



Лабораторна робота №13

Імпорт логів із Wazuh у Splunk.

Мета: Метою роботи є освоєння процесу інтеграції системи моніторингу безпеки Wazuh із платформою аналітики Splunk шляхом налаштування передачі подій, створення індексу для логів Wazuh, перевірки надходження даних.

Інструменти: гіпервізор VirtualBox, модель комп'ютерної мережі.

Теоретичні відомості

У попередніх роботах створено стенове середовище у VirtualBox, що складається з чотирьох хостів:

Serv-G-N-1 (Windows Server 2022) – контролер домену з ролями AD DS, DNS і DHCP. Налаштовано Wazuh Agent для локального моніторингу ресурсів.

Serv-G-N-5 (Ubuntu Server 24.04) – сервер на налаштовано Wazuh Agent для локального моніторингу ресурсів.

Serv-G-N-7 (Amazon Linux 2023) – сервер Wazuh Appliance, що містить компоненти Wazuh Server (Manager, API) та Elasticsearch + Kibana (Dashboard)

Serv-G-N-9 (Ubuntu Server 24.04) – сервер Splunk Free — локальний індексатор та аналітична платформа для збору, обробки та візуалізації журналів безпеки. Сервер призначений для прийому логів з операційних систем та сервісів мережі, а також для інтеграції з Wazuh через Wazuh Splunk App.

Мережеве середовище забезпечує взаємодію між вузлами з доменною інфраструктурою для подальшого моніторингу її елементів.

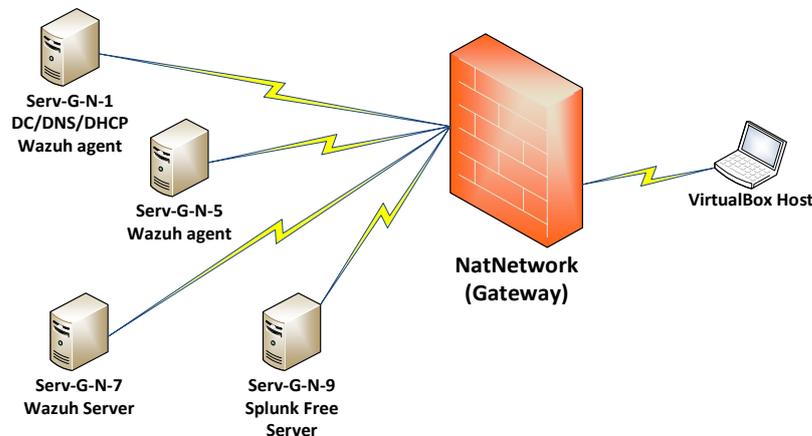


Рис. 13.1. Топологія мережі

Коротка теорія (Wazuh – Splunk: Filebeat, HEC, Wazuh App)

Інтеграція Wazuh зі Splunk дозволяє збирати всі події безпеки, які генерують Wazuh-агенти, і відображати їх у Splunk у вигляді дашбордів, пошукових запитів і звітів. Щоб це працювало, у Wazuh використовується Filebeat, а у Splunk — HTTP Event Collector (HEC) та спеціальний застосунок Wazuh App.

❖ Filebeat у Wazuh Appliance

У Wazuh Appliance (Serv-G-N-7) використовується компонент Filebeat, який читає лог-файли менеджера Wazuh, упаковує події у JSON та надсилає їх у зовнішні системи SIEM (Splunk, Elasticsearch, Logstash).

З точки зору інтеграції важливо що Filebeat підтримує Splunk HEC як транспорт, а Wazuh вже містить шаблон для Splunk у своєму пакеті.

❖ Splunk HEC (HTTP Event Collector)

HEC (HTTP Event Collector) — це механізм Splunk, який приймає події через HTTP/HTTPS. Для інтеграції потрібно увімкнути HEC у Splunk, створити токен доступу (authentication token), дозволити прийом JSON-подій, відкрити порт (зазвичай 8088/tcp або інший, якщо змінили).

Після цього Filebeat зможе надсилати події без прямої установки Forwarder-а на сервері.

❖ Wazuh Splunk App (візуалізація)

Wazuh App for Splunk — це офіційний застосунок, який містить готові дашборди для FIM (File Integrity Monitoring), аналізу активних відповідей (Active Response), аналітики подій агентів, візуалізацій MITRE ATT&CK та модулів Syscheck, Rootcheck, Vulnerability Detector та ін.



Він працює поверх Splunk Search і не вимагає додаткової конфігурації — лише правильного sourcetype та index.

❖ Як відбувається передача подій (спрощена схема)

Wazuh Manager генерує події (FIM, Syscheck, Rootcheck, інтеграції, агенти). Filebeat читає їх із log-файлів та формує JSON. Filebeat надсилає події через HTTP до Splunk HEC endpoint. Splunk приймає події та розміщує їх у вказаному індексі (наприклад, wazuh). Wazuh Splunk App відображає їх у вигляді дашбордів.

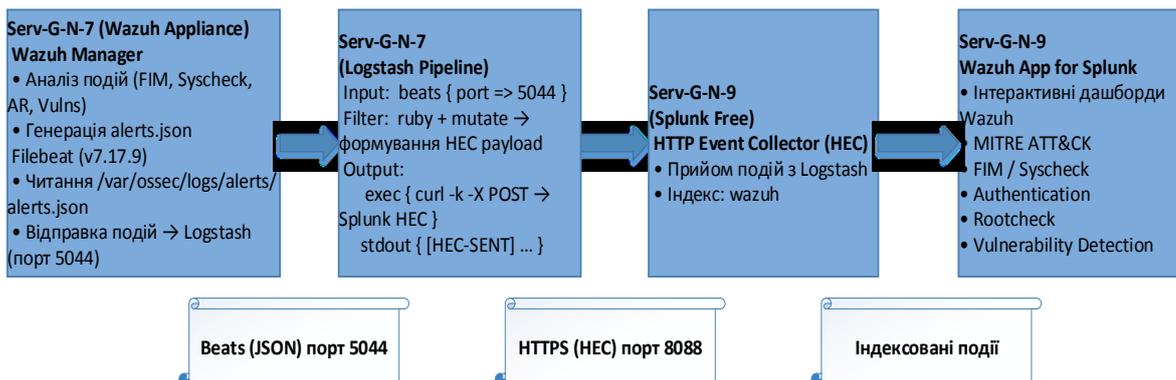


Рис. 13.2. Архітектура інтеграції Wazuh–Logstash–Splunk

Основні риси, що визначають взаємодію Wazuh та Splunk у процесі побудови централізованої системи безпекового моніторингу:

- ❖ Universal Forwarder не використовується. Передавання подій здійснюється безпосередньо через Filebeat, що працює на Wazuh Manager.
- ❖ Комунікація відбувається через один порт – 8088/TCP. Це порт прийому подій Splunk HEC (HTTP Event Collector).
- ❖ Для подій Wazuh створюється окремий індекс у Splunk. Це забезпечує структурованість даних та спрощує роботу з дашбордами.
- ❖ Після надходження подій активується модуль “Wazuh App for Splunk”. Додаток автоматично буде візуалізації й дашборди для аналізу безпеки.

Підготовка Splunk (увімкнення HEC)

Перед увімкненням HEC створимо робочий індекс з іменем wazuh (через Splunk CLI):

```
sudo -u splunk /opt/splunk/bin/splunk add index wazuh
```

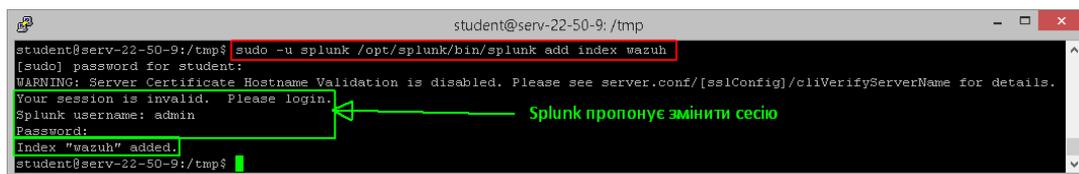


Рис. 13.3. Створення індексу wazuh у CLI

Виконуємо послідовність дій для увімкнення HTTP Event Collector (HEC).

