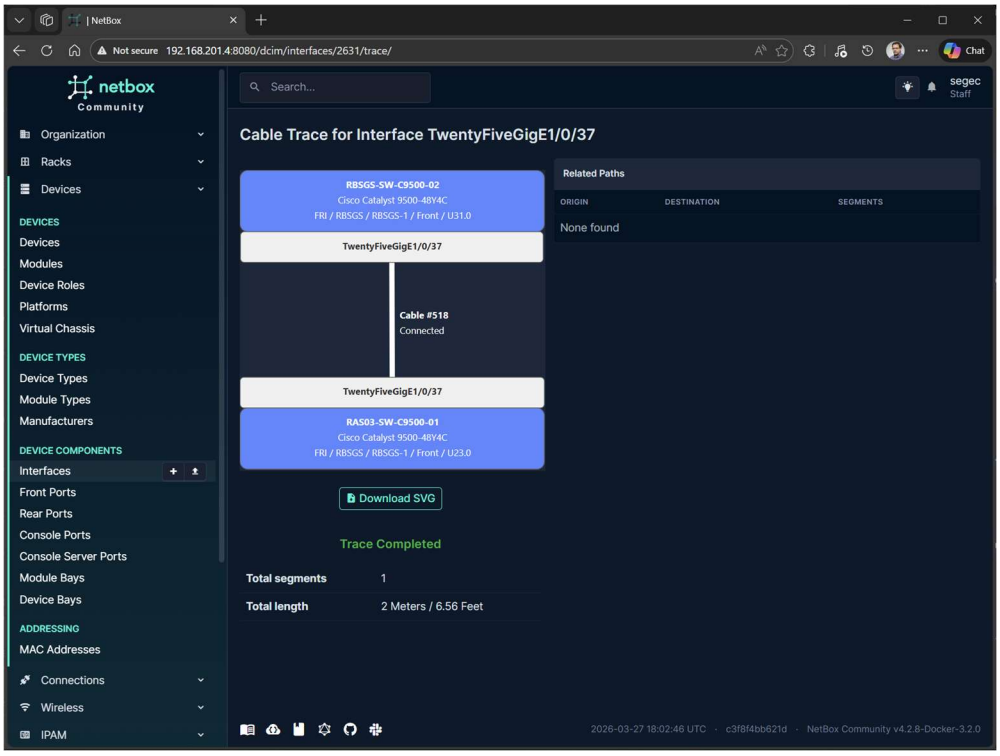


Figure 7 Cable tracing

2.8.4 Current status and limitations of NetBox in the PoC

Within the PoC, NetBox was successfully deployed and given a basic configuration as a functional tool for recording infrastructure data. In this state, it provides a usable basis for managing selected objects and for orientation within the environment of the proposed solution. From the PoC perspective, it was confirmed that NetBox is suitable for complementing the SOC solution with infrastructure context and for supporting work with assets and addressing.

On the other hand, it should be noted that its use in the PoC remained at the level of basic record-keeping and partial context enrichment. Not all of the planned integration links to the other components of the solution were completed, and so at this stage NetBox does not function as a fully automated source of enrichment data. For the time being, its role is rather a supporting one, laying the groundwork for further extension in future stages of the project.

For this reason, NetBox in the PoC should be understood as a functional, but not yet fully developed, component. Nevertheless, its deployment demonstrated its practical usability in the proposed solution and confirmed the suitability of including it in the wider architecture of the SOC service.

2.8.5 DFIR IRIS

DFIR-IRIS is used within the INNOSOC PoC solution as an open-source tool designed for the management and recording of security incidents (case management). Within our PoC, it serves as the central place where alerts from the Wazuh system are received and subsequently processed as incidents (cases). It enables the recording of artefacts, notes and tasks, the changing of incident statuses and the coordination of analysts' work.

2.8.6 DFIR IRIS deployment model in the OpenStack environment

The system is deployed in the OpenStack environment on a virtual machine running the Ubuntu 24.04 operating system. DFIR-IRIS runs in containers using Docker Compose, which allows simple deployment, administration and scaling of the solution.

