Poznajemy Grayloga i instalacja

Opowieść, instalacja w Dockerze, logowanie kontem z AD.

- Graylog: jedno okno na logi z całej firmy
- Instalacja Grayloga w Dockerze: Graylog, MongoDB i OpenSearch
- Graylog: logowanie kontem z Active Directory

Graylog: jedno okno na logi z całej firmy

Poziom: podstawy **Dotyczy: Graylog 7.1** **Czas: ~15 min**

Ta strona otwiera książkę o Graylogu: skąd wziął się pomysł zbierania logów w jednym miejscu, dlaczego akurat Graylog i co czeka na kolejnych stronach.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
RFC	Request for Comments: oficjalny dokument opisujący standard internetowy, na przykład sposób działania danego protokołu; w ITIL także Request for Change, czyli wniosek o zmianę
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera

Skąd się wzięło

> **Pojęcie: syslog.** Standardowy sposób, w jaki systemy uniksowe i urządzenia sieciowe zapisują i przesyłają dalej komunikaty o zdarzeniach: kto się zalogował, jaki proces się uruchomił, którą regułę firewalla zastosowano. Syslog nie jest programem, tylko formatem i protokołem, więc rozumie go niemal każdy system i każde urządzenie sieciowe od czterdziestu lat.

Napisał go Eric Allman na początku lat osiemdziesiątych na Uniwersytecie Kalifornijskim w Berkeley, jako część swojego programu Sendmail do obsługi poczty elektronicznej. Sendmail potrzebował sposobu na zapisywanie własnych komunikatów o błędach, a Allman zaprojektował go na tyle uniwersalnie, że inne programy zaczęły z niego korzystać przy okazji. Format okazał się na tyle użyteczny, że stał się nieformalnym standardem całego Uniksa, a później doczekał się oficjalnej specyfikacji w RFC 3164, a następnie w nowszym, bardziej ustrukturyzowanym RFC 5424.

Syslog rozwiązuje jednak tylko połowę problemu: mówi, jak log ma wyglądać i jak go wysłać, ale nie mówi, gdzie go szukać ani jak go przeszukiwać. Administrator dziesięciu czy stu serwerów, z których każdy pisze własny plik `/var/log/messages`, wciąż musi wiedzieć, na który serwer się

zalogować, zanim zada jakiekolwiek pytanie. Tę drugą połowę, przeszukiwalny magazyn i wygodny interfejs nad surowym strumieniem syslogowym, dopisała dopiero kolejna generacja narzędzi.

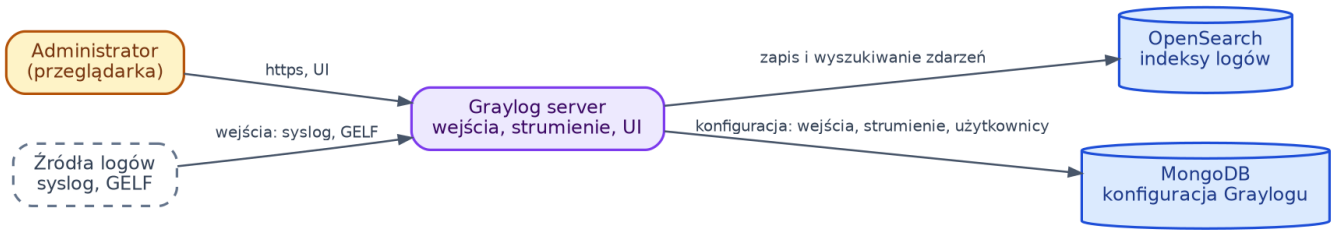
> **Pojęcie: Graylog.** Serwer centralnego logowania z własnym interfejsem webowym: przyjmuje zdarzenia z wielu źródeł naraz, indeksuje je w silniku wyszukiwania i pozwala budować zapytania, pulpity i alerty bez logowania się na żaden z monitorowanych serwerów. Graylog nie zastępuje syslogu, tylko dokłada mu drugą połowę: miejsce, w którym wszystkie te komunikaty w końcu się spotykają.

Graylog napisał Lennart Koopmann, programista z Hamburga, który zaczął projekt jako prywatną inicjatywę pod nazwą Torch około 2009 i 2010 roku, pracując wtedy w serwisie społecznościowym XING. Projekt szybko przerósł status hobby: w 2012 roku powstała spółka, w 2013 roku produkt przyjął nazwę Graylog, a w 2014 roku firma przeniosła się do Houston w Teksasie, gdzie działa do dziś jako Graylog, Inc. Od 2016 roku obok darmowej wersji Open istnieje płatna wersja Enterprise z dodatkowymi funkcjami, do których jeszcze wrócimy.

Silnikiem wyszukiwania pod spodem był przez lata Elasticsearch. W styczniu 2021 roku firma Elastic zmieniła licencję Elasticsearch na taką, która przestała być wolnym oprogramowaniem w rozumieniu społeczności open source, więc Amazon odpowiedział własnym forkiem sprzed zmiany licencji, nazwanym OpenSearch. Graylog poszedł tą samą drogą: od wersji 5.0 z 2022 roku OpenSearch jest domyślnym i docelowym silnikiem, a stary Elasticsearch traktowany jest jako ścieżka do migracji, nie jako cel sam w sobie.