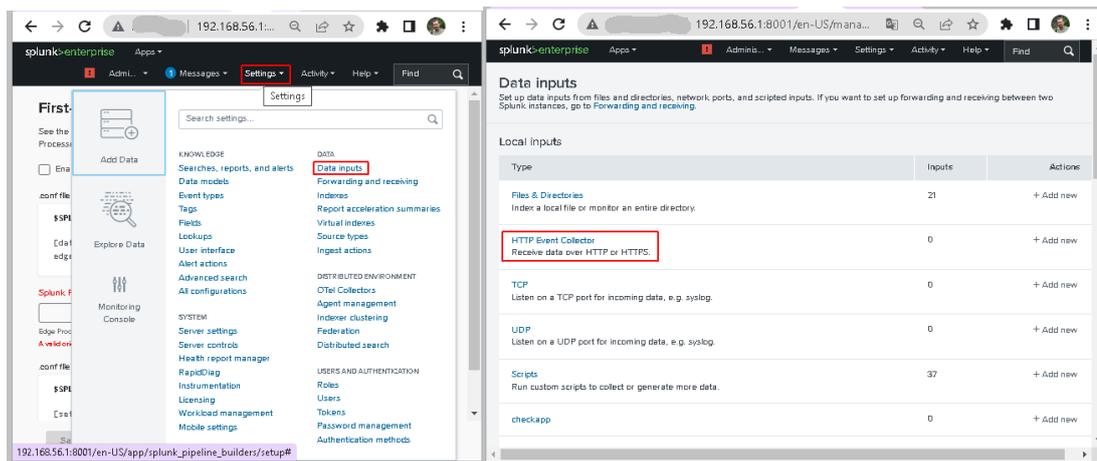


Рис. 13.4. Вхід до меню HEC



Входимо до Splunk Web, переходимо у меню Settings – Data Inputs. У рядку “Splunk REST API Origin” вводимо адресу розгорнутого на хості Serv-G-N-9 Splunk у відповідності до формату HTTPS, навіть якщо він розгорнутий локально <https://<IP адреса твого сплунк сервера>:8088>. Splunk внутрішньо використовує HTTPS для свого REST API на порту 8089 — незалежно від того, чи веб-інтерфейс доступний по HTTPS.

У випадку, наведеному на рис. 13.4 ця адреса відповідає <https://192.168.50.11:8088>. Натискаємо Save внизу вікна та потрапляємо у меню Splunk Data inputs/Local inputs де можливо увімкнути HEC. У списку обираємо HTTP Event Collector.

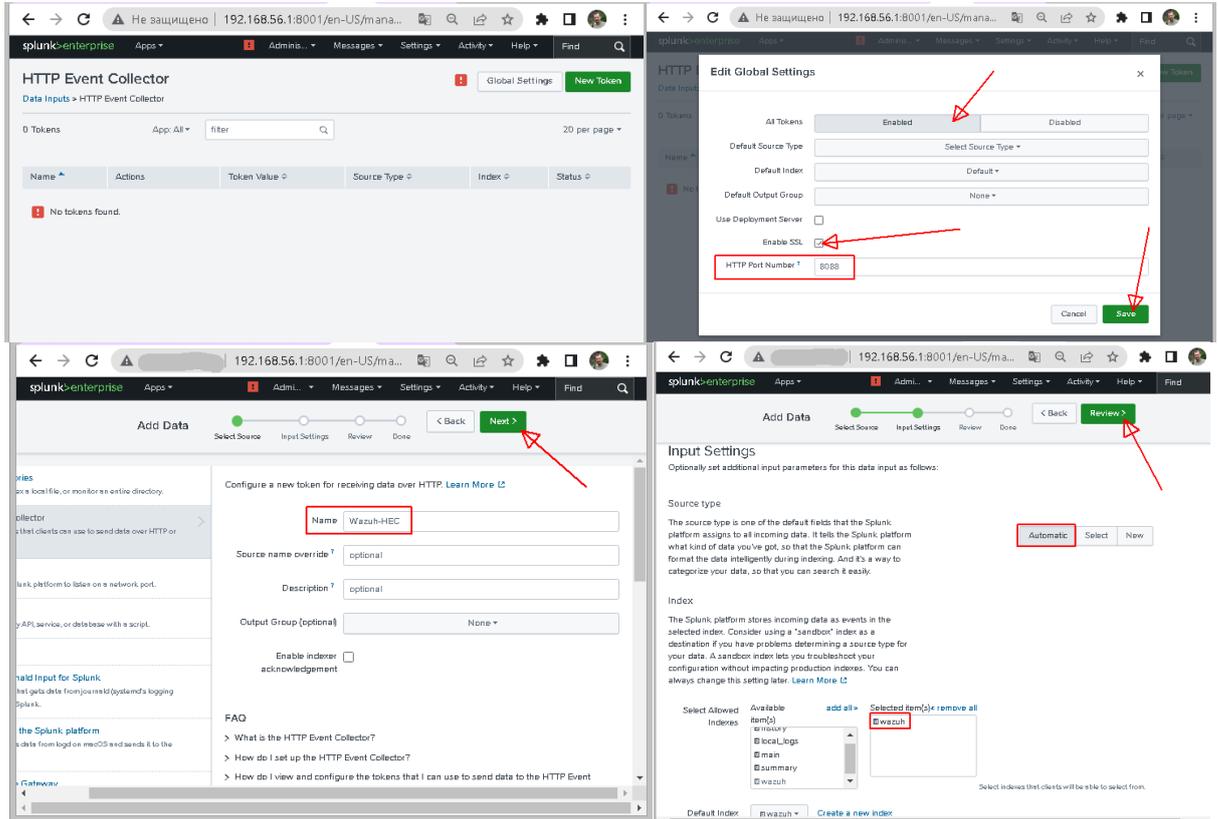


Рис. 13.5. Генерація токена Wazuh-HEC

Натискаємо Global Settings та обираємо All Tokens – Enables, Enable SSL = Yes та HTTP Port Number = 8088 (за замовчуванням) і зберігаємо зміни (Save).

У цьому ж меню натискаємо New Token та потрапляємо у вікно «Configure a new token for receiving data over HTTP» Вводимо назву токена Wazuh-HEC. Переходимо далі у полі Source type залишаємо Automatic та обираємо у полі Index створений на початку цього пункту методики у CLI індекс wazuh. Якщо індекс не обрати, Splunk не прийматиме події. Послідовно обираємо пункти Review та Submit.

Переглянути, перевірити стан та скопіювати Token Value (він знадобиться Filebeat) можливо по кнопці Show у пункті Settings – Data Inputs – HTTP Event Collector (рис.13.6).

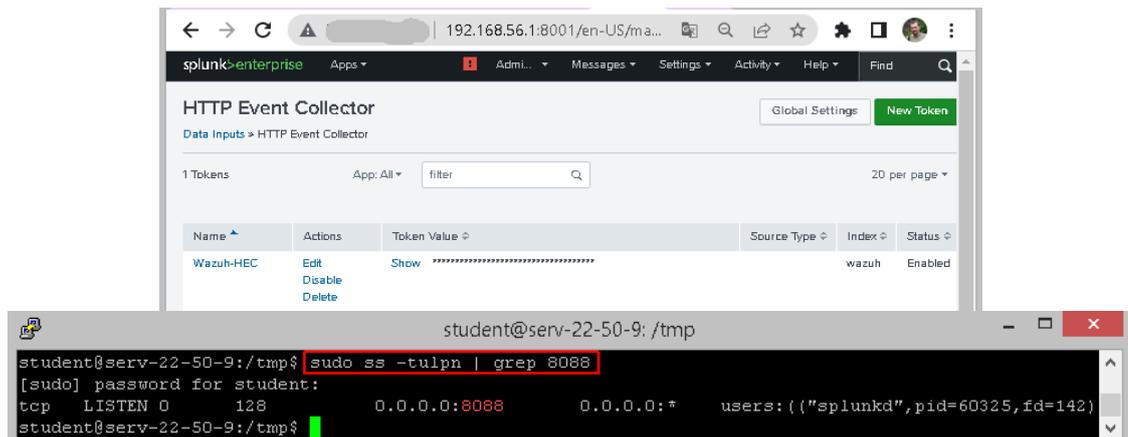


Рис. 13.6. Перегляд Settings – Data Inputs – HTTP Event Collector та перевірка “чи слухається порт 8088”



Перевіряємо прослуховування порту в системі на Splunk-сервері (Ubuntu):

```
sudo ss -tulpn | grep 8088
```

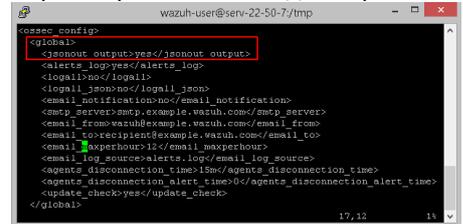
Таким чином, ми отримали очікуваний результат – Splunkd слухає порт 8088 (рис.13.6). Виконані дії у цьому пункті: НЕС увімкнули, порт прослуховується, токен є.

Інтеграція Wazuh – Splunk через Universal Forwarder

Ми використовуємо Wazuh Manager (Appliance), який поставляється з Filebeat, що жорстко прив'язаний до Elasticsearch. Така конфігурація Wazuh не підтримує output.http. Встановимо для вирішення цієї проблеми Splunk Universal Forwarder.