Technical specification of the DFIR-IRIS deployment:

- Environment – OpenStack
- Instance name – ubuntu-24.04-kis
- IP address – 192.168.20.45 (internal IP in OpenStack), 192.168.4.97 (external IP)
- vCPU – 4
- RAM – 8 GB
- Disk – 40 GB
- OS – Ubuntu 24.04

The solution is built on a container architecture in which the individual components (web interface, backend and database layer) run separately in Docker containers. The entire application is started using the command:

docker compose up -d

DFIR-IRIS provides a web interface through which the analyst works with incidents, while the backend handles the application logic and communication with the database, where all incidents, artefacts and the change history are stored

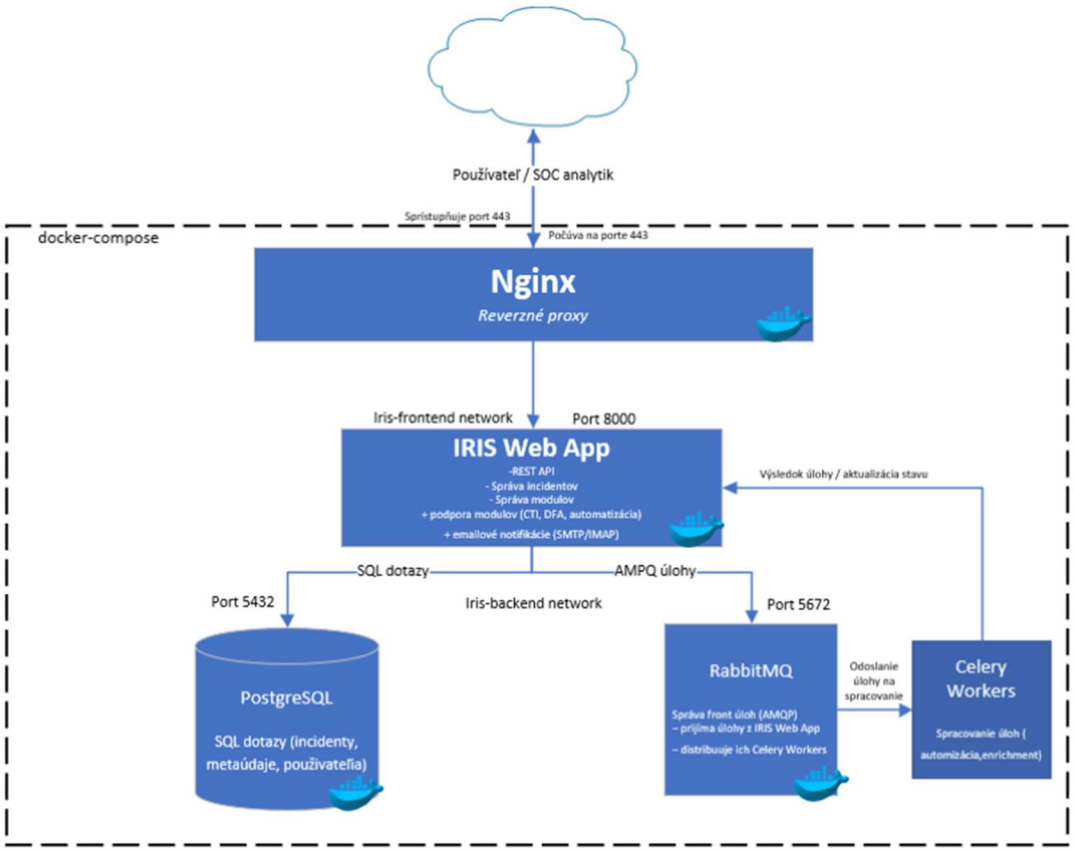


Figure 2 DFIR-IRIS architecture

The figure illustrates the architecture of DFIR-IRIS and the processing of incidents within the INNOSOC environment. The solution is based on a containerized architecture in which the individual components communicate via internal services.

The frontend is the web interface for analysts, which is used to work with incidents (cases), artefacts and tasks. Requests from the interface are processed by the backend, which provides the application logic and the management of the incident workflow. The backend communicates with the database layer, where all information about incidents is stored, including status, users, artefacts and the timeline. This architecture enables consistent record-keeping and retrospective tracing of the operations performed.

