STUDIA INFORMATICA 2010
Volume 31 Number 2B (90)

Mariusz FRYDRYCH, Wojciech HORZELSKI
Wyższa Szkoła Informatyki w Łodzi, Uniwersytet Łódzki, Wydział Matematyki i Informatyki

# *DBMAIL* – SYSTEM POCZTY ELEKTRONICZNEJ O ARCHITEKTURZE BAZODANOWEJ

**Streszczenie.** W pracy przedstawiono system poczty elektronicznej, w którym fragment zbioru użytkowników, uprawnień, jak i tzw. skrzynek pocztowych jest zrealizowany w całości po stronie bazy danych, a nie w natywnym systemie plików, tak jak to występuje w klasycznych systemach pocztowych. Każda z wiadomości jest rozkładana na kilka części charakterystycznych dla poczty elektronicznej i umieszczana w schemacie relacyjnej bazy danych. Z systemem sprzęgnięte są dwa interfejsy protokołów odbioru poczty (dla MUA) IMAP i POP3, dwa dla serwerów (MTA) SMTP i LMTP oraz interfejsy do konfiguracji ustawień i zarządzania zasobami systemu. Omawiany system *DBmail* jest projektem z otwartym źródłem (open source). System został przetestowany i wdrożony w jednostce średniej wielkości z około 15 tysiącami użytkowników w warunkach produkcyjnych.

**Słowa kluczowe**: poczta elektroniczna, open source

# *DBMAIL* - DATABASE ARCHITECTURE E-MAIL SYSTEM

**Summary**. The paper presents an electronic mail system where a storage part is entirely implemented on the database system. The presented system *DBmail* is an open-source project which has been tested and implemented in an average volume production environment of about 15 thousands of users.

**Keywords**: electronic mail, open source

## 1. Struktura systemu *DBmail*

Poczta elektroniczna jest jedną z najbardziej powszechnych usług współczesnych sieci komputerowych. Zaawansowane systemy poczty elektronicznej umożliwiają użytkownikom dostęp do zasobów firmowych w dowolnym czasie i z dowolnego miejsca. Nieodzownym atrybutem takich systemów jest zapewnienie odpowiedniego poziomu bezpieczeństwa, dostępności oraz

niezawodności. Należy przy tym dążyć, aby taki system był jak najłatwiejszy w administracji i najbardziej opłacalny. W dzisiejszym świecie trudno wyobrazić sobie funkcjonowanie jakiejkolwiek instytucji bez własnego systemu obsługi poczty elektronicznej.

Systemy poczty elektronicznej zbudowane są zazwyczaj z następujących składników [1]:

- serwery poczty *MTA* (*Mail Transfer Agent*), np. *Exim, SendMail, Posfix, Qmail, Exchange*
- klienci poczty *MUA* (*Mail User Agent*), np. *Thunderbird, Evolution, Microsoft Outlook*

Serwery *MTA* wykorzystują protokół komunikacyjny poczty wychodzącej *SMTP* (*Simple Mail Transfer Protocol*), z kolei klienci poczty *MUA* zarówno protokół serwerowy (*SMTP*), jak i protokoły tzw. poczty przychodzącej *POP3* (*Post Office Protocol ver. 3*) i *IMAP* (*Internet Message Access Protocol*) [2]. Prowadzi to czasami do nieprecyzyjnego rozumienia problemu transferu poczty elektronicznej przez jej mniej doświadczonych użytkowników. Mianowicie, ten sam protokół *SMTP* służy zarówno do transferu poczty przez serwery (*MTA*), jak i jej wysyłania przez użytkowników, za pomocą klientów (*MUA*). Jest to po części zrozumiałe, ponieważ nie wykształcił się jeszcze „wydelegowany" protokół wysyłania poczty przez klientów pocztowych. Co prawda, niektóre serwery *MTA* (np. *Exim*) „potrafią" odróżnić rodzaj klienta po drugiej stronie, tzn. czy jest to inny serwer *MTA* czy klient *MUA*, obsługując klienta w wyspecjalizowanym trybie tzw. *submission mode* [3 i 4]. Do takiej obsługi protokołu *SMTP* [6] już dawno wydelegowano oddzielny port TCP 587 (poza portem 25), co nawet znalazło zastosowanie w polityce TP SA blokującej od 1 grudnia 2009 roku port 25 na wyjściu w usłudze Neostrada. W ten sposób zachęca się użytkowników do używania portu *submission mode* 587, co ma na celu m.in. ograniczenie niechcianej poczty.