Jak to działa w jednym obrazku

Graylog sam w sobie nie przechowuje logów. To warstwa aplikacyjna, która przyjmuje zdarzenia na wejściach, przepuszcza je przez reguły klasyfikujące, a samo przechowywanie i przeszukiwanie oddaje silnikowi OpenSearch. Własną konfigurację, czyli kto ma jakie konto i jak nazywają się wejścia oraz reguły, trzyma osobno, w bazie MongoDB.



Graylog przyjmuje logi ze źródeł, zapisuje i przeszukuje je w OpenSearch, a swoją konfigurację trzyma w MongoDB (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> Piątek, 15:40. Ktoś skasował plik `cennik_2026_Q4.xlsx` z udziału `\\fs01\Ksiegowosc`, a nikt nie przyznaje się do winy. Tomek, drugi tydzień w pracy, dostaje zadanie „sprawdź w logach, kto to zrobił". Zanim w firmie stanął Graylog, oznaczało to piętnaście osobnych sesji SSH i piętnaście różnych `grep` po plikach `/var/log`, z których część i tak już zdążyła się zrotować i zniknąć.

Zespół IT Zielonej długo radził sobie z logami tak, jak radzi sobie większość małych firm: każdy serwer trzymał swoje zdarzenia lokalnie, a przeszukanie czegokolwiek ponad jedną maszynę oznaczało ręczne kopiowanie plików albo, gorzej, logowanie się po kolei wszędzie tam, gdzie problem mógł zostawić ślad. Przy piętnastu serwerach produkcyjnych i dwóch administratorach ta metoda przestała się skalować na długo przed tym, zanim ktokolwiek zdążył to świadomie zauważyć.

Graylog zmienia to jednym mechanizmem: każdy serwer i każda aplikacja, opisane osobno na stronie Wysyłka logów z hostów, wysyłają swoje zdarzenia od razu, w chwili ich powstania, do jednego centralnego miejsca. Tomek zamiast piętnastu sesji SSH wpisuje jedno zapytanie w przeglądarce i w kilka sekund ma odpowiedź, z którego konta i o której godzinie plik zniknął, niezależnie od tego, na którym z piętnastu serwerów zdarzenie faktycznie powstało.

Mapa tej książki

Książka o Graylogu składa się z dziesięciu stron: od instalacji, przez codzienne wyszukiwanie, po kopię zapasową i aktualizację.

Strona	Na jakie pytanie odpowiada
Graylog: jedno okno na logi z całej firmy	Skąd wziął się Graylog i po co go mamy
Instalacja Grayloga w Dockerze: Graylog, MongoDB i OpenSearch	Jak postawić trzy kontenery i połączyć je w jeden działający system
Graylog: logowanie kontem z Active Directory	Jak podłączyć logowanie przez LDAP i co z tego jest darmowe, a co płatne
Po co centralne logowanie i jak to działa	Jaki problem w ogóle rozwiązuje centralne logowanie, niezależnie od narzędzia
Graylog: wejścia, strumienie, wyszukiwanie	Jak Graylog klasyfikuje i przeszukuje przychodzące zdarzenia

Strona	Na jakie pytanie odpowiada
Wysyłka logów z hostów	Jak skonfigurować rsyslog, journald i Windows, żeby logi w ogóle dotarły
Kopia zapasowa i aktualizacja Grayloga	Jak bezpiecznie zrobić backup trzech baz naraz i podnieść wersję
Graylog: codzienna praca z logami	Jak szukać, dzielić logi na strumienie z retencją, budować panele i alerty z powiadomieniem e-mail oraz reguły potoku
Graylog: przeprowadzka z docker01 na log01	Jak przenieść Grayloga na własny serwer bez zmian na hostach i z planem powrotu
Graylog: częste problemy	Co robić, gdy brakuje logów, czas jest przesunięty, dysk OpenSearch się zapełnił albo pola zniknęły

Powiązane: Po co centralne logowanie i jak to działa · Architektura Zabbixa: serwer, proxy, agent, frontend

Źródła: en.wikipedia.org/wiki/Graylog · graylog.org, sekcja „About" i historia firmy · RFC 5424 (The Syslog Protocol) · go2docs.graylog.org, „Upgrading to OpenSearch"

Instalacja Grayloga w Dockerze: Graylog, MongoDB i OpenSearch

Typ: procedura

Dotyczy: Graylog 7.1.9, MongoDB 8.0, OpenSearch 2.19.6 (sprawdzone na go2docs.graylog.org i hub.docker.com 28.09.2026)

Czas: ~40 min

Efekt: trzy działające kontenery, Graylog, MongoDB i OpenSearch, połączone w jeden system dostępny po WWW pod `logi.zielona.example`, gotowy na pierwsze wejścia i pierwsze logowanie.