2.8.7 Basic integration links of the DFIR IRIS component in INNOSOC

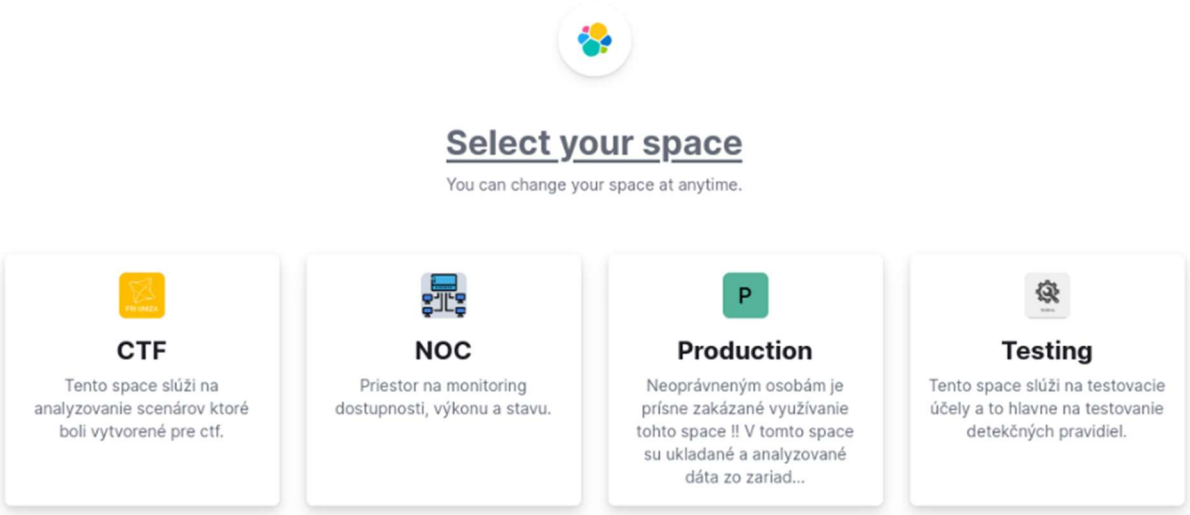
DFIR-IRIS supports integration via a REST API and webhooks. Within the PoC, it is connected to the Wazuh system, from which it receives alerts in JSON format and processes them as incidents or as updates to existing cases. The Shuffle system is connected to DFIR-IRIS and will serve as the automation layer for operations on incidents, such as enrichment, case updates or triggering responses. This connection is implemented and its functionality has been verified through testing.

Services currently connected and tested using DFIR-IRIS:

- Wazuh – receiving alerts and creating incidents (JSON / webhook)
- Shuffle – automation and updating of incidents via the API

2.9 Kibana (ELK)

Within the INNOSOC PoC solution, Kibana is used as the graphical interface in the ELK stack. At FRI UNIZA it is deployed at a URL in the FRI domain. The logical division of rules, access and other aspects is implemented using Spaces. We have the following Spaces deployed. Their role / use can also be seen in the figure.



Main features of Kibana used in the PoC:

- Discover – viewing records,
- Dashboards (Analytics) – displaying records in clear charts,
- Dashboards (Security) - operation of rules,
- Rules - management of detection rules,
- Alerts,
- Dev Tools - administration console,
- Stack Monitoring,
- Stack Management.

The entire stack is managed in Stack Managment.

2.9.1 Dashboards created

Dashboards in Kibana are used to visualize data from Elasticsearch by means of charts, maps and filters, thereby providing an overview of the monitored data.