Перед встановленням та першим запуском SplunkUF переконаємось, що JSON-вивід увімкнений у файлі `/var/ossec/etc/ossec.conf`. Перевіряємо, що присутній блок:

```
<global>
  <jsonout_output>yes</jsonout_output>
</global>
```



Перезапускаємо Wazuh Manager та переконаємось, що файл з логами існує:

```
sudo systemctl restart wazuh-manager
sudo ls -l /var/ossec/logs/alerts/alerts.json
```

Оскільки подальші кроки лабораторної роботи передбачають роботу зі службами та конфігураційними файлами Wazuh, а також читання файлів журналів у каталозі `/var/ossec/logs/alerts`, необхідно забезпечити користувачу робочої станції право доступу до цих файлів. У Wazuh Appliance власником файлів Wazuh є користувач `wazuh` і група `wazuh`. За замовчуванням каталоги логів мають права:

```
drwxrwx--- wazuh:wazuh /var/ossec/logs
drwxr-x--- wazuh:wazuh /var/ossec/logs/alerts
```

Тому користувачі, які не входять до групи `wazuh`, не можуть читати файли `alerts.json`. Щоб надати права, додаємо робочого користувача до групи `wazuh`:

```
sudo usermod -aG wazuh wazuh-user
```

Після цього потрібно виконати повторний вхід до системи або команду:

```
su - wazuh-user
```

Після оновлення прав доступ до логів можна перевірити командою:

```
sudo ls -lh /var/ossec/logs/alerts/alerts.json
```

Визначаємо архітектуру нашого віртуального хосту та вантажимо з офіційної сторінки продукту https://www.splunk.com/en_us/download/universal-forwarder.html Splunk Universal Forwarder (RPM) у каталог `/tmp` серверу `Serv-G-N-7` (Amazon Linux 2023), скопіювавши офіційне посилання (рис. 13.7).

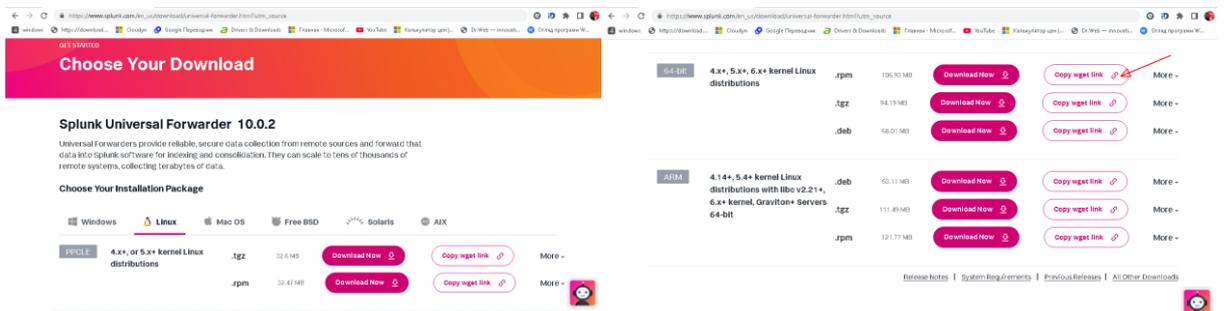


Рис. 13.7. Отримання лінку для завантаження пакету Splunk Universal Forwarder (RPM) 64-bit

```
cd /tmp
wget -O splunkforwarder-10.0.2-e2d18b4767e9.x86_64.rpm
```

```
https://download.splunk.com/products/universalforwarder/releases/10.0.2/linux/splunkforwarder-10.0.2-e2d18b4767e9.x86_64.rpm
```

Якщо `wget` повідомить про помилку 404, отримуємо на сторінці актуальне посилання. В різних версіях змінюється `build-id`, тому 404 — нормальна ситуація.

При виникненні незрозумілих помилок у роботі чи запуску Splunk Universal Forwarder на BM Wazuh Appliance (Amazon Linux 2023) рекомендовано виконати повне чисте видалення Splunk UF та повторити встановлення. Процедура видалення Splunk Universal Forwarder на BM Wazuh Appliance (Amazon Linux 2023) описана у додатку 1 цього документу.

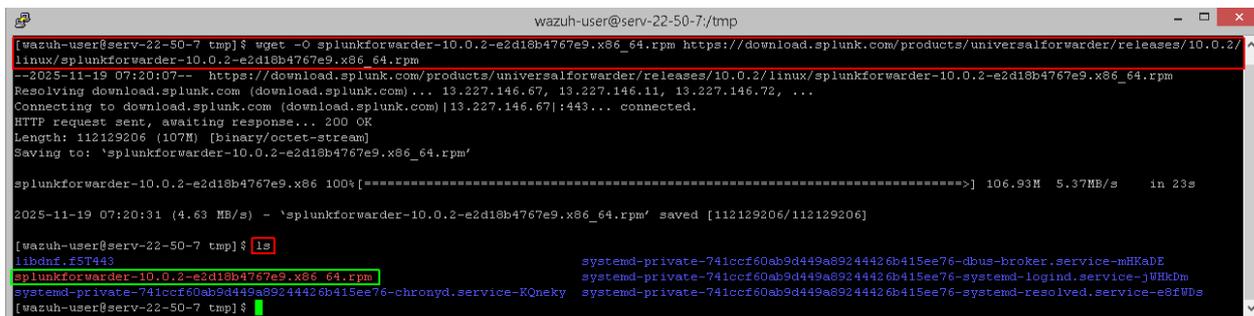


Рис. 13.8. Завантаження пакету Splunk Universal Forwarder (RPM) 64-bit

Перевіряємо, що RPM-пакет завантажено без збоїв та встановлюємо його, підставивши у команду реальне ім'я завантаженого файлу:

```
sudo rpm -ivh splunkforwarder.rpm
```

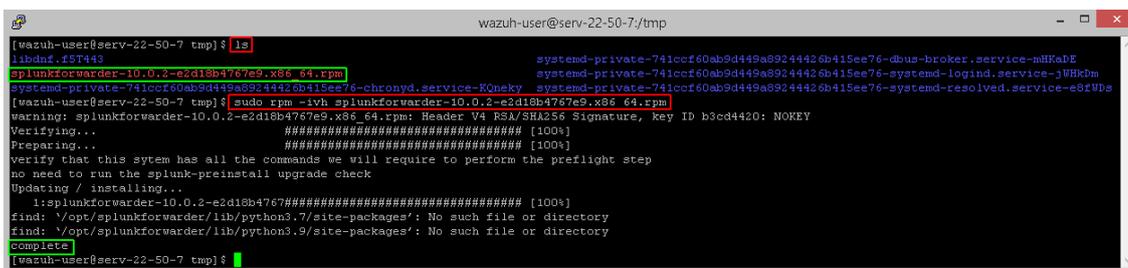


Рис. 13.9. Встановлення пакету Splunk Universal Forwarder (RPM) 64-bit

Автоматично встановлення Splunk UF відбувається у каталог **/opt/splunkforwarder**. Після цього є база-структура каталогів UF.Splunk



Рис. 13.10. Базова структура каталогів Splunk UF

UF автоматично створить користувача splunkfwd, під яким працюватиме служба. Надаємо йому права на Splunk UF каталоги:

```
sudo chown -R splunkfwd:splunkfwd /opt/splunkforwarder
```

Виконуємо перший запуск UF з прийняттям ліцензії. UF згенерує базові сертифікати, створить каталоги конфігурацій і підготує систему до підключення до Deployment Server.

```
sudo -u splunkfwd /opt/splunkforwarder/bin/splunk start --accept-license
```

При виконанні цієї команди Splunk попросить створити адміністратора CLI, що ми виконуємо. Пропоноване ім'я адміністратора splunker, у якості паролю використовуємо будь-який сильний пароль

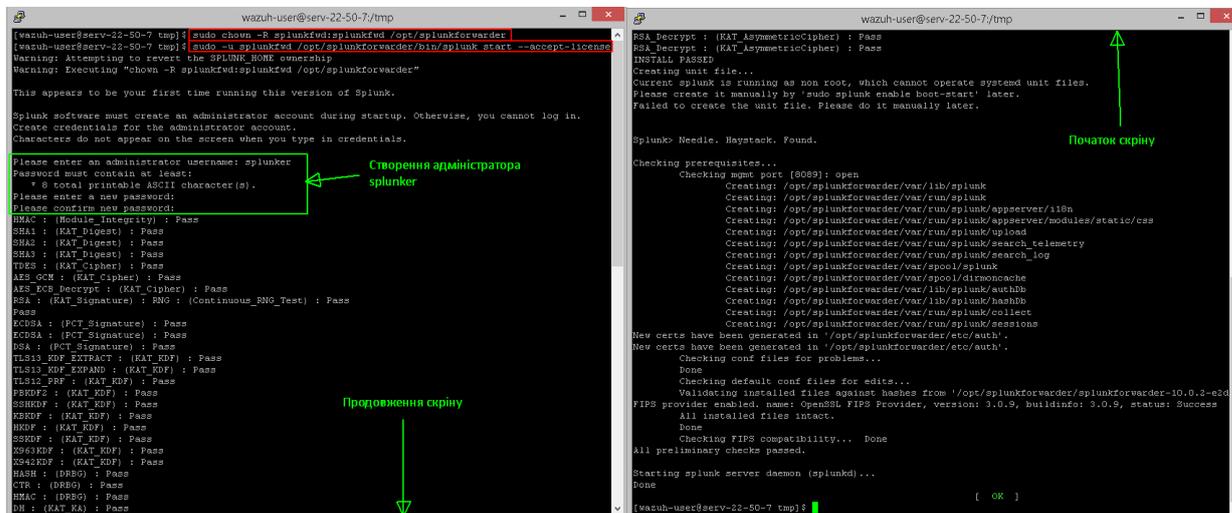


Рис. 13.11. Перший запуск Splunk Universal Forwarder



Створюємо файл `/opt/splunkforwarder/etc/system/local/deploymentclient.conf` – базову конфігурацію Deployment Client, що задає адресу Deployment Server, до якого UF буде підключатися. У якості `<IP_Deployment_Server>` «підставляємо» реальну IP-адресу Splunk серверу (Serv-G-N-9):

```
[deployment-client]
disabled = false

[target-broker:deploymentServer]
targetUri = <IP_Deployment_Server>:8089
```



