Poczta: Postfix, MailCow, Zimbra

Droga wiadomości, konfiguracja Postfiksa, kolejka, diagnostyka dostarczania oraz platformy pocztowe.

- Poznajemy pocztę
  - Poczta: najstarsza usługa internetu, która wciąż potrafi zepsuć piątek
- Jak działa poczta
  - Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka
- Postfix
  - Postfix: konfiguracja
  - Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie
  - Diagnostyka dostarczania poczty
- Platformy pocztowe
  - Zimbra

Poznajemy pocztę

Opowieść i mapa książki.

Poczta: najstarsza usługa internetu, która wciąż potrafi zepsuć piątek

Poziom: podstawy **Dotyczy: poczta elektroniczna, MailCow Dockerized**

Czas: ~15 min

Ta strona otwiera książkę o poczcie: skąd wzięła się poczta elektroniczna, dlaczego jest starsza niż sam internet w dzisiejszym rozumieniu, i dlaczego Zielona zamiast trzymać ją w Zimbrze, przenosi się na MailCow.

Skróty użyte na tej stronie (15)

Skrót	Co znaczy
DKIM	DomainKeys Identified Mail: cyfrowy podpis wiadomości e-mail
DMARC	Domain-based Message Authentication, Reporting and Conformance: reguła domeny mówiąca, co robić z mailem, który nie przeszedł SPF i DKIM
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
IBM	International Business Machines: firma, w której laboratorium powstał model relacyjny i pierwszy działający silnik SQL
IMAP	Internet Message Access Protocol: protokół, którym klient pocztowy pobiera wiadomości z serwera, zostawiając je tam zamiast kasować
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
LDAP	Lightweight Directory Access Protocol: protokół do odpytywania katalogu użytkowników (AD, FreeIPA)
MTA	Mail Transfer Agent: serwer przekazujący pocztę dalej, np. Postfix
PDF	Portable Document Format: format dokumentu, który wygląda tak samo na każdym komputerze, np. zakres akredytacji jednostki na stronie PCA

Skrót	Co znaczy
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
SMTP	Simple Mail Transfer Protocol: protokół wysyłania poczty
SPF	Sender Policy Framework: rekord DNS mówiący, które serwery mogą wysyłać pocztę domeny
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera
WWW	World Wide Web: strony i aplikacje otwierane w przeglądarce po HTTP albo HTTPS

Skąd się wzięło

> **Pojęcie: poczta elektroniczna (e-mail).** Sposób przesyłania wiadomości tekstowych między komputerami, starszy niż strony WWW, starszy niż sam internet w dzisiejszym znaczeniu tego słowa. Pierwsze systemy poczty elektronicznej działały już w latach sześćdziesiątych, ale tylko między użytkownikami tego samego komputera: wiadomość zostawało się w cudzym katalogu domowym na tej samej maszynie, nikt nigdzie jej nie wysyłał.

To, co dziś rozumiemy jako pocztę, czyli wiadomość wędrującą między różnymi komputerami w sieci, powstało w 1971 roku. Ray Tomlinson, inżynier pracujący dla firmy BBN nad siecią ARPANET (poprzedniczką internetu), połączył istniejący już program do poczty lokalnej z programem do przesyłania plików między maszynami. Potrzebował sposobu zapisania, do kogo i na jakim komputerze wysyła wiadomość, więc sięgnął po znak `@`, który w tamtym czasie służył głównie do oznaczania cen na klawiaturach maszyn do pisania i praktycznie nie pojawiał się w nazwach użytkowników. Adres w formacie `użytkownik@komputer` rozdzielał od razu dwie rzeczy: kto jest odbiorcą i gdzie go szukać. Sam Tomlinson po latach żartował, że nie pamięta treści pierwszej testowej wiadomości, prawdopodobnie był to ciąg losowych znaków z górnego rzędu klawiatury.

Przez kolejną dekadę każdy ośrodek akademicki pisał swój własny format przesyłania poczty, co czyniło komunikację między różnymi sieciami koszmarem. Porządek zaprowadził dopiero **Pojęcie: SMTP (Simple Mail Transfer Protocol).** Protokół tekstowy do przekazywania wiadomości między serwerami, opisany w 1982 roku przez Jona Postela w dokumencie RFC 821, obowiązujący (po wielu poprawkach) do dziś. Więcej o samym SMTP i drodze wiadomości piszemy na stronie Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka.

Pierwszym powszechnie używanym programem realizującym SMTP po stronie serwera był Sendmail, napisany przez Erica Allmana z Uniwersytetu Kalifornijskiego w Berkeley. Sendmail zdominował internet na dwie dekady, ale zapłacił za to legendarnie nieczytelnym plikiem konfiguracyjnym: reguły trasowania poczty zapisywało się w składni tak gęstej, że administratorzy

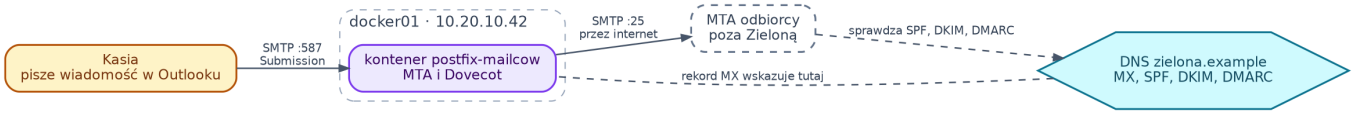
przechowywali gotowe fragmenty jak zaklęcia, nie do końca rozumiejąc, dlaczego działają. W 1998 roku Wietse Venema, holenderski badacz bezpieczeństwa pracujący wtedy dla IBM, opublikował Postfixa: MTA zaprojektowanego od zera z myślą o bezpieczeństwie i czytelnej konfiguracji. Postfix, opisany szerzej na stronie Postfix: konfiguracja, do dziś jest jednym z najpopularniejszych silników pocztowych na świecie i stoi pod maską większości gotowych pakietów, łącznie z MailCow.

Wraz z rosnącym internetem rosła też skala nadużyć: pole `From:` można było przez dekady wpisać dowolnie, bez żadnej weryfikacji. Fala spamu i phishingu wymusiła powstanie trzech mechanizmów uwierzytelniających nadawcę: SPF (2000), DKIM (2004, połączenie dwóch konkurencyjnych propozycji Yahoo i Cisco) oraz DMARC (2012, spinający oba poprzednie). Opisujemy je szczegółowo na stronie Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka: bez nich instalacja pocztowa trafia niemal automatycznie do spamu u odbiorców. Rekordy, które Zielona publikuje dla nich w strefie `zielona.example`, drogę DMARC do `p=reject` i szyfrowanie między serwerami opisuje osobna książka o domenie: strony SPF, DKIM i DMARC: trzy podejścia do poczty i systemy wysyłające w imieniu domeny, Droga DMARC od p=none do p=reject z raportami parsedmarc na docker01 i MTA-STS i TLS-RPT: szyfrowanie poczty między serwerami, a BIMI informacyjnie.

Prowadzenie własnego serwera pocztowego od zera, czyli ręczne złożenie Postfixa, Dovecota (serwera IMAP), filtra antyspamowego, webmaila i ochrony przed nadużyciami w jeden spójny system, jest projektem na tygodnie pracy i kolejne miesiące utrzymania. Stąd powstały gotowe pakiety pocztowe: komercyjna Zimbra Collaboration Suite (od 2005 roku, dziś rozwijana przez Synacor i Zextras) oraz otwartoźródłowy MailCow Dockerized, opisany krócej na stronie MailCow, łączący te same komponenty w zestaw kontenerów Docker uruchamianych jednym poleceniem.

Jak to działa w jednym obrazku

Wiadomość Kasi, wysłana z Outlooka do klienta spoza Zielonej, trafia najpierw do kontenera Postfixa wewnątrz MailCow na `docker01`, a stamtąd przez internet do serwera odbiorcy. Po drodze serwer odbiorcy sprawdza w DNS domeny `zielona.example` trzy rzeczy: czy adres IP, z którego przyszło połączenie, ma prawo wysyłać pocztę w imieniu tej domeny (SPF), czy podpis kryptograficzny w nagłówku się zgadza (DKIM) i co zrobić, jeśli któraś z tych dwóch kontroli zawiedzie (DMARC).



Droga wiadomości od Kasi przez MailCow na docker01 do serwera odbiorcy, z kontrolą DNS po drodze (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> Piątek, 15:50. Pani Halina wysyła do biura rachunkowego skan zamkniętego miesiąca z załączonym plikiem PDF. Mija dziesięć minut, telefon od księgowej: „nic nie doszło, sprawdziłam nawet spam". Tomek loguje się do starej Zimbry, przegląda kolejkę: wiadomość wisi z błędem `550 5.7.1 SPF check failed`, bo adres IP biurowego łącza od miesiąca nie jest już wpisany w rekordzie SPF domeny, odkąd dostawca internetu zmienił pulę adresów. Nikt tego nie zauważył, bo Zimbra nie ma żadnego czytelnego miejsca, w którym pokazałaby, że coś jest nie tak z autoryzacją domeny.

Zimbra działa w Zielonej od lat i działa poprawnie jako sam magazyn skrzynek, ale jej panel administracyjny rozdziela konfigurację MTA, LDAP i antyspamu na osobne narzędzia wiersza poleceń (`zmprov`, `zmcontrol`, edycję plików konfiguracyjnych ręcznie), opisane na stronie Zimbra. Diagnoza problemu takiego jak wyżej wymaga znajomości wewnętrznego, zmodyfikowanego Postfixa Zimbry i osobnych komend do każdego komponentu.

MailCow rozwiązuje ten sam zestaw problemów jednym webowym panelem administracyjnym: kolejka poczty, reguły antyspamowe silnika Rspamd, status DKIM i SPF danej domeny oraz dziennik zdarzeń każdej skrzynki widoczne są w jednym miejscu, bez logowania się po SSH. Do tego dochodzi mniejszy narzut utrzymania: aktualizacja to jeden skrypt (`update.sh`), a kopia zapasowa to jeden skrypt (`backup_and_restore.sh`), oba opisane na istniejącej stronie MailCow. To dlatego zespół IT Zielonej przenosi pocztę firmową z wygaszanej Zimbry na MailCow, zaczynając od instalacji opisanej na kolejnej stronie tej książki.

Mapa tej książki

Książka o poczcie łączy ogólną wiedzę o tym, jak działa poczta w ogóle, ze szczegółami dwóch konkretnych silników: Postfixa jako samodzielnego MTA i MailCow jako gotowego pakietu, którym Zielona zastępuje starą Zimbrę.

Strona	Na jakie pytanie odpowiada
Poczta: najstarsza usługa internetu, która wciąż potrafi zepsuć piątek	Skąd wzięła się poczta elektroniczna i dlaczego Zielona przechodzi z Zimbry na MailCow

Strona	Na jakie pytanie odpowiada
Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka	Kto jest kim na drodze wiadomości i co sprawdzają SPF, DKIM, DMARC
Postfix: konfiguracja	Jak skonfigurować Postfixa jako serwer wysyłający powiadomienia dalej
Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie	Jak sprawdzić, dlaczego wiadomość utknęła
Diagnostyka dostarczania	Jak czytać logi, gdy wiadomość nie dochodzi
MailCow	Jakie kontenery składają się na MailCow i jak wykonać kopię zapasową
Instalacja MailCow na docker01 od zera	Jak postawić MailCow od zera i podłączyć domenę `zielona.example`
MailCow: konta z Active Directory	Jak podłączyć logowanie do MailCow kontem domenowym
Zimbra	Jak obsłużyć starą pocztę, zanim zostanie wygaszona
MailCow: serwer pocztowy w kontenerach, który Zielona wybrała zamiast Zimbry	Osobna książka o MailCow: codzienna praca, kopia i aktualizacja oraz częste problemy

MailCow ma od niedawna własną książkę, którą otwiera strona z ostatniego wiersza tabeli: tam są codzienna praca, kopia zapasowa, aktualizacja i częste problemy.