Dashboards created:

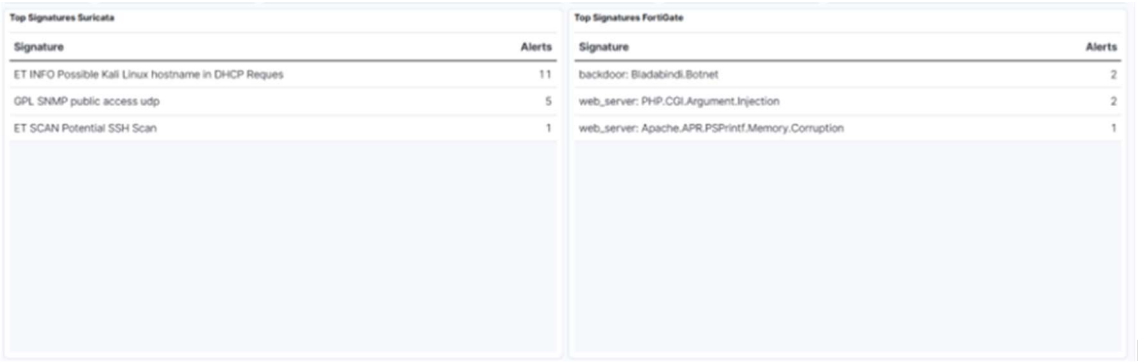
- Suricata UNIZA
- Arkime UNIZA
- F_kibana_dashboards
- Researchers' Night
- IDS DMZ: Visualizes data from IDS sensors in the demilitarized zones and makes it possible to compare detections between Suricata and Fortigate.

- Login Activity Monitoring: Focuses on successful and failed logins
- Stack Load: Used to monitor RAM usage by the individual Elasticsearch, Logstash and Kibana nodes



IDS

DMZ dashboard, alerts



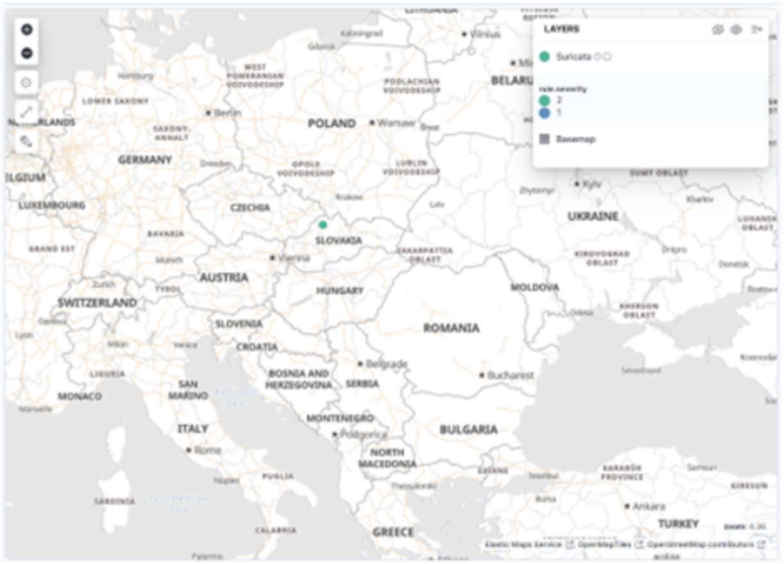
IDS

DMZ dashboard, top signatures



IDS

DMZ dashboard, top source IPs



IDS DMZ dashboard, map

2.9.2 Technical parameters and verification of functionality

During deployment, we defined for Kibana the maximum amount of RAM it can work with. We set it to 8 GB. Kibana does not have a RAM limit defined by default; by assigning a limit, we ensured that Kibana does not allocate all of the memory for its own use. We chose 8 GB to provide sufficient memory resources when displaying complex visualizations.

Kibana is able to verify the TLS certificate of Elasticsearch, which ensures encrypted communication. Kibana authenticates to Elasticsearch when connecting using a username and password; the default user accounts are used. Authentication is configured in the main Kibana configuration file. To increase security, the default user's password is not stored in so-called "plaintext" form; instead, we use a keystore to store the password, as with the other tools. Network communication is secured using iptables, with which we defined firewall rules. User accounts with roles designated for this purpose are used to log in to the Kibana UI, i.e. so-called role-based access control (RBAC).