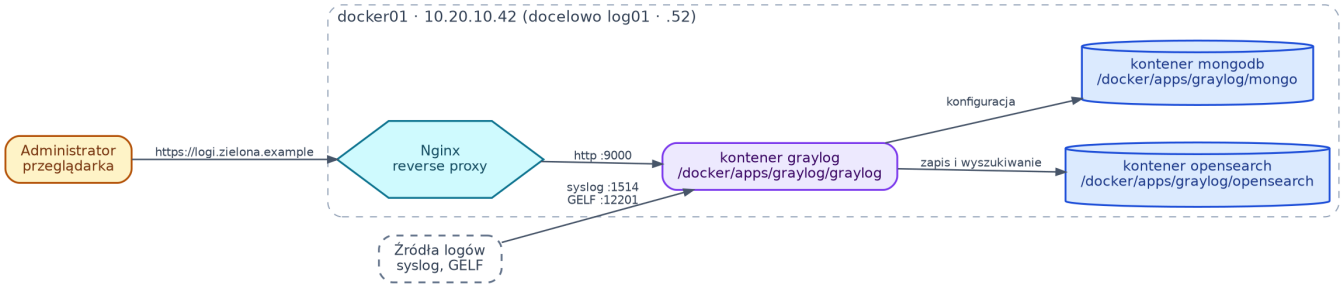
Skróty użyte na tej stronie (6)

Skrót	Co znaczy
API	Application Programming Interface: sposób, w jaki programy rozmawiają ze sobą
CLI	Command Line Interface: praca przez wpisywanie komend w terminalu
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
SHA256	funkcja skrótu kryptograficznego dająca 256-bitowy odcisk danych
WebSocket	stałe, dwukierunkowe połączenie przeglądarki z serwerem, przez które Home Assistant na żywo odświeża panele; serwer pośredniczący musi przekazywać nagłówki Upgrade i Connection
WWW	World Wide Web: strony i aplikacje otwierane w przeglądarce po HTTP albo HTTPS

Zanim zaczniemy

Komendy wykonujemy z CLI bash jako user z uprawnieniami sudo albo root, na maszynie z zainstalowanym Dockerem i wtyczką Compose. Graylog sam w sobie to tylko jeden z trzech kontenerów: potrzebuje MongoDB na konfigurację i OpenSearch na same logi, opisane szerzej na stronie Graylog: wejścia, strumienie, wyszukiwanie.

> W naszym środowisku dane aplikacji trzymamy w `/docker/apps/<nazwa>`, a plik Compose w `/docker/compose/<nazwa>/compose.yaml`. To konwencja, nie wymóg Dockera: można ją zmienić, byle w całej firmie była jedna. Katalog `/docker` to osobny wolumen `vgDocker/lvDocker`, żeby pełny dysk aplikacji nie zatrzymał systemu.



Graylog, MongoDB i OpenSearch jako trzy kontenery na docker01, za Nginksem, ze źródłami logów wchodzącymi na porty 1514 i 12201 (kliknięcie otwiera pełny rozmiar)

Wymaganie	Uwaga
Docker Engine i wtyczka Compose	sprawdzamy `docker compose version`
`vm.max_map_count` od `262144` w górę	wymóg OpenSearch, ustawiamy w kroku 1
Wolny port `9000` na hoście	interfejs i API Graylogu, zwykle za Nginksem, nie wystawiany wprost na świat
Porty `1514` i `12201` wolne, tylko jeśli faktycznie ich użyjemy	pierwsze wejście syslog i GELF, patrz Graylog: wejścia, strumienie, wyszukiwanie
Wpis DNS `logi.zielona.example` na `docker01`	patrz Świat przykładów: Zielona po adres
Certyfikat dla `logi.zielona.example`	z CA Zielonej albo Let's Encrypt, do Nginksa w kroku 5

Uważaj: obrazy zawsze przypinamy do konkretnej wersji, nigdy do `latest`. OpenSearch dodatkowo ma górną granicę: Graylog 7.1.x wspiera OpenSearch tylko do wersji 2.19.x, a wersje 3.0 i nowsze nie są przez niego obsługiwane, więc podmiana tagu na `latest` przy aktualizacji potrafi po cichu wystawić kombinację, której Graylog w ogóle nie rozumie.

> Dziś Graylog stoi na `docker01` razem z Gitea i resztą aplikacji Zielonej. Docelowo przenosimy go na dedykowany `log01` (10.20.10.52): przenosiny to kopia katalogów `/docker/apps/graylog` i `/docker/compose/graylog` na nowy host, zmiana wpisu DNS `logi.zielona.example` i przepięcie Nginksa, nie osobna instalacja od zera.

Przebieg

1. Ustawienie vm.max_map_count na hoście

OpenSearch odmawia startu, jeśli system nie pozwala procesowi zmapować odpowiednio dużej liczby obszarów pamięci. Ustawiamy to trwale, żeby przetrwało restart hosta:

```
sudo tee /etc/sysctl.d/99-opensearch.conf <<< "vm.max_map_count=262144"
sudo sysctl --system
```

Przykład:

```
* Applying /etc/sysctl.d/99-opensearch.conf ...
vm.max_map_count = 262144
```

2. Katalogi na dane, zgodnie z konwencją ścieżek

```
sudo mkdir -p /docker/apps/graylog/mongo /docker/apps/graylog/opensearch
/docker/apps/graylog/graylog /docker/apps/graylog/snapshots
sudo mkdir -p /docker/compose/graylog
```

Kontenery OpenSearch i Grayloga działają jako zwykli użytkownicy, nie jako root, więc katalog założony przez `mkdir` (właściciel root) blokuje im zapis i kontener nie wstaje. Numery użytkowników sprawdzamy w obrazach, nie przepisujemy z przykładu:

```
docker run --rm --entrypoint id opensearchproject/opensearch:2.19.6
docker run --rm --entrypoint id graylog/graylog:7.1.9
```

```
sudo chown -R NumerUID:NumerGID /docker/apps/graylog/opensearch /docker/apps/graylog/snapshots
sudo chown -R NumerUID:NumerGID /docker/apps/graylog/graylog
```

> (gdzie `NumerUID:NumerGID` to numery z wyniku poprzedniej komendy, osobne dla OpenSearch i osobne dla Grayloga)

MongoDB nie wymaga tego kroku: obraz sam nadaje właściciela katalogowi `/data/db`. Pełny opis z przykładowym wynikiem jest na stronie Graylog: przeprowadzka z docker01 na log01.

