Mateusz Gaweł

Zespół Szkół im. ks. S. Staszica w Tarnobrzegu

POCZTA ELEKTRONICZNA, PROTOKÓŁ SMTP PRZYKŁADY KOMUNIKACJI

Streszczenie

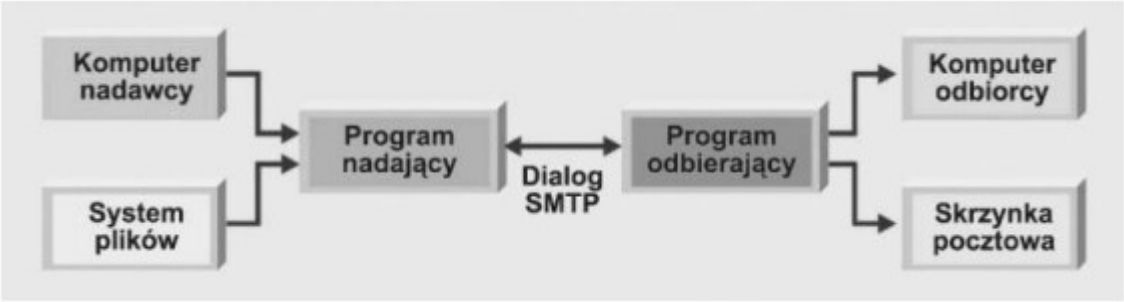
Praca zawiera informacje dotyczące komunikacji z serwerem poczty elektronicznej. Poniżej opisany został protokół SMTP (Simple Mail Transfer Protocol), który używany jest do przekazywania poczty elektronicznej w sieci Internet. W pracy przedstawiono sposoby komunikacji z wykorzystaniem protokołu SMTP oraz zaprezentowano przykład wysłania wiadomości przez terminal oraz skrypt napisany w języku PHP. Ponadto w pracy znaleźć można informacje dotyczące bezpieczeństwa podczas korzystania z protokołu SMTP.

1. WSTĘP

Poczta elektroniczna (e-mail) została wynaleziona w 1965 roku. Początkowo ta usługa służyła jedynie do przesyłania wiadomości między użytkownikami jednego komputera. Już w 1971 roku e-mail został wykorzystany do przesyłania widomości tekstowych między wieloma komputerami. Początkowo poczta była obsługiwana przez protokoły CPYNET, FTP (File Transfer Protocol), UUCP (Unix to Unix Copy). W roku 1982 wprowadzony został protokół SMTP, który używany jest do dzisiaj.[1]

1.1. Protokół SMTP

SMTP to prosty protokół komunikacyjny, który używany jest do przesyłania poczty w Internecie. Do komunikacji używa on prostych poleceń tekstowych. Taka struktura poleceń jest mocną stroną SMTP. Wdrożenie tego protokołu jest proste, ponieważ wykorzystuje on inny powszechny protokół TCP. Możliwe jest tworzenie serwerów pocztowych na każdej platformie. Dzięki temu sieć pocztowa w Internecie jest rozbudowana. Dziś SMTP jest najczęściej używany jako protokół poczty wychodzącej. Szczegółowe informacje o protokole SMTP można znaleźć w normach RFC (Request for Comments).[2]Autoryzacja i bezpieczeństwo



Rysunek 1 - Schemat funkcjonowania SMTP[2]

1.2. Autoryzacja i bezpieczeństwo

SMTP do autoryzacji używa rozszerzenia SMTP AUTH. Pozwala ona na dwie podstawowe metody logowania: AUTH PLAIN i AUTH LOGIN. Każda z tych metod do komunikacji między klientem a serwerem używa kodowania Base64. [3]

- SMTP LOGIN – aby zalogować się tym sposobem, należy osobno przekonwertować nazwę użytkownika i hasło do formatu Base64.
- SMTP PLAIN – to logowanie również wymaga uprzedniego zakodowania danych, tylko, że jako jednego ciągu znaków. Hasło i login powinny być zakodowane w formie: „\0login\0hasło”, gdzie „\0”to bajt zerowy. Ponadto w tym sposobie uwierzytelniania musimy pamiętać, że przed znakami specjalnymi, np. „@”, należy użyć znaku ucieczki „\”.