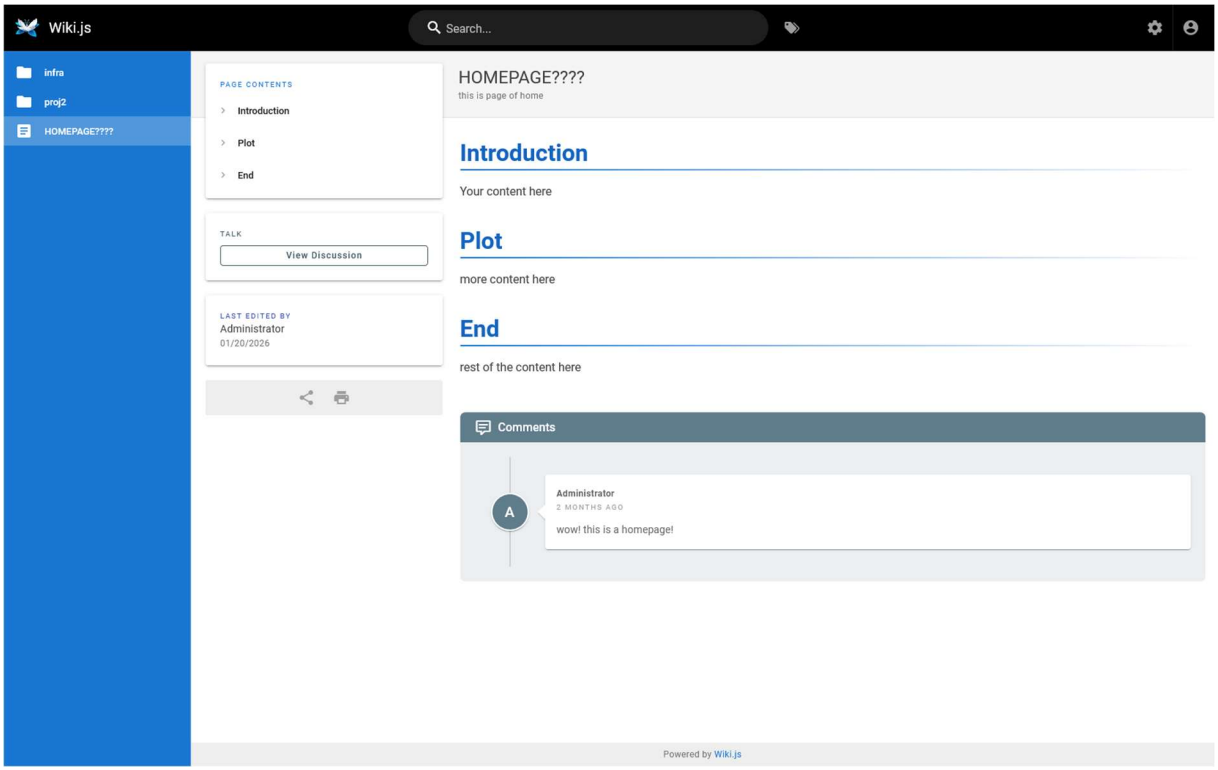
Kibana is monitored by Zabbix, LibreNMS and also by Metricbeat. Through such monitoring, we track the availability and stability of the service.