3. Wygenerowanie sekretów Grayloga

> **Pojęcie: GRAYLOG_PASSWORD_SECRET.** Losowy ciąg znaków, którym Graylog szyfruje hasła i sesje przechowywane w MongoDB. Musi mieć przynajmniej 16 znaków, w praktyce generujemy dłuższy i nie zmieniamy go już później, bo zmiana unieważnia wszystkie zapisane hasła.

```
< /dev/urandom tr -dc 'A-Za-z0-9' | head -c96; echo
```

Przykład:

```
qT8fRZmN2vLxKp...(96 znaków)
```

> **Pojęcie: GRAYLOG_ROOT_PASSWORD_SHA2.** Skrót SHA256 hasła konta `admin`, jedynego konta zapisanego wprost w konfiguracji, a nie w bazie. Graylog nigdy nie przechowuje tego hasła jawnym tekstem, więc wpisujemy tu wynik komendy, nie samo hasło:

```
echo -n 'HasloAdministratora' | sha256sum | cut -d' ' -f1
```

Przykład:

```
8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a91
```

Oba wyniki wklejamy do pliku `/docker/compose/graylog/.env`, obok pliku `compose.yaml`, nie w nim samym:

```
GRAYLOG_PASSWORD_SECRET=qT8fRZmN2vLxKp...(96 znaków)
GRAYLOG_ROOT_PASSWORD_SHA2=8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a91
```

4. Plik compose.yaml

Plik `/docker/compose/graylog/compose.yaml`:

```
services:
  mongodb:
    image: mongo:8.0
    container_name: graylog-mongodb
    restart: always
    volumes:
      - /docker/apps/graylog/mongo:/data/db

  opensearch:
    image: opensearchproject/opensearch:2.19.6
    container_name: graylog-opensearch
    restart: always
    environment:
      - discovery.type=single-node
      - plugins.security.disabled=true
      - bootstrap.memory_lock=true
      - OPENSEARCH_JAVA_OPTS=-Xms1g -Xmx1g
      - path.repo=/usr/share/opensearch/snapshots
    ulimits:
      memlock:
        soft: -1
        hard: -1
      nofile:
        soft: 65536
        hard: 65536
    volumes:
      - /docker/apps/graylog/opensearch:/usr/share/opensearch/data
      - /docker/apps/graylog/snapshots:/usr/share/opensearch/snapshots

  graylog:
    image: graylog/graylog:7.1.9
    container_name: graylog
    restart: always
    depends_on:
      - mongodb
      - opensearch
    environment:
      - GRAYLOG_PASSWORD_SECRET=${GRAYLOG_PASSWORD_SECRET}
```

```
      - GRAYLOG_ROOT_PASSWORD_SHA2=${GRAYLOG_ROOT_PASSWORD_SHA2}
      - GRAYLOG_HTTP_BIND_ADDRESS=0.0.0.0:9000
      - GRAYLOG_HTTP_EXTERNAL_URI=https://logi.zielona.example/
      - GRAYLOG_ELASTICSEARCH_HOSTS=http://opensearch:9200
      - GRAYLOG_MONGODB_URI=mongodb://mongodb:27017/graylog
    volumes:
      - /docker/apps/graylog/graylog:/usr/share/graylog/data
    ports:
      - "9000:9000/tcp"
      - "1514:1514/tcp"
      - "1514:1514/udp"
      - "12201:12201/tcp"
      - "12201:12201/udp"
```

Ustawienie `path.repo` i wolumen `snapshots` przygotowują miejsce na migawki OpenSearch. Bez nich późniejsza rejestracja repozytorium kopii (opisana na stronie Kopia zapasowa i aktualizacja Grayloga) zwróci błąd, a dodanie ich po fakcie wymaga odtworzenia kontenera. Jeśli migawek nie planujemy, możemy te dwie linie pominąć.

Port `9200` OpenSearch celowo nie trafia na listę `ports`: Graylog rozmawia z nim po wewnętrznej sieci Dockera, a `plugins.security.disabled=true` wyłącza wbudowane uwierzytelnianie tego portu, więc nie wystawiamy go na hosta, tym bardziej na świat. Porty `1514` i `12201` zostawiamy zmapowane tylko wtedy, gdy rzeczywiście zamierzamy z nich korzystać: dokładamy albo usuwamy pojedynczą parę linii, zależnie od tego, które wejście skonfigurujemy w interfejsie Graylogu.