W niniejszym artykule przedstawiono analizę system *DBmail* (system jest typu *open-source*) oraz opisano jego wdrożenie w warunkach produkcyjnych. Architektura tego systemu oparta jest na systemie bazy danych sprzęgniętej za pomocą dobrze zdefiniowanych interfejsów z elementami „klasycznej" poczty elektronicznej. Cała logika użytkowników, ról, skrzynek, aliasów, przekazywania (*forwarding*) oraz medium magazynowania wiadomości wraz ze strukturą nagłówków i załączników została tu zaimplementowana w dość złożonej strukturze relacyjnej bazy danych.

Opisany system doskonale się sprawdza w środowisku poczty „korporacyjnej" czy „hostingowej", tzn. w środowisku ze znaczną ilością użytkowników, w którym zajmowane stanowiska nie posiadają cech trwałości, co prowadzi do wykształcenia relacji pomiędzy „prawdziwymi" użytkownikami a ich rolami, zwanymi „wirtualnymi" użytkownikami systemu. Środowiska tego typu są charakterystyczne dla instytucji o charakterze edukacyjnym (np. wyższe uczelnie, instytucję szkoleniowe).

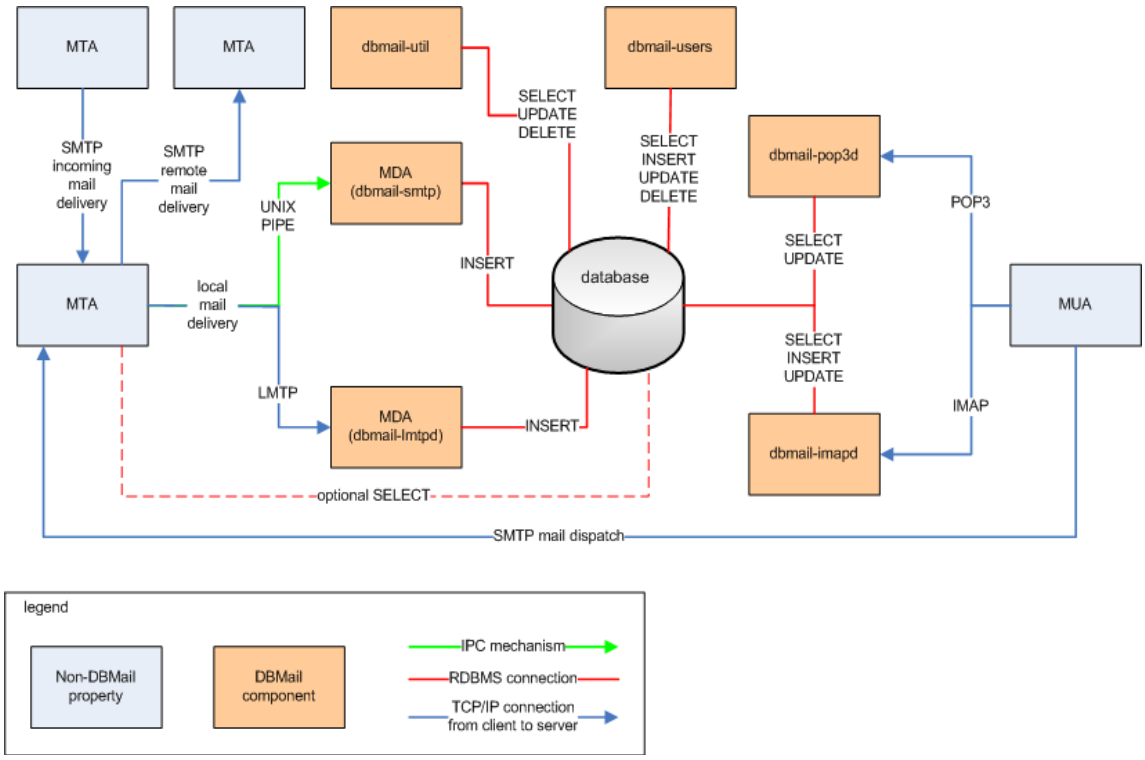
Tak zbudowany system można w bardzo prosty sposób wpisać w strukturę klastrową. Mianowicie, wiele serwerów *MTA* może być w relacji z wieloma bazami danych, a cały klaster skła-

da się w warstwie logicznej z połączenia tych elementów za pomocą interfejsów dostarczonych przez opisywany tutaj system *DBmail*.