Powiązane: Active Directory: konta, grupy, jednostki organizacyjne · Certyfikaty SSL: wystawienie, konwersja, weryfikacja · SPF, DKIM i DMARC: trzy podejścia do poczty i systemy wysyłające w imieniu domeny

Źródła: en.wikipedia.org/wiki/Ray_Tomlinson · en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol · en.wikipedia.org/wiki/Sendmail · en.wikipedia.org/wiki/Postfix_(software) · docs.mailcow.email

Jak działa poczta

Droga wiadomości, SPF, DKIM i DMARC.

Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka

Poziom: podstawy **Dotyczy: dowolny serwer pocztowy (Postfix, MailCow, Zimbra)**

Czas: ~20 min

Zanim zaczniemy konfigurować którykolwiek serwer pocztowy, ustalamy wspólny słownik — kto jest kim na drodze wiadomości i co faktycznie sprawdzają SPF, DKIM i DMARC.

Skróty użyte na tej stronie (17)

Skrót	Co znaczy
DKIM	DomainKeys Identified Mail: cyfrowy podpis wiadomości e-mail
DMARC	Domain-based Message Authentication, Reporting and Conformance: reguła domeny mówiąca, co robić z mailem, który nie przeszedł SPF i DKIM
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP
IMAP	Internet Message Access Protocol: protokół, którym klient pocztowy pobiera wiadomości z serwera, zostawiając je tam zamiast kasować
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
MDA	Mail Delivery Agent: program wkładający pocztę do skrzynki
MTA	Mail Transfer Agent: serwer przekazujący pocztę dalej, np. Postfix
MTA-STS	Mail Transfer Agent Strict Transport Security: zasada opublikowana przez domenę, że poczta do niej ma być wysyłana wyłącznie szyfrowanym połączeniem z serwerem o ustalonej nazwie
MUA	Mail User Agent: program pocztowy użytkownika, np. Outlook
MX	Mail Exchanger: rekord DNS wskazujący serwer poczty domeny
POP3	starszy protokół pobierania poczty na dysk lokalny, w odróżnieniu od IMAP
PTR	rekord DNS odwrotny: zamienia adres IP na nazwę
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
SMTP	Simple Mail Transfer Protocol: protokół wysyłania poczty

Skrót	Co znaczy
SPF	Sender Policy Framework: rekord DNS mówiący, które serwery mogą wysyłać pocztę domeny
TLS-RPT	SMTP TLS Reporting: rekord _smtp._tls z adresem, na który serwery nadawców wysyłają raz na dobę raport o udanych i nieudanych szyfrowanych połączeniach z naszymi serwerami poczty (RFC 8460)
TXT	typ rekordu DNS przechowującego dowolny tekst, wykorzystywany między innymi do publikowania reguł SPF, DKIM i DMARC

Z życia wzięte

□□ Z ŻYCIA WZIĘTE

Pan Wojtek podczas audytu pyta, dlaczego wiadomość z fałszywym adresem `prezes@zielona.example` w polu „Od" przeszła bez ostrzeżenia do skrzynki księgowej. Zespół sprawdza rekordy DNS i widzi poprawny SPF, ale brak jakiegokolwiek DMARC dla domeny. SPF sam w sobie nie broni pola widocznego w kliencie pocztowym, więc dopiero dodanie DMARC z polityką `quarantine` zaczyna faktycznie łapać takie próby.

Droga jednej wiadomości

Pojęcie: MUA (Mail User Agent). Program, którym użytkownik pisze i czyta pocztę — klient pocztowy typu Outlook, Thunderbird, albo aplikacja webmail. MUA nie dostarcza wiadomości samodzielnie, tylko przekazuje ją dalej protokołem SMTP do serwera.

Pojęcie: MTA (Mail Transfer Agent). Serwer odpowiedzialny za przekazywanie wiadomości między serwerami pocztowymi po protokole SMTP — to jest rola, którą pełnią Postfix, opisany na stronie Postfix: konfiguracja, oraz komponent MTA wewnątrz MailCow i Zimbry. MTA odbiera wiadomość od nadawcy (albo od innego MTA) i albo dostarcza ją lokalnie, albo przekazuje dalej w stronę serwera docelowego, ustalanego na podstawie rekordu MX domeny odbiorcy.

Pojęcie: MDA (Mail Delivery Agent). Komponent, który bierze wiadomość dostarczoną przez MTA do lokalnej domeny i zapisuje ją do faktycznej skrzynki odbiorcy — w formacie takim jak Maildir albo mbox, skąd MUA odczytuje ją przez protokół IMAP albo POP3. MTA i

MDA bywają częścią tego samego pakietu (Postfix ma wbudowany prosty MDA) albo osobnymi programami połączonymi ze sobą (np. Dovecot jako MDA obok Postfixa jako MTA).

Pełna droga wiadomości wysłanej z klienta pocztowego do skrzynki odbiorcy w innej domenie wygląda więc tak: MUA nadawcy → MTA nadawcy → (SMTP przez internet) → MTA odbiorcy → MDA odbiorcy → skrzynka → MUA odbiorcy (przez IMAP/POP3).



MUA nadawcy przekazuje wiadomość swojemu MTA, ten po SMTP przekazuje ją MTA odbiorcy, a MDA zapisuje ją do skrzynki, skąd MUA odbiorcy czyta ją przez IMAP lub POP3 (kliknięcie otwiera pełny rozmiar)

Czym jest kolejka

Pojęcie: kolejka. Miejsce, w którym MTA tymczasowo przechowuje wiadomości między odebraniem ich od nadawcy a skutecznym dostarczeniem dalej — wiadomość nie jest przekazywana natychmiast i bezpośrednio z gniazda nadawcy do gniazda odbiorcy, tylko najpierw zapisywana na dysku, a dopiero potem MTA próbuje ją dostarczyć, ewentualnie ponawiając próby, jeśli serwer docelowy jest chwilowo niedostępny. Kolejka jest tym, co pozwala poczcie działać mimo tymczasowych awarii sieci czy przeciążenia serwera odbiorcy — mechanizm ten opisujemy szczegółowo na przykładzie Postfixa na stronie Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie.

Protokół SMTP i porty

Pojęcie: SMTP (Simple Mail Transfer Protocol). Protokół tekstowy, którym jeden MTA przekazuje wiadomość do drugiego, oraz którym MUA wysyła nową wiadomość do swojego serwera. Ten sam protokół obsługuje trzy różne porty, w zależności od tego, kto z kim rozmawia i czy połączenie jest od razu szyfrowane.

Port	Nazwa	Zastosowanie

25/tcp	SMTP	komunikacja między serwerami MTA (relay) — tym portem jeden serwer pocztowy przekazuje wiadomość do drugiego po drodze do odbiorcy
465/tcp	SMTPS	wysyłka od klienta (MUA) do serwera z szyfrowaniem TLS nawiązywanym od razu przy połączeniu (implicit TLS)
587/tcp	Submission	wysyłka od klienta (MUA) do serwera, połączenie zaczyna się jako zwykłe, nieszyfrowane, a szyfrowanie włącza się komendą `STARTTLS` w trakcie sesji

Uważaj: port 25 jest tym, który blokuje większość dostawców internetu dla zwykłych łączy domowych i wiele serwerów aplikacyjnych w chmurze domyślnie — to jest powód, dla którego serwer aplikacyjny wysyłający pocztę przez zewnętrzny przekaźnik łączy się z nim zwykle na porcie 587 albo 465, a nie na 25, co opisujemy szerzej na stronie Postfix: konfiguracja.

Koperta a nagłówki

> **Pojęcie: koperta SMTP (envelope).** Adresy nadawcy i odbiorcy przekazywane w samym protokole SMTP komendami `MAIL FROM` i `RCPT TO`, używane przez serwery do faktycznego routingu wiadomości — to jest adres, na który serwer próbuje dostarczyć wiadomość i na który wraca powiadomienie o błędzie dostarczenia, jeśli coś pójdzie nie tak.

> **Pojęcie: nagłówki wiadomości (headers).** Pola `From:`, `To:`, `Subject:` widoczne w treści samej wiadomości, wyświetlane użytkownikowi przez klienta pocztowego. Koperta i nagłówki nie muszą się pokrywać — to normalny mechanizm używany np. przy wysyłce w imieniu wielu adresów (`Send As` opisany na stronie MailCow), ale jest też mechanizmem nadużywanym w phishingu, gdzie nagłówek `From:` pokazuje zaufany adres, a koperta wskazuje zupełnie inny serwer nadawcy.

SPF, DKIM, DMARC

> **Pojęcie: SPF (Sender Policy Framework).** Rekord DNS typu TXT publikowany przez właściciela domeny, wymieniający, które serwery mają prawo wysyłać pocztę w imieniu tej domeny — serwer odbiorcy sprawdza adres IP, z którego faktycznie przyszło połączenie

SMTP, i porównuje go z listą w rekordzie SPF domeny z koperty (`MAIL FROM`). SPF sprawdza więc tylko, czy nadawca połączył się z dozwolonego adresu — nic nie mówi o integralności treści wiadomości.

Przykład rekordu SPF:

```
v=spf1 ip4:203.0.113.10 include:_spf.zielona.example -all
```

Pojęcie: DKIM (DomainKeys Identified Mail). Mechanizm podpisu kryptograficznego dodawanego do nagłówków wiadomości przez serwer wysyłający, weryfikowanego przez klucz publiczny opublikowany w rekordzie DNS TXT domeny nadawcy. W przeciwieństwie do SPF, DKIM sprawdza integralność samej treści — jeśli wiadomość zostanie zmieniona po drodze (nawet przez pośredniczący serwer), podpis przestaje się zgadzać.

Pojęcie: DMARC (Domain-based Message Authentication, Reporting and Conformance). Rekord DNS TXT mówiący odbiorcy, co zrobić z wiadomością, która nie przejdzie SPF ani DKIM (albo przejdzie, ale niezgodnie z domeną z nagłówka `From:`) — możliwe polityki to `none` (nic nie rób, tylko raportuj), `quarantine` (do spamu) i `reject` (odrzuć). DMARC dodatkowo wiąże wynik SPF/DKIM z domeną widoczną w nagłówku `From:`, czego żaden z tych dwóch mechanizmów osobno nie robi — to jest mechanizm, który realnie broni przed podszywaniem się pod domenę w polu widocznym dla użytkownika.

DMARC opisuje dziś RFC 9989 z maja 2026 roku, który zastąpił RFC 7489: rekord nadal zaczyna się od `v=DMARC1`, ale znacznik `pct` przeszedł do historii, a doszły `np` i `t`. Rekordy dla Zielonej i systemów wysyłających w jej imieniu są na stronie SPF, DKIM i DMARC: trzy podejścia do poczty i systemy wysyłające w imieniu domeny, droga od `p=none` do `p=reject` na stronie Droga DMARC od p=none do p=reject z raportami parsedmarc na docker01, a szyfrowanie między serwerami (MTA-STS, TLS-RPT) na stronie MTA-STS i TLS-RPT: szyfrowanie poczty między serwerami, a BIMI informacyjnie.