Додаємо UF у systemd, щоб він стартував при завантаженні системи від користувача splunkfwd.

```
sudo /opt/splunkforwarder/bin/splunk enable boot-start -user splunkfwd
```

Перший виклик Enable boot-start для systemd встановлює unit systemd, але якщо splunkd вже запущений, потрібно його зупинити (`kill -f splunkd`) перед повторним викликом та перезавантажити:

```
sudo kill -f splunkd
sudo /opt/splunkforwarder/bin/splunk enable boot-start -user splunkfwd
sudo systemctl daemon-reload
```

Виконуємо контроль запуску через systemd: старт, зупинка, перезапуск та перевірка статусу UF

```
sudo systemctl start SplunkForwarder
sudo systemctl stop SplunkForwarder
sudo systemctl restart SplunkForwarder
sudo systemctl status SplunkForwarder
```

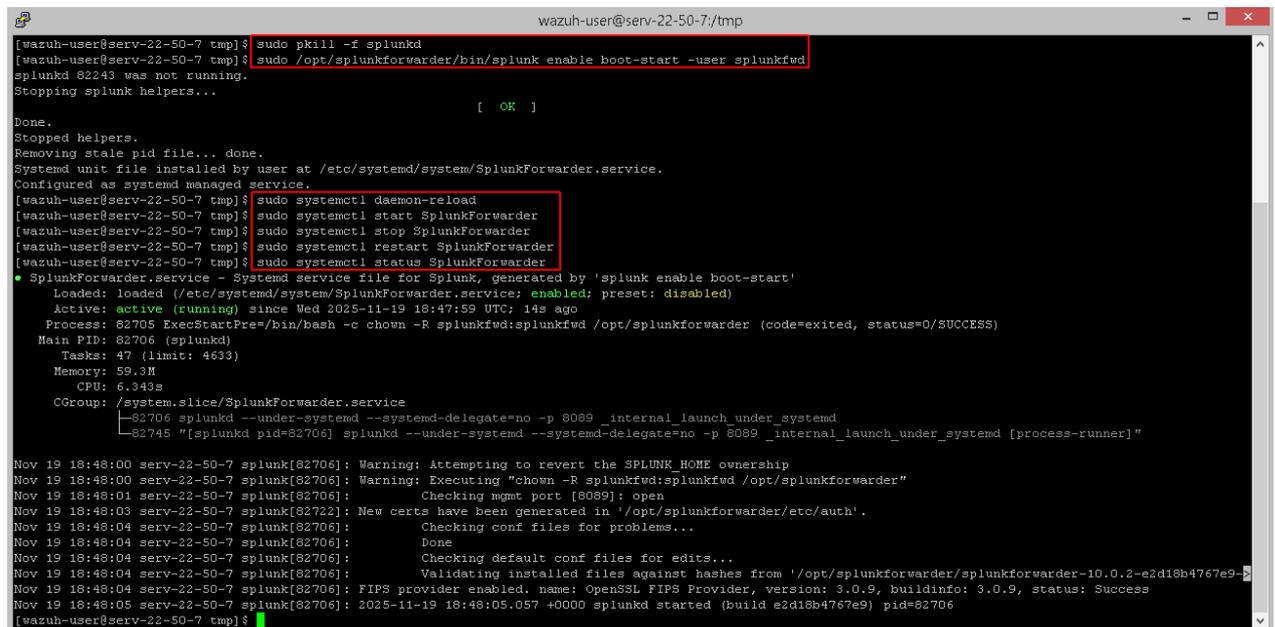


Рис. 13.12. Налаштування та запуск служби Splunk Universal Forwarder

На рис. 13.12 показано налаштування запуску Splunk Forwarder під systemd, служба активна (active, running), основний процес splunkd запущений, права користувача splunkfwd коректні. Тобто, з системної точки зору UF повністю готовий до роботи.

Виконуємо виправлення прав на каталоги (рекомендовано після будь-якого редагування конфігу)

```
sudo chown -R splunkfwd:splunkfwd /opt/splunkforwarder
```

Переходимо до наступного пункту методички.

Автоматичне архівування логів Wazuh для стабільної роботи Filebeat

Filebeat не вміє працювати з вкладеною структурою директорій у модулі wazuh. Щоб уникнути падіння Filebeat через старі логи, можна автоматично переміщати та архівувати старі каталоги Wazuh. Створюємо файл скрипта `/usr/local/bin/wazuh_archive.sh`, куди вставляємо наступний лістинг:

```
#!/bin/bash
ALERTS_DIR="/var/ossec/logs/alerts"
ARCHIVE_DIR="/var/ossec/logs/archives"
DATE=$(date +%Y-%m-%d-%H%M)
mkdir -p "$ARCHIVE_DIR"
for d in "$ALERTS_DIR"/*; do
```



```
[ -d "$d" ] || continue
BASENAME=$(basename "$d")
tar -czf "$ARCHIVE_DIR/alerts-$BASENAME-$DATE.tgz" -C "$ALERTS_DIR" "$BASENAME"
rm -rf "$d"
done
```

Зберігаємо файл та робимо скрипт виконуваним:

```
sudo chmod +x /usr/local/bin/wazuh_archive.sh
```

ВМ, що ми використовуємо (Amazon Linux 2023) не має встановленого пакету cronie або crontabs. Встановлюємо його, активуємо та запускаємо сервіс:

```
sudo dnf install cronie -y
sudo systemctl enable crond
sudo systemctl start crond
sudo systemctl status crond
```

Для налаштування автоматичного запуску скрипта (через cron), відкриваємо cron для root (щоб скрипт мав права на всі файли Wazuh):

```
sudo crontab -e
```

та додаємо рядок, щоб скрипт `/usr/local/bin/wazuh_archive.sh` запускався щодня о 23:50:

```
50 23 * * * /usr/local/bin/wazuh_archive.sh
```

Зберігаємо файл cron та тестуємо роботу скрипта вручну:

```
sudo /usr/local/bin/wazuh_archive.sh
```

Переконаємось, що старі каталоги з `/var/ossec/logs/alerts/` перемістились у `/var/ossec/logs/archives/` у вигляді `.tgz`.

```
wazuh-user@serv-22-50-7:~$ sudo systemctl enable crond
[wazuh-user@serv-22-50-7 ~]$ sudo systemctl start crond
[wazuh-user@serv-22-50-7 ~]$ sudo systemctl status crond
● crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; preset: enabled)
   Active: active (running) since Sat 2025-11-22 18:30:45 UTC; 8s ago
     Main PID: 30377 (crond)
       Tasks: 1 (limit: 5836)
      Memory: 996.0K
         CPU: 13ms
       CGroup: /system.slice/crond.service
              └─30377 /usr/sbin/crond -n

Nov 22 18:30:45 serv-22-50-7 crond[30377]: (CRON) STARTUP (1.5.7)
Nov 22 18:30:45 serv-22-50-7 crond[30377]: (CRON) INFO (Syslog will be used instead of sendmail.)
Nov 22 18:30:45 serv-22-50-7 crond[30377]: (CRON) INFO (RANDOM_DELAY will be scaled with factor 97% if used.)
Nov 22 18:30:45 serv-22-50-7 crond[30377]: (CRON) INFO (running with inotify support)
[wazuh-user@serv-22-50-7 ~]$ sudo crontab -e
no crontab for root - using an empty one
crontab: installing new crontab
[wazuh-user@serv-22-50-7 ~]$ sudo /usr/local/bin/wazuh_archive.sh
[wazuh-user@serv-22-50-7 ~]$ sudo ls /var/ossec/logs/alerts/
alerts.json  alerts.log
[wazuh-user@serv-22-50-7 ~]$ sudo ls /var/ossec/logs/archives/
2025 alerts-2025-11-22-1834.tgz  archives.log
[wazuh-user@serv-22-50-7 ~]$
```

Рис. 13.13. Налаштування автоматичного архівування логів Wazuh

Встановлення та Logstash на Wazuh Appliance (Amazon Linux 2023).