Nie do przecenienia jest też kwestia bezpieczeństwa tak skonstruowanego systemu, przede wszystkim użytkownicy nie mają dostępu do systemu plikowego, a jedynie dostęp pośredni przez mechanizmy bazy danych. Taka architektura znacznie upraszcza zarządzanie, archiwizację oraz migrację systemu, co z kolei znacznie ułatwia pracę administratorom. Pewnym mankamentem opisywanego systemu, w jego obecnej wersji, jest brak warstwy kryptograficznej. Jednak ten problem można łatwo rozwiązać instalując w krytycznych miejscach przepływu danych (np. *dbmail-imapd* czy *dbmail-pop3d*) szyfrujące tunele SSL, np. program z otwartym źródłem *stunnel* [4 i 5].

W trakcie projektowania systemu *DBmail* szczególny nacisk położony został na następujące cechy: skalowalność, zarządzalność, wydajność, bezpieczeństwo i elastyczność.

W celu realizacji tych cech zaprojektowano system pocztowy o budowie modularnej:

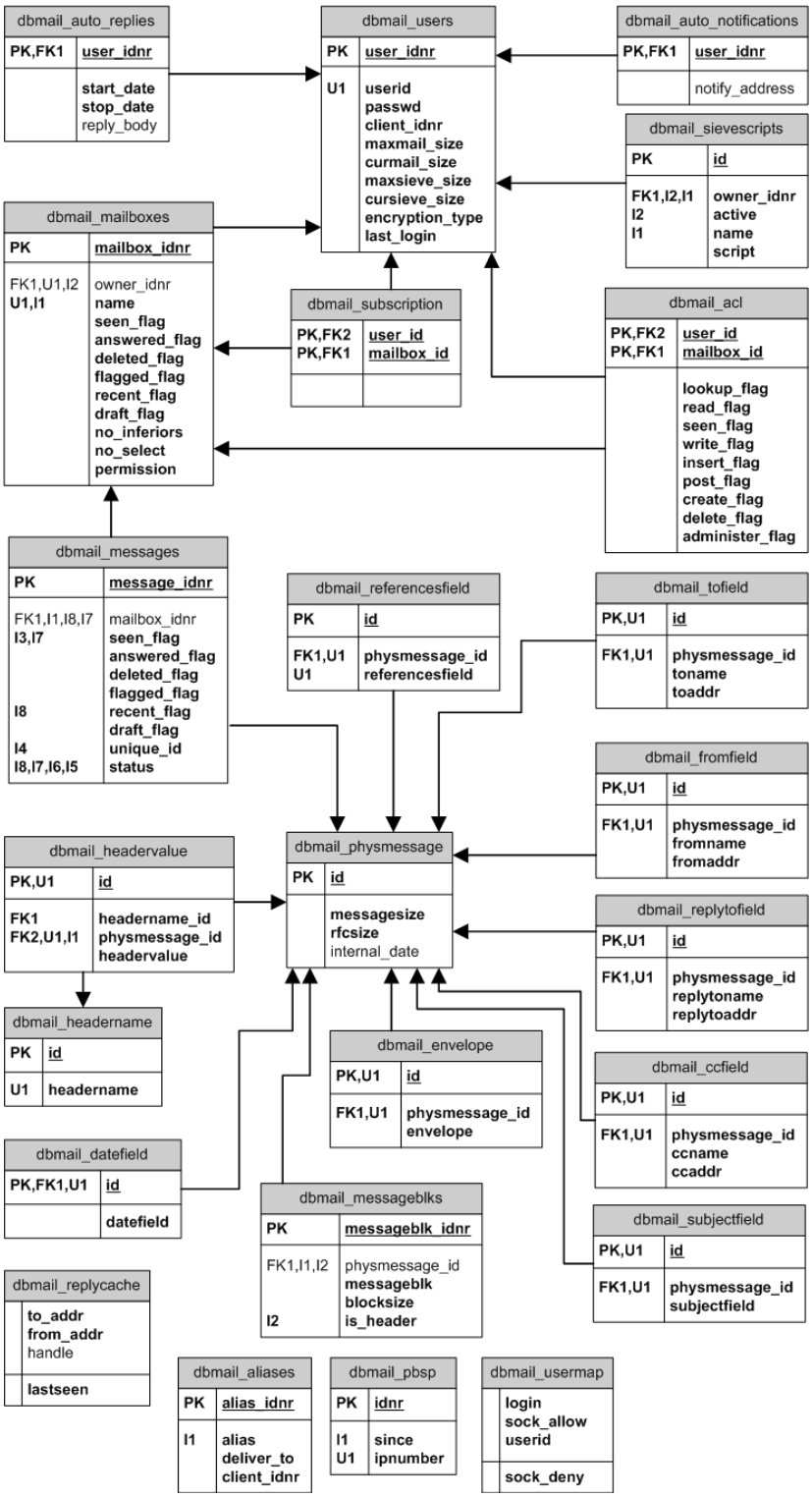



Rys. 1. Schemat systemu *DBmail*
Fig. 1. *DBmail* diagram