Jak już wspomniano SMTP do kodowania używa formatu Base64. Nie jest to bezpieczne rozwiązanie. Dlatego do SMTP wprowadzono standard TLS/SSL, który pozwala na zapewnieniu poufności i integralności transmisji danych oraz uwierzytelnieniu serwera i klienta. Standard opiera się na szyfrowaniu asymetrycznym oraz certyfikatach X.509. Standardowy serwer SMTP bez szyfrowania zazwyczaj pracuje na porcie 25, natomiast bezpieczne połączenie nawiązywane jest portem 587. Dodatkowo rozwijane są ulepszone wersje SMTP, zwiększające bezpieczeństwo i funkcjonalność, np. ESMTP.

1.3. Polecenia SMTP

Użytkownik może komunikować się z serwerem SMTP za pomocą poniższych poleceń. Większość z komend do działania potrzebuje wpisania dodatkowego parametru. Połączenie z serwerem SMTP z użyciem poniższych poleceń zostanie opisane w następnych rozdziałach.

- HELO – polecenie, które ma za zadanie „przywitać" serwer SMTP. Bez wpisania tego polecenia, nie możemy korzystać z dobrodziejstw protokołu
- EHLO – polecenie, które działa jak powyższe, z tą różnicą, że po wpisaniu tej komendy uruchamiane są dodatkowe mechanizmy, np. SMTP AUTH;
- AUTH LOGIN/PLAIN – po tej komendzie należy wpisać odpowiednio zakodowane dane logowania (zależne od metody autoryzacji);
- MAIL FROM: – za pomocą tego polecenia ustawiamy adres nadawcy wiadomości;
- RCPT TO: – komenda, którą ustawiamy adres odbiorcy e-maila;
- DATA – po wpisaniu tego polecenia zaczynamy tworzenie treści właściwej wiadomości;
- TO: – ponownie adres odbiorcy (zostanie wyświetlony u odbiorcy wiadomości);
- FROM: – adres nadawcy, który zostanie użyty w podobnym celu jak po wpisaniu adresu po komendzie „TO:"
- SUBJECT: – polecenie, które pozwala na wpisanie tematu wiadomości. Po wpisaniu i wysłaniu komendy „SUBJECT:" można wpisać nagłówki i treść wiadomości. Robimy to czystym tekstem, bez żadnych poleceń. Gdy skończymy pisać naszą wiadomość wpisujemy kropkę w nowej linii. Dla serwera SMTP jest to znak, że nasza wiadomość została skończona. E-Mail zostaje wysłany do adresata;
- QUIT – polecenie, które kończy naszą sesję z serwerem SMTP.

Jeżeli będziemy potrzebowali pomocy, możemy użyć polecenia „HELP".

SMTP rozpoznaje również inne polecenia, jednak te które znajdują się powyżej w zupełności wystarczą do prawidłowego wysłania prostej wiadomości elektronicznej.

1.4. Komunikaty SMTP

Po wysłaniu polecenia do serwera SMTP otrzymujemy od niego odpowiedź. Odpowiedzi najczęściej otrzymujemy pod postacią kodu. Pod poszczególnymi z nich mogą zdefiniowane być informacje, powiadomienia oraz błędy. Poniżej znajdują się wzory (gdzie X, Y, Z to dowolne liczby) dla występujących komunikatów oraz przykłady kodów.[4][5]

- 1YZ – przyjęcie komendy;
- 2YZ – operacja zakończona sukcesem;
- 3YZ – oczekiwanie na następne dane;
- 4YZ – niepoprawne polecenie;
- 5YZ – błąd krytyczny;
- X0Z – błąd składni polecenia;
- X1Z – odpowiedź na polecenie informacyjne;
- X2Z – kody związane z połączeniem;
- X5Z – aktualny stan serwera SMTP.