Mechanizm	Co sprawdza	Gdzie żyje
SPF	czy adres IP nadawcy jest na liście dozwolonych dla domeny koperty	rekord DNS TXT
DKIM	czy treść wiadomości nie została zmieniona po podpisaniu	rekord DNS TXT (klucz publiczny) + nagłówek `DKIM-Signature`
DMARC	co zrobić, gdy SPF/DKIM zawiodą, i czy wynik zgadza się z domeną z `From:`	rekord DNS TXT

Rekordy MX i PTR

> **Pojęcie: rekord MX (Mail Exchanger).** Rekord DNS domeny odbiorcy, wskazujący nazwę serwera (albo kilku serwerów z priorytetami), do którego inne MTA mają wysyłać pocztę dla tej domeny — to właśnie ten rekord sprawdza MTA nadawcy, żeby w ogóle wiedzieć, dokąd się połączyć.

Przykład:

```
zielona.example.    3600    IN    MX    10 poczta.zielona.example.
```

> **Pojęcie: rekord PTR (reverse DNS).** Rekord odwrotny, przypisujący nazwę domenową do adresu IP — w przeciwieństwie do zwykłego rekordu A (nazwa → IP), PTR działa w drugą stronę (IP → nazwa) i jest utrzymywany przez właściciela zakresu adresów IP, nie przez właściciela domeny. Wiele serwerów pocztowych na świecie odrzuca albo mocno punktuje ujemnie połączenia z adresów IP bez poprawnie skonfigurowanego PTR, traktując jego brak jako sygnał serwera niewłaściwie skonfigurowanego albo pochodzącego z puli adresów dynamicznych typowych dla spamerów.

Częste błędy

Traktowanie SPF jako wystarczającego zabezpieczenia. SPF sam w sobie nie chroni przed podszywaniem się pod adres widoczny w kliencie pocztowym (nagłówek `From:`) — chroni tylko adres z koperty, który użytkownik zwykle w ogóle nie widzi. Pełna ochrona wymaga DMARC wiążącego oba mechanizmy z widoczną domeną nadawcy.

Brak rekordu PTR dla serwera wysyłającego. Wiadomości wysyłane z serwera bez poprawnego PTR trafiają do spamu albo są odrzucane przez sporą część odbiorców, mimo poprawnie skonfigurowanego SPF i DKIM — PTR ustawia dostawca łącza albo hostingu, nie da się go dodać samodzielnie w DNS własnej domeny.

□□ Ćwiczenia

Ćwiczenie 1 — sprawdzenie rekordów własnej domeny

```
dig +short TXT zielona.example | grep spf
dig +short MX zielona.example
dig +short -x 203.0.113.10
```

Sprawdzamy kolejno rekord SPF, rekord MX i rekord PTR dla wybranego adresu IP wysyłającego pocztę.

Odpowiedź

Pierwsza komenda pokazuje politykę SPF domeny, druga serwer(y) przyjmujące pocztę dla tej domeny, trzecia nazwę domenową przypisaną do adresu IP — brak wyniku w trzeciej komendzie oznacza brak skonfigurowanego PTR dla tego adresu.

Pytania kontrolne

1. Dlaczego wiadomość może przejść SPF, a mimo to zostać odrzucona przez DMARC?

Odpowiedź

SPF sprawdza domenę z koperty (`MAIL FROM`), a DMARC dodatkowo wymaga, żeby domena, która przeszła SPF (lub DKIM), zgadzała się z domeną widoczną w nagłówku `From:` — jeśli te dwie domeny się różnią, DMARC uznaje to za niezgodność, nawet gdy SPF samo w sobie przeszło pozytywnie.

Powiązane: SPF, DKIM i DMARC: trzy podejścia do poczty i systemy wysyłające w imieniu domeny · Droga DMARC od p=none do p=reject z raportami parsedmarc na docker01 · MTA-STS i TLS-RPT: szyfrowanie poczty między serwerami, a BIMI informacyjnie · Postfix: konfiguracja · Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie · Diagnostyka dostarczania

Źródła: RFC 5321 (SMTP) · RFC 7208 (SPF) · RFC 6376 (DKIM) · RFC 9989 (DMARC, maj 2026, zastąpił RFC 7489; zmiany opisuje strona o SPF, DKIM i DMARC w książce o domenie)

Postfix

Konfiguracja, kolejka i diagnostyka dostarczania.

Postfix: konfiguracja

Poziom: średni **Dotyczy: Postfix 3.x (Debian/Ubuntu i RHEL/Rocky)** **Czas: ~25 min**

Najczęstszy przypadek, jaki konfigurujemy, to nie pełny serwer pocztowy, tylko serwer aplikacyjny, który ma tylko wysyłać powiadomienia dalej przez zewnętrzny przekaźnik — ta strona pokazuje gotowy komplet dla takiego scenariusza.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
DKIM	DomainKeys Identified Mail: cyfrowy podpis wiadomości e-mail
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
MX	Mail Exchanger: rekord DNS wskazujący serwer poczty domeny
RHEL	Red Hat Enterprise Linux: komercyjna dystrybucja Linuksa
SMTP	Simple Mail Transfer Protocol: protokół wysyłania poczty

Fiszka

TYLKO ODCZYT

Podejrzenie faktycznie działającej konfiguracji: postconf -n

```
sudo postconf -n
```

> **Pojęcie: postconf -n.** Właściwy sposób oglądania konfiguracji Postfixa — wypisuje wyłącznie parametry, które faktycznie zostały jawnie ustawione w `main.cf`, pomijając setki parametrów pozostawionych na wartości domyślnej. Czytanie samego pliku `main.cf` przez `cat` pokazuje to samo, ale `postconf -n` jest odporne na komentarze, białe znaki i kolejność linii — dwie różnie sformatowane konfiguracje dające ten sam efekt wyglądają w `postconf -n` identycznie.

Przykład:

```
compatibility_level = 3.6
inet_interfaces = loopback-only
mydestination = $myhostname, localhost.$mydomain, localhost
myhostname = app01.ipa.zielona.example
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
myorigin = /etc/mailname
relayhost = [AdresPrzekaznika]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_tls_security_level = encrypt
smtpd_banner = $myhostname ESMTP $mail_name
```

□□ TYLKO ODCZYT

Sprawdzenie wartości jednego parametru: postconf

```
postconf mydestination
```

Pytamy o wartość dokładnie jednego parametru, wliczając wartość domyślną, jeśli nie został jawnie ustawiony — wygodniejsze niż filtrowanie wyniku `postconf -n`, gdy szukamy parametru, którego nikt jawnie nie nadpisał.

Przykład:

```
mydestination = $myhostname, localhost.$mydomain, localhost
```

□□ ZMIENIA SYSTEM

Wczytanie nowej konfiguracji: systemctl reload

```
sudo systemctl reload postfix.service
```

Nakazujemy działającej instancji ponownie wczytać `main.cf` bez zrywania trwających sesji SMTP — standardowe zakończenie edycji `main.cf`, wykonywane po każdej zmianie.

SMTP = Simple Mail Transfer Protocol: protokół wysyłania poczty · **systemctl** = system control: sterowanie usługami systemd

Z życia wzięte

Z ŻYCIA WZIĘTE

Ola zgłasza w środę, że nowa aplikacja na `app01` nie wysyła powiadomień o błędach, a `postconf -n` wygląda identycznie jak na działającym serwerze. Tomek sprawdza `smtp_sasl_password_maps` i widzi, że plik `sasl_passwd` został zmieniony ręcznie, ale nikt nie uruchomił potem `postmap`. Po przebudowaniu bazy `.db` powiadomienia wychodzą do przekaźnika bez dalszych zmian w konfiguracji.

Czym są main.cf i master.cf

Pojęcie: main.cf. Główny plik konfiguracyjny instancji Postfixa, w standardowej instalacji leżący w `/etc/postfix/main.cf` — każdy parametr to jedna linia `nazwa = wartość`. To jest plik, który edytujemy w zdecydowanej większości zmian konfiguracyjnych.

Pojęcie: master.cf. Plik definiujący, jakie usługi (demony) Postfix uruchamia i na jakich portach nasłuchują — `smtp`, `submission` (port 587), `smtps` (port 465), lokalny `pickup`, `cleanup` i inne. Zmieniamy go rzadziej niż `main.cf`, głównie gdy chcemy włączyć albo dostosować nasłuch na porcie 587/465, albo uruchomić drugą, niezależną instancję Postfixa na tej samej maszynie.

Najważniejsze parametry main.cf

Parametr	Do czego służy
`myhostname`	pełna nazwa FQDN tej instancji, używana m.in. w bannerze SMTP i jako baza dla `myorigin` / `mydestination`
`mydomain`	domena tej instancji, domyślnie wyliczana z `myhostname` przez odcięcie pierwszego segmentu

Parametr	Do czego służy
`myorigin`	domena, jaka pojawia się jako nadawca w wiadomościach wysyłanych lokalnie bez pełnego adresu — zwykle wskazuje plik `/etc/mailname` albo wprost `$mydomain`
`inet_interfaces`	interfejsy sieciowe, na których Postfix nasłuchuje; `loopback-only` ogranicza nasłuch do `127.0.0.1`, co jest typowe dla serwera aplikacyjnego niepełniącego roli publicznego MTA
`mynetworks`	zakresy adresów IP, którym Postfix ufa i pozwala przekazywać pocztę dalej (relay) bez uwierzytelnienia
`relayhost`	adres zewnętrznego serwera, przez który Postfix przekazuje całą wychodzącą pocztę zamiast dostarczać ją bezpośrednio
`smtp_sasl_auth_enable`	włączenie uwierzytelniania SASL przy łączeniu się z `relayhost`
`smtp_sasl_password_maps`	wskazanie pliku z danymi logowania do przekaźnika (`sasl_passwd`)
`message_size_limit`	maksymalny rozmiar pojedynczej wiadomości w bajtach, wliczając załączniki
`smtpd_recipient_restrictions`	reguły decydujące, które wiadomości przychodzące Postfix w ogóle przyjmie

Konfiguracja serwera aplikacyjnego wysyłającego przez przekaźnik

> **Pojęcie: przekaźnik (relay, smart host).** Zewnętrzny serwer SMTP, przez który lokalny Postfix przekazuje całą wychodzącą pocztę, zamiast próbować dostarczyć ją bezpośrednio do serwerów MX odbiorców — typowe rozwiązanie dla serwera aplikacyjnego, który generuje powiadomienia (raporty, alerty, potwierdzenia), ale sam nie jest pełnoprawnym serwerem pocztowym wystawionym do internetu, nie ma własnego DKIM ani dobrej reputacji IP.

To jest najczęstszy scenariusz, jaki konfigurujemy — serwer aplikacyjny bez własnej domeny pocztowej, którego jedynym zadaniem jest bezpiecznie i wiarygodnie przekazać pocztę dalej przez przekaźnik firmowy albo zewnętrznego dostawcę SMTP.