System składa się z następujących modułów:

- serwerów poczty przychodzącej *IMAP* (*dbmail-imapd*) i *POP3* (*dbmailpop3d*),
- serwera (*dbmail-lmtpd*) protokołu komunikacyjnego *LMTP* (uproszczony protokół *SMTP*),
- interfejsu dla protokołu *SMTP* (*dbmail-smtp*),
- modułu do zarządzania użytkownikami i ich zasobami (*dbmail-users*),
- modułu do zarządzania zasobami baz danych (*dbmail-util*),

- narzędzia do migracji systemu (*dbmail-export*),
- sterowników do silnika baz danych: *Sqlite*, *MySQL*, *PostgreSQL*.

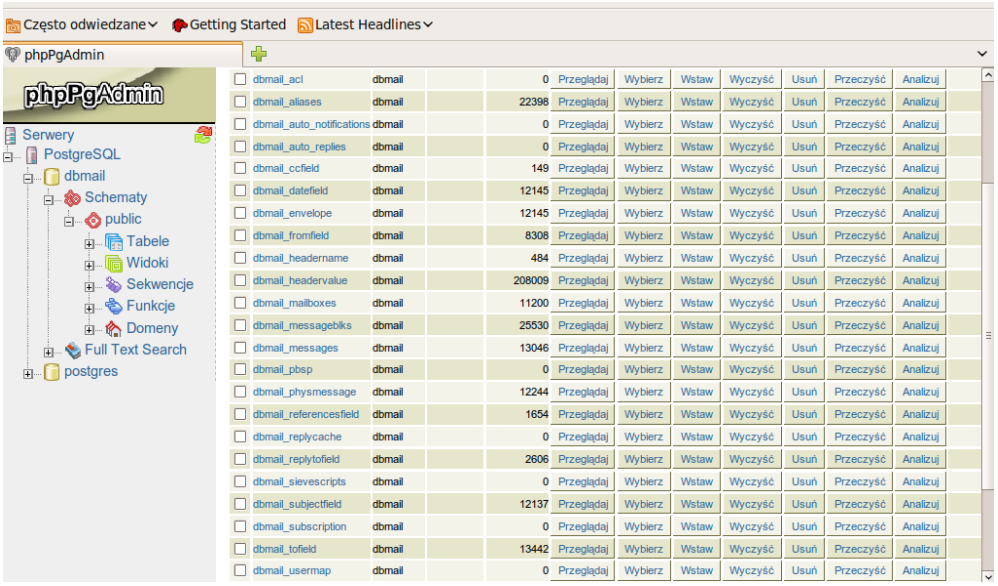



Rys. 2. Diagram relacyjnej bazy danych systemu *DBmail*
Fig. 2. *DBmail* relational databases diagram

Fundamentalnym elementem systemu jest ostatni z tych składników, tzn. interfejs pośredniczący pomiędzy warstwą poczty elektronicznej a silnikiem baz danych. Przykładowo

w implementacji sterownika do silnika *PostgreSQL*, relacyjna baza danych składa się z 23 tabel. Szczegółowa budowa bazy przedstawiona została na schemacie (rys. 2).

