Cyberprotection Systems

Laboratory Work 1. Event Gathering and Correlation

(MEMORY)

Name - SHUBHAM NEGI

1. Topology configuration

Describe the configured topology (machines, O.S), as well as the technologies used (VirtualBox, Dockers, etc).

The configured topology – The whole system is setup in **VirtualBox** with all the virtual machines as described below

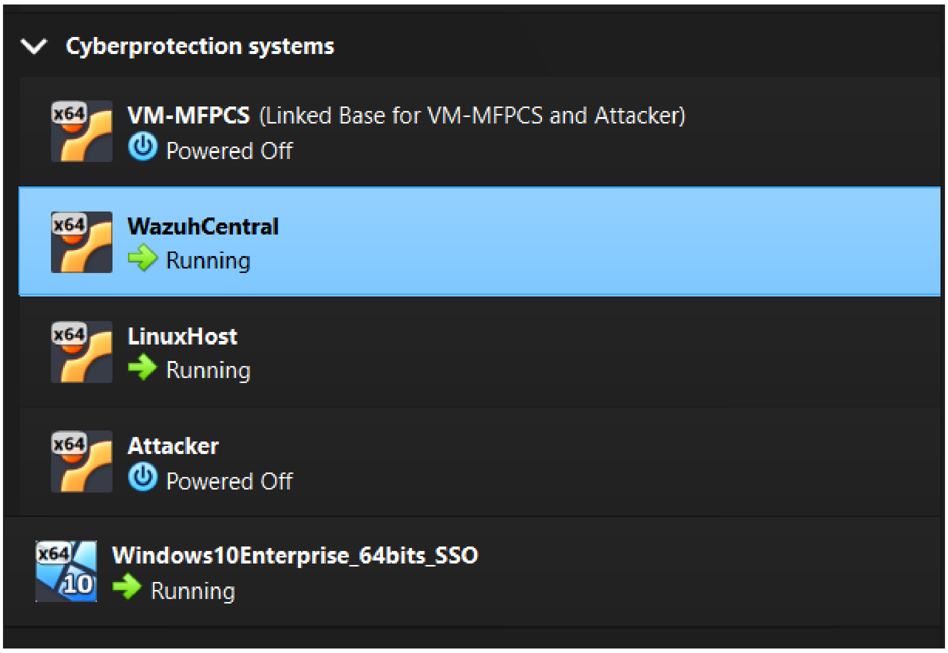
Network Interface – **intent** (**10.10.10.0/24**)

WazuhCentral – **10.10.10.1**, Linux OS (Lubuntu)

LinuxHost – **10.10.10.2**, Linux OS (Lubuntu)

Windows10Host – **10.10.10.3**, Windows 10 OS

Put a screenshot (or some) to show them running.



Cyberprotection systems

VM-MFPCS (Linked Base for VM-MFPCS and Attacker)
Powered Off

WazuhCentral
Running

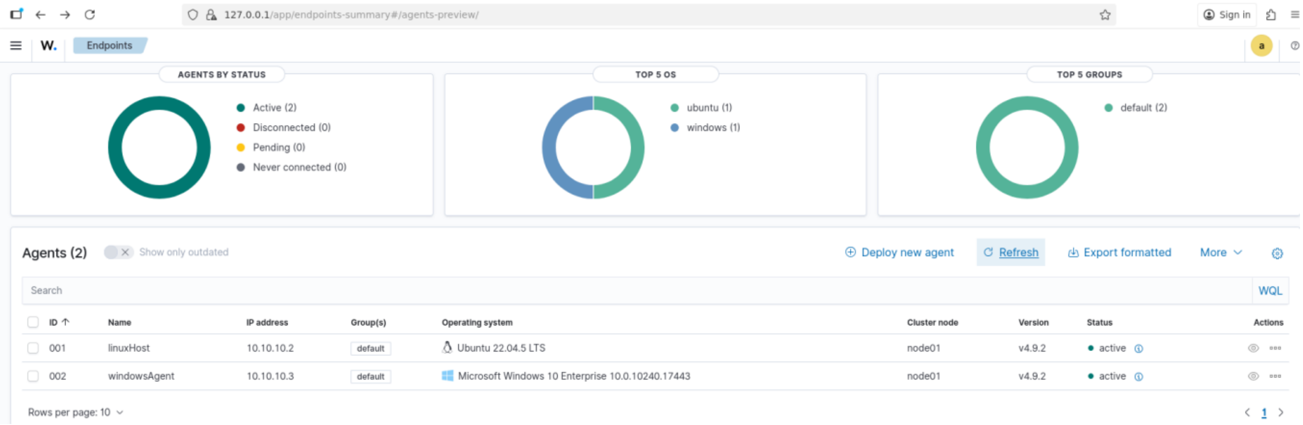
LinuxHost
Running

Attacker
Powered Off

Windows10Enterprise_64bits_SSO
Running

2. Wazuh setting

Show a screenshot (or some) where it can be seen the deployed agents in Wazuh. Explain it.

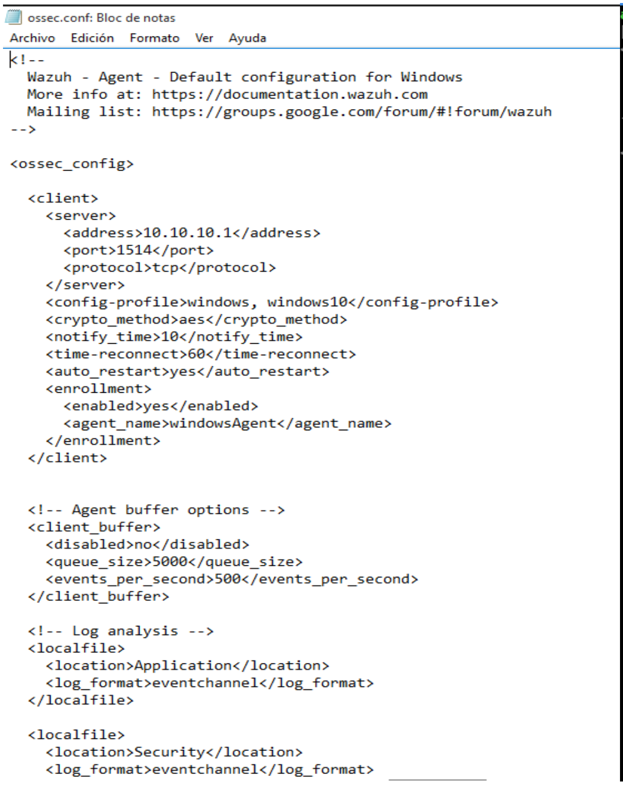


ID ↑	Name	IP address	Group(s)	Operating system	Cluster node	Version	Status	Actions
001	linuxHost	10.10.10.2	default	Ubuntu 22.04.5 LTS	node01	v4.9.2	● active	
002	windowsAgent	10.10.10.3	default	Microsoft Windows 10 Enterprise 10.0.10240.17443	node01	v4.9.2	● active	