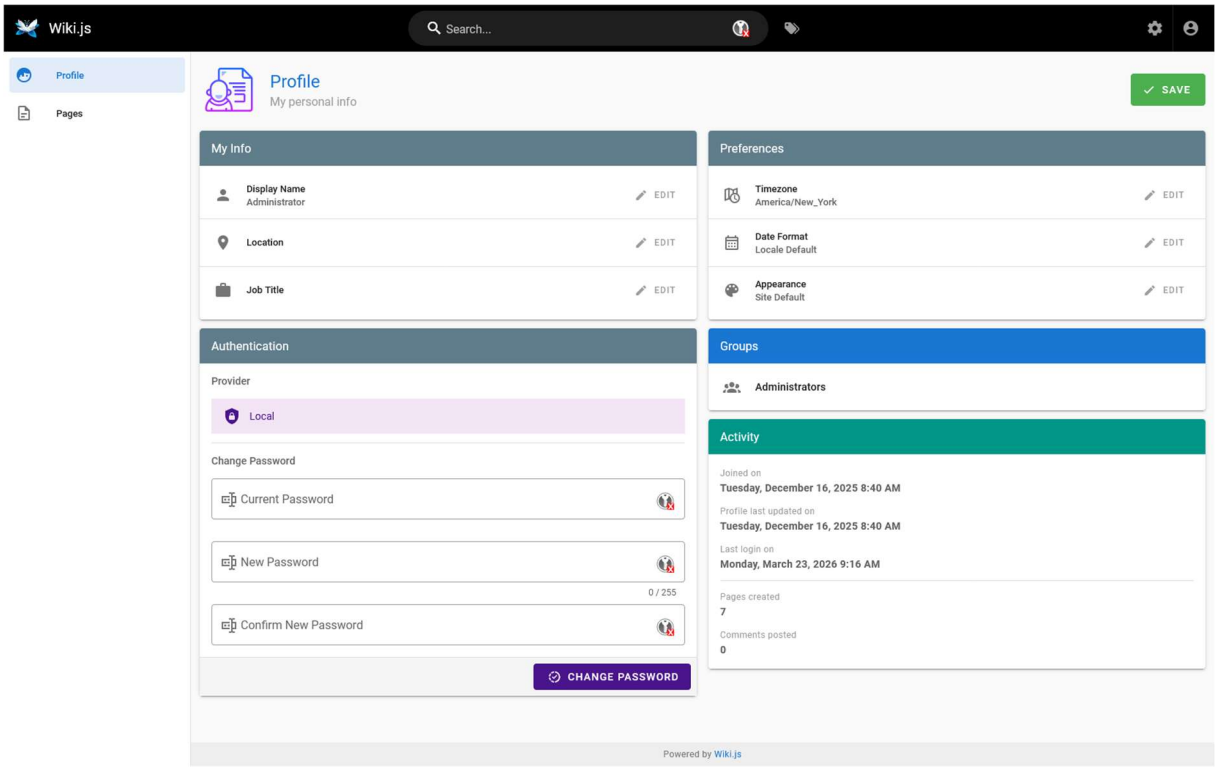
The Kibana service also runs in its own LXC container on the same physical server as the rest of the ELK stack. The container has the IP address 192.168.2.188, which is accessible only from a specific IP range. The Kibana directory contains the configuration file as well as certificates with the elasticsearch certification authority to secure encrypted communication. Example of the kibana.yml configuration file:

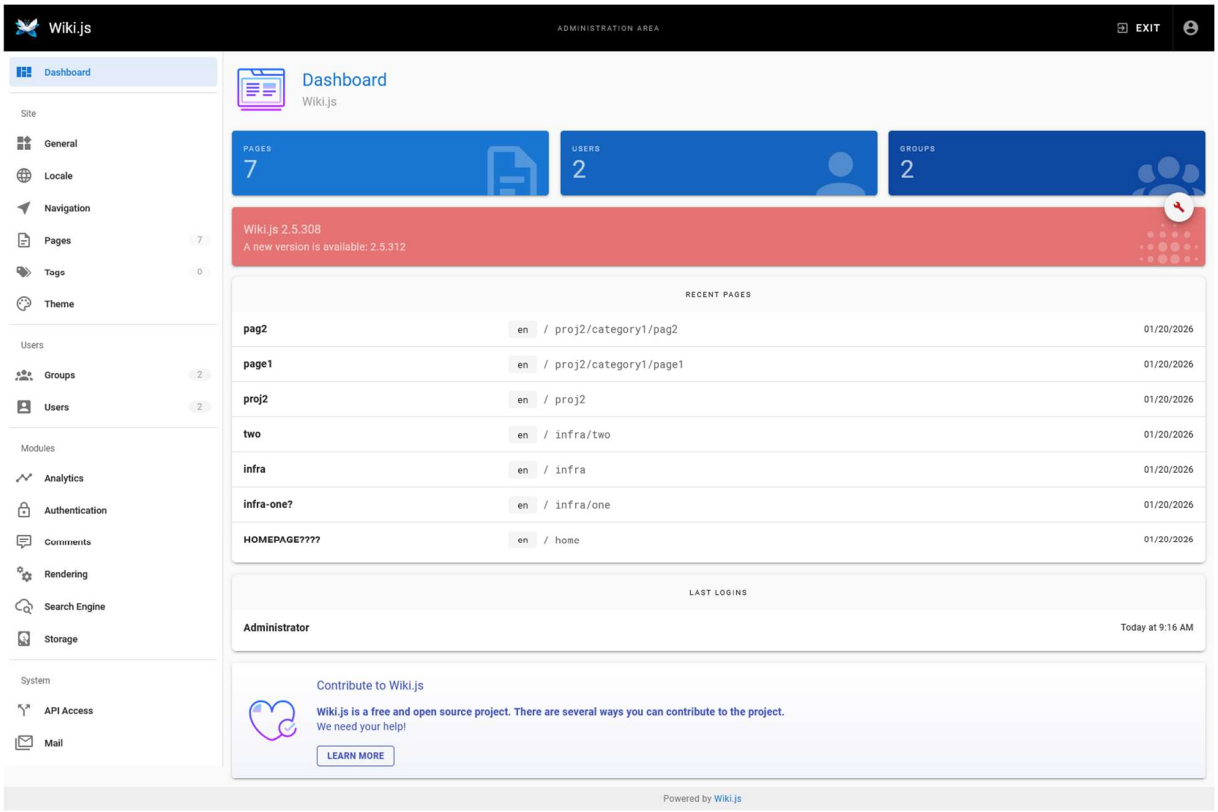
```
server.port: 5601
server.host: "0.0.0.0"
server.publicBaseUrl: "https://[internal domain]"
server.ssl.enabled: false
elasticsearch.hosts: ["https://192.168.2.178:9200"]
elasticsearch.username: "kibana_system"
elasticsearch.requestTimeout: 180000
```