Logstash постачається через репозиторій Elastic. Виконуємо додавання репозиторію. Завантажуємо і встановлюємо ключ GPG та створюємо репозиторій для Elastic:

```
sudo rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

Команда створення репозиторію для Elastic:

```
sudo tee /etc/yum.repos.d/elastic.repo <<EOF
[elastic-8.x]
name=Elastic repository for 8.x packages
baseurl=https://artifacts.elastic.co/packages/8.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
EOF
```



```
wazuh-user@serv-22-50-7:/tmp
[wazuh-user@serv-22-50-7 tmp]$ sudo rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
[wazuh-user@serv-22-50-7 tmp]$ sudo tee /etc/yum.repos.d/elastic.repo <<EOF
[elastic-8.x]
name=Elastic repository for 8.x packages
baseurl=https://artifacts.elastic.co/packages/8.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
EOF
[elastic-8.x]
name=Elastic repository for 8.x packages
baseurl=https://artifacts.elastic.co/packages/8.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
[wazuh-user@serv-22-50-7 tmp]$ yum list logstash
```

Команда створення репозиторію для Elastic

Рис. 13.14. Встановлення ключа та створення репозиторію для Elastic

Перевіряємо доступність пакету Logstash (рис.13.15):

```
yum list logstash
```

Для встановлення Logstash виконуємо:

```
sudo yum install -y logstash
```

Після встановлення перевіряємо, чи з'явилися каталоги:

```
ls -l /etc/logstash
```

```
ls -l /etc/logstash/conf.d
```

```
wazuh-user@serv-22-50-7:/tmp
[wazuh-user@serv-22-50-7 tmp]$ yum list logstash
Amazon Linux 2023 repository                               740 kB/s | 47 MB   01:05
Amazon Linux 2023 Kernel Livepatch repository            33 kB/s | 29 kB   00:00
Elastic repository for 8.x packages                       5% [====]
Elastic repository for 8.x packages                       5% [====]
Elastic repository for 8.x packages                       5% [====]
Elastic repository for 8.x packages                       9% [=====]
Elastic repository for 8.x packages                       9% [=====]
Elastic repository for 8.x packages                       9% [=====]
Elastic repository for 8.x packages                       10% [=====]
Elastic repository for 8.x packages                       12% [=====]
Elastic repository for 8.x packages                       12% [=====]
Elastic repository for 8.x packages                       12% [=====]
Elastic repository for 8.x packages                       1.4 MB/s | 140 MB 01:38
EL-2023.9.20251014 - Wazuh                               2.3 MB/s | 42 MB  00:17
Last metadata expiration check: 0:00:01 ago on Fri Nov 21 07:55:33 2025.
Available Packages
logstash.aarch64                                         1:8.19.7-1      elastic-8.x
logstash.x86_64                                         1:8.19.7-1      elastic-8.x
```

Рис. 13.15. Доступність пакету Logstash

Wazuh – Logstash – Splunk інтеграція потребує правильного формування подій у специфічному форматі, який сприймає Splunk HTTP Event Collector (HEC). Для цього в Logstash необхідно встановити спеціалізований плагін `logstash-output-exec`, що спрощує процес форматування та передачу даних. Встановлення плагіну та перевірка виконуються командами:

```
sudo /usr/share/logstash/bin/logstash-plugin install logstash-output-exec
```

```
sudo /usr/share/logstash/bin/logstash-plugin list | grep output-exec
```

```
wazuh-user@serv-22-50-7:~
[wazuh-user@serv-22-50-7 ~]$ sudo /usr/share/logstash/bin/logstash-plugin install logstash-output-exec
Using bundled JDK: /usr/share/logstash/jdk
Validating logstash-output-exec
Resolving mixin dependencies
Installing logstash-output-exec
Installation successful
[wazuh-user@serv-22-50-7 ~]$ sudo /usr/share/logstash/bin/logstash-plugin list | grep output-exec
Using bundled JDK: /usr/share/logstash/jdk
logstash-output-exec
[wazuh-user@serv-22-50-7 ~]$
```

Рис. 13.16. Встановлення плагіну `logstash-output-exec`

На попередньому кроці до каталогу `/etc/logstash` встановлено пакет, а каталог `/etc/logstash/conf.d` призначений для створення pipeline, який ми описуємо у конфігураційному файлі `wazuh-to-splunk.conf`

Вміст файлу конфігурації pipeline `/etc/logstash/conf.d/wazuh-to-splunk.conf` з коментарями приведено у додатку 3 цієї методики. Редагуємо файл за приведеним зразком, замінюючи значення `token_Wazuh-HEC` на власне, яке копіюємо з WEB-UI Splunk по кнопці Show у пункті Settings – Data Inputs – HTTP Event Collector (рис.13.6). Не забуваємо змінити IP-адресу Splunk-серверу (`192.168.50.11`) та ім'я хосту (`serv-22-50-7`), де розгорнуто Wazuh-Logstash.



У додатку 3 параметри, які необхідно змінити, виділені *таким кольором* ☺, а коментарі виділено *таким кольором* ☺

Конфігураційний файл Logstash описує pipeline, що складається з трьох основних секцій — input, filter та output.

Секція **input** відповідає за приймання подій від Filebeat. У конфігурації використовується вхідний плагін beats, який відкриває порт 5044 та приймає структуровані JSON-події, що надходять від модуля Wazuh-Agent.

Секція **filter** виконує попередню обробку отриманих даних перед їхньою передачею до Splunk. Для цього застосовуються:

- фільтр **ruby** — для формування Unix-мітки часу, необхідної HEC;
- фільтр **mutate** — для видалення службових полів, які не повинні потрапляти у Splunk;
- додатковий блок **ruby** — для побудови структурованого JSON-об'єкта у форматі, що відповідає вимогам Splunk HEC (time, host, index, sourcetype, event). Результат формування зберігається у полі `hec_payload`.

Секція **output** забезпечує відправлення події до Splunk. У ній використовується плагін `hec`, який викликає команду `curl` для виконання HTTPS-запиту до Splunk HTTP Event Collector. Дані передаються у вигляді JSON-рядка з поля `hec_payload`, сформованого у секції `filter`. Для контролю та налагодження передбачений додатковий вихід `stdout` з виведенням сформованого HEC-повідомлення у консоль.

Вмикаємо Logstash у автозапуск, запускаємо та переглядаємо статус (рис. 13.17):

```
sudo systemctl enable logstash
sudo systemctl start logstash
sudo systemctl status logstash
```

```
wazuh-user@serv-22-50-7/tmp
[wazuh-user@serv-22-50-7 tmp]$ sudo systemctl enable logstash
[wazuh-user@serv-22-50-7 tmp]$ sudo systemctl start logstash
[wazuh-user@serv-22-50-7 tmp]$ sudo systemctl status logstash
● logstash.service - logstash
   Loaded: loaded (/usr/lib/systemd/system/logstash.service; enabled; preset: disabled)
   Active: active (running) since Fri 2025-11-21 09:10:44 UTC; 39s ago
     Main PID: 87877 (java)
       Tasks: 15 (limit: 4633)
      Memory: 396.1M
         CPU: 37.422s
    CGroup: /system.slice/logstash.service
            └─87877 /usr/share/logstash/jdk/bin/java -Xms1g -Xmx1g -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djruby.compile.invokedynamic=true -XX:+HeapDumpOnCrash

Nov 21 09:10:44 serv-22-50-7 logstash[87877]: Using bundled JDK: /usr/share/logstash/jdk
[wazuh-user@serv-22-50-7 tmp]$
```

Рис. 13.17. Вмикання автозапуску, запуск та перегляд статусу Logstash

Заміна стандартної версії Filebeat, запуск та тестування конвейера.

Стандартна версія Filebeat, що постачається у складі VM Wazuh appliance, може бути несумісною з актуальними модулями Logstash або використовуваними плагінами. Тому перед подальшим налаштуванням необхідно замінити вбудовану версію Filebeat на стабільну та сумісну редакцію. Імпорт GPG-ключа та додавання репозиторію Elastic 7.x було виконано на одному з попередніх кроків (рис. 13.14). Видаляємо вбудовану версію Filebeat:

```
sudo dnf remove filebeat -y
sudo dnf install filebeat-7.17.9 -y
```

Налаштовуємо Filebeat для відправки логів у Logstash. Для цього виконуємо бекап файлу конфігурації Filebeat та створюємо новий файл `/etc/filebeat/filebeat.yml` мінімальної конфігурації, приведені у додатку 4, таблиця 13.1.

```
sudo mv /etc/filebeat/filebeat.yml ~ filebeat.yml.bkp
```

Filebeat міг зберегти старі дані (state), які не збігаються з новою версією. Щоб запобігти такій помилці конфігурації виконуємо видалення наступних файлів:

```
sudo rm -rf /var/lib/filebeat/{registry,registry-filebeat,meta.json}
```

Перевіряємо конфігурацію Filebeat:

```
sudo filebeat test config
```

Має бути **Config OK**. Переконаємось, що Filebeat встановлено як системний сервіс:

```
which filebeat
```

Очікується `/usr/bin/filebeat`. Якщо помилок не виникло, виконуємо повну перевірку конфігу:

```
sudo /usr/share/logstash/bin/logstash --config.test_and_exit -f /etc/logstash/conf.d/wazuh-to-splunk.conf
```

Має бути: **Configuration OK** (рис.13.18).



Перезапускаємо Logstash, дивимось статус (має бути активний) та перевіряємо чи «слухає» порт 5044. Перевірку порту виконуємо через 40-50 сек після перезапуску logstash. Існує висока ймовірність затримки.