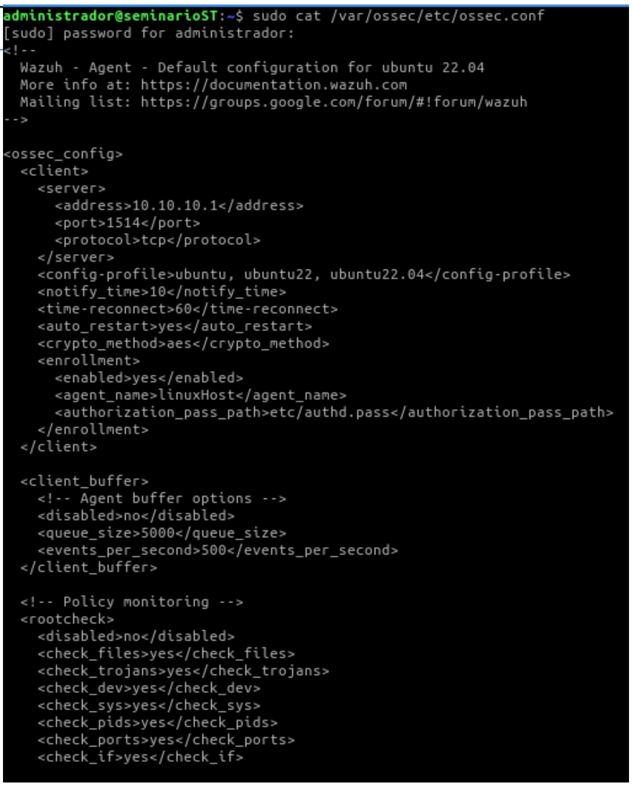
This is the screenshot of the endpoint-summary tab page in the wazuh central machine where we can see that currently the agents active are LinuxHost and WindowsAgent. Both of these machines have wazuh-agents installed and the Wazuh service being run. If we turn off any of these machines we can see the current status of the agents/machines in the dashboard.

3. Wazuh agents

Show a screenshot (or some) with the configuration files of the agents.



```
<!--
  Wazuh - Agent - Default configuration for Windows
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>

  <client>
    <server>
      <address>10.10.10.1</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>windows, windows10</config-profile>
    <crypto_method>aes</crypto_method>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
    <enrollment>
      <enabled>yes</enabled>
      <agent_name>windowsAgent</agent_name>
    </enrollment>
  </client>


  <!-- Agent buffer options -->
  <client_buffer>
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

  <!-- Log analysis -->
  <localfile>
    <location>Application</location>
    <log_format>eventchannel</log_format>
  </localfile>

  <localfile>
    <location>Security</location>
    <log_format>eventchannel</log_format>
```

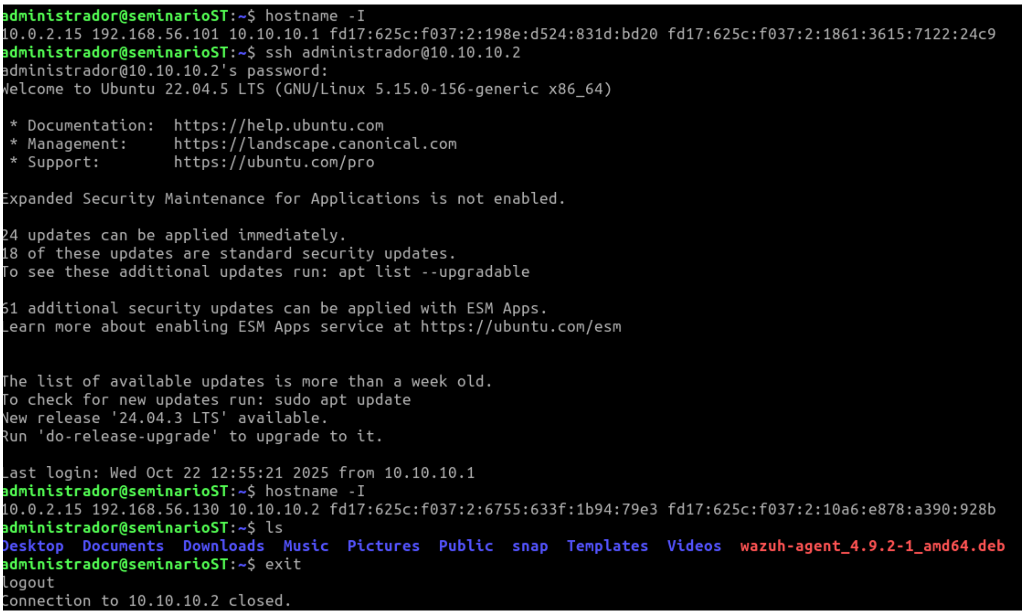


```
administrador@seminarioST:~$ sudo cat /var/ossec/etc/ossec.conf
[sudo] password for administrador:
<!--
  Wazuh - Agent - Default configuration for ubuntu 22.04
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <client>
    <server>
      <address>10.10.10.1</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>ubuntu, ubuntu22, ubuntu22.04</config-profile>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
    <crypto_method>aes</crypto_method>
    <enrollment>
      <enabled>yes</enabled>
      <agent_name>linuxHost</agent_name>
      <authorization_pass_path>etc/authd.pass</authorization_pass_path>
    </enrollment>
  </client>

  <client_buffer>
    <!-- Agent buffer options -->
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
  </client_buffer>

  <!-- Policy monitoring -->
  <rootcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>yes</check_dev>
    <check_sys>yes</check_sys>
    <check_pids>yes</check_pids>
    <check_ports>yes</check_ports>
    <check_if>yes</check_if>
```