Teraz czas na dokładne wyjaśnienie kodów SMTP:

- 211 – informacje o systemie lub odpowiedź na polecenie HELP;
- 214 – odpowiedź na polecenie HELP;
- 220 – SMTP jest gotowy;
- 221 – SMTP zakończył połączenie;
- 235 – autoryzacja zakończona powodzeniem;
- 250 – zadanie zakończone sukcesem;
- 251 – użytkownik nie odnaleziony, przekazano do XYZ;
- 334 – komunikaty związane z autoryzacją (prośby o login i hasło);

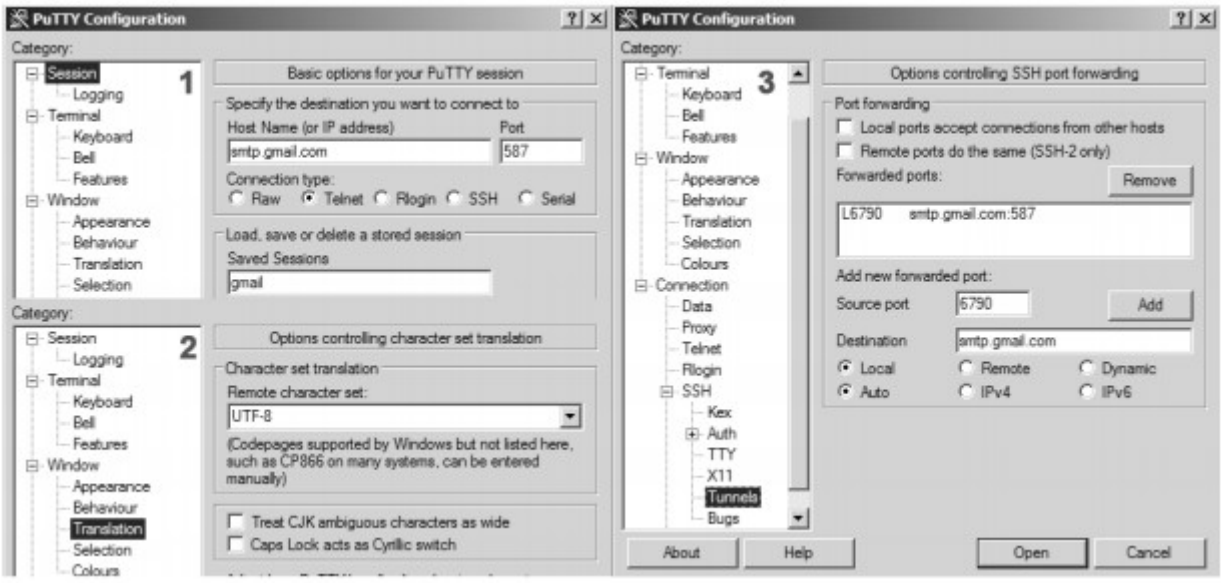
- 354 – wpisz treść maila wraz z tematem, nagłówkami itp. (zakończ kropką w nowej linii);
- 421 – SMTP nie jest gotowy;
- 450 – skrzynka pocztowa niedostępna;
- 451 – wybrane polecenie nie zostało wykonane, błąd w systemie;
- 452 – wybrane polecenie nie zostało wykonane, brak miejsca na dysku;
- 500 – błąd składni, komenda nie rozpoznana;
- 501 – błąd w składni argumentów w poleceniu;
- 502 – nieznana komenda;
- 503 – zła kolejność poleceń;
- 504 – nie wpisano argumentów dla polecenia;
- 521 – serwer SMTP nie akceptuje wiadomości;
- 530 – dostęp zabroniony;
- 550 – akcja nie wykonana, skrzynka niedostępna;
- 551 – użytkownik nie został odnaleziony;
- 552 – polecenie nie zostało wykonane, przekroczony przydział dysku;
- 553 – polecenie nie zostało wykonane, nazwa skrzynki pocztowej niedozwolona;
- 554 – wysłanie wiadomości nie powiodło się.

SMTP jest prostym protokołem, z którym komunikacja odbywa się za pomocą poleceń. Powyższe i wcześniej opisane właściwości przydadzą się w kolejnym rozdziale.

2. PRZYKŁAD WYSŁANIA WIADOMOŚCI

2.1. Telnet

W tym rozdziale pokaże jak wygląda komunikacja użytkownika z serwerem SMTP. Do prezentacji użyję programu Putty oraz serwera SMTP poczty Gmail.com. Ten serwer ze względów bezpieczeństwa używa standardu TLS/SSL. Z tego powodu poczta Google'a używa portu 587.