```
sudo systemctl restart logstash
sudo systemctl status logstash
sudo ss -tulnp | grep 5044
```

```
wazuh-user@serv-22-50-7-~$ sudo filebeat test config
Config OK
wazuh-user@serv-22-50-7-~$ which filebeat
/usr/bin/filebeat
wazuh-user@serv-22-50-7-~$ sudo /usr/share/logstash/bin/logstash --config.test_and_exit -f /etc/logstash/conf.d/wazuh-to-splunk.conf
Using bundled JDK: /usr/share/logstash/jdk
WARNING: Could not find logstash.yml which is typically located in $LS_HOME/config or /etc/logstash. You can specify the path using --path.settings. Continuing using the defaults
Could not find log4j2 configuration at path /usr/share/logstash/config/log4j2.properties. Using default config which logs errors to the console
[WARN ] 2025-11-23 17:35:13.090 [main] runner - Starting from version 9.0, running with superuser privileges is not permitted unless you explicitly set 'allow_superuser' to true. Proceeding acknowledging the possible security risks
[WARN ] 2025-11-23 17:35:13.102 [main] runner - NOTICE: Running Logstash as a superuser is strongly discouraged as it poses a security risk. Set 'allow_superuser' to false for better security.
[WARN ] 2025-11-23 17:35:13.140 [main] runner - 'pipeline.buffer.type' setting is not explicitly defined. Before moving to 9.x set it to 'heap' and tune heap size upwrd, or set it to 'direct' to maintain existing behavior.
[INFO ] 2025-11-23 17:35:13.143 [main] runner - Starting Logstash ("logstash.version">"8.19.7", "jruby.version">"jruby 9.4.9.0 (3.1.4) 2024-11-04 547c6b150c OpenJDK 64-Bit Server VM 21.0.9+10-LTS on 21.0.9+10-LTS sindy #11 [888 64-bitmx]")
[INFO ] 2025-11-23 17:35:13.148 [main] runner - JVM bootstrap flags: [-Xms1g, -Xmx1g, -Djava.awt.headless=true, -Dfile.encoding=UTF-8, -Djruby.compile.invokedynamic=true, -XX:+HeapDumpOnOutOfMemoryError, -Djava.security.egd=file:/dev/urandom, -Dlog4j2.isThreadContextMapInheritable=true, -Djruby.regexp.interruptible=true, -Djdk.io.File.enableADS=true, --add-exports=jdk.compiler/com.sun.tools.javac.api=ALL-UNNAMED, --add-exports=jdk.compiler/com.sun.tools.javac.file=ALL-UNNAMED, --add-exports=jdk.compiler/com.sun.tools.javac.parser=ALL-UNNAMED, --add-exports=jdk.compiler/com.sun.tools.javac.tree=ALL-UNNAMED, --add-exports=jdk.compiler/com.sun.tools.javac.util=ALL-UNNAMED, --add-opens=java.base/java.security=ALL-UNNAMED, --add-opens=java.base/java.io=ALL-UNNAMED, --add-opens=java.base/java.nio.channels=ALL-UNNAMED, --add-opens=java.base/java.management/sun.management=ALL-UNNAMED, -Dio.netty.allocation.maxOrder=1]
[INFO ] 2025-11-23 17:35:13.755 [main] StreamReadConstraintsUtil - Jackson default value override 'logstash.jackson.stream-read-constraints.max-string-length' configured to '200000000' (logstash default)
[INFO ] 2025-11-23 17:35:13.758 [main] StreamReadConstraintsUtil - Jackson default value override 'logstash.jackson.stream-read-constraints.max-number-length' configured to '10000' (logstash default)
[INFO ] 2025-11-23 17:35:13.759 [main] StreamReadConstraintsUtil - Jackson default value override 'logstash.jackson.stream-read-constraints.max-nesting-depth' configured to '1000' (logstash default)
[WARN ] 2025-11-23 17:35:14.610 [LogStash:Runner] multilocal - Ignoring the 'pipelines.yml' file because modules or command line options are specified
[INFO ] 2025-11-23 17:35:16.618 [LogStash:Runner] Reflections - Reflections took 606 ms to scan 1 uris, producing 150 keys and 530 values
[INFO ] 2025-11-23 17:35:18.514 [LogStash:Runner] javapipeline - Pipeline 'main' is configured with 'pipeline.ecs_compatibility: v8' setting. All plugins in this pipeline will default to 'ecs_compatibility => v8' unless explicitly configured otherwise.
Configuration OK
[INFO ] 2025-11-23 17:35:18.520 [LogStash:Runner] runner - Using config.test_and_exit mode. Config Validation Result: OK. Exiting Logstash
wazuh-user@serv-22-50-7-~$
```

Рис. 13.18. Перевірка коректності конфігурації Filebeat та Logstash

```
wazuh-user@serv-22-50-7-~$ sudo systemctl restart logstash
wazuh-user@serv-22-50-7-~$ sudo systemctl status logstash
● logstash.service - logstash
   Loaded: loaded (/usr/lib/systemd/system/logstash.service; enabled; preset: disabled)
   Active: active (running) since Sat 2025-11-22 17:35:34 UTC; 8s ago
     Main PID: 2949 (java)
       Tasks: 14 (limit: 5836)
      Memory: 162.4M
         CPU: 9.454s
    CGroup: /system.slice/logstash.service
            └─2949 /usr/share/logstash/jdk/bin/java -Xms1g -Xmx1g -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djruby.compile.invokedynamic=true -XX:+HeapDumpOnO

Nov 22 17:35:34 serv-22-50-7 logstash[2949]: Using bundled JDK: /usr/share/logstash/jdk
wazuh-user@serv-22-50-7-~$ sudo ss -tulnp | grep 5044
tcp LISTEN 0 4096          *:5044          *:                users: ({"java", pid=2949, id=97})
wazuh-user@serv-22-50-7-~$
```

Рис. 13.19. Перевірка коректності запуску Logstash

Перезапускаємо Filebeat та перевіряємо його повний стан і логи:

```
sudo systemctl restart filebeat
sudo systemctl status filebeat -l
```

```
wazuh-user@serv-22-50-7-~$ sudo systemctl restart filebeat
wazuh-user@serv-22-50-7-~$ sudo systemctl status filebeat -l
● filebeat.service - Filebeat sends log files to Logstash or directly to Elasticsearch.
   Loaded: loaded (/etc/systemd/system/filebeat.service; enabled; preset: disabled)
   Active: active (running) since Fri 2025-12-05 17:11:40 UTC; 19s ago
     Docs: https://www.elastic.co/products/beats/filebeat
     Main PID: 69300 (filebeat)
       Tasks: 7 (limit: 5836)
      Memory: 32.1M
         CPU: 236ms
    CGroup: /system.slice/filebeat.service
            └─69300 /usr/share/filebeat/bin/filebeat -c /etc/filebeat/filebeat.yml -e

Dec 05 17:11:41 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:41.728Z      INFO      [crawler]      beater/crawler.go:117      starting input, keys present
Dec 05 17:11:41 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:41.729Z      WARN      [cfgwarn]      log/input.go:89      DEPRECATED: Log input. Use Filestre
Dec 05 17:11:41 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:41.729Z      INFO      [input]        log/input.go:171      Configured paths: [/var/ossec/logs/s
Dec 05 17:11:41 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:41.729Z      INFO      [crawler]      beater/crawler.go:148      Starting input (ID: 869896530)
Dec 05 17:11:41 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:41.729Z      INFO      [crawler]      beater/crawler.go:106      Loading and starting inputs
Dec 05 17:11:51 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:51.739Z      INFO      [input.harvester] log/harvester.go:309      Harvester started for
Dec 05 17:11:52 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:52.740Z      INFO      [publisher_pipeline_output] pipeline/output.go:143      Connecting
Dec 05 17:11:52 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:52.740Z      INFO      [publisher]    pipeline/retry.go:219      retryer: send await signal
Dec 05 17:11:52 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:52.740Z      INFO      [publisher]    pipeline/retry.go:223      done
Dec 05 17:11:52 serv-22-50-7 filebeat[69300]: 2025-12-05T17:11:52.740Z      INFO      [publisher_pipeline_output] pipeline/output.go:151      Connection
```

Рис. 13.20. Перезапуск filebeat після редагування.

Переглядаємо журнал у пошуках маркерів успішної роботи logstash (рис.13.21)

```
sudo journalctl -u logstash -f
```

Існує 3 групи маркерів, кожен з яких сигналізує про окремий етап успішної роботи:

- ✓ У журналі присутні рядки з [HEC-SENT] Це означає, що Logstash успішно сформував коректний HEC JSON.
- ✓ Присутні повідомлення {"text": "Success", "code": 0} Це означає, що Splunk HTTP Event Collector прийняв подію.
- ✓ Повторювані цикли HEC-SENT + Success при різних подіях. Це означає стабільну, безперебійну роботу інтеграції.



Додаток 1.

Повне чисте видалення Splunk Universal Forwarder Wazuh Appliance (Amazon Linux 2023)

Процедура видалення Splunk Universal Forwarder приведена на випадок необхідності «відкату» та реконфігурування побудованої системи. У описаній методиці ця процедура не використовується.

Зупиняємо UF і видаляємо systemd unit

```
sudo systemctl stop SplunkForwarder
sudo systemctl disable SplunkForwarder
sudo rm -f /etc/systemd/system/SplunkForwarder.service
sudo rm -f /usr/lib/systemd/system/SplunkForwarder.service
sudo systemctl daemon-reload
```

Видаляємо rpm-пакет UF. Якщо rpm скаже “not installed” — значить вже видалено, і це ок.