4. Event generation

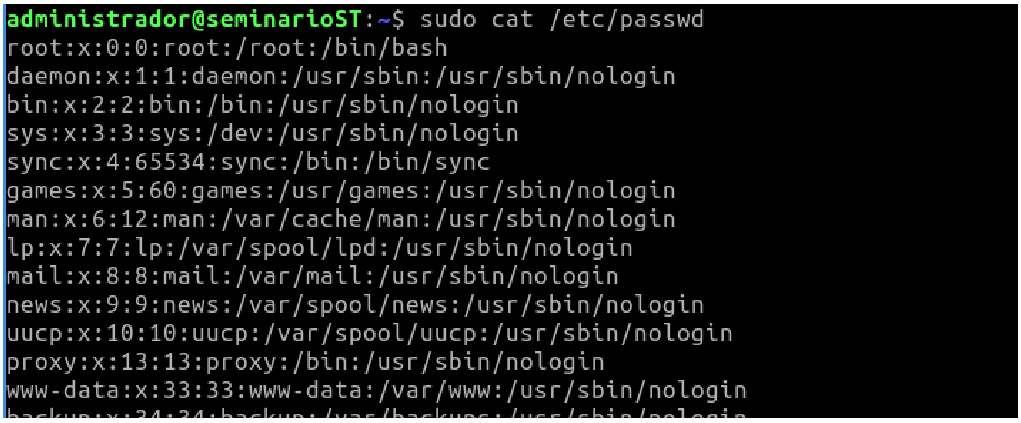
a) Show a screenshot (or some) in which the SSH connection task (from Practical part 1.1) is shown; i.e. present the SSH connection, the actions performed, the log events produced, and the generated events in Wazuh. Explain what it is shown.

Below are the two screenshots from the wazuh central machine where I logged into linuxHost via ssh. The first screenshot shows the IP address before and after the login to confirm that the login was successful



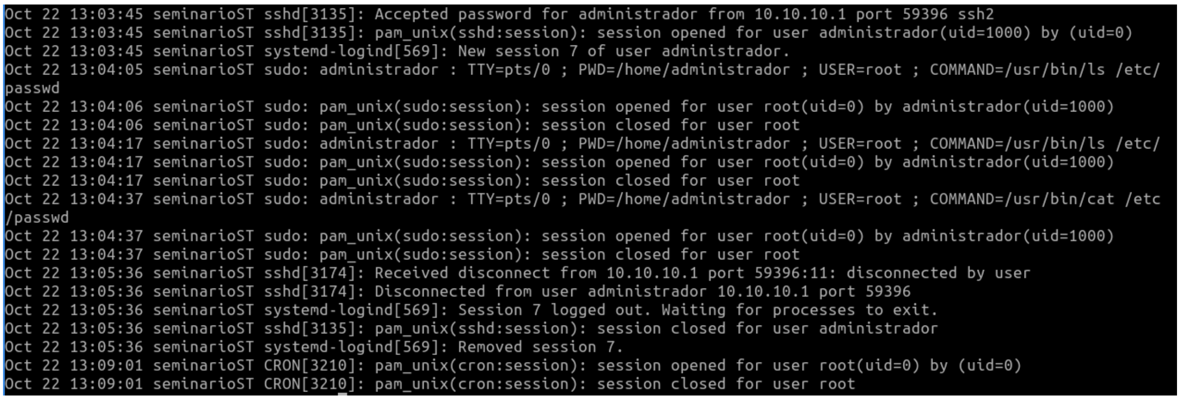
```
administrador@seminarioST:~$ hostname -I
10.0.2.15 192.168.56.101 10.10.10.1 fd17:625c:f037:2:198e:d524:831d:bd20 fd17:625c:f037:2:1861:3615:7122:24c9
administrador@seminarioST:~$ ssh administrador@10.10.10.2
administrador@10.10.10.2's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-156-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

24 updates can be applied immediately.
18 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

61 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '24.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Oct 22 12:55:21 2025 from 10.10.10.1
administrador@seminarioST:~$ hostname -I
10.0.2.15 192.168.56.130 10.10.10.2 fd17:625c:f037:2:6755:633f:1b94:79e3 fd17:625c:f037:2:10a6:e878:a390:928b
administrador@seminarioST:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos  wazuh-agent_4.9.2-1_amd64.deb
administrador@seminarioST:~$ exit
logout
Connection to 10.10.10.2 closed.
```

This screenshot is where I ran the command to list the restricted passwd file in the linuxHost via sudo.



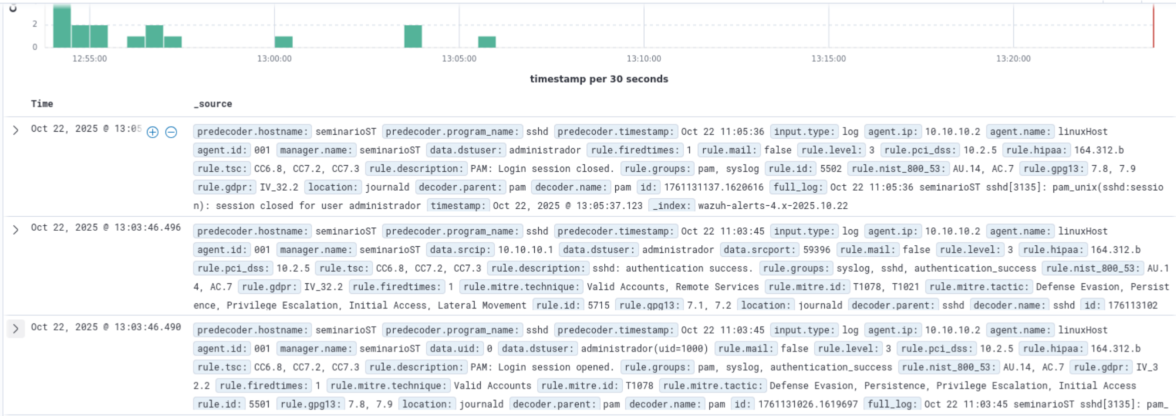
```
administrador@seminarioST:~$ sudo cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

Now the below screenshot shows the auth.log contents in the linuxHost machine for the above ssh session. As it is visible that the ssh connection was established from the wazuh central machine(10.10.10.1) and the cat command which I ran using sudo as root is also being logged in the subsequent lines of the log file and finally the session closed part is being logged.