Zarządzanie bazą możne odbywać się poprzez odpowiednie dla wybranego systemu narzędzia (np. rys 3. przedstawia *phpPgAdmin* dla bazy *PostgreSQL*). Z łatwością można zauważyć, że mamy oddzielne tabele dla użytkowników, ich skrzynek, folderów, aliasów, przekierowań, nagłówków i kopert wiadomości oraz „fizycznej" treści wraz z załącznikami. Pozwala to na transformację, z pozoru amorficznego, obiektu, jaką jest wiadomość *email*, na w pełni zestrukturyzowny element relacyjnej bazy danych. Interpretowanie zasobów poczty elektronicznej w postaci obiektów bazy danych skutkuje znakomitą wydajnością systemu pocztowego, m.in. szybkim wyszukiwaniem kontekstowym, łatwym i efektywnym zarządzaniem, skalowalnością i elastycznością.




Rys. 3. Widok *phpPgAdmin* bazy *PostgreSQL*
Fig. 3. *phpPgAdmin* view of *PostgreSQL* database

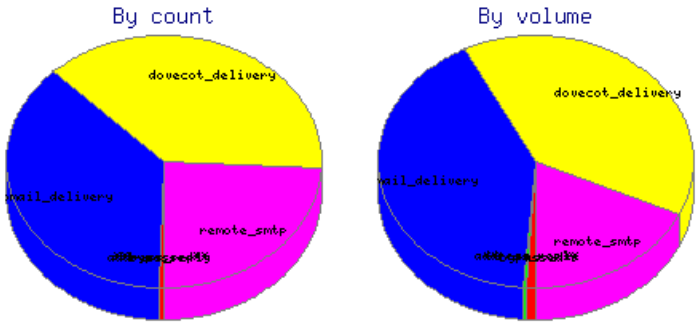
## 2. Wdrożenie i analiza wydajności systemu

System *DBmail* wdrożono w celach testowych w jednostce (wyższa uczelnia) z około 15 tysiącami użytkowników w warunkach produkcyjnych, jako równoległy system pocztowy. W tym celu wykorzystano komputer wyposażony w dwurdzeniowy procesor *AMD64* o mocy 1.86 GHz z pamięcią RAM 2GB, na którym został zainstalowany system operacyjny FreeBSD-7.2-STABLE.

System *DBmail* testowany był w okresie miesiąca równolegle z podstawowym systemem pocztowym. Obydwa systemy pocztowe były równomiernie obciążone, co ilustruje statystyka serwera *Exim* przedstawiona na rys. 4.

**Deliveries by Transport**

| | Volume | Messages |
|---|---|---|
| **bypassed** | 63MB | 1164 |
| address_reply | 15MB | 81 |
| dbmail_delivery | 2586MB | 90915 |
| dovecot_delivery | 2462MB | 93582 |
| remote_smtp | 1137MB | 59259 |




Rys. 4. Porównanie obciążenia systemów w okresie testu
Fig. 4. Comparison of the load during the test of system

System *DBmail-2.2.11* sprzęgnięto z serwerem *MTA Exim* w wersji 4.69 i badano jego zachowanie z bazami danych *Sqlite3 3.6.19*, *PostgreSQL 8.4.2* oraz *MySQL 5.0.89*.