Aplikacja wysyła pocztę do lokalnego Postfixa na loopback, a ten przekazuje ją dalej przez uwierzytelniony przekaźnik SMTP zamiast łączyć się bezpośrednio z internetem (kliknięcie otwiera pełny rozmiar)

Ustawienie własnej nazwy hosta

```
sudo postconf -e "myhostname = NazwaSerwera.zielona.example"
```

Ograniczenie nasłuchu do interfejsu lokalnego

```
sudo postconf -e "inet_interfaces = loopback-only"
```

Serwer aplikacyjny nie ma przyjmować poczty z zewnątrz, tylko lokalnie od procesów działających na tej samej maszynie (np. cron, aplikacja wysyłająca powiadomienia) — ograniczenie nasłuchu do `loopback-only` zamyka Postfix na porcie 25 dla całej reszty sieci.

Wskazanie przekaźnika: relayhost

```
sudo postconf -e "relayhost = [AdresPrzekaznika]:587"
```

Nawiasy kwadratowe wokół adresu wyłączają wyszukiwanie rekordu MX dla tego adresu — Postfix łączy się z nim bezpośrednio, tak jakby to była nazwa hosta, a nie domena pocztowa.

Włączenie uwierzytelniania SASL wobec przekaźnika

```
sudo postconf -e "smtp_sasl_auth_enable = yes"
sudo postconf -e "smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd"
sudo postconf -e "smtp_sasl_security_options = noanonymous"
sudo postconf -e "smtp_tls_security_level = encrypt"
```

`smtp_sasl_security_options = noanonymous` zabrania Postfixowi próby połączenia bez uwierzytelnienia, gdyby z jakiegoś powodu dane logowania okazały się niedostępne — wolimy błąd dostarczenia niż wysłanie nieuwierzytelnionej próby, którą przekaźnik i tak odrzuci.

Utworzenie pliku z danymi logowania do przekaźnika: sasl_passwd

```
echo "[AdresPrzekaznika]:587 NazwaUzytkownika:HasloUzytkownika" | sudo tee
/etc/postfix/sasl_passwd
```

Format pliku to jedna linia na przekaźnik: adres i port w nawiasach kwadratowych, spacja, `login:hasło`.

Zbudowanie bazy danych z pliku sasl_passwd: postmap

```
sudo postmap /etc/postfix/sasl_passwd
```

> **Pojęcie: postmap.** Postfix nie czyta plików typu `hash:` bezpośrednio w formacie tekstowym — `postmap` kompiluje plik tekstowy do binarnej bazy danych (`sasl_passwd.db`) w formacie zadeklarowanym w nazwie mapy (`hash`, `btree`). Po każdej zmianie samego pliku tekstowego trzeba ponownie wykonać `postmap`, inaczej Postfix nadal korzysta ze starej zawartości bazy.

Przykład (komenda przy powodzeniu nic nie wypisuje, sprawdzamy kod wyjścia):

```
echo $?
```

```
0
```

Ograniczenie uprawnień pliku z hasłami

```
sudo chmod 0600 /etc/postfix/sasl_passwd /etc/postfix/sasl_passwd.db
```

Plik zawiera hasło w postaci jawnej — ograniczamy odczyt wyłącznie do właściciela (roota), tak samo jak każdy inny plik z poświadczeniami na serwerze.

Wczytanie kompletu zmian

```
sudo systemctl reload postfix.service
```

Limit rozmiaru wiadomości: message_size_limit

```
sudo postconf -e "message_size_limit = 26214400"
```

Wartość podajemy w bajtach — powyższe 26214400 to 25 MB, typowy limit zgodny z ograniczeniami większości zewnętrznych przekaźników i skrzynek odbiorczych. Wiadomość przekraczająca ten limit jest odrzucana już przez lokalny Postfix, zanim w ogóle trafi do kolejki wychodzącej.

Ograniczenia dla wiadomości przychodzących: smtpd_recipient_restrictions

```
smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated,
reject_unauth_destination
```

> **Pojęcie: smtpd_recipient_restrictions.** Uporządkowana lista reguł sprawdzana przy każdej próbie dostarczenia wiadomości do lokalnej instancji, sprawdzana od lewej do prawej — pierwsza reguła, która jednoznacznie rozstrzyga (zezwól albo odrzuć), kończy sprawdzanie. `permit_mynetworks` przepuszcza połączenia z zaufanych sieci zdefiniowanych w `mynetworks`, `permit_sasl_authenticated` przepuszcza klientów, którzy poprawnie się uwierzytelnili, a `reject_unauth_destination` na końcu odrzuca wszystko, co próbuje przekazać pocztę dalej (relay) bez spełnienia żadnego z wcześniejszych warunków — to jest reguła zamykająca, chroniąca przed użyciem naszego serwera jako otwartego przekaźnika (open relay) dla całego internetu.

Wielointancyjność Postfixa na jednym serwerze

> **Pojęcie: instancja Postfixa.** Osobny, niezależnie skonfigurowany zestaw plików (`main.cf`, `master.cf`, własny `sasl_passwd`) i osobna usługa systemd, działające obok siebie na tej samej maszynie, każda nasłuchująca na innym porcie lokalnym. Serwery aplikacyjne obsługujące kilku różnych klientów albo kilka niezależnych integracji pocztowych czasem uruchamiają po kilka takich instancji zamiast jednej wspólnej konfiguracji — każda z osobnym `relayhost`, osobnymi danymi logowania i osobną nazwą nadawcy.

Typowy układ katalogów przy kilku instancjach na jednym serwerze:

```
/etc/postfix/              — instancja domyślna, usługa postfix.service
/etc/postfix_klientA/      — druga instancja, usługa postfix_klientA.service
/etc/postfix_klientB/      — trzecia instancja, usługa postfix_klientB.service
```

Każda instancja ma własny plik `main.cf` pod swoją ścieżką, własną bazę `sasl_passwd` i własną jednostkę systemd o nazwie odpowiadającej nazwie katalogu — komendy `postmap`, `postconf -c SciezkaDoPliku` i `systemctl reload` trzeba wtedy wykonać osobno dla każdej instancji, której zmiana dotyczy.

```
sudo postmap /etc/postfix_klientA/sasl_passwd
sudo systemctl reload postfix_klientA.service
```

Uważaj: przy kilku instancjach na jednej maszynie zawsze sprawdzamy najpierw, której instancji dotyczy dana zmiana — część konfiguracji skrzynek bywa używana w więcej niż jednej instancji naraz, więc zmiana zrobiona tylko w jednej z nich, gdy powinna dotyczyć kilku, kończy się połowicznym, trudnym do zdiagnozowania efektem.

Częste błędy

Zapomniany postmap po zmianie sasl_passwd. Plik tekstowy jest już poprawny, ale Postfix nadal używa starych danych logowania, bo baza `.db` nie została przebudowana — każda zmiana pliku typu `hash:` wymaga ponownego `postmap`.

Brak reject_unauth_destination na końcu smtpd_recipient_restrictions. Serwer staje się otwartym przekaźnikiem, przez który dowolny nadawca z internetu potrafi wysyłać pocztę do dowolnego odbiorcy — to jest błąd konfiguracyjny prowadzący do umieszczenia adresu IP serwera

na czarnych listach w ciągu godzin.

Edycja main.cf bezpośrednio zamiast przez postconf -e. Ręczna edycja pliku jest w porządku, ale łatwo o literówkę w nazwie parametru, którą Postfix po prostu zignoruje bez ostrzeżenia — `postconf -e` sprawdza nazwę parametru względem znanej listy i od razu zwraca błąd przy literówce.

□□ Ćwiczenia

> Ćwiczenia zakładają lokalną instalację Postfixa i dostęp do dowolnego testowego konta SMTP z uwierzytelnianiem (np. skrzynka testowa na własnej domenie).

Ćwiczenie 1 — podgląd rzeczywistej konfiguracji

```
sudo postconf -n | sort
```

Wypisujemy i porównujemy z tabelą wyżej, które z najważniejszych parametrów są w naszej instalacji jawnie ustawione, a które pozostały na wartości domyślnej.

Odpowiedź

Świeża instalacja Postfixa zwykle ma ustawione tylko kilka parametrów (`myhostname`, `mydestination`, `mynetworks`, `inet_interfaces`) — reszta z tabeli, w tym cały blok `smtp_sasl_*` i `relayhost`, pojawia się dopiero po ręcznej konfiguracji przekaźnika.

Powiązane: Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka · Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie · Diagnostyka dostarczania

Źródła: `man 5 postconf` · `man 1 postmap` · dokumentacja postfix.org, rozdział „STANDARD_CONFIGURATION_README" i „SASL_README"

Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie

Poziom: średni **Dotyczy: Postfix 3.x** **Czas: ~25 min**

Sprawdzanie, dlaczego wiadomość utknęła, i porządkowanie kolejki z niewysłanych maili jest jedną z najczęstszych czynności przy utrzymaniu serwera pocztowego — każda komenda niżej dostaje własną fiszkę z realistycznym wynikiem.

Skróty użyte na tej stronie (1)

Skrót	Co znaczy
DNS	Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP

Fiszka

TYLKO ODCZYT

Podgląd kolejki: mailq

```
mailq
```

> **Pojęcie: mailq.** Najprostsza komenda podglądu kolejki, dostępna na każdej instalacji Postfixa — używamy jej wyłącznie, gdy na serwerze działa tylko jedna instancja, bo `mailq` nie przyjmuje parametru wskazującego inną instancję niż domyślna.

Przykład:

```
-Queue ID-  --Size-- ----Arrival Time---- -Sender/Recipient-------
3F2C1A4B821     4821 Wed Sep 24 09:12:03  powiadomienia@zielona.example
```

```
(connect to mail.example.net[203.0.113.44]:25: Connection timed out)
                                        odbiorca@example.net

7A9D3E1C440     1203 Wed Sep 24 09:14:51  raporty@zielona.example
                                        drugi.odbiorca@example.net

-- 2 Kbytes in 2 Requests.
```

mailq = podgląd kolejki niewysłanych wiadomości Postfiksa (skrót do postqueue -p)

TYLKO ODCZYT

Podgląd kolejki poleceniem natywnym: postqueue -p

```
sudo postqueue -p
```

Odpowiednik `mailq` w rodzinie komend `postqueue`, dający identyczny wynik — różnica w tym, że `postqueue` obsługuje parametr `-c` do wskazania innej instancji Postfixa, co opisujemy niżej przy wielointancyjności.

Przykład:

```
-Queue ID-  --Size-- ----Arrival Time---- -Sender/Recipient-------
3F2C1A4B821     4821 Wed Sep 24 09:12:03  powiadomienia@zielona.example
(connect to mail.example.net[203.0.113.44]:25: Connection timed out)
                                        odbiorca@example.net

-- 1 Kbytes in 1 Request.
```

postqueue = podgląd i opróżnianie kolejki Postfiksa

TYLKO ODCZYT

Podgląd kolejki innej instancji

```
sudo postqueue -c /etc/postfix_klientA -p
```

Jeśli na serwerze działa kilka instancji Postfixa, opisanych na stronie Postfix: konfiguracja, każda z osobnym katalogiem konfiguracji, parametr `-c` wskazuje, której instancji dotyczy zapytanie — bez niego `postqueue` odpytuje domyślną instancję pod `/etc/postfix`, nawet jeśli

faktycznie interesuje nas inna.

postqueue = podgląd i opróżnianie kolejki Postfiksa

ZMIENIA SYSTEM

Wymuszenie natychmiastowej próby dostarczenia: postqueue -f

```
sudo postqueue -f
```

> **Pojęcie: flush kolejki.** Nakazujemy Postfixowi natychmiastową próbę dostarczenia wszystkich wiadomości znajdujących się w `deferred`, zamiast czekać na kolejny automatyczny cykl wynikający z narastającego czasu odłożenia — używane typowo zaraz po naprawieniu przyczyny niepowodzenia (np. przywróceniu połączenia sieciowego albo poprawieniu rekordu DNS odbiorcy), żeby nie czekać kilku godzin na kolejną automatyczną próbę.