5. Pierwsze uruchomienie

```
cd /docker/compose/graylog
docker compose up -d
```

Przykład:

```
[+] Running 4/4
 ✔ Network graylog_default       Created
 ✔ Container graylog-mongodb     Started
 ✔ Container graylog-opensearch  Started
 ✔ Container graylog             Started
```

Pierwszy start trwa dłużej niż zwykły restart: Graylog czeka, aż OpenSearch faktycznie odpowie, i dopiero potem zakłada własne indeksy. Sprawdzamy log, dopóki nie zobaczymy potwierdzenia gotowości:

```
docker compose logs -f graylog
```

Przykład:

```
graylog  | 2026-09-28T10:02:41.331Z INFO [InputSetupService] Triggering launching persisted
inputs
graylog  | 2026-09-28T10:02:41.402Z INFO [ServerBootstrap] Graylog server up and running.
```

6. Reverse proxy: Nginx przed Graylogiem

Fragment vhostu `/etc/nginx/conf.d/logi.zielona.example.conf`, z nagłówkami wymaganymi przez WebSocket, bo interfejs Graylogu odświeża wyniki wyszukiwania w czasie rzeczywistym właśnie po WebSocket, opisanym na stronie Reverse proxy i równoważenie obciążenia:

```
server {
    listen 443 ssl;
    server_name logi.zielona.example;

    ssl_certificate     /etc/nginx/certy/logi.zielona.example.crt;
    ssl_certificate_key /etc/nginx/certy/logi.zielona.example.key;

    location / {
        proxy_pass http://10.20.10.42:9000;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }
}
```

Weryfikacja

Sprawdzamy, że wszystkie trzy kontenery działają:

```
docker compose ps
```

Przykład:

```
NAME                IMAGE                                    STATUS
graylog             graylog/graylog:7.1.9                    Up 3 minutes
graylog-mongodb     mongo:8.0                                 Up 3 minutes
graylog-opensearch  opensearchproject/opensearch:2.19.6      Up 3 minutes
```

Sprawdzamy stan zdrowia Grayloga przez wbudowany punkt kontrolny API, ten sam, którego używa load balancer:

```
curl -s http://localhost:9000/api/system/lbstatus
```

Przykład:

```
ALIVE
```

◫ ZRZUT EKRANU DO UZUPEŁNIENIA

Zrzut do uzupełnienia: pierwszy ekran logowania pod `https://logi.zielona.example`, z widocznym polem „Username" i „Password" oraz numerem wersji Graylogu w stopce.

Wycofanie zmian

Zatrzymanie i usunięcie kontenerów, bez kasowania danych z wolumenów:

```
docker compose down
```

Pełne usunięcie razem z danymi, tylko gdy naprawdę chcemy zacząć od zera:

```
docker compose down -v
sudo rm -rf /docker/apps/graylog
```

Gdy coś nie zadziała

Objaw	Przyczyna	Naprawa
Kontener `graylog-opensearch` restartuje się w pętli, w logu `max virtual memory areas vm.max_map_count too low`	krok 1 pominięty albo nie przetrwał restartu hosta	`sysctl vm.max_map_count`, w razie potrzeby powtarzamy krok 1

Objaw	Przyczyna	Naprawa
Kontener `graylog` bez końca czeka, w logu brak `Graylog server up and running`	OpenSearch jeszcze nie odpowiada albo `GRAYLOG_ELASTICSEARCH_HOSTS` wskazuje zły adres	`docker compose logs opensearch`, sprawdzamy nazwę usługi w `compose.yaml`
`curl http://localhost:9000/api/system/lbstatus` zwraca błąd połączenia	kontener `graylog` jeszcze się uruchamia albo port `9000` zajęty przez inną usługę	`docker compose logs graylog`, `ss -tlnp \`
Logowanie kontem `admin` odrzucone mimo poprawnego hasła	`GRAYLOG_ROOT_PASSWORD_SHA2` policzony z innego hasła niż wpisywane w formularzu	liczymy skrót ponownie krokiem 3, poprawiamy `.env`, `docker compose up -d`
Strona `https://logi.zielona.example` wisi przy ładowaniu wyników wyszukiwania	brak nagłówków `Upgrade`/`Connection` w Nginksie	dokładamy blok WebSocket z kroku 6

Powiązane: Po co centralne logowanie i jak to działa · Graylog: wejścia, strumienie, wyszukiwanie · Konwencja Dockera w Zielonej · Reverse proxy i równoważenie obciążenia

Źródła: go2docs.graylog.org, rozdział „Docker Installation with Self-Managed OpenSearch" · go2docs.graylog.org, „Compatibility Matrix" · opensearch.org, dokumentacja instalacji Docker · endoflife.date/graylog i endoflife.date/opensearch (sprawdzone 28.09.2026)

Graylog: logowanie kontem z Active Directory

Typ: procedura **Dotyczy: Graylog 7.1, Active Directory na dc01/dc02** **Czas: ~30 min**

Efekt: logowanie do Graylogu kontem domenowym z Active Directory, bez zakładania konta lokalnego dla każdej nowej osoby. Po drodze sprawdzamy uczciwie, gdzie kończy się darmowa wersja Open, a zaczyna płatny Enterprise.

