

Systemd i narzędzie journalctl



1 Czym jest systemd

systemd to nowoczesny **system init** i **menedżer usług** w systemach Linux.

Zastąpił stare mechanizmy (`SysVinit`, `rc.local`, `upstart`) i pełni kluczową rolę w:

- uruchamianiu procesów systemowych podczas startu systemu,
- monitorowaniu usług (`systemctl`),
- zbieraniu logów systemowych (`journald`),
- zarządzaniu sesjami użytkowników (`logind`),
- kontrolowaniu montowania, sieci, zasilania itd.

Dzięki `systemd`, Linux ma **centralny system kontroli i rejestracji zdarzeń** – co jest bardzo istotne z punktu widzenia **bezpieczeństwa, audytu i reagowania na incydenty**.

2 systemd-journald i narzędzie journalctl

`systemd-journald` to komponent `systemd`, który:









- zbiera logi z jądra (`kernel`),
- odbiera komunikaty z usług (`sshd`, `sudo`, `login`, `polkit`, itp.),
- zapisuje logi w formacie binarnym (z metadanymi: PID, UID, UNIT, IP, itp.),
- umożliwia ich odczyt i filtrowanie przez narzędzie `journalctl`.

3 Dlaczego to ważne w cyberbezpieczeństwie

Z perspektywy bezpieczeństwa, `journalctl` to **główne źródło dowodów i informacji o aktywności systemu**:

- wykrywanie prób logowania nieudanych / nieautoryzowanych (`sshd`, `su`, `sudo`),
- śledzenie eskalacji uprawnień,
- analiza aktywności root'a,
- wykrywanie błędów usług (np. `crash`, `exploit attempt`),
- analiza incydentów po włamaniu.









Typowe polecenia journalctl dla analizy bezpieczeństwa

Cel	Polecenie	Opis
 Wszystkie logi bezpieczeństwa (ostatnie 24h)	<code>sudo journalctl -p err..alert --since "24 hours ago"</code>	Filtruje błędy i alerty z ostatniej doby
 Próby logowania przez SSH	<code>sudo journalctl -u ssh -p info --since "7 days ago"</code>	Logi z usługi SSH (np. Failed password, Invalid user)
 Nieudane logowania (różne źródła)	<code>sudo journalctl egrep -i "Failed password Invalid user authentication failure permission denied"</code>	egrep Filtruje po błędnych wpisach
 Akcje użytkownika root	<code>sudo journalctl _UID=0 --since yesterday</code>	Wszystko, co wykonywał użytkownik root
 Eskalacja uprawnień (sudo, su)	<code>sudo journalctl -t sudo -t su --since today</code>	Analiza użycia poleceń administracyjnych
 Błędy PAM i autoryzacji	<code>sudo journalctl -t pam_unix -p err --since today</code>	Sprawdza błędy modułu PAM
 Próby połączeń SSH z zewnątrz	<code>sudo journalctl -u ssh grep -i " from "</code>	pokazuje wpisy usługi SSH zawierające słowo „from”
 Historia z danego bootu (aktualnej sesji)	<code>sudo journalctl -b</code>	Logi od ostatniego uruchomienia systemu
 Zdarzenia jądra (kernel)	<code>sudo journalctl -k -p warning..alert</code>	Ostrzeżenia i błędy systemowe (np. sterowniki, kernel panic)

Najczęściej używane przełączniki journalctl

Przełącznik	Znaczenie	Przykład użycia	Zastosowanie (cyberbezpieczeństwo)
-u <unit>	Filtruje logi konkretnej usługi systemd (np. <code>ssh</code> , <code>sudo/su</code> , <code>pam_unix</code> , ...)	<code>journalctl -u ssh</code>	Analiza logowań i błędów SSH
-t <tag>	Filtruje po tagu (np. <code>sudo</code> , <code>su</code> , <code>pam_unix</code>)	<code>journalctl -t sudo</code>	Śledzenie użycia poleceń sudo
--since "<czas>"	Początek zakresu czasowego	<code>journalctl --since "2025-10-10 12:00"</code>	Analiza logów od konkretnej daty/godziny
--until "<czas>"	Koniec zakresu czasowego	<code>journalctl --until "2025-10-16"</code>	Ograniczenie przedziału czasowego
-b	Logi z aktualnego uruchomienia systemu (boot)	<code>journalctl -b</code>	Analiza zdarzeń od ostatniego restartu
-k	Logi jądra (kernel)	<code>journalctl -k -p warning</code>	Wykrywanie błędów sprzętu, kernela
-p <priorytet>	Filtr wg poziomu ważności (<code>emerg</code> , <code>alert</code> , <code>crit</code> , <code>err</code> , <code>warning</code> , <code>notice</code> , <code>info</code> , <code>debug</code>)	<code>journalctl -p err..alert</code>	Pokazuje tylko błędy i ostrzeżenia
-f	„Follow” – wyświetla logi na żywo (jak <code>tail -f</code>)	<code>journalctl -u ssh -f</code>	Monitorowanie prób logowania w czasie rzeczywistym
--no-pager	Wyłącza przeglądarkę (<code>less</code>), wypisuje bez zatrzymywania	<code>journalctl -u ssh --no-pager</code>	Do eksportu lub skryptów
--output <format> lub -o	Określa format wyjścia (<code>short</code> , <code>json</code> , <code>json-pretty</code> , <code>cat</code>)	<code>journalctl -o json-pretty</code>	Eksport logów do analizy lub SIEM
--disk-usage	Pokazuje ile miejsca zajmują logi	<code>journalctl --disk-usage</code>	Kontrola zużycia przestrzeni przez <code>journald</code>
--vacuum-time=<czas>	Usuwa logi starsze niż podany czas	<code>sudo journalctl --vacuum-time=14d</code>	Rotacja logów – ograniczenie retencji
_UID=<id>	Filtruje po ID użytkownika	<code>journalctl _UID=0</code>	Śledzenie działań użytkownika root
_PID=<id>	Filtruje logi konkretnego procesu	<code>journalctl _PID=1234</code>	Analiza zachowania jednego procesu
_COMM=<nazwa>	Filtruje po nazwie procesu	<code>journalctl _COMM=sshd</code>	Dokładniejsze filtrowanie po nazwie binarki
--grep "<regex>"	Wbudowany filtr regex (od systemd 239+)	<code>journalctl --grep "Failed password"</code>	Szybsze wyszukiwanie wzorców bez <code>grep</code>
-n <liczba>	Pokazuje ostatnie N wpisów	<code>journalctl -n 50</code>	Szybki podgląd ostatnich zdarzeń
--list-boots	Lista wszystkich uruchomień systemu	<code>journalctl --list-boots</code>	Audyt restartów i incydentów systemowych
-xe	Tryb szczegółowy z kontekstem błędów	<code>journalctl -xe</code>	Diagnostyka przy awariach lub błędach usług