Przykład (komenda przy powodzeniu nic nie wypisuje):

```
sudo postqueue -f && echo "flush wykonany"
```

```
flush wykonany
```

DNS = Domain Name System: książka telefoniczna internetu, zamienia nazwy na adresy IP · **postqueue** = podgląd i opróżnianie kolejki Postfiksa

TYLKO ODCZYT

Podgląd treści konkretnej wiadomości w kolejce: postcat -q

```
sudo postcat -q 3F2C1A4B821
```

> **Pojęcie: postcat.** Wypisuje pełną treść wiadomości zidentyfikowanej numerem `IdKolejki`, wraz z kopertą (nadawca, odbiorcy) i nagłówkami — przydatne, gdy chcemy sprawdzić, czy wiadomość utknięta w kolejce ma poprawną treść i poprawnych odbiorców, zanim zdecydujemy, czy wymusić dostarczenie, czy ją usunąć.

Przykład:

```
*** ENVELOPE RECORDS deferred/3/3F2C1A4B821 ***
message_size:            4821             820               1
0            4821
message_arrival_time: Wed Sep 24 09:12:03 2026
sender: powiadomienia@zielona.example
named_attribute: rewrite_context=local
recipient: odbiorca@example.net
*** MESSAGE CONTENTS deferred/3/3F2C1A4B821 ***
Received: from localhost (localhost [127.0.0.1])
 by app01.ipa.zielona.example (Postfix) with ESMTP id 3F2C1A4B821
 for <odbiorca@example.net>; Wed, 24 Sep 2026 09:12:03 +0200 (CEST)
From: Powiadomienia <powiadomienia@zielona.example>
To: odbiorca@example.net
Subject: Raport dobowy
*** HEADER EXTRACTED ***
*** MESSAGE FILE END ***
```

□□ OSTROŻNIE · NIEODWRACALNE

Usunięcie pojedynczej wiadomości z kolejki: postsuper -d

```
sudo postsuper -d 3F2C1A4B821
```

> **Pojęcie: postsuper.** Narzędzie administracyjne do zarządzania kolejką — w odróżnieniu od `postqueue`, które tylko podgląda i wymusza próby dostarczenia, `postsuper` faktycznie usuwa, wstrzymuje albo zwalnia wiadomości. Usunięcie wiadomości tym poleceniem jest nieodwracalne — Postfix nie wysyła przy tym żadnego powiadomienia do nadawcy o niepowodzeniu dostarczenia, wiadomość po prostu znika z kolejki.

Przykład:

```
postsuper: 3F2C1A4B821: removed
postsuper: Deleted: 1 message
```

□□ **postsuper** = superużytkownik kolejki Postfiksa: usuwanie i wstrzymywanie maili

□□ OSTROŻNIE · NIEODWRACALNE

Usunięcie całej kolejki: postsuper -d ALL

```
sudo postsuper -d ALL
```

Uważaj: ta komenda usuwa bezpowrotnie wszystkie wiadomości znajdujące się we wszystkich katalogach kolejki (`incoming`, `active`, `deferred`, `hold`) — używamy jej wyłącznie w sytuacji, w której świadomie decydujemy, że żadna z zalegających wiadomości nie ma już wartości (np. po długiej awarii, gdy treść powiadomień jest już nieaktualna), nigdy jako rutynowego porządkowania.

Przykład:

```
postsuper: Deleted: 47 messages
```

□□ **postsuper** = superużytkownik kolejki Postfiksa: usuwanie i wstrzymywanie maili

□□ **TYLKO ODCZYT**

Wstrzymanie pojedynczej wiadomości: postsuper -h

```
sudo postsuper -h 3F2C1A4B821
```

> **Pojęcie: wstrzymanie (hold).** Przenosi wiadomość do katalogu `hold`, wyłączając ją z automatycznych prób dostarczenia bez usuwania — używane, gdy chcemy zbadać podejrzaną wiadomość albo wstrzymać wysyłkę do czasu wyjaśnienia sprawy po stronie odbiorcy, nie tracąc przy tym samej wiadomości.

Przykład:

```
postsuper: 3F2C1A4B821: placed on hold
postsuper: Placed on hold: 1 message
```

□□ **postsuper** = superużytkownik kolejki Postfiksa: usuwanie i wstrzymywanie maili

□□ **TYLKO ODCZYT**

Zwolnienie wstrzymanej wiadomości: postsuper -H

```
sudo postsuper -H 3F2C1A4B821
```

Wielka litera `-H` w odróżnieniu od małej `-h` — zwraca wiadomość z powrotem do kolejki `deferred`, gdzie ponownie zaczyna podlegać automatycznym próbom dostarczenia.

Przykład:

```
postsuper: 3F2C1A4B821: released from hold
postsuper: Released from hold: 1 message
```

postsuper = superużytkownik kolejki Postfiksa: usuwanie i wstrzymywanie maili

OSTROŻNIE · NIEODWRACALNE

Usunięcie wszystkich wiadomości od danego nadawcy: jednolinijkowiec

```
mailq | tail -n +2 | awk 'BEGIN{RS="";FS="\n"} /powiadomienia@zielona\.example/{print $1}' | awk '{print $1}' | tr -d '*!' | sudo postsuper -d -
```

Pojęcie: jednolinijkowiec czyszczący. `postsuper` nie ma wbudowanego filtra „usuń po nadawcy" — komenda składa taki filtr sama, przetwarzając wynik `mailq` na bloki (jeden blok na wiadomość, rozdzielone pustą linią przez `RS=""`), wyszukując bloki zawierające interesujący nas adres, wyciągając z nich sam identyfikator kolejki (pierwsza kolumna pierwszej linii bloku) i przekazując listę identyfikatorów do `postsuper -d -`, gdzie `-` każe czytać identyfikatory ze standardowego wejścia zamiast z argumentów.

Przykład:

```
postsuper: 3F2C1A4B821: removed
postsuper: 9B1E4F2A017: removed
postsuper: Deleted: 2 messages
```

mailq = podgląd kolejki niewysłanych wiadomości Postfiksa (skrót do postqueue -p)

OSTROŻNIE · NIEODWRACALNE

Usunięcie wszystkich wiadomości do danego odbiorcy: jednolinijkowiec

```
mailq | tail -n +2 | awk 'BEGIN{RS="";FS="\n"} /odbiorca@example\.net/{print $1}' | awk '{print $1}' | tr -d '*!' | sudo postsuper -d -
```

Ten sam mechanizm co wyżej, tylko wzorzec dopasowania to adres odbiorcy zamiast nadawcy — w wyniku `mailq` obie role widoczne są w tym samym bloku wiadomości, więc filtr działa identycznie niezależnie od tego, czy szukamy po `Sender` czy po `Recipient`.

mailq = podgląd kolejki niewysłanych wiadomości Postfiksa (skrót do postqueue -p)

Z życia wzięte

Z ŻYCIA WZIĘTE

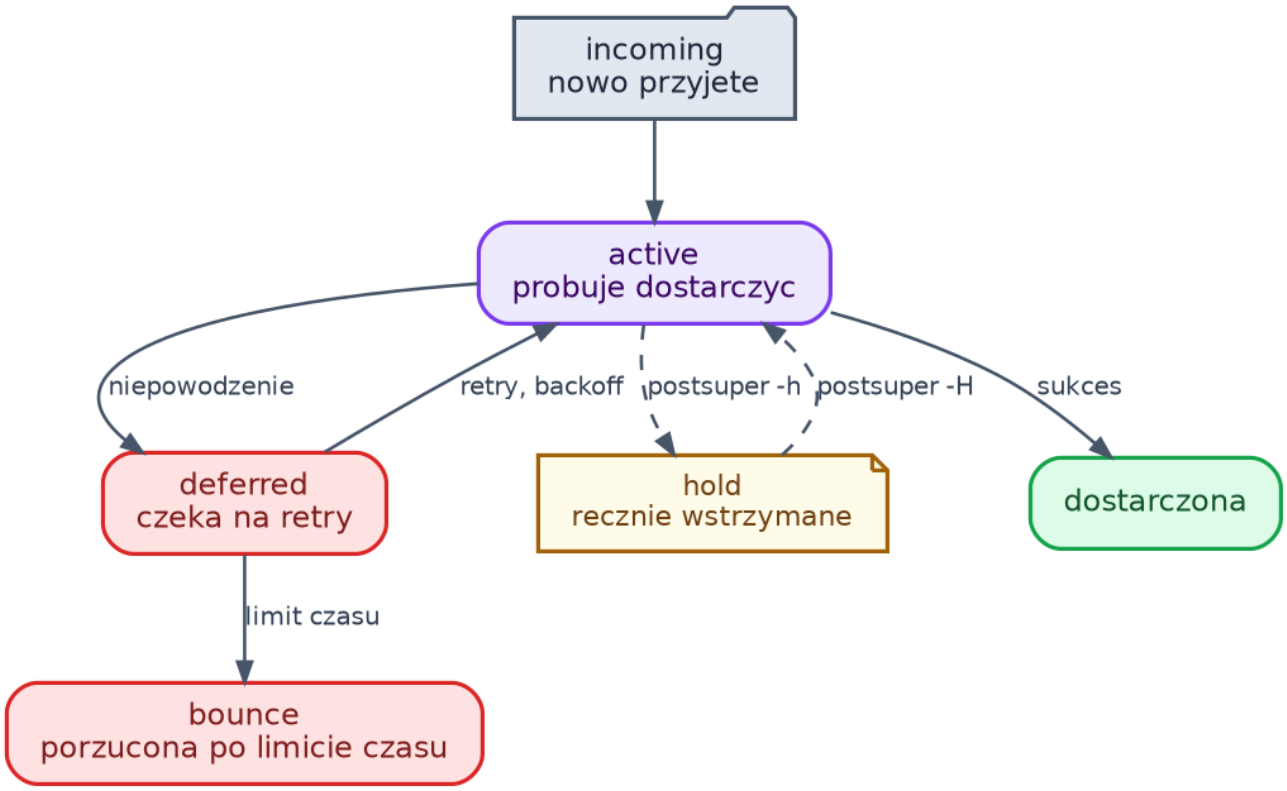
W poniedziałek po długim weekendzie Tomek widzi w `mailq` na `app01` kilkadziesiąt zaległych powiadomień o raportach z minionych dni, których treść jest już nieaktualna. Zamiast wymuszać dostarczenie każdej z osobna, sprawdza najpierw `postcat -q` na kilku z nich, żeby upewnić się, że rzeczywiście nikt na nie nie czeka. Dopiero potem czyści całą kolejkę `postsuper -d ALL`, zamiast robić to odruchowo za każdym razem, gdy `mailq` pokazuje więcej niż kilka pozycji.