Demony oczekujące na połączenia z *MUA dbmail-imapd* i *dbmailpop3d* oddzielono szyfrującym tunelem *SSL stunnel* w celu podwyższenia stopnia bezpieczeństwa. Sprzężenia systemu z serwerem *MTA Exim* dokonano na poziomie transportu za pomocą protokołu *LMTP* (*Local Mail TransferProtocol*) poprzez lokalne gniazdo uniksowe */var/tmp/dbmail-lmtpd.socket*. W tym celu wykorzystano element systemu *dbmail dbmail-lmtpd* jako tzw. *LDA* (*Local Delivery Agent*) lub *MDA* (*Mail Delivery Agent*). Osiągnięto to przez następującą konfigurację (fragment pliku konfiguracyjnego *Exim*) [4]:

```
localuser_dbmail:
                driver = accept
                transport = dbmail_delivery
                unseen = true

localuser_dovecot:
                driver = accept
                check_local_user
                transport = dovecot_delivery
                cannot_route_message = Unknown user
```

oraz odpowiednie konfiguracje transportów (*Exim*):

```
dovecot_delivery:
                driver = pipe
                command = /usr/local/libexec\
                          /dovecot/deliver
                message_prefix =
                message_suffix =
                log_output
                delivery_date_add
                envelope_to_add
                return_path_add
                group = mail
dbmail_delivery:
                driver = lmtp
                socket = /var/tmp/dbmail-lmtpd.socket
                delivery_date_add
                envelope_to_add
```

```
return_path_add
user = mailnull
group = mail
```

Dla celów testowania systemu, oprócz normalnej pracy z „prawdziwą" pocztą, system był testowany za pomocą programu generującego około 100 wiadomości na sekundę o losowej treści *test-mail.c*.

Przy bardzo dużym obciążeniu (praktycznie nie spotykanym w rzeczywistych warunkach pracy), system *DBmail* zachowywał dużą wydajność przy współpracy z bazą danych *PostgreSQL 8.4.2*, natomiast z bazami *MySQL* i *Sqlite3* następowała blokada bazy po kilkunastu wiadomościach. Serwer *MTA* zwracał tzw. „przejściowy" błąd o kodzie 430 (*transient negative completion reply*), w wyniku czego wiadomość wędrowała do kolejki *Exima*, by w następnym cyklu (tj. po około 30 minutach) zostać poprawnie przetworzona. Ilustrują to odpowiednie fragmenty dzienników systemu (kolejka *Exima*):

```
some-host# exim -Mvl 1NgSt4-0005ZA-CE
          2010-02-14 02:03:02 Received from some-user@some-host
          U=some-user P=local
          S=2277 T="test 0000000100 some-host"
2010-02-14 02:03:04 dbmailuser@some-host
          R=localuser_dbmail
          T=dbmail_delivery defer (-46):
          LMTP error after end of data:
          430 Message not received
```

oraz informacja interfejsu *dbmail-lmtpd*

```
some-host# tail -f /var/log/dbmail.err
Feb 14 02:13:18 some-host dbmail-lmtpd[20390]:
          Error:[sql] dbsqlite.c,db_query(+330):
          sqlite3_get_table failed:
          database is locked
Feb 14 02:13:18 some-host dbmail-lmtpd[20389]:
          Debug:[sql] dbsqlite.c,db_query(+324):
          database locked, retrying...
Feb 14 02:13:18 some-host dbmail-lmtpd[20390]:
          Debug:[db] dbmodule.c,db_query(+145):
          last query took [0] secondo
```

Przy jednoczesnej obsłudze 1024 listów *DBmail* pracujący z bazą danych *PostgreSQL* pozostawił w kolejce niespełna 50 wiadomości, natomiast przy wykorzystaniu *Sqlite3* zakolejkowanych zostało prawie 1000 wiadomości. Wyniki takie zostały osiągnięte przy niemalże identycznym obciążeniu systemu (co ilustrują poniższe fragmenty plików opisujących obciążenie zasobów systemowych):

```
#PostgreSQL
real 191.O69
user 1.34
sys 3.86
      1404  maximum resident set size
        78  average shared memory size
       444  average unshared data size
       114  average unshared stack size
```

```
      260190  page reclaims
           0  page faults
           0  swaps
           3  block input operations
           8  block output operations
           0  messages sent
           0  messages received
           0  signals received
       12011  voluntary context switches
       48243  involuntary context switches

# Sqlite3
real 175.21
user 1.18
sys 3.68
        1404  maximum resident set size
          76  average shared memory size
         436  average unshared data size
         111  average unshared stack size
      260190  page reclaims
           0  page faults
           0  swaps
           0  block input operations
           0  block output operations
           0  messages sent
           0  messages received
           0  signals received
       10989  voluntary context switches
       13388  involuntary context switches
```

## 3. Wnioski

Przedstawiony tu system *DBmail* stanowi funkcjonalne rozwiązanie do zarządzania pocztą użytkowników. Przeprowadzone w środowisku produkcyjnym testy potwierdziły jego skuteczność oraz wydajność przy zachowaniu cech założonych przy projektowaniu systemu.