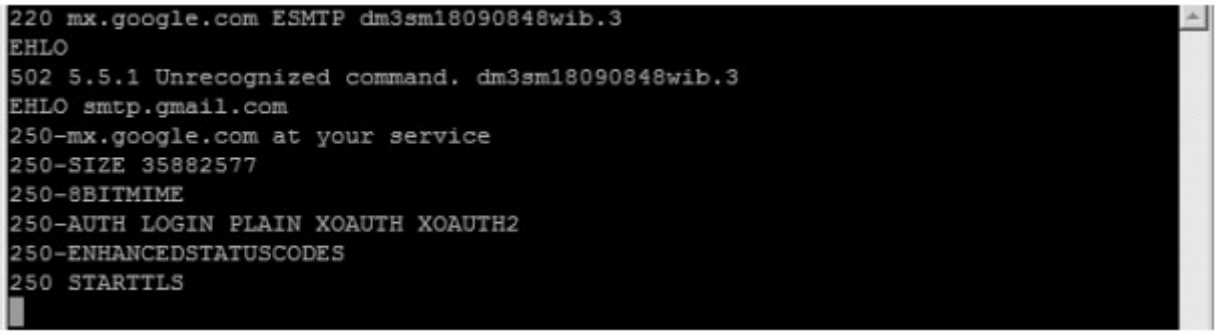


Rysunek 2 - Konfiguracja Putty do połączenia z smtp.gmail.com

Po wprowadzeniu danych możemy połączyć się z serwerem SMTP. Jeśli podane informacje są poprawne powinniśmy ujrzeć ekran podobny do tego poniżej:

Rysunek 3 - Udane połączenie z serwerem SMTP

Można zauważyć, że na ekranie powitalnym pojawił się kod odpowiedzi 220, który oznacza gotowość usługi.Teraz czas na „przywitanie" serwera za pomocą polecenia „EHLO". Jak widać najpierw wpisano komendę bez argumentów, żeby można było zobaczyć kod błędu.



Rysunek 4 - Powitanie serwera SMTP

Teraz można rozpocząć procedurę autoryzacji. W przykładzie użyję metody AUTH LOGIN. Po wpisaniu tego polecenia ukazał się ciąg znaków „VXNlcm5hbWU6". Jest to zakodowany w Base64 tekst „Username:". Nazwę użytkownika poczty również musimy wprowadzić w tym kodowaniu. W tym wypadku jest to „Z29ybmlrc210cEBnbWFpbC5jb20=" (zakodowany tekst „gorniksmtp@gmail.com").

Rysunek 5 - Logowanie AUTH LOGIN

Po wprowadzeniu nazwy logowania pojawia się prośba o hasło („Password:", w Base64: „UGFzc3dvcmQ6"). Hasło użyte w tym przykładzie to zakodowane „gorniksmtp" („Z29ybmlrc210cA==" w Base64). Po udanym logowaniu otrzymujemy komunikat o powodzeniu autoryzacji wraz z odpowiednim kodem.

Rysunek 6 - Poprawne logowanie

Teraz czas kolejną część tego przykładu – określenie nadawcy i odbiorcy. Otrzymujemy kolejne komunikaty potwierdzające wykonanie akcji.

Rysunek 7 - Określenie nadawcy i odbiorcy

Gdy mamy ustalone dane adresowe możemy zacząć tworzyć wiadomość (polecenie „DATA"). Najpierw po poleceniu „SUBJECT:" wpisujemy temat wiadomości, następnie adres e-mail odbiorcy („TO:"), który ma wyświetlić się po odebraniu wiadomości. Ostatnim krokiem przed wpisaniem treścią właściwej wiadomości jest umieszczenie nagłówków. Przykładowo umieszczę nagłówek, który pozwoli na wysłanie wiadomości z polskimi znakami. Teraz można zacząć komponowanie swojego e-maila. Po zakończeniu tworzenia wiadomości przechodzimy do nowej linii i wpisujemy kropkę, którą również zatwierdzamy. E-mail powinien zostać wysłany.

```
DATA
354  Go ahead dm3sm18090848wib.3
SUBJECT: TEST SMTP
TO: <gorniksmtp@gmail.com>
Content-type: text/html; charset=utf-8
Witaj, właśnie przeprowadzam test SMTP.
.
250 2.0.0 OK 1348511888 dm3sm18090848wib.3
```