Czym jest kolejka Postfiksa

Pojęcie: kolejka Postfiksa. Zbiór katalogów na dysku, w których Postfix przechowuje wiadomości między odebraniem ich a skutecznym dostarczeniem, wprowadzony ogólnie na stronie Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka. W standardowej instalacji katalogi te leżą pod `/var/spool/postfix/` i każdy z nich odpowiada innemu etapowi przetwarzania wiadomości.

Katalog	Rola
`incoming`	wiadomości dopiero co przyjęte, jeszcze przed przetworzeniem przez `cleanup`
`active`	wiadomości aktualnie przetwarzane, którymi zajmuje się demon `qmgr`, próbujące zostać dostarczone w tej chwili
`deferred`	wiadomości, których dostarczenie się nie powiodło i czekają na kolejną próbę po czasie odłożenia (backoff)

Katalog	Rola
`hold`	wiadomości ręcznie wstrzymane, niebrane pod uwagę do dostarczenia, dopóki ktoś ich nie zwolni



Wiadomość wchodzi przez incoming do active, stamtąd trafia do dostarczonych albo do deferred na retry, aż w końcu do bounce po limicie czasu, a hold to osobna, ręczna ścieżka (kliknięcie otwiera pełny rozmiar)

Wiadomość, która nie zostanie dostarczona za pierwszym razem (np. serwer odbiorcy chwilowo niedostępny), trafia z `active` do `deferred` i tam czeka na kolejną próbę wysyłaną automatycznie w odstępach rosnących wykładniczo, aż do przekroczenia maksymalnego czasu w kolejce, po którym Postfix odsyła nadawcy komunikat o niepowodzeniu dostarczenia (bounce).

Wymuszenie i czyszczenie przy kilku instancjach naraz

Na serwerze z kilkoma instancjami Postfixa, opisanymi na stronie Postfix: konfiguracja, każda instancja ma własną, w pełni niezależną kolejkę — flush jednej z nich nie dotyka pozostałych.

```
sudo postqueue -c /etc/postfix_klientA -f
```

Przeładowanie wszystkich instancji naraz, gdy nazwy usług systemd współdzielą wspólny wzorzec:

```
sudo systemctl reload postfix*
```

Częste błędy

Użycie mailq zamiast postqueue -c przy wielu instancjach. `mailq` zawsze odpytuje domyślną instancję pod `/etc/postfix` — na serwerze z kilkoma instancjami wygląda to na pustą kolejkę, mimo że w innej instancji zalega dziesiątki wiadomości.

postsuper -d ALL wykonane bez rozeznania. Usuwa bezpowrotnie także wiadomości, które za chwilę i tak zostałyby poprawnie dostarczone (np. po tymczasowej awarii sieci) — przed tak szerokim usunięciem sprawdzamy przyczynę zalegania komendą `postcat -q` na kilku przykładowych wiadomościach.

Flush wykonany zanim naprawiono przyczynę problemu. `postqueue -f` nie naprawia niczego samo z siebie — wymuszona próba dostarczenia bez usunięcia rzeczywistej przyczyny (zły DNS, zablokowany port, błędne dane logowania do przekaźnika) kończy się dokładnie tym samym błędem i ponownym odłożeniem do `deferred`.

□□ Ćwiczenia

> Ćwiczenia zakładają lokalną instalację Postfixa skonfigurowaną tak, by dostarczenie na chwilę się nie powiodło (np. wskazanie nieistniejącego przekaźnika).

Ćwiczenie 1 — pełny cykl: zaległość, podgląd, wymuszenie

```
mail -s "Test" test@nieistniejaca-domena.invalid <<< "tresc testowa"
sleep 5
mailq
```

Wysyłamy testową wiadomość do nieistniejącej domeny, sprawdzamy, że trafiła do `deferred`, a następnie próbujemy `postqueue -f`.

Odpowiedź

Wiadomość pojawia się w wyniku `mailq` z komunikatem błędu DNS (np. `Host or domain name not found`) — `postqueue -f` w tym przypadku nic nie zmieni, bo przyczyna (nieistniejąca domena) nie ustąpi sama, więc po pewnym czasie Postfix odeśle nadawcy komunikat o trwałym niepowodzeniu dostarczenia.

Powiązane: Postfix: konfiguracja · Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka · Diagnostyka dostarczania

Źródła: `man 1 postqueue` · `man 1 postsuper` · `man 1 postcat` · dokumentacja postfix.org, rozdział „QSHAPE_README"

Diagnostyka dostarczania poczty

Poziom: średni **Dotyczy: Postfix 3.x** **Czas: ~20 min**

Zanim zaczniemy podejrzewać konfigurację, sprawdzamy log — niemal każde niepowodzenie dostarczenia zostawia w nim jasny ślad, o ile wiemy, czego szukać.

Skróty użyte na tej stronie (8)

Skrót	Co znaczy
IP	Internet Protocol: adres komputera w sieci; w elektryce: stopień ochrony obudowy (Ingress Protection)
RBL	Realtime Blackhole List: lista adresów IP, z których szła niechciana poczta; wiele serwerów odrzuca pocztę z adresów znajdujących się na takiej liście
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
RHEL	Red Hat Enterprise Linux: komercyjna dystrybucja Linuksa
SMTP	Simple Mail Transfer Protocol: protokół wysyłania poczty
SPF	Sender Policy Framework: rekord DNS mówiący, które serwery mogą wysyłać pocztę domeny
TCP	Transmission Control Protocol: protokół sieciowy z potwierdzaniem dostarczenia
TLS	Transport Layer Security: szyfrowanie połączeń, np. w HTTPS

Fiszka

TYLKO ODCZYT

Śledzenie logu na żywo: tail -f

```
sudo tail -f /var/log/maillog
```

Widzimy kompletny przebieg jednej udanej wiadomości jako kilka kolejnych linii.

Przykład:

```
Sep 24 09:12:03 NazwaSerwera postfix/smtpd[18402]: 3F2C1A4B821:
client=localhost[127.0.0.1]
Sep 24 09:12:03 NazwaSerwera postfix/cleanup[18405]: 3F2C1A4B821: message-
id=<20260924071203.3F2C1A4B821@NazwaSerwera.zielona.example>
Sep 24 09:12:03 NazwaSerwera postfix/qmgr[1421]: 3F2C1A4B821:
from=<powiadomienia@zielona.example>, size=4821, nrcpt=1 (queue active)
Sep 24 09:12:04 NazwaSerwera postfix/smtp[18410]: 3F2C1A4B821:
to=<odbiorca@example.net>, relay=mail.example.net[203.0.113.20]:25, delay=1.2,
delays=0.1/0/0.6/0.5, dsn=2.0.0, status=sent (250 2.0.0 Ok: queued as A1B2C3D4)
Sep 24 09:12:04 NazwaSerwera postfix/qmgr[1421]: 3F2C1A4B821: removed
```

□□ **tail** = pokaż koniec pliku, z flagą -f śledzi plik na żywo

□□ TYLKO ODCZYT

Śledzenie jednej wiadomości po queue id: grep

```
sudo grep '3F2C1A4B821' /var/log/maillog
```

Przykład:

```
Sep 24 09:12:03 NazwaSerwera postfix/smtpd[18402]: 3F2C1A4B821:
client=localhost[127.0.0.1]
Sep 24 09:12:03 NazwaSerwera postfix/qmgr[1421]: 3F2C1A4B821:
from=<powiadomienia@zielona.example>, size=4821, nrcpt=1 (queue active)
Sep 24 09:12:04 NazwaSerwera postfix/smtp[18410]: 3F2C1A4B821:
to=<odbiorca@example.net>, relay=none, delay=0.1, delays=0.1/0/0/0, dsn=4.4.1,
status=deferred (connect to mail.example.net[203.0.113.44]:25: Connection timed out)
```

Uważaj: przy dużym ruchu ten sam krótki identyfikator queue id bywa reużyty przez Postfix po jego zwolnieniu z poprzedniej wiadomości — przy diagnozowaniu starszych wpisów warto dodatkowo filtrować po znaczniku czasu, żeby mieć pewność, że wszystkie znalezione linie dotyczą tej samej, a nie dwóch różnych wiadomości o tym samym identyfikatorze wydanym w różnych dniach.

□□ **grep** = global regular expression print: szukaj wzorca w tekście

TYLKO ODCZYT

Wysyłka testowa: swaks

```
swaks --to odbiorca@example.net --from powiadomienia@zielona.example --server localhost
--port 587 --auth LOGIN --auth-user NazwaUzytkownika --auth-password HasloUzytkownika --
tls
```

Przykład:

```
-> EHLO NazwaKlienta
<-  250-poczta.zielona.example
<-  250-STARTTLS
<-  250 AUTH LOGIN PLAIN
-> STARTTLS
<-  220 2.0.0 Ready to start TLS
-> AUTH LOGIN
<-  334 VXNlcm5hbWU6
-> ...
<-  235 2.7.0 Authentication successful
-> MAIL FROM:<powiadomienia@zielona.example>
<-  250 2.1.0 Ok
-> RCPT TO:<odbiorca@example.net>
<-  250 2.1.5 Ok
-> DATA
<-  354 End data with <CR><LF>.<CR><LF>
-> .
<-  250 2.0.0 Ok: queued as B7E1F209A3
```

swaks = Swiss Army Knife for SMTP: program do ręcznego wysyłania wiadomości testowych i podglądu rozmowy z serwerem poczty

TYLKO ODCZYT

Wysyłka testowa: sendmail

```
echo -e "Subject: Test\n\nTresc testowa" | sendmail -v odbiorca@example.net
```

Prostsza alternatywa bez zależności od dodatkowego pakietu — flaga `-v` włącza tryb gadatliwy, pokazujący przebieg lokalnego przekazania wiadomości do kolejki Postfixa.

Przykład:

```
odbiorca@example.net... Connecting to local...
odbiorca@example.net... Sent (3F2C1A4B821 Message accepted for delivery)
```

echo = od angielskiego echo: wypisuje podany tekst na standardowe wyjście, często łączone z przekierowaniem do pliku

TYLKO ODCZYT

Ręczna sesja SMTP przez telnet (port 25, bez szyfrowania)

```
telnet localhost 25
```

Ręczne wpisywanie komend SMTP (`EHLO`, `MAIL FROM`, `RCPT TO`, `DATA`) — przydatne do zweryfikowania, czy Postfix w ogóle odpowiada na porcie 25 i jakim bannerem, zanim zaczniemy podejrzewać coś bardziej złożonego.

SMTP = Simple Mail Transfer Protocol: protokół wysyłania poczty · **RCPT** = komenda protokołu SMTP wskazująca adres odbiorcy wiadomości

TYLKO ODCZYT

Ręczna sesja SMTP z STARTTLS na porcie 587: openssl s_client

```
openssl s_client -connect localhost:587 -starttls smtp
```

Przykład:

```
250 2.0.0 Ready to start TLS
depth=0 CN = poczta.zielona.example
verify return:1
```

SMTP = Simple Mail Transfer Protocol: protokół wysyłania poczty · **STARTTLS** = włączenie szyfrowania w trakcie już nawiązanej, nieszyfrowanej sesji pocztowej · **openssl** = zestaw narzędzi do certyfikatów i szyfrowania

Z życia wzięte

Z ŻYCIA WZIĘTE

W czwartek Ola zgłasza, że raport wysyłany co noc z `app01` nie dochodzi do jednego konkretnego klienta, a do reszty owszem. Tomek filtruje `maillog` po adresie tego odbiorcy i widzi powtarzający się kod `550 5.7.1 Service unavailable, client blacklisted`. Sprawdzenie adresu IP `app01` na publicznych listach RBL potwierdza wpis sprzed kilku miesięcy, o którym nikt w zespole nie wiedział.