Skróty użyte na tej stronie (8)

Skrót	Co znaczy
AD	Active Directory: katalog Microsoftu z kontami użytkowników, komputerów i grup w firmie
CA	Certificate Authority: urząd certyfikacji, który podpisuje certyfikaty
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
JRE	Java Runtime Environment: środowisko uruchomieniowe Javy, potrzebne do odpalenia programów napisanych w tym języku, jak sam Graylog
JVM	Java Virtual Machine: maszyna wirtualna, w której działają programy w Javie
LDAP	Lightweight Directory Access Protocol: protokół do odpytywania katalogu użytkowników (AD, FreeIPA)
LDAPS	LDAP szyfrowany TLS, zwykle port 636
TLS	Transport Layer Security: szyfrowanie połączeń, np. w HTTPS

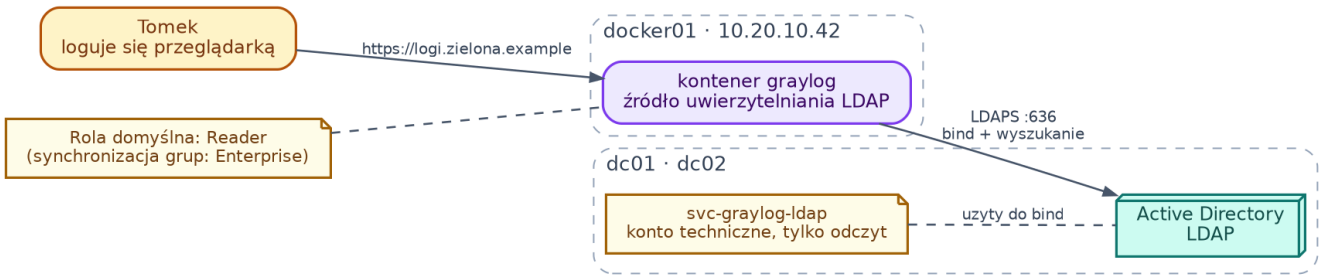
Zanim zaczniemy

Z ŻYCIA WZIĘTE

Ola dostaje dostęp do Graylogu, żeby sama sprawdzała logi swoich aplikacji zamiast pytać za każdym razem zespół IT. Loguje się tym samym kontem domenowym, którym otwiera pocztę i loguje się do komputera, bez osobnego hasła do zapamiętania i bez maila z tymczasowym hasłem, który trzeba zmienić przy pierwszym logowaniu.

Pojęcie: LDAP. Protokół odpytywania katalogu z kontami, grupami i innymi obiektami organizacyjnymi, opisany szerzej przy okazji Gitea na stronie Logowanie kontem z Active Directory. Active Directory na `dc01` i `dc02` udostępnia swoje dane właśnie przez LDAP, a Graylog, zamiast trzymać osobną bazę haseł, o każdym logowaniu pyta katalog, czy takie konto istnieje i czy podane hasło jest poprawne.

Pojęcie: usługa uwierzytelniania (authentication service). Konfiguracja w Graylogu opisująca, jak i gdzie łączyć się z katalogiem: adres serwera, poświadczenia konta technicznego do wyszukiwania, filtr określający, które konta wolno logować, i rolę, jaką dostają automatycznie po pierwszym logowaniu.



Graylog łączy się z Active Directory na dc01 po LDAPS przez konto techniczne, a każde zsynchronizowane konto dostaje tę samą rolę domyślną (kliknięcie otwiera pełny rozmiar)

Wymaganie	Uwaga
Konto techniczne w AD do bind LDAP	zakładamy w kroku 1, prawo tylko do odczytu katalogu
LDAPS na porcie 636 osiągalny z `docker01` do `dc01`/`dc02`	sprawdzamy zaporą, zanim zaczniemy konfigurację w Graylogu
Certyfikat firmowego CA zaufany przez JVM wewnątrz kontenera	bez tego kroku 4 połączenie LDAPS kończy się błędem certyfikatu
Decyzja, jaką rolę domyślną dostają zsynchronizowane konta	patrz sekcja niżej o różnicy między Open i Enterprise

Uważaj: konto techniczne wskazane przy uwierzytelnianiu systemowym powinno mieć wyłącznie prawo odczytu katalogu, na tej samej zasadzie najmniejszych uprawnień, na jakiej opieramy konta monitorujące bazy danych. Graylog nie potrzebuje żadnego prawa zapisu w Active Directory do samego uwierzytelniania użytkowników.

Co jest w Open, a co wymaga Enterprise

To rozróżnienie decyduje, jak dokładnie skonfigurujemy krok 3, więc sprawdzamy je, zanim usiądziemy do formularza. Samo uwierzytelnianie LDAP i Active Directory, czyli sprawdzenie loginu i hasła w katalogu, jest częścią darmowego Graylog Open i działa dokładnie tak, jak opisuje ta strona.

Synchronizacja grup, czyli automatyczne mapowanie grup z Active Directory na role albo zespoły (teams) wewnątrz Graylogu, jest funkcją wyłącznie Graylog Enterprise: dokumentacja producenta wprost zaznacza przy tym ekranie, że wymaga ważnej licencji Enterprise. W wersji Open każde konto zsynchronizowane z katalogu dostaje jedną, tę samą rolę domyślną skonfigurowaną w polu „Default Roles", niezależnie od tego, do jakich grup faktycznie należy w Active Directory.