Przykłady poleceń journalctl

Cel	Polecenie
 Wykrycie nieudanych logowań	<code>sudo journalctl -u ssh --since "24 hours ago" --grep "Failed password"</code>
 Analiza użycia sudo	<code>sudo journalctl -t sudo --since today</code>
 Próby logowania na root	<code>sudo journalctl -u ssh --grep "for root"</code>
 Błędy bezpieczeństwa (PAM, auth)	<code>sudo journalctl -p err -t pam_unix</code>
 Logi z danego okresu	<code>sudo journalctl --since "2025-10-14" --until "2025-10-16"</code>
 Tylko błędy z całego systemu	<code>sudo journalctl -p err..alert --no-pager</code>
 Logi z poprzedniego uruchomienia	<code>sudo journalctl -b -1</code>
 Eksport pełnych logów do pliku	<code>sudo journalctl --since "2025-10-01" > system_logs.txt</code>

Przykładowy skrypt

```
OUTPUT="security_report.csv"
DAYS=7
echo "Typ błędu,Data,Użytkownik,Adres IP" > "$OUTPUT"

# Pobierz logi z ostatnich X dni (pełny dostęp wywołując skrypt z sudo)
journalctl --since "$DAYS days ago" -t sshd -t su --no-pager |
while read -r line; do
    # Failed password
    # Sep 16 06:45:45 skrypty-lab sshd[33081]: Failed password for invalid user vaagrant from
    # zapis ${line:0:15} powoduje pobranie pierwszych 15 znaków z linii (czyli tutaj data i godzina)
    # składnia ${variable:offset:length}, offset to pozycja początkowa (licząc od 0), length to długość
    if [[ "$line" =~ Failed\ password\ for\ (invalid\ user\ )?([a-zA-Z0-9._-]+)\ from\ ([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+) ]]; then
        echo "Failed password,${line:0:15},${BASH_REMATCH[2]},${BASH_REMATCH[3]}" >> "$OUTPUT"

    # Invalid user
    # Sep 16 06:45:45 skrypty-lab sshd[33081]: Invalid user vaagrant from
    elif [[ "$line" =~ Invalid\ user\ ([a-zA-Z0-9._-]+)\ from\ ([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+) ]]; then
        echo "Invalid user,${line:0:15},${BASH_REMATCH[1]},${BASH_REMATCH[2]}" >> "$OUTPUT"

    # Próby logowania na root
    # Sep 16 06:45:45 skrypty-lab sshd[33081]: Failed password for root from
    elif [[ "$line" =~ Failed\ password\ for\ root\ from\ ([0-9]+\.[0-9]+\.[0-9]+\.[0-9]+) ]]; then
        echo "Failed password for root,${line:0:15},root,${BASH_REMATCH[1]}" >> "$OUTPUT"

    # Użycie sudo
    # Oct 16 12:09:29 shell-lab sudo:      root : TTY=pts/3 ; PWD=/root ; USER=root ; COMMAND=/bin/ls
    elif [[ "$line" =~ sudo: ]]; then
        echo "sudo usage,${line:0:15},-, -" >> "$OUTPUT"

    # Próby su
    # Oct 16 11:40:53 shell-lab su[3246]: pam_unix(su-1:auth): authentication failure; logname=vagrant uid=1000 euid=0 tty=/dev/pts/3 ruser=vagrant rhost= user=root
    elif [[ "$line" =~ pam_unix.*authentication\ failure.*user=([a-zA-Z0-9._-]+) ]]; then
        echo "pam_unix auth failure,${line:0:15},${BASH_REMATCH[1]},-" >> "$OUTPUT"
    fi
done
```