Czytanie /var/log/maillog

Pojęcie: maillog. Log, do którego Postfix (oraz Dovecot i inne komponenty pocztowe na tej samej maszynie) zapisuje każdy etap obsługi wiadomości — na RHEL/Rocky domyślnie `/var/log/maillog`, na Debianie/Ubuntu zwykle `/var/log/mail.log`. Każda linia zaczyna się znacznikiem czasu, nazwą hosta i nazwą procesu, który ją zapisał (`postfix/smtpd`, `postfix/qmgr`, `postfix/smtp`).

Śledzenie jednej wiadomości po queue id

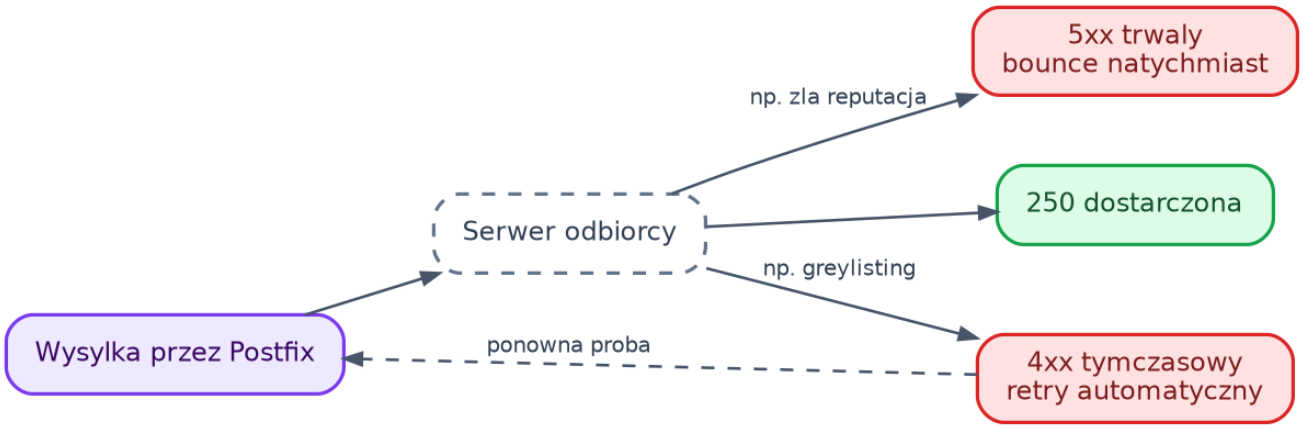
Pojęcie: queue id. Identyfikator wiadomości nadawany przez Postfix w momencie przyjęcia jej do kolejki (widoczny też w wyniku `mailq` i `postqueue -p`, opisanych na stronie Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie) — ten sam identyfikator pojawia się we wszystkich liniach loga dotyczących danej wiadomości, aż do jej usunięcia z kolejki, co pozwala prześledzić całą jej drogę jednym poleceniem.

Kody odpowiedzi 4xx a 5xx

Pojęcie: kod DSN (Delivery Status Notification). Trzycyfrowy kod zwracany przez serwer odbiorcy (albo przez sam Postfix przy odrzuceniu lokalnym), którego pierwsza cyfra od razu mówi, czy problem jest tymczasowy, czy trwały.

Klasa kodu	Znaczenie	Zachowanie Postfixa
4xx (np. 450, 451, 421)	błąd tymczasowy — serwer odbiorcy prosi o ponowienie próby później	wiadomość trafia do `deferred` i Postfix próbuje ponownie automatycznie
5xx (np. 550, 554)	błąd trwały — serwer odbiorcy odrzuca wiadomość jednoznacznie i bezwarunkowo	Postfix nie ponawia próby, od razu odsyła nadawcy komunikat o niepowodzeniu (bounce)

Uważaj: rozróżnienie 4xx/5xx jest decyzją serwera odbiorcy, nie naszą — ten sam realny problem (np. greylisting) różni dostawcy sygnalizują różnymi kodami, więc samą liczbę zawsze czytamy razem z treścią komunikatu tekstowego, który za nią stoi.



Kod 4xx od serwera odbiorcy wraca do Postfixa jako sygnał do ponownej próby, kod 5xx kończy się natychmiastowym bounce bez ponawiania (kliknięcie otwiera pełny rozmiar)

Najczęstsze przyczyny odrzuceń

> **Pojęcie: greylisting.** Technika antyspamowa, w której serwer odbiorcy przy pierwszej próbie od nieznanej wcześniej trójki (adres IP nadawcy, adres nadawcy, adres odbiorcy) celowo odpowiada tymczasowym błędem 4xx, licząc na to, że prawdziwy serwer pocztowy — w odróżnieniu od większości oprogramowania spamowego — spróbuje ponownie po chwili, zgodnie ze specyfikacją SMTP. Postfix robi to automatycznie, więc greylisting jest niewidoczny dla użytkownika poza kilkuminutowym opóźnieniem pierwszej wiadomości do nowego odbiorcy.

Przyczyna	Typowy komunikat w logu	Co sprawdzamy

brak albo zły SPF	`550 5.7.1 SPF check failed`	rekord TXT SPF domeny nadawcy, opisany na stronie Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka
brak rekordu PTR	`450 4.7.1 Client host rejected: cannot find your reverse hostname`	rekord PTR adresu IP wysyłającego serwera u dostawcy łącza
greylisting	`450 4.2.0 Greylisted, please try again in 180 seconds`	nic nie robimy — Postfix ponowi próbę sam, problem znika po kilku minutach
zła reputacja adresu IP	`550 5.7.1 Service unavailable, client blacklisted`	sprawdzenie adresu IP na publicznych listach RBL (np. Spamhaus)
limit rozmiaru	`552 5.3.4 Message size exceeds fixed maximum message size`	`message_size_limit` po naszej stronie albo limit skrzynki odbiorcy

Test wysyłki z wiersza poleceń

> **Pojęcie: swaks.** Narzędzie zbudowane specjalnie do testowania sesji SMTP — w odróżnieniu od zwykłego klienta pocztowego pokazuje pełny przebieg rozmowy z serwerem linia po linii, wliczając moment negocjacji `STARTTLS` i uwierzytelnienia, co czyni je dużo wygodniejszym narzędziem diagnostycznym niż wysłanie zwykłej wiadomości testowej i czekanie na skutek.

> **Pojęcie: STARTTLS.** W odróżnieniu od portu 465, gdzie szyfrowanie zaczyna się natychmiast po nawiązaniu połączenia TCP, port 587 zaczyna sesję jako zwykłe, nieszyfrowane połączenie, a klient dopiero w trakcie rozmowy wysyła komendę `STARTTLS`, żądając przejścia na szyfrowanie. Flaga `-starttls smtp` w `openssl s_client` odwzorowuje dokładnie ten mechanizm, w przeciwieństwie do zwykłego połączenia TLS, które na porcie 587 zakończyłoby się błędem, bo serwer oczekuje najpierw zwykłego tekstu `EHLO`.

Częste błędy

Traktowanie greylistingu jako awarii. Pojedyncza wiadomość odrzucona z kodem 450 i komunikatem o ponowieniu próby za kilka minut to normalne działanie mechanizmu antyspamowego odbiorcy, nie usterka po naszej stronie — Postfix sam ją dostarczy przy kolejnej automatycznej próbie, bez żadnej interwencji.

Testowanie przez port 587 bez -starttls. `openssl s_client -connect host:587` bez flagi `-starttls smtp` kończy się błędem negocjacji TLS, bo serwer po nawiązaniu połączenia na tym porcie wysyła zwykły tekstowy banner SMTP, a nie od razu rozpoczyna handshake TLS.

Ignorowanie treści komunikatu przy kodzie 550. Sam kod mówi tylko „odrzucone trwale" — realna przyczyna (zła reputacja, SPF, zbyt duży rozmiar, nieistniejąca skrzynka) zawsze jest w tekście towarzyszącym kodowi, który trzeba przeczytać, a nie tylko sprawdzić pierwszą cyfrę.

□□ Ćwiczenia

> Ćwiczenia zakładają lokalną instalację Postfixa z dostępem do `/var/log/maillog`.

Ćwiczenie 1 — pełne prześledzenie jednej wiadomości

```
echo "test" | mail -s "sledzenie" odbiorca@example.net
sleep 2
qid=$(grep -oP '(?<=postfix/qmgr\[\d{1,6}\]: )[A-F0-9]+(?=: from=)' /var/log/maillog | tail -1)
grep "$qid" /var/log/maillog
```

Wysyłamy wiadomość testową, wyciągamy jej queue id z ostatniego wpisu `qmgr` i filtrujemy cały log po tym identyfikatorze.

Odpowiedź

Wynik powinien pokazać komplet linii dla tej wiadomości: przyjęcie przez `smtpd`, nadanie message-id przez `cleanup`, umieszczenie w kolejce przez `qmgr`, próbę dostarczenia przez `smtp` (lub `local`) i finalne usunięcie z kolejki po powodzeniu.

Powiązane: Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie · Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka · Postfix: konfiguracja

Źródła: `man 1 swaks` · dokumentacja postfix.org, rozdział „DSN_README" · RFC 5321

Platformy pocztowe

MailCow i Zimbra.

Zimbra

Poziom: podstawy **Dotyczy: Zimbra Collaboration Suite** **Czas: ~15 min**

Zimbra to kolejny kompletny serwer pocztowy, podobnie jak MailCow, ale zbudowany jako jeden pakiet instalowany na maszynie, nie jako zestaw kontenerów — poniżej same podstawy, którymi realnie się posługujemy.

Skróty użyte na tej stronie (2)

Skrót	Co znaczy
LDAP	Lightweight Directory Access Protocol: protokół do odpytywania katalogu użytkowników (AD, FreeIPA)
MTA	Mail Transfer Agent: serwer przekazujący pocztę dalej, np. Postfix

Fiszka

TYLKO ODCZYT

Przełączenie się na konto zimbra

```
sudo su - zimbra
```

> **Pojęcie: konto zimbra.** Wszystkie komendy administracyjne Zimbry (`zmcontrol`, `zmprov`, `zmqstat`) uruchamiamy z poziomu tego konta systemowego, nie z poziomu roota — sama usługa też działa na tym koncie, więc uprawnienia do jej plików i gniazd są z nim spójne.

su = substitute user: przełącz się na innego użytkownika

TYLKO ODCZYT

Sprawdzenie statusu wszystkich usług: zmcontrol status

```
zmcontrol status
```

Wypisujemy stan każdego komponentu Zimbry na danym serwerze — MTA, mailbox, LDAP, antyspam, w zależności od tego, które role ten konkretny serwer pełni.