```
Oct 22 13:03:45 seminarioST sshd[3135]: Accepted password for administrador from 10.10.10.1 port 59396 ssh2
Oct 22 13:03:45 seminarioST sshd[3135]: pam_unix(sshd:session): session opened for user administrador(uid=1000) by (uid=0)
Oct 22 13:03:45 seminarioST systemd-logind[569]: New session 7 of user administrador.
Oct 22 13:04:05 seminarioST sudo: administrador : TTY=pts/0 ; PWD=/home/administrador ; USER=root ; COMMAND=/usr/bin/ls /etc/
passwd
Oct 22 13:04:06 seminarioST sudo: pam_unix(sudo:session): session opened for user root(uid=0) by administrador(uid=1000)
Oct 22 13:04:06 seminarioST sudo: pam_unix(sudo:session): session closed for user root
Oct 22 13:04:17 seminarioST sudo: administrador : TTY=pts/0 ; PWD=/home/administrador ; USER=root ; COMMAND=/usr/bin/ls /etc/
Oct 22 13:04:17 seminarioST sudo: pam_unix(sudo:session): session opened for user root(uid=0) by administrador(uid=1000)
Oct 22 13:04:17 seminarioST sudo: pam_unix(sudo:session): session closed for user root
Oct 22 13:04:37 seminarioST sudo: administrador : TTY=pts/0 ; PWD=/home/administrador ; USER=root ; COMMAND=/usr/bin/cat /etc
/passwd
Oct 22 13:04:37 seminarioST sudo: pam_unix(sudo:session): session opened for user root(uid=0) by administrador(uid=1000)
Oct 22 13:04:37 seminarioST sudo: pam_unix(sudo:session): session closed for user root
Oct 22 13:05:36 seminarioST sshd[3174]: Received disconnect from 10.10.10.1 port 59396:11: disconnected by user
Oct 22 13:05:36 seminarioST sshd[3174]: Disconnected from user administrador 10.10.10.1 port 59396
Oct 22 13:05:36 seminarioST systemd-logind[569]: Session 7 logged out. Waiting for processes to exit.
Oct 22 13:05:36 seminarioST sshd[3135]: pam_unix(sshd:session): session closed for user administrador
Oct 22 13:05:36 seminarioST systemd-logind[569]: Removed session 7.
Oct 22 13:09:01 seminarioST CRON[3210]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Oct 22 13:09:01 seminarioST CRON[3210]: pam_unix(cron:session): session closed for user root
```

Below are the events generated in the wazuh dashboard.



```
Time                        _source
> Oct 22, 2025 @ 13:05      predecoder.hostname: seminarioST predecoder.program_name: sshd predecoder.timestamp: Oct 22 11:05:36 input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost
                            agent.id: 001 manager.name: seminarioST data.dstuser: administrador rule.firedtimes: 1 rule.mail: false rule.level: 3 rule.pci_dss: 10.2.5 rule.hipaa: 164.312.b
                            rule.tsc: CC6.8, CC7.2, CC7.3 rule.description: PAM: Login session closed. rule.groups: pam, syslog rule.id: 5502 rule.nist_800_53: AU.14, AC.7 rule.gpg13: 7.8, 7.9
                            rule.gdpr: IV_32.2 location: journald decoder.parent: pam decoder.name: pam id: 1761131137.1620616 full_log: Oct 22 11:05:36 seminarioST sshd[3135]: pam_unix(sshd:sessio
                            n): session closed for user administrador timestamp: Oct 22, 2025 @ 13:05:37.123 _index: wazuh-alerts-4.x-2025.10.22
> Oct 22, 2025 @ 13:03:46.496  predecoder.hostname: seminarioST predecoder.program_name: sshd predecoder.timestamp: Oct 22 11:03:45 input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost
                            agent.id: 001 manager.name: seminarioST data.srcip: 10.10.10.1 data.dstuser: administrador data.srcport: 59396 rule.mail: false rule.level: 3 rule.hipaa: 164.312.b
                            rule.pci_dss: 10.2.5 rule.tsc: CC6.8, CC7.2, CC7.3 rule.description: sshd: authentication success. rule.groups: syslog, sshd, authentication_success rule.nist_800_53: AU.1
                            4, AC.7 rule.gdpr: IV_32.2 rule.firedtimes: 1 rule.mitre.technique: Valid Accounts, Remote Services rule.mitre.id: T1078, T1021 rule.mitre.tactic: Defense Evasion, Persist
                            ence, Privilege Escalation, Initial Access, Lateral Movement rule.id: 5715 rule.gpg13: 7.1, 7.2 location: journald decoder.parent: sshd decoder.name: sshd id: 176113102
> Oct 22, 2025 @ 13:03:46.490  predecoder.hostname: seminarioST predecoder.program_name: sshd predecoder.timestamp: Oct 22 11:03:45 input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost
                            agent.id: 001 manager.name: seminarioST data.uid: 0 data.dstuser: administrador(uid=1000) rule.mail: false rule.level: 3 rule.pci_dss: 10.2.5 rule.hipaa: 164.312.b
                            rule.tsc: CC6.8, CC7.2, CC7.3 rule.description: PAM: Login session opened. rule.groups: pam, syslog, authentication_success rule.nist_800_53: AU.14, AC.7 rule.gdpr: IV_3
                            2.2 rule.firedtimes: 1 rule.mitre.technique: Valid Accounts rule.mitre.id: T1078 rule.mitre.tactic: Defense Evasion, Persistence, Privilege Escalation, Initial Access
                            rule.id: 5501 rule.gpg13: 7.8, 7.9 location: journald decoder.parent: pam decoder.name: pam id: 1761131026.1619697 full_log: Oct 22 11:03:45 seminarioST sshd[3135]: pam_
```

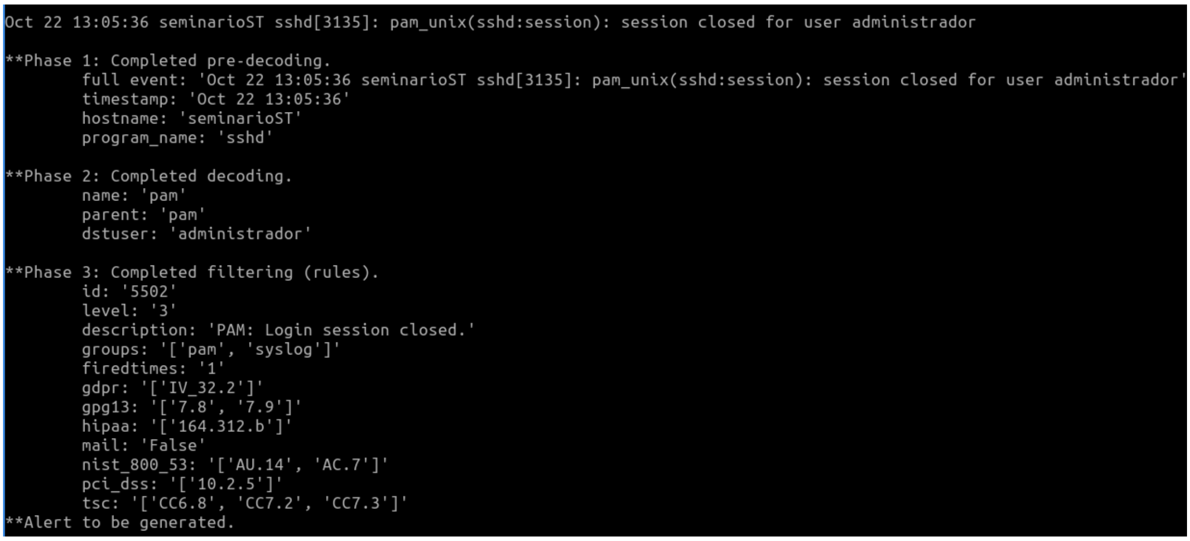
b) Relate at least three events from the dashboard to the corresponding lines in the auth.log file.

The first line in the auth.log corresponds to the second event in the dashboard where the authentication success has been shown as password accepted. The wazuh event also logs the complete log line and therefore its easy to see the corresponding log in the auth.log file.

The second line in the auth.log corresponds to the third event in the dashboard and the third event in dashboard which shows the logout of the ssh session corresponds to the fourth last line in the auth.log file.

c) Analyze whether the dashboard displays additional information that was not present in the original logs. Investigate how Wazuh is able to provide enriched data and document your findings.

Additional details like the mitre technique, id, tactic are being provided to check further from the MITRE ATTACK framework perspective and get more details to help generated any documentations or reports easily. The Specific articles or sections of the privacy laws like GDPR and HIPAA are also provided additionally to see which particular sections are violated with the alert or event generated.