W praktyce w Zielonej oznacza to, że wszyscy logujący się przez LDAP dostają w Open rolę `Reader` (obowiązkową, bez niej nikt nie otworzy interfejsu), a podniesienie kogoś do roli `Admin` wykonujemy ręcznie, z poziomu System → Users, po pierwszym zsynchronizowaniu jego konta. To jeden dodatkowy klik na osobę, nie automatyzacja przez grupę, ale wciąż mniej pracy niż zakładanie całego konta lokalnego od zera.

Przebieg

1. Konto techniczne w Active Directory

Zakładamy konto techniczne do bind LDAP, z hasłem nieulegającym wygaśnięciu i bez prawa jego zmiany przez samo konto:

```
New-ADUser -Name "svc-graylog-ldap" `
  -SamAccountName "svc-graylog-ldap" `
  -UserPrincipalName "svc-graylog-ldap@ad.zielona.example" `
  -Path "OU=KontaTechniczne,DC=ad,DC=zielona,DC=example" `
  -AccountPassword (Read-Host -AsSecureString "Haslo konta") `
  -Enabled $true `
  -PasswordNeverExpires $true `
  -CannotChangePassword $true
```

2. Zaufanie do firmowego CA w JVM

Graylog działa na maszynie wirtualnej Javy (JVM), a JVM ma własny magazyn zaufanych certyfikatów, niezależny od systemowego magazynu hosta. Certyfikat CA Zielonej musi trafić właśnie tam, inaczej połączenie LDAPS kończy się błędem certyfikatu mimo poprawnej reszty konfiguracji. Najpierw sprawdzamy, gdzie w obrazie leży JRE:

```
docker exec graylog printenv JAVA_HOME
```

Przykład:

```
/usr/lib/jvm/temurin-17-jre
```

Kopiujemy certyfikat do kontenera i dokładamy go do magazynu `cacerts` tym samym `keytool`, którego używa sam Graylog, wskazując ścieżkę wyliczoną w poprzednim kroku:

```
docker cp CA-Zielona.crt graylog:/tmp/ca-zielona.crt
docker exec graylog keytool -importcert -trustcacerts -noprompt \
  -alias ca-zielona \
  -file /tmp/ca-zielona.crt \
  -keystore "$(docker exec graylog printenv JAVA_HOME)/lib/security/cacerts" \
  -storepass changeit
docker restart graylog
```

> Po przenosinach urzędu według strony Przenosiny urzędu z dc01 bez przerwy: dwa urzędy równolegle i wygaszenie CA-Zielona dokładamy tu w ten sam sposób zaufanie do `Zielona-Root-CA`, a `CA-Zielona.crt` zostaje obok do wygaszenia starego urzędu.

Uważaj: `changeit` to domyślne hasło magazynu `cacerts` w niemal każdej dystrybucji Javy, nie hasło specyficzne dla Graylogu. Jeśli obraz zmienił je przy budowaniu, `keytool` zgłosi błąd złego hasła i sprawdzamy je w dokumentacji obrazu, zamiast zgadywać kolejne wartości.

> Import przez `keytool` do działającego kontenera zmienia plik `cacerts` wewnątrz warstwy kontenera, a nie na dysku hosta. Przy każdym odtworzeniu kontenera (`docker compose up -d` po zmianie tagu obrazu albo `docker compose down`) certyfikat znika i LDAPS znów kończy się błędem `unable to find valid certification path`. Taki import traktujemy jako szybki test, a na stałe robimy trwały magazyn zaufania z poniższego podkroku.

Trwały magazyn zaufania: truststore w katalogu danych

Przygotowujemy kopię magazynu z dopisanym CA Zielonej i zapisujemy ją w katalogu danych Grayloga, który i tak jest wolumenem na hoście (`/docker/apps/graylog/graylog`). Certyfikat `/tmp/ca-zielona.crt` jest już w kontenerze z poprzedniego polecenia.

```
docker exec graylog sh -c 'cp "$JAVA_HOME/lib/security/cacerts" /tmp/truststore.jks && keytool -importcert -trustcacerts -noprompt -alias ca-zielona -file /tmp/ca-zielona.crt -keystore /tmp/truststore.jks -storepass changeit'
docker cp graylog:/tmp/truststore.jks /docker/apps/graylog/graylog/truststore.jks
```

Plik musi należeć do użytkownika Grayloga (numery sprawdzamy tak, jak w kroku z `chown` na stronie Instalacja Grayloga w Dockerze: Graylog, MongoDB i OpenSearch):

```
sudo chown NumerUID:NumerGID /docker/apps/graylog/graylog/truststore.jks
```

W `compose.yaml`, w usłudze `graylog`, dopisujemy do `environment` zmienną, która wskazuje JVM własny magazyn zamiast domyślnego:

```
    environment:
      - GRAYLOG_SERVER_JAVA_OPTS=-Xms1g -Xmx1g -server -XX:+UseG1GC -XX:-OmitStackTraceInFastThrow -Djavax.net.ssl.trustStore=/usr/share/graylog/data/truststore.jks -Djavax.net.ssl.trustStorePassword=changeit
```

Zmienna zastępuje domyślne opcje JVM obrazu, dlatego wpisujemy w niej od razu także opcje pamięci (wartości `-Xms` i `-Xmx` dopasowujemy do swojej instalacji, jak na stronie Graylog: częste problemy). Magazyn jest kopią systemowego `cacerts` z jednym dopisanym certyfikatem, więc inne połączenia TLS Grayloga nadal działają. Po zmianie odtwarzamy kontener: `docker compose up -d graylog`. Teraz certyfikat przeżywa każde odtworzenie, bo leży na hoście. Po aktualizacji obrazu Grayloga odświeżamy kopię `cacerts`, żeby nie zostać ze starą listą zaufanych urzędów.