Przykład:

```
Host zimbra01.ipa.zielona.example
        antispam                Running
        antivirus                Running
        ldap                     Running
        mailbox                  Running
        mta                      Running
        stats                    Running
```

MTA = Mail Transfer Agent: serwer przekazujący pocztę dalej, np. Postfix · **LDAP** = Lightweight Directory Access Protocol: protokół do odpytywania katalogu użytkowników (AD, FreeIPA) · **zmcontrol** = sterowanie usługami serwera pocztowego Zimbra (start/stop/status)

TYLKO ODCZYT

Zatrzymanie i uruchomienie usług: zmcontrol stop / start

```
zmcontrol stop
zmcontrol start
```

Zatrzymujemy i uruchamiamy cały komplet komponentów Zimbry na danym serwerze naraz — używane głównie przy pracach serwisowych wymagających pełnego zatrzymania usługi, rzadziej przy zwykłej zmianie konfiguracji, która zwykle wystarcza przeładować pojedynczy komponent.

zmcontrol = sterowanie usługami serwera pocztowego Zimbra (start/stop/status)

TYLKO ODCZYT

Sprawdzenie wersji: zmcontrol -v

```
zmcontrol -v
```

Przykład:

```
Release 10.0.9.GA.4179.UBUNTU20.64
```

zmcontrol = sterowanie usługami serwera pocztowego Zimbra (start/stop/status)

TYLKO ODCZYT

Zarządzanie kontami i domenami: zmprov

```
zmprov sp AdresSkrzynki@zielona.example NoweHaslo
```

> **Pojęcie: zmprov.** Narzędzie wiersza poleceń do zarządzania obiektami w LDAP Zimbry — kontami, domenami, listami dystrybucyjnymi. `sp` (set password) zmienia hasło wskazanej skrzynki bez potrzeby logowania się do panelu administracyjnego, przydatne przy pracy skryptowej albo masowej zmianie haseł.

Przykład (komenda przy powodzeniu nic nie wypisuje):

```
zmprov sp uzytkownik@zielona.example NoweHaslo123! && echo "haslo zmienione"
```

```
haslo zmienione
```

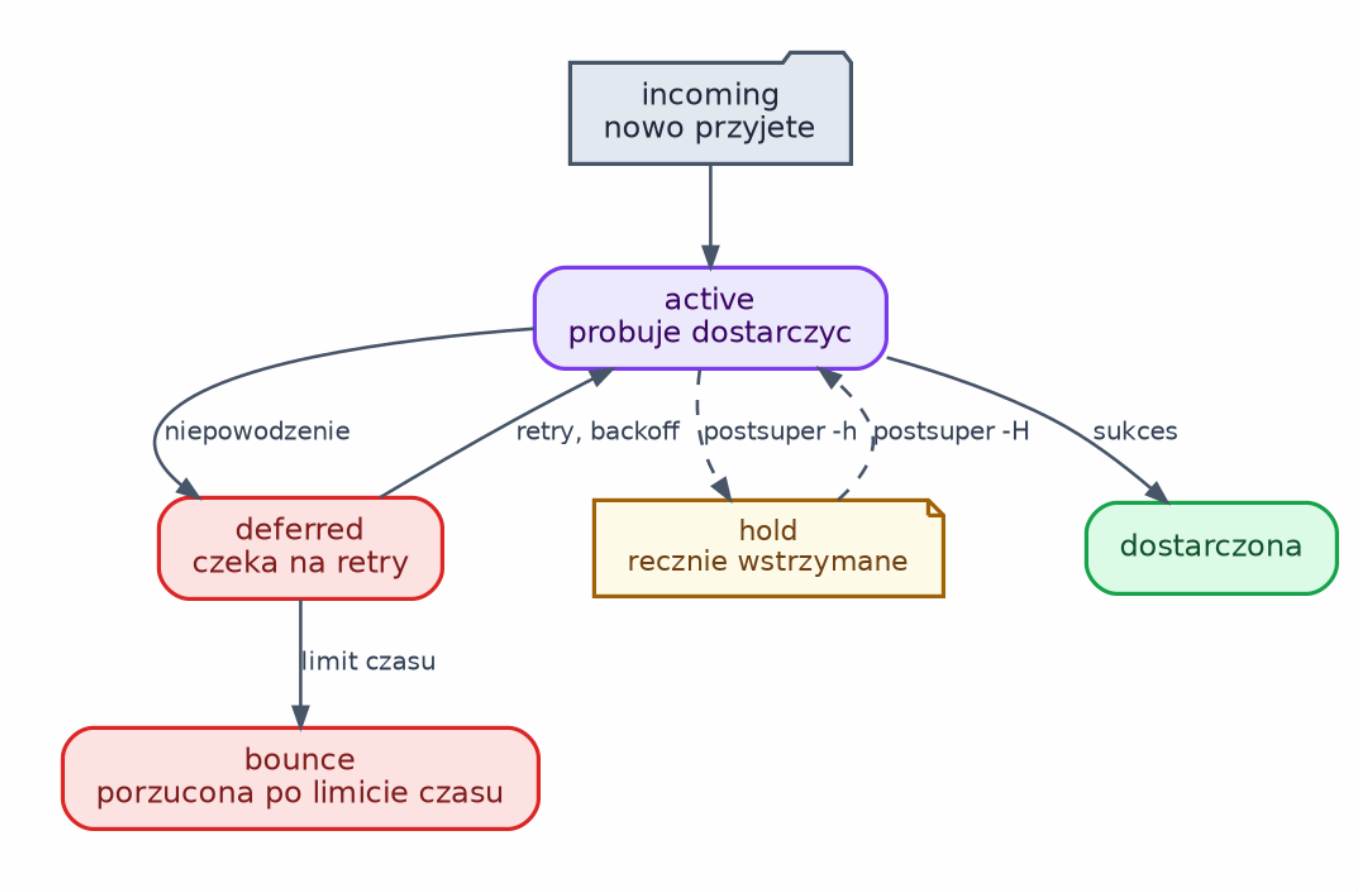
LDAP = Lightweight Directory Access Protocol: protokół do odpytywania katalogu użytkowników (AD, FreeIPA) · **zmprov** = Zimbra provisioning: zarządzanie kontami i domenami w Zimbrze z linii poleceń

TYLKO ODCZYT

Podgląd kolejki pocztowej: zmqstat

```
zmqstat
```

> **Pojęcie: kolejka Zimbry.** Zimbra wykorzystuje wewnętrznie zmodyfikowanego Postfixa jako MTA, więc mechanizm kolejki jest koncepcyjnie identyczny z tym opisanym na stronie Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie — `zmqstat` to wygodny skrót pokazujący podsumowanie liczby wiadomości w poszczególnych stanach kolejki, bez konieczności wywoływania natywnych komend Postfixa ze zmienioną ścieżką instalacji.



Kolejka wewnętrznego Postfixa Zimbry przechodzi przez te same stany incoming, active, deferred i hold co zwykły Postfix (kliknięcie otwiera pełny rozmiar)

Przykład:

```
active=0
deferred=3
hold=0
incoming=0
```

Do pełnego podglądu treści konkretnej wiadomości w kolejce sięgamy po natywne narzędzie Postfixa spod ścieżki instalacji Zimbry:

```
/opt/zimbra/common/sbin/postqueue -p
```

MTA = Mail Transfer Agent: serwer przekazujący pocztę dalej, np. Postfix

TYLKO ODCZYT

Podgląd logów: /var/log/zimbra.log

```
sudo tail -f /var/log/zimbra.log
```

Główny log zdarzeń Zimbry, wspólny dla większości komponentów — pierwsze miejsce, w którym szukamy śladu po błędzie logowania, problemie z dostarczeniem albo awarii komponentu.

tail = pokaż koniec pliku, z flagą -f śledzi plik na żywo

Z życia wzięte

Z ŻYCIA WZIĘTE

Zespół trzyma jeszcze `zimbra01` uruchomiony wyłącznie dla kilku starych skrzynek archiwalnych, które czekają na migrację do MailCow. W poniedziałek pani Halina zgłasza, że nie widzi starej korespondencji sprzed dwóch lat w nowej skrzynce, bo nikt jej nie powiedział, że to osobny, jeszcze niewygaszony system. Tomek loguje się na `zimbra01` kontem `zimbra`, znajduje wiadomości przez `zmprov` i przekierowuje ją do właściwego archiwum, zamiast czekać na pełną migrację.

Czym jest Zimbra

Pojęcie: Zimbra. Pakiet serwera pocztowego łączący MTA (własny, zmodyfikowany Postfix), serwer LDAP z danymi kont i domen, silnik antyspamowy (SpamAssassin) i webmail — instalowany jako jeden komplet na serwerze Linux, uruchamiany na dedykowanym koncie systemowym `zimbra`. W środowiskach z kilkoma serwerami Zimbry role bywają rozdzielone — jeden serwer pełni funkcję MTA i bramy pocztowej, inny trzyma skrzynki (mailbox), jeszcze inny LDAP — więc pierwszym krokiem diagnozy jest zawsze ustalenie, którą rolę pełni dany serwer.

Dodanie użytkownika do reguły blokującej spam

Zimbra pozwala blokować wiadomości od konkretnego nadawcy dla całej instalacji przez edycję pliku reguł odrzuceń wewnętrznego Postfixa, a nie przez panel administracyjny — zmiana wymaga ręcznego przeładowania mapy.

```
sudo su - zimbra
vi /opt/zimbra/postfix-*/conf/postfix_reject_sender
```

Dodajemy linię z adresem nadawcy do zablokowania, w formacie zgodnym z resztą pliku, po czym przebudowujemy mapę:

```
postmap /opt/zimbra/postfix-*/conf/postfix_reject_sender
```

Kopia zapasowa

Zimbra nie ma jednego wbudowanego skryptu kopii zapasowej równoważnego `backup_and_restore.sh` z MailCow — kopię zapasową wykonujemy zwykle na poziomie systemu plików (katalog `/opt/zimbra/`) oraz eksportu LDAP, dobierając zakres do architektury konkretnego wdrożenia (pojedynczy serwer albo rozdzielone role MTA/mailbox/LDAP). Przy środowisku z rozdzielonymi rolami kopię LDAP wykonujemy zawsze na serwerze pełniącym tę rolę, nie na serwerze MTA.

Częste błędy

Wykonywanie komend zmprov/zmcontrol jako root. Większość poleceń administracyjnych Zimbry wymaga uruchomienia z konta `zimbra` — próba wywołania ich jako root kończy się błędem braku uprawnień do gniazd i plików należących do tego konta.

Mylenie roli serwera w środowisku z rozdzielonymi komponentami. Sprawdzenie kolejki albo restart usługi na serwerze pełniącym rolę LDAP zamiast MTA nic nie da — `zmcontrol status` na początku każdej diagnozy pokazuje, które role dany serwer faktycznie pełni.

□□ Ćwiczenia

Ćwiczenie 1 — sprawdzenie ról własnego serwera

```
sudo su - zimbra -c "zmcontrol status"
```

Sprawdzamy, które komponenty Zimbry działają na danym serwerze, zanim spróbujemy wykonać jakąkolwiek zmianę konfiguracji.

Odpowiedź

Wynik pokazuje listę komponentów ze statusem `Running` albo `Stopped` — jeśli brakuje w niej np. `mta`, oznacza to, że ten serwer nie pełni roli bramy pocztowej i zmiany w konfiguracji Postfixa trzeba wykonać na innym serwerze środowiska.

Powiązane: Kolejka Postfiksa: podgląd, wymuszenie, czyszczenie · Jak działa poczta: MTA, MDA, MUA, SMTP, kolejka · MailCow

Źródła: dokumentacja Zimbra (zimbra.com/documentation) · `zmprov --help` · `zmcontrol --help`