```
Oct 22 13:05:36 seminarioST sshd[3135]: pam_unix(sshd:session): session closed for user administrador

**Phase 1: Completed pre-decoding.
        full event: 'Oct 22 13:05:36 seminarioST sshd[3135]: pam_unix(sshd:session): session closed for user administrador'
        timestamp: 'Oct 22 13:05:36'
        hostname: 'seminarioST'
        program_name: 'sshd'

**Phase 2: Completed decoding.
        name: 'pam'
        parent: 'pam'
        dstuser: 'administrador'

**Phase 3: Completed filtering (rules).
        id: '5502'
        level: '3'
        description: 'PAM: Login session closed.'
        groups: '['pam', 'syslog']'
        firedtimes: '1'
        gdpr: '['IV_32.2']'
        gpg13: '['7.8', '7.9']'
        hipaa: '['164.312.b']'
        mail: 'False'
        nist_800_53: '['AU.14', 'AC.7']'
        pci_dss: '['10.2.5']'
        tsc: '['CC6.8', 'CC7.2', 'CC7.3']'
**Alert to be generated.
```

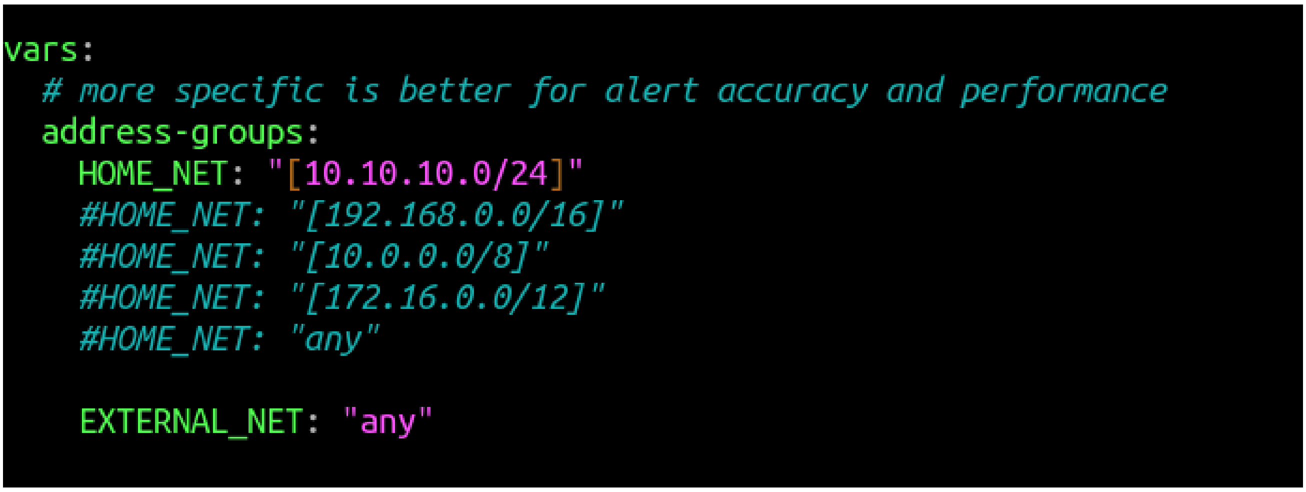
Using wazuh-logtest tool we can see how the logs are processed in wazuh like above and this explains how wazuh is able to provide the additional details or compliance mappings. All of these are based on first identifying the type of decoder used based on the log and then applying rules based on the ruleset mappings. For example, based on the PAM or ssh rule fires, it automatically provides all the matched details based on the mappings to the framework already provided in the rule's description.

d) Identify three different decoders used and the rule sets involved:

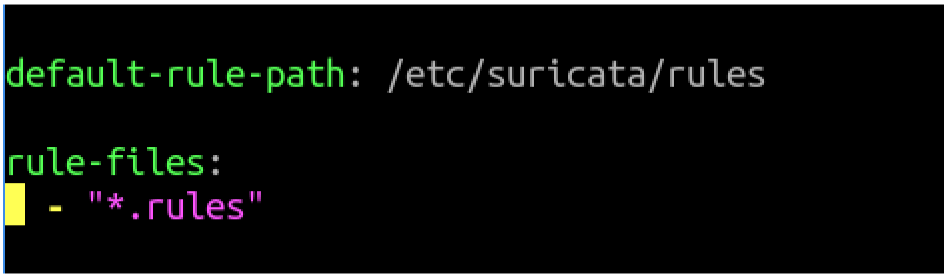
- Decoders used: Identify the decoders applied to the logs to normalize the information.
  In the three events, there are two decoders used – **pam** and **sshd**
- Rule sets involved: Indicate the rules that were triggered and their severity level.
  - rule: **5502** - PAM: Login session closed. Severity level – **3**
  - rule: **5715** - sshd: authentication success. Severity level – **3**
  - rule: **5501** - PAM: Login session opened. Severity level – **3**

5. Suricata installation

Show a screenshot (or some) where it can be seen the Suricata folders and configuration file (/etc/suricata/suricata.yaml) contents.



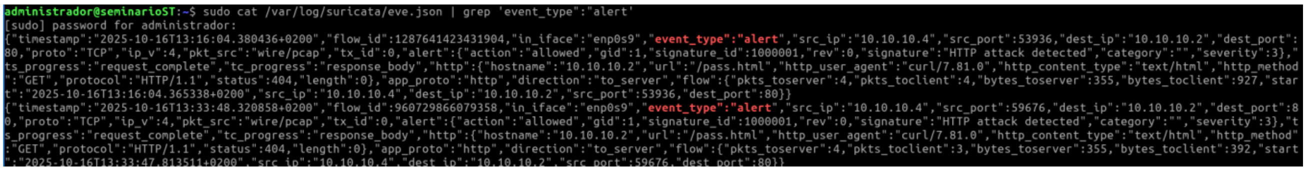
```
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[10.10.10.0/24]"
    #HOME_NET: "[192.168.0.0/16]"
    #HOME_NET: "[10.0.0.0/8]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"