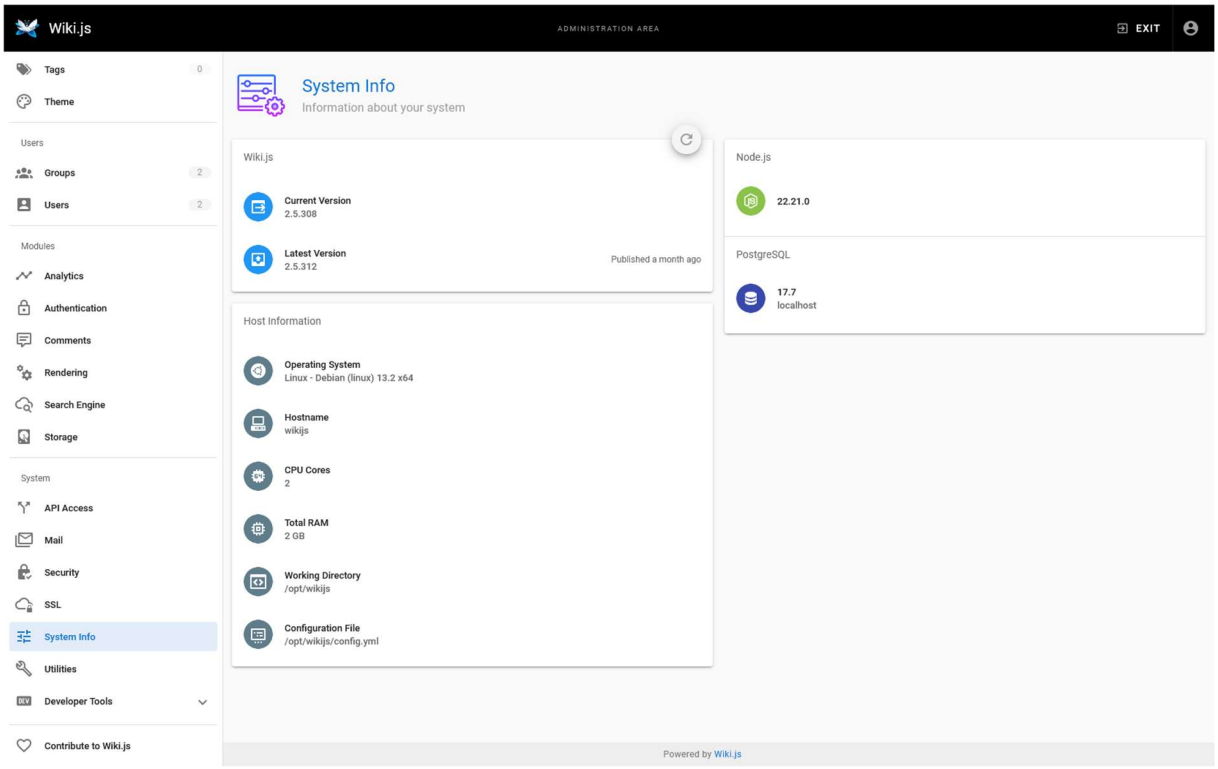
```
elasticsearch.ssl.certificateAuthorities:                                    [
"/etc/kibana/certs/kibana/elasticsearch-ca.pem" ]
logging:
  appenders:
    file:
      type: file
      fileName: /var/log/kibana/kibana.log
      layout:
        type: json
  root:
    appenders:
      - default
      - file
pid.file: /run/kibana/kibana.pid
xpack.encryptedSavedObjects.encryptionKey: [key1]
xpack.reporting.encryptionKey: [key2]
xpack.security.encryptionKey: [key3]
monitoring.kibana.collection.enabled: false
monitoring.ui.enabled: true
monitoring.ui.elasticsearch.hosts: ["https://192.168.2.178:9200"]
     monitoring.ui.ccs.enabled: false
```