W systemach poczty typu „korporacyjnego" czy „hostingowego" jako końcowego transportu serwera *MTA* (tzw. *LDA* lub *MDA*) korzystniej jest użyć elementu systemu leżącego „bliżej" serwera odbioru poczty (jak IMAP i POP3), ponieważ ten element rozkłada, indeksuje i przechowuje wiadomości w skrzynkach i folderach użytkowników. Wygenerowane indeksy są potem wykorzystywane przez bliźniaczy element systemu przy przeglądaniu i organizacji folderów pocztowych przez użytkowników. Takie postępowanie znacznie skraca obsługę poczty przez klientów *MUA*. Ponadto taka architektura systemu pozwala w łatwy sposób zbudować bardziej skomplikowaną strukturę klastra pocztowego.

W wyniku przeprowadzonych testów w warunkach bardzo dużego obciążenia pocztą, integracja systemu *DBmail* z silnikiem bazy danych *PostgreSQL* jest zdecydowanie najkorzystniejszym rozwiązaniem. Natomiast jeśli natychmiastowe dostarczenie wiadomości do skrzynki użytkownika nie jest krytyczne, lepszy będzie silnik bazy danych *Sqlite3*, ze względu na bardzo prostą administrację bazą (jeden plik) i wręcz trywialną ewentualną migrację na maszyny o odmiennej architekturze czy inne systemy operacyjne (plik bazy *Sqlite3* jest przenośny). Baza da-

nych *MySQL* ma te same mankamenty, co baza *Sqlite3* (tj. blokowanie się pod dużym obciążeniem), natomiast nie ma tych zalet, co silnik *Sqlite3*, jak chociażby wspomniana łatwość migracji, dlatego jej stosowanie nie wydaje się zasadne.

## BIBLIOGRAFIA

1. Hazel P.: The Exim SMTP Mail Server. UIT Cambridge Ltd., 2007.
2. Wood D.: Programming Internet Mail. O'Reilly, 1999.
3. Hughes L.: Internet e-mail Protocols, Standards and Implementation. Artech House Publishers, 1998.
4. Loshin P.: Essential Email Standards: RFCs and Protocols Made Practical. John Wiley and Sons, 1999.
5. Rhoton J.: Programmer's Guide to Internet Mail: SMTP, POP, IMAP, and LDAP. Digital Press, 1999.
6. Klensin J.: Simple Mail Transfer Protocol (RFC 2821), 2001.

Recenzenci: Dr hab. inż. Andrzej Chydziński, prof. Pol. Śląskiej
Dr inż. Paweł Kasprowski

Wpłynęło do Redakcji 2 stycznia 2009 r.

## Abstract

The paper presents an electronic mail system, in which a set of users, their rights, and storage part is entirely implemented on the database side, rather than in the native file system as in the classic postal systems. Each message is distributed to a number of components specific to e-mail and laden in the relational database schema. The system of two coupled interfaces receive e-mail protocols (for MUA) IMAP and POP3 servers, and two for the MTA, SMTP and LMTP and interfaces to configure the settings and resources management system. The present system, *DBmail* is a project with an open source and has been tested and implemented in an average volume of about 15 thousands of users in production environment.

**Adresy**

Mariusz FRYDRYCH: Wyższa Szkoła Informatyki w Łodzi, Rzgowska 17a, 93-008 Łodź, Uniwersytet Łódzki, Wydział Matematyki i Informatyki, ul. Banacha 22, 90-238 Łódź, Polska, frydrych@wsinf.edu.pl.

Wojciech HORZELSKI: Uniwersytet Łódzki, Wydział Matematyki i Informatyki, ul. Banacha 22, 90-238 Łódź, Polska, horzel@imul.uni.lodz.pl.