    EXTERNAL_NET: "any"
```



```
default-rule-path: /etc/suricata/rules

rule-files:
  - "*.rules"
```

6. Suricata events

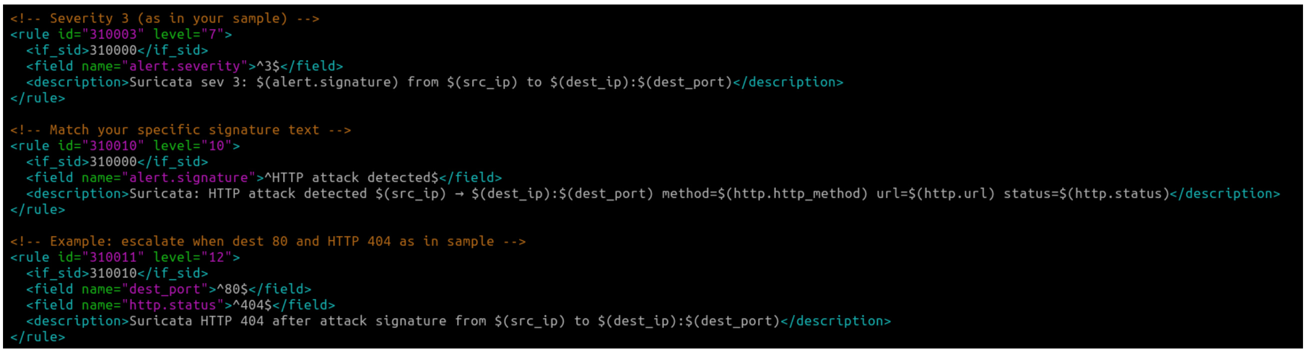
Generate some events and show a screenshot (or some) where it can be seen the contents of Suricata events json file.



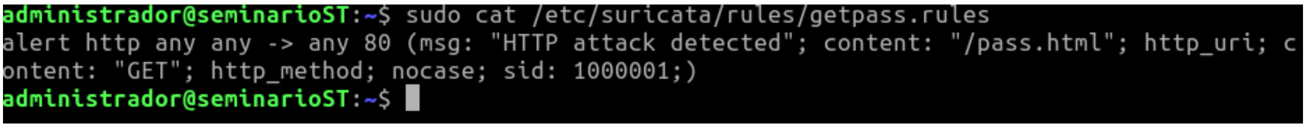
```
administrador@seminarioST:~$ sudo cat /var/log/suricata/eve.json | grep 'event_type":"alert'
[sudo] password for administrador:
{"timestamp":"2025-10-16T13:16:04.380436+0200","flow_id":1287641423431904,"in_iface":"enp0s9","event_type":"alert","src_ip":"10.10.10.4","src_port":53936,"dest_ip":"10.10.10.2","dest_port":
80,"proto":"TCP","ip_v":4,"pkt_src":"wire/pcap","tx_id":0,"alert":{"action":"allowed","gid":1,"signature_id":1000001,"rev":0,"signature":"HTTP attack detected","category":"","severity":3},"
ts_progress":"request_complete","tc_progress":"response_body","http":{"hostname":"10.10.10.2","url":"/pass.html","http_user_agent":"curl/7.81.0","http_content_type":"text/html","http_method
":"GET","protocol":"HTTP/1.1","status":404,"length":0},"app_proto":"http","direction":"to_server","flow":{"pkts_toserver":4,"pkts_toclient":4,"bytes_toserver":355,"bytes_toclient":927,"star
t":"2025-10-16T13:16:04.365338+0200","src_ip":"10.10.10.4","dest_ip":"10.10.10.2","src_port":53936,"dest_port":80}}
{"timestamp":"2025-10-16T13:33:48.320858+0200","flow_id":960729866079358,"in_iface":"enp0s9","event_type":"alert","src_ip":"10.10.10.4","src_port":59676,"dest_ip":"10.10.10.2","dest_port":8
0,"proto":"TCP","ip_v":4,"pkt_src":"wire/pcap","tx_id":0,"alert":{"action":"allowed","gid":1,"signature_id":1000001,"rev":0,"signature":"HTTP attack detected","category":"","severity":3},"t
s_progress":"request_complete","tc_progress":"response_body","http":{"hostname":"10.10.10.2","url":"/pass.html","http_user_agent":"curl/7.81.0","http_content_type":"text/html","http_method"
:"GET","protocol":"HTTP/1.1","status":404,"length":0},"app_proto":"http","direction":"to_server","flow":{"pkts_toserver":4,"pkts_toclient":3,"bytes_toserver":355,"bytes_toclient":392,"start
":"2025-10-16T13:33:47.813511+0200","src_ip":"10.10.10.4","dest_ip":"10.10.10.2","src_port":59676,"dest_port":80}}
```

7. Suricata rule for “HTTP Attack Detected”

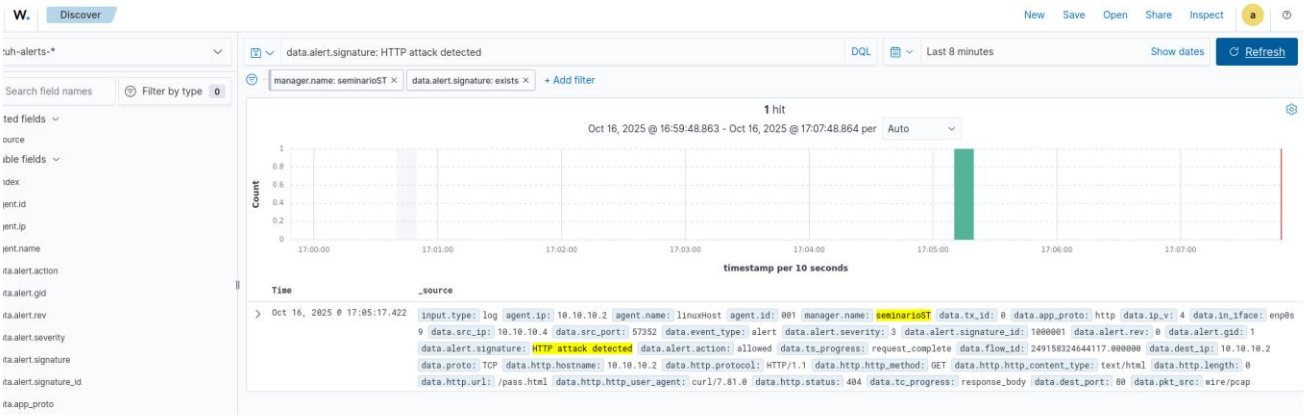
Show with screenshots the created rule. Access the protected file and show the event detection in Suricata (json file), and in Wazuh (generated event).