```
sudo rpm -e splunkforwarder
```

Видаляємо каталоги Splunk UF

```
sudo rm -rf /opt/splunkforwarder
sudo rm -rf /opt/splunk
```

Видаляємо створених користувачів. UF створює splunkfwd та splunker.

```
sudo userdel -r splunkfwd
sudo userdel -r splunker
```

Якщо скаже “user does not exist” — це нормально.

Додаток 2.

Видалення сертифікату на Wazuh Appliance (Amazon Linux 2023)

Процедура видалення сертифікату приведена на випадок необхідності його заміни. У описаній методиці ця процедура не використовується.

Перед видаленням не потрібно alias, перевіряємо, що він існує

```
sudo /usr/share/logstash/jdk/bin/keytool -list \
  -keystore /usr/share/logstash/jdk/lib/security/cacerts \
  -storepass changeit | grep splunk-ca
```

Якщо рядок знайдено — alias точно є і видаляємо не потрібний сертифікат

```
sudo /usr/share/logstash/jdk/bin/keytool \
  -delete \
  -alias splunk-ca \
  -keystore /usr/share/logstash/jdk/lib/security/cacerts \
  -storepass changeit
```



Приклад налаштування файлу конфігурації pipeline /etc/logstash/conf.d/wazuh-to-splunk.conf

```
input {
  beats {
    port => 5044
  }
}

filter {
  # 1 — Конвертація timestamp у Unix time (цілі секунди)
  ruby {
    code => '
    event.set("[@metadata][hec_time]", event.get("@timestamp").to_i)
    '
  }
  # 2 — Очищення зайвих кореневих полів
  mutate {
    remove_tag => ["_jsonparsefailure"]
    remove_field => ["@version", "tags", "ecs", "[agent]", "[input]", "[log]", "[predecoder]", "[manager]", "[host]", "type",
    "path", "@timestamp"]
  }
  # 3 — Формування Splunk HEC JSON
  ruby {
    code => '
    require "json"
    clean_event_json = event.to_json
    hec_payload = {
      "time" => event.get("[@metadata][hec_time]"),
      "host" => "serv-22-50-7",
      "index" => "wazuh",
      "sourcetype" => "wazuh_alerts",
      "event" => clean_event_json
    }
    event.set("hec_payload", hec_payload.to_json)
    '
  }
}

output {
  if [hec_payload] {
    # 4 — Відправка через exec → curl
    exec {
      command => "curl -k -s -X POST -H 'Authorization: Splunk token_Wazuh-HEC' -H 'Content-Type: application/json'
      'https://192.168.50.11:8088/services/collector/event' -d '{{hec_payload}}'"
    }
    # 5 — Логування у stdout для аудиту
    stdout {
      codec => line {
        format => '[HEC-SENT] '{{hec_payload}}'
      }
    }
  }
}
```



Приклад налаштування Filebeat для відправки логів у Logstash. Файл конфігурації Filebeat /etc/filebeat/filebeat.yml

Таблиця 13.1.

Конфігураційний файл після редагування

```
# Wazuh - Filebeat configuration file

output.logstash:
  hosts: ["127.0.0.1:5044"]

filebeat.inputs:
  - type: log
    enabled: true
    paths:
      - /var/ossec/logs/alerts/alerts.json
    json.keys_under_root: true
    json.add_error_key: true

setup.template.json.enabled: true
setup.template.json.path: '/etc/filebeat/wazuh-template.json'
setup.template.json.name: 'wazuh'
setup.ilm.overwrite: true
setup.ilm.enabled: false

logging.level: info
logging.to_files: true
logging.files:
  path: /var/log/filebeat
  name: filebeat
  keepfiles: 7
  permissions: 0644

logging.metrics.enabled: false
```



Імпорт CA Splunk у Logstash

У додатку описана процедура додавання кореневого сертифікату CA Splunk (CN=SplunkCommonCA) у Java truststore Logstash. Підключаємось по SSH до серверу Splunk Serv-G-N-9 [192.168.50.11]

Зберігаємо сертифікати, які видає порт НЕС (8088) у файлі **/tmp/splunk-full-chain.pem**:

```
openssl s_client -connect 192.168.50.11:8088 -showcerts </dev/null \  
| sed -n '/BEGIN CERTIFICATE/,/END CERTIFICATE/p' > /tmp/splunk-full-chain.pem
```

Розбиваємо ланцюжок збережених сертифікатів на окремі файли (рис.13.14):

```
csplit -f cert -b "%02d.pem" /tmp/splunk-full-chain.pem "/-----BEGIN CERTIFICATE-----/" "{*}"
```

```
student@serv-22-50-9: /tmp
student@serv-22-50-9:/tmp$ openssl s_client -connect 192.168.50.11:8088 -showcerts </dev/null \
| sed -n '/BEGIN CERTIFICATE/,/END CERTIFICATE/p' > /tmp/splunk-full-chain.pem
Can't use SSL_get_servername
depth=1 C = US, ST = CA, L = San Francisco, O = Splunk, CN = SplunkCommonCA, emailAddress = support@splunk.com
verify error:num=19:self-signed certificate in certificate chain
verify return:1
depth=1 C = US, ST = CA, L = San Francisco, O = Splunk, CN = SplunkCommonCA, emailAddress = support@splunk.com
verify return:1
depth=0 CN = SplunkServerDefaultCert, O = SplunkUser
verify return:1
DONE
student@serv-22-50-9:/tmp$ ls
snap-private-tmp
splunk-full-chain.pem
systemd-private-a332162210044734ac3375e8f1d36dfd-fwupd.service-jC8i1M
systemd-private-a332162210044734ac3375e8f1d36dfd-ModemManager.service-zhm59n
systemd-private-a332162210044734ac3375e8f1d36dfd-polkit.service-qGOLJo
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-logind.service-c7uN4S
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-resolved.service-je6Ny1
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-timesyncd.service-D88FVf
systemd-private-a332162210044734ac3375e8f1d36dfd-upower.service-QYukyM
student@serv-22-50-9:/tmp$ csplit -f cert -b "%02d.pem" /tmp/splunk-full-chain.pem "/-----BEGIN CERTIFICATE-----/" "{*}"
0
1184
1265
student@serv-22-50-9:/tmp$ ls
cert-00.pem
cert-01.pem
cert-02.pem
snap-private-tmp
splunk-full-chain.pem
systemd-private-a332162210044734ac3375e8f1d36dfd-fwupd.service-jC8i1M
systemd-private-a332162210044734ac3375e8f1d36dfd-ModemManager.service-zhm59n
systemd-private-a332162210044734ac3375e8f1d36dfd-polkit.service-qGOLJo
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-logind.service-c7uN4S
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-resolved.service-je6Ny1
systemd-private-a332162210044734ac3375e8f1d36dfd-systemd-timesyncd.service-D88FVf
systemd-private-a332162210044734ac3375e8f1d36dfd-upower.service-QYukyM
student@serv-22-50-9:/tmp$
```

Рис. 13.14. Отримання файлів сертифікатів серверу Splunk Serv-G-N-9 [192.168.50.11]

В результаті описаній дій (рис.13.14) у каталозі /tmp серверу збережено три файли сертифікатів. Визначаємо, кореневий сертифікат по вмісту поля subject у виводі команд:

```
openssl x509 -in cert-01.pem -noout -text  
openssl x509 -in cert-02.pem -noout -text
```

```
student@serv-22-50-9:/tmp$ openssl x509 -in cert-01.pem -noout -text
Certificate:
  Data:
    Version: 1 (0x0)
    Serial Number:
      99:54:03:79:33:74:f8:b1:56:b7:b1:83:15:6a:e4:95:15:2d:23:2e
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C = US, ST = CA, L = San Francisco, O = Splunk, CN = SplunkCommonCA,
    emailAddress = support@splunk.com
    Validity
      Not Before: Nov 15 14:11:41 2025 GMT
      Not After : Nov 14 14:11:41 2028 GMT
    Subject: CN = SplunkServerDefaultCert, O = SplunkUser
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
      Modulus:
        00:b8:e8:92:b6:af:21:71:da:36:56:b5:1d:db:b8:
        5e:51:5b:44:92:c6:b3:72:e3:e4:5d:09:ec:67:ae:
        77:71:bc:e5:e7:25:95:40:f2:59:74:a9:da:3b:3d:
        40:0d:4b:16:4d:bd:4c:9c:06:65:ac:83:12:2e:df:
        35:e7:e8:76:b6:47:36:3d:d9:77:55:e8:73:d4:78:
        13:e4:09:9b:26:89:b1:44:1e:a3:11:29:e0:e8:af:
        cb:93:4b:94:00:d7:48:32:13:01:f8:2d:72:dff:a6:37:
        cc:73:af:6c:eb:bd:79:8d:d1:0c:23:cb:d9:bb:ab:
        f0:5a:fc:c9:29:b6:3e:9e:e0:59:40:79:93:c3:0a:
        df:24:b5:4a:b4:59:6c:33:11:d8:86:28:67:d4:27:
        6f:c9:0e:0c:21:90:00:ed:60:c9:b5:61:dc:9a:69:
        1c:3f:62:14:8b:77:65:2e:8c:dd:45:88:c9:45:c1:
        3f:d0:2d:69:14:95:4a:7c:6f:85:21:1b:46:0a:3b:
  -----BEGIN CERTIFICATE-----
  MIICpDCCBQAwggEiMBQGA1UEAwwuSplunkServerDefaultCert
  -----END CERTIFICATE-----