3. Konfiguracja usługi uwierzytelniania w interfejsie

W System → Authentication → Create service → Active Directory wypełniamy formularz zgodnie z tabelą.

Pole	Wartość w Zielonej	Uwaga
Title	`AD Zielona`	dowolna, widoczna tylko w panelu administracyjnym

Pole	Wartość w Zielonej	Uwaga
Server Address	`dc01.ad.zielona.example:636`	drugi kontroler `dc02` dopisujemy jako kolejny adres, jeśli formularz na to pozwala
Security Options	LDAPS	port 636, ruch szyfrowany od razu przy nawiązaniu połączenia
System User DN	`CN=svc-graylog-ldap,OU=KontaTechniczne,DC=ad,DC=zielona,DC=example`	konto techniczne z kroku 1
System Password	hasło konta technicznego	to samo hasło co przy `New-ADUser`
Search Base DN	`OU=Uzytkownicy,DC=ad,DC=zielona,DC=example`	jednostka organizacyjna z kontami pracowników
Search Pattern	`(&(objectClass=user)(sAMAccountName={0}))`	`{0}` zastępuje login wpisany na ekranie logowania
Name Attribute	`sAMAccountName`	login w starym formacie, ten sam co do stacji Windows
Full Name Attribute	`displayName`	imię i nazwisko wyświetlane w interfejsie
Email Attributes	`mail`	adres do powiadomień
Default Roles	`Reader`	obowiązkowe: bez roli Reader zsynchronizowane konto nie otworzy interfejsu

◫ ZRZUT EKRANU DO UZUPEŁNIENIA

Zrzut do uzupełnienia: ekran System → Authentication po dodaniu usługi, wiersz „AD Zielona" ze statusem Aktywna i typem Active Directory.

Weryfikacja

Testujemy logowanie kontem domenowym na ekranie logowania Graylogu, używając loginu `ola.wisniewska` i hasła domenowego, zamiast konta `admin`.

Sprawdzamy z poziomu interfejsu, że konto rzeczywiście trafiło na listę użytkowników, w System → Users, z kolumną „Client" pokazującą nazwę usługi `AD Zielona` zamiast „internal".

◫ ZRZUT EKRANU DO UZUPEŁNIENIA

Zrzut do uzupełnienia: ekran System → Users po pierwszym logowaniu Oli przez LDAP, z widocznym kontem `ola.wisniewska` i źródłem `AD Zielona`.

Wycofanie zmian

Wyłączenie usługi uwierzytelniania bez usuwania importowanych kont, gdy chcemy tymczasowo wrócić do logowania lokalnego: w System → Authentication odznaczamy „Enable service" przy usłudze `AD Zielona`, zamiast ją usuwać.

Pełne usunięcie usługi: w System → Authentication, przy usłudze `AD Zielona`, wybieramy „Delete service". Konta, które zdążyły się już zalogować, zostają na liście użytkowników jako konta bez działającego źródła uwierzytelniania, aż administrator usunie je ręcznie.

Gdy coś nie zadziała

Objaw	Przyczyna	Co robimy
`LDAP Result Code 49 Invalid Credentials` w logu Graylogu	zły System User DN albo hasło konta technicznego	sprawdzamy DN i hasło komendą `ldapsearch` z hosta `docker01` przed poprawką w Graylogu
`unable to find valid certification path to requested target`	certyfikat firmowego CA niezaufany w magazynie `cacerts`	wykonujemy krok 2, dokładamy CA do keystore i restartujemy kontener
LDAPS działał, a po odtworzeniu kontenera znów zgłasza błąd certyfikatu	CA zaimportowane `keytool` do działającego kontenera zniknęło razem ze starym kontenerem	przechodzimy na trwały truststore z kroku 2 (`GRAYLOG_SERVER_JAVA_OPTS` i plik na hoście)
Konto istnieje w AD, ale logowanie odrzucone	Search Pattern bez `{0}` albo konto poza jednostką organizacyjną wskazaną w Search Base DN	sprawdzamy, czy `{0}` jest we wzorcu i czy konto rzeczywiście leży pod wskazanym `OU`
Konto loguje się, ale widzi tylko odczyt, mimo że powinno być administratorem	brak synchronizacji grup w wersji Open, opisany wyżej	podnosimy rolę ręcznie w System → Users, albo rozważamy licencję Enterprise przy większej liczbie takich wyjątków

Powiązane: Instalacja Grayloga w Dockerze: Graylog, MongoDB i OpenSearch · Logowanie kontem z Active Directory · Active Directory: konta, grupy, jednostki organizacyjne

Źródła: go2docs.graylog.org, rozdział „Active Directory and LDAP Authentication" · graylog.org, „Graylog Feature List" (Open vs Enterprise) · community.graylog.org, wątek o imporcie certyfikatu do magazynu Javy w kontenerze Graylog