```
<!-- Severity 3 (as in your sample) -->
<rule id="310003" level="7">
  <if_sid>310000</if_sid>
  <field name="alert.severity">^3$</field>
  <description>Suricata sev 3: $(alert.signature) from $(src_ip) to $(dest_ip):$(dest_port)</description>
</rule>

<!-- Match your specific signature text -->
<rule id="310010" level="10">
  <if_sid>310000</if_sid>
  <field name="alert.signature">^HTTP attack detected$</field>
  <description>Suricata: HTTP attack detected $(src_ip) → $(dest_ip):$(dest_port) method=$(http.http_method) url=$(http.url) status=$(http.status)</description>
</rule>

<!-- Example: escalate when dest 80 and HTTP 404 as in sample -->
<rule id="310011" level="12">
  <if_sid>310010</if_sid>
  <field name="dest_port">^80$</field>
  <field name="http.status">^404$</field>
  <description>Suricata HTTP 404 after attack signature from $(src_ip) to $(dest_ip):$(dest_port)</description>
</rule>
```



```
administrador@seminarioST:~$ sudo cat /etc/suricata/rules/getpass.rules
alert http any any -> any 80 (msg: "HTTP attack detected"; content: "/pass.html"; http_uri; c
ontent: "GET"; http_method; nocase; sid: 1000001;)
administrador@seminarioST:~$ 
```

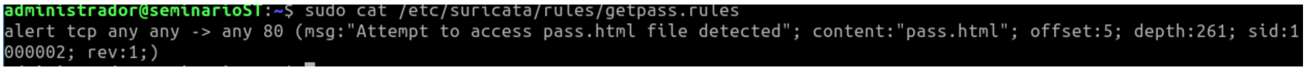