student@serv-22-50-9:/tmp$ openssl x509 -in cert-02.pem -noout -text
Certificate:
  Data:
    Version: 1 (0x0)
    Serial Number:
      9d:1c:13:7c:b6:3f:c5:e3
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C = US, ST = CA, L = San Francisco, O = Splunk, CN = SplunkCommonCA,
    emailAddress = support@splunk.com
    Validity
      Not Before: Jan 30 20:26:54 2017 GMT
      Not After : Jan 28 20:26:54 2027 GMT
    Subject: C = US, ST = CA, L = San Francisco, O = Splunk, CN = SplunkCommonCA,
    emailAddress = support@splunk.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
      Modulus:
        00:cc:1f:65:b5:51:06:93:bd:d0:bc:f9:71:5e:d0:
        34:a8:c5:bf:46:b0:d0:95:03:05:27:f0:bf:b5:74:
        55:25:d4:2c:99:c5:b8:5a:c6:84:4e:d7:9e:59:98:
        7c:02:0a:33:3b:5e:9a:ad:c8:e9:49:ab:58:bf:7:
        2f:e0:b9:94:02:1e:d3:de:26:56:55:31:cb:22:ra:1:
        4d:58:b4:95:53:ad:a0:d1:79:27:a3:7c:52:81:01:
        0f:4a:f1:0a:04:da:99:45:9b:f5:49:6b:f1:df:ea:
        06:1e:0b:4d:0c:06:b8:a9:38:f4:1f:22:ee:8e:a5:
        03:01:11:57:b1:1b:c1:3d:6f:d5:7c:39:28:98:78:
        fd:6f:c2:3e:2b:7a:8e:53:04:2d:20:e5:2b:fa:82:
        b1:d6:d3:35:27:b5:4d:37:93:5f:94:bc:3d:35:d1:
        ed:95:f6:e5:c3:46:2c:e7:12:39:43:1b:b7:ad:c0:
  -----BEGIN CERTIFICATE-----
  MIICpDCCBQAwggEiMBQGA1UEAwwuSplunkCommonCA
  -----END CERTIFICATE-----
```

Рис. 13.15. Перегляд інформації про сертифікати серверу Splunk Serv-G-N-9 [192.168.50.11]

Перегляд інформації про сертифікати дає наступне:

- ❖ **/tmp/cert-01.pem** — серверний сертифікат Splunk
- ❖ **/tmp/cert-02.pem** — кореневий CA Splunk (SplunkCommonCA). Саме його треба імпортувати.

Переіменовуємо кореневий CA:

```
mv /tmp/cert-02.pem /tmp/splunk-ca.pem
```



Копіюємо Splunk CA на Logstash-сервер. Замініть адресу 192.168.50.9 на адресу свого Serv-G-N-7 у рядку та виконайте на Splunk-сервері:

```
scp -o StrictHostKeyChecking=no /tmp/splunk-ca.pem wazuh-user@192.168.50.9:/tmp/splunk-ca.pem
```

Далі користувач вводить пароль wazuh. Це – стандартний пароль для Wazuh Appliance (рис.13.16).

```
student@serv-22-50-9:/tmp$ mv /tmp/cert-02.pem /tmp/splunk-ca.pem
student@serv-22-50-9:/tmp$ scp -o StrictHostKeyChecking=no /tmp/splunk-ca.pem wazuh-user@192.168.50.9:/tmp/splunk-ca.pem
wazuh-user@192.168.50.9's password:
splunk-ca.pem
student@serv-22-50-9:/tmp$
```

Рис. 13.16. Копіювання кореневого сертифікату CA Splunk (CN=SplunkCommonCA) на сервер Serv-22-50-7

Залишаємо Splunk-сервер та переходимо у SSH-сесію на Logstash-сервері, він же Wazuh Appliance, він же Serv-G-N-7 (192.168.50.9) ☺. Створюємо каталог та переміщуємо до нього завантажений сертифікат:

```
sudo mkdir -p /etc/logstash/certs
sudo mv /tmp/splunk-ca.pem /etc/logstash/certs/splunk-ca.pem
sudo chmod 644 /etc/logstash/certs/splunk-ca.pem
```

Перевіряємо місце розташування truststore та імпортуємо Splunk CA у Java truststore Logstash

```
sudo find /usr/share/logstash -name cacerts
```

```
wazuh-user@serv-22-50-7:~$ ls /tmp
hyperfdata_logstash
hyperfdata_root
splunk-ca.pem
systemd-private-e9b31bd8241840f8be68edda1ff0c334-chrond.service-QL8kCc
systemd-private-e9b31bd8241840f8be68edda1ff0c334-chrond.service-QL8kCc
[wazuh-user@serv-22-50-7 ~]$ sudo mkdir -p /etc/logstash/certs
[wazuh-user@serv-22-50-7 ~]$ sudo mv /tmp/splunk-ca.pem /etc/logstash/certs/splunk-ca.pem
[wazuh-user@serv-22-50-7 ~]$ sudo chmod 644 /etc/logstash/certs/splunk-ca.pem
[wazuh-user@serv-22-50-7 ~]$ sudo find /usr/share/logstash -name cacerts
/usr/share/logstash/jdk/lib/security/cacerts
[wazuh-user@serv-22-50-7 ~]$ sudo /usr/share/logstash/jdk/bin/keytool \
  -importcert \
  -alias splunk-ca \
  -file /etc/logstash/certs/splunk-ca.pem \
  -keystore /usr/share/logstash/jdk/lib/security/cacerts \
  -storepass changeit \
  -noprompt
Warning: use -cacerts option to access cacerts keystore
Certificate was added to keystore
[wazuh-user@serv-22-50-7 ~]$
```

Рис. 13.17. Імпорт кореневого сертифікату CA Splunk на сервері Wazuh/Logstash

Імпортуємо CA:

```
sudo /usr/share/logstash/jdk/bin/keytool \
  -importcert \
  -alias splunk-ca \
  -file /etc/logstash/certs/splunk-ca.pem \
  -keystore /usr/share/logstash/jdk/lib/security/cacerts \
  -storepass changeit \
  -noprompt
```

Logstash довіряє Splunk.



Запуск / перезапуск служб Wazuh (Manager + Dashboard + Indexer)

Перевірка стану всіх сдужб Wazuh

```
sudo systemctl status wazuh-manager --no-pager -l  
sudo systemctl status wazuh-dashboard --no-pager -l  
sudo systemctl status wazuh-indexer --no-pager -l
```

Коректний перезапуск (рекомендований порядок) Indexer (OpenSearch) – Manager – Dashboard:

```
sudo systemctl restart wazuh-indexer  
sudo systemctl restart wazuh-manager  
sudo systemctl restart wazuh-dashboard
```

Якщо Wazuh Manager не стартує (timeout / зависання), зупиняємо Manager, перевіряємо що залишилось “живим” та запускаємо Manager заново

```
sudo systemctl stop wazuh-manager  
ps aux | grep wazuh | head -n 50  
sudo systemctl start wazuh-manager  
sudo systemctl status wazuh-manager --no-pager -l
```

Діагностуємо помилку запуску Manager. Якщо старт з помилкою:

```
sudo journalctl -xeu wazuh-manager.service --no-pager | tail -n 80
```

А також основний лог Wazuh:

```
sudo tail -n 80 /var/ossec/logs/ossec.log
```

Якщо нічого не допомагає, виконуємо повний “reset-перезапуск” всього Wazuh (жорсткий сценарій)

```
sudo systemctl stop wazuh-dashboard  
sudo systemctl stop wazuh-manager  
sudo systemctl stop wazuh-indexer  
sudo systemctl start wazuh-indexer  
sudo systemctl start wazuh-manager  
sudo systemctl start wazuh-dashboard
```

Корисні посилання

- Wazuh. Splunk integration.
<https://documentation.wazuh.com/current/integrations-guide/splunk/index.html>
- Integrating Wazuh and Splunk
<https://www.linkedin.com/pulse/integrating-wazuh-splunk-vaibhav-karayath-p2gnc>
- Integrations guide: Elastic, OpenSearch, Splunk, Amazon Security Lake
<https://documentation.wazuh.com/current/integrations-guide/index.html>
- wazuh / wazuh-splunk
<https://github.com/wazuh/wazuh-splunk>
- IBM Storage Defender. Splunk® Configuration Guide
https://www.ibm.com/docs/en/SSDR5G6_prod/pdf/splunk_config_guide.pdf