2.10 Wiki.js




Wiki.js
Search...
Profile
Pages
Profile
My personal info
SAVE
My Info
Display Name
Administrator
EDIT
Location
EDIT
Job Title
EDIT
Preferences
Timezone
America/New_York
EDIT
Date Format
Locale Default
EDIT
Appearance
Site Default
EDIT
Authentication
Provider
Local
Change Password
Current Password
New Password
0 / 255
Confirm New Password
CHANGE PASSWORD
Groups
Administrators
Activity
Joined on
Tuesday, December 16, 2025 8:40 AM
Profile last updated on
Tuesday, December 16, 2025 8:40 AM
Last login on
Monday, March 23, 2026 9:16 AM
Pages created
7
Comments posted
0
Powered by Wiki.js


Wiki.js
ADMINISTRATION AREA
EXIT
Dashboard
Site
General
Locale
Navigation
Pages
Tags
Theme
Users
Groups
Users
Modules
Analytics
Authentication
Comments
Rendering
Search Engine
Storage
System
API Access
Mail
Dashboard
Wiki.js
PAGES
7
USERS
2
GROUPS
2
Wiki.js 2.5.308
A new version is available: 2.5.312
RECENT PAGES
pag2 en / proj2/category1/pag2 01/20/2026
page1 en / proj2/category1/page1 01/20/2026
proj2 en / proj2 01/20/2026
two en / infra/two 01/20/2026
infra en / infra 01/20/2026
infra-one? en / infra/one 01/20/2026
HOMEPAGE???? en / home 01/20/2026
LAST LOGINS
Administrator Today at 9:16 AM
Contribute to Wiki.js
Wiki.js is a free and open source project. There are several ways you can contribute to the project.
We need your help!
LEARN MORE
Powered by Wiki.js



Plus, in the rest of the documentation there is also some screenshot of the e-mail settings or something like that, I think...

3 DEPLOYMENT AND BASIC CONFIGURATION OF THE POC

Briefly describe here how the PoC was deployed in the OpenStack environment, how the VMs were used and where Docker was used. Do not go into detail as in V14, just summarize the actual state.

In my opinion we have exhausted this with the content above – for each component we have the deployment model, the virtual machines and services used, and the basic integration links ... here I don't know whether we will come up with anything more... there won't be the energy.... In my opinion we have it above.. For the components

3.1 Deployment model

3.2 Virtual machines and services used

3.3 Basic integration links

4 CONCLUSION

4.1 XX

Funded by the European Union
NextGenerationEU
RECOVERY AND RESILIENCE PLAN
GOVERNMENT OFFICE OF THE SLOVAK REPUBLIC