```
Oct 16, 2025 @ 17:05:17.422   input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost agent.id: 001 manager.name: seminarioST data.tx_id: 0 data.app_proto: http data.ip_v: 4 data.in_iface: enp0s
                              9 data.src_ip: 10.10.10.4 data.src_port: 57352 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000001 data.alert.rev: 0 data.alert.gid: 1
                              data.alert.signature: HTTP attack detected data.alert.action: allowed data.ts_progress: request_complete data.flow_id: 249158324644117.000000 data.dest_ip: 10.10.10.2
                              data.proto: TCP data.http.hostname: 10.10.10.2 data.http.protocol: HTTP/1.1 data.http.http_method: GET data.http.http_content_type: text/html data.http.length: 0
                              data.http.url: /pass.html data.http.http_user_agent: curl/7.81.0 data.http.status: 404 data.tc_progress: response_body data.dest_port: 80 data.pkt_src: wire/pcap
```

8. Suricata rule for “Attempt to Access pass.html File Detected”

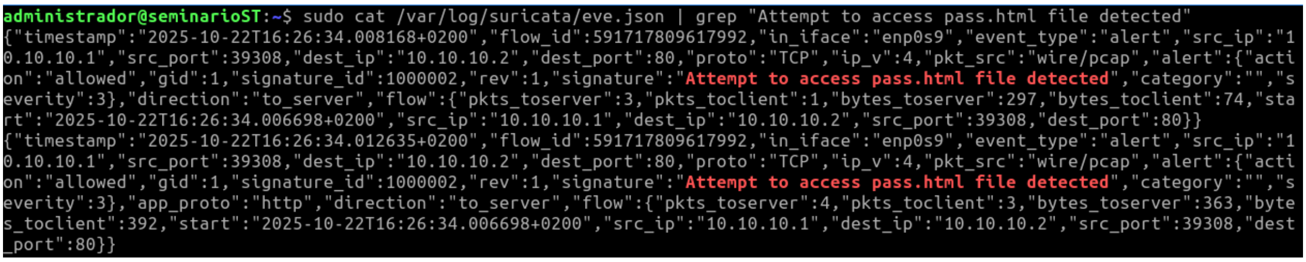
Show with screenshots the created rule. Access the protected file and show the event detection in Suricata (json file), and in Wazuh (generated event).

Rule – *alert tcp any any -> any 80 (msg:"Attempt to access pass.html file detected"; content:"pass.html"; offset:5; depth:261; sid:1000002; rev:1;)*



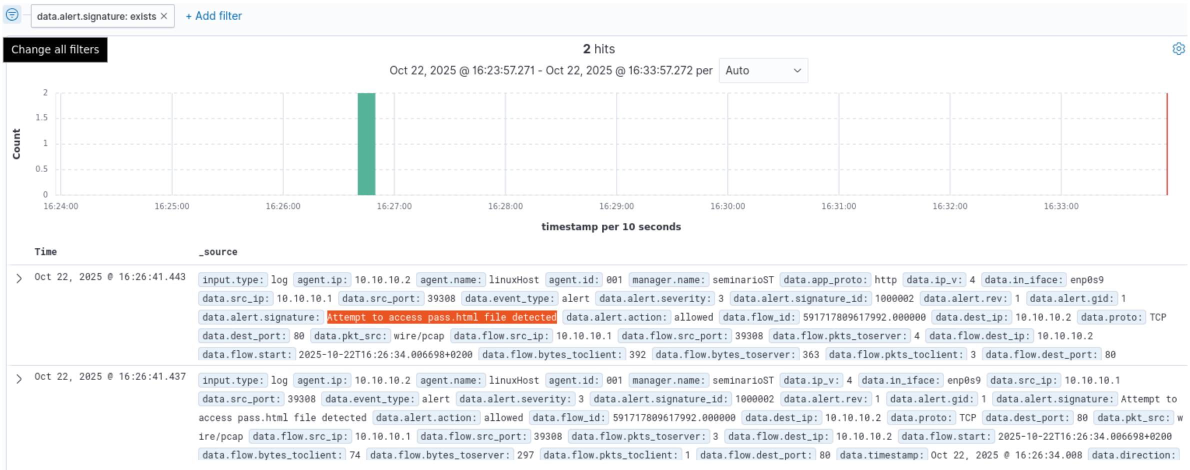
```
administrador@seminarioST:~$ sudo cat /etc/suricata/rules/getpass.rules
alert tcp any any -> any 80 (msg:"Attempt to access pass.html file detected"; content:"pass.html"; offset:5; depth:261; sid:1
000002; rev:1;)
```

Detection in Suricata eve.json file



```
administrador@seminarioST:~$ sudo cat /var/log/suricata/eve.json | grep "Attempt to access pass.html file detected"
{"timestamp":"2025-10-22T16:26:34.008168+0200","flow_id":591717809617992,"in_iface":"enp0s9","event_type":"alert","src_ip":"1
0.10.10.1","src_port":39308,"dest_ip":"10.10.10.2","dest_port":80,"proto":"TCP","ip_v":4,"pkt_src":"wire/pcap","alert":{"acti
on":"allowed","gid":1,"signature_id":1000002,"rev":1,"signature":"Attempt to access pass.html file detected","category":"","s
everity":3},"direction":"to_server","flow":{"pkts_toserver":3,"pkts_toclient":1,"bytes_toserver":297,"bytes_toclient":74,"sta
rt":"2025-10-22T16:26:34.006698+0200","src_ip":"10.10.10.1","dest_ip":"10.10.10.2","src_port":39308,"dest_port":80}}
{"timestamp":"2025-10-22T16:26:34.012635+0200","flow_id":591717809617992,"in_iface":"enp0s9","event_type":"alert","src_ip":"1
0.10.10.1","src_port":39308,"dest_ip":"10.10.10.2","dest_port":80,"proto":"TCP","ip_v":4,"pkt_src":"wire/pcap","alert":{"acti
on":"allowed","gid":1,"signature_id":1000002,"rev":1,"signature":"Attempt to access pass.html file detected","category":"","s
everity":3},"app_proto":"http","direction":"to_server","flow":{"pkts_toserver":4,"pkts_toclient":3,"bytes_toserver":363,"byte
s_toclient":392,"start":"2025-10-22T16:26:34.006698+0200","src_ip":"10.10.10.1","dest_ip":"10.10.10.2","src_port":39308,"dest
_port":80}}
```

Wazuh alert generated



```
Oct 22, 2025 @ 16:26:41.443   input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost agent.id: 001 manager.name: seminarioST data.app_proto: http data.ip_v: 4 data.in_iface: enp0s9
                              data.src_ip: 10.10.10.1 data.src_port: 39308 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000002 data.alert.rev: 1 data.alert.gid: 1
                              data.alert.signature: Attempt to access pass.html file detected data.alert.action: allowed data.flow_id: 591717809617992.000000 data.dest_ip: 10.10.10.2 data.proto: TCP
                              data.dest_port: 80 data.pkt_src: wire/pcap data.flow.src_ip: 10.10.10.1 data.flow.src_port: 39308 data.flow.pkts_toserver: 4 data.flow.dest_ip: 10.10.10.2
                              data.flow.start: 2025-10-22T16:26:34.006698+0200 data.flow.bytes_toclient: 392 data.flow.bytes_toserver: 363 data.flow.pkts_toclient: 3 data.flow.dest_port: 80

Oct 22, 2025 @ 16:26:41.437   input.type: log agent.ip: 10.10.10.2 agent.name: linuxHost agent.id: 001 manager.name: seminarioST data.ip_v: 4 data.in_iface: enp0s9 data.src_ip: 10.10.10.1
                              data.src_port: 39308 data.event_type: alert data.alert.severity: 3 data.alert.signature_id: 1000002 data.alert.rev: 1 data.alert.gid: 1 data.alert.signature: Attempt to
                              access pass.html file detected data.alert.action: allowed data.flow_id: 591717809617992.000000 data.dest_ip: 10.10.10.2 data.proto: TCP data.dest_port: 80 data.pkt_src: w
                              ire/pcap data.flow.src_ip: 10.10.10.1 data.flow.src_port: 39308 data.flow.pkts_toserver: 3 data.flow.dest_ip: 10.10.10.2 data.flow.start: 2025-10-22T16:26:34.006698+0200
                              data.flow.bytes_toclient: 74 data.flow.bytes_toserver: 297 data.flow.pkts_toclient: 1 data.flow.dest_port: 80 data.timestamp: Oct 22, 2025 @ 16:26:34.008 data.direction:
```