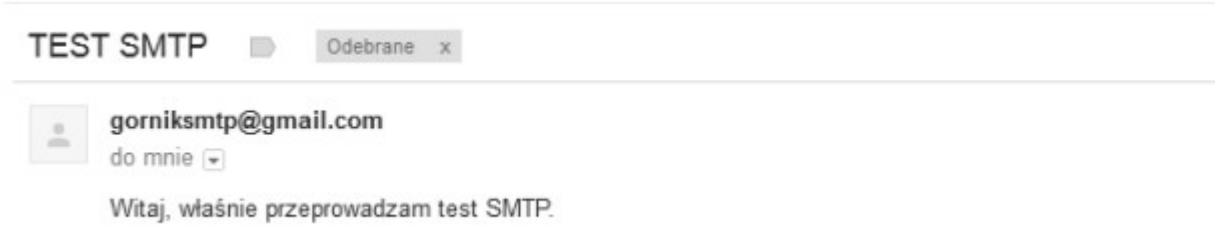
Rysunek 8 - Tworzenie wiadomości oraz jej wysłanie

Teraz zostało już nam tylko zakończenie połączenia. Po udanym wysłaniu wiadomości wpisujemy polecenie „QUIT”:

```
QUIT
```

Rysunek 9 - Kończenie pracy z serwerem SMTP

Teraz możemy zobaczyć naszą wiadomość:



Rysunek 10 - Wiadomość wysłana przez SMTP

2.2. Sniffowanie SMTP z TLS/SSL

SMTP bez szyfrowania nie jest bezpieczne. Wykorzystanie standardu TLS/SSL do zabezpieczenia tego protokołu pozwala na bezpieczną transmisję danych. W tym podrozdziale do prezentacji użyto dwóch programów – Klienta pocztowego oraz programu do sniffowania sieci Wireshark. O to co udało się podsłuchać:

Source	Destination	Protocol	Length	Info
173.194.70.109	192.168.2.2	SMTP	96	S: 220 mx.google.com ESMTP v3sm4409341wiy.5
192.168.2.2	173.194.70.109	SMTP	65	C: EHLO BALU
173.194.70.109	192.168.2.2	SMTP	178	S: 250-mx.google.com at your service | 250-SIZE 35882577 | 250-8BITMIME |
192.168.2.2	173.194.70.109	SMTP	64	C: STARTTLS
173.194.70.109	192.168.2.2	SMTP	84	S: 220 2.0.0 Ready to start TLS
173.194.70.109	192.168.2.2	SMTP	97	S: 220 mx.google.com ESMTP bn7sm4399346wib.8
192.168.2.2	173.194.70.109	SMTP	65	C: EHLO BALU
173.194.70.109	192.168.2.2	SMTP	178	S: 250-mx.google.com at your service | 250-SIZE 35882577 | 250-8BITMIME |
192.168.2.2	173.194.70.109	SMTP	64	C: STARTTLS
173.194.70.109	192.168.2.2	SMTP	84	S: 220 2.0.0 Ready to start TLS

Rysunek 11 - Wiadomość wysłana przez klienta pocztowego

Jak widać no powyższym zrzucie ekranu śledzenie pakietów kończy się przy komunikacie „Ready to start TLS”. Szyfrowanie SMTP spełnia swoje zadanie, ponieważ żadne prywatne dane po stronie serwera nie zostały ujawnione.

Uruchamiając swój własny lub wybierając zewnętrzny serwer SMTP, warto zadbać o bezpieczeństwo. Należy pamiętać o tym, żeby posiadał on moduł szyfrowania TLS/SSL. W innym przypadku administratora serwera lub nawet klienta może spotkać niespodzianka w postaci zdobycia przez osobę niepowołaną dostępu do prywatnych wiadomości.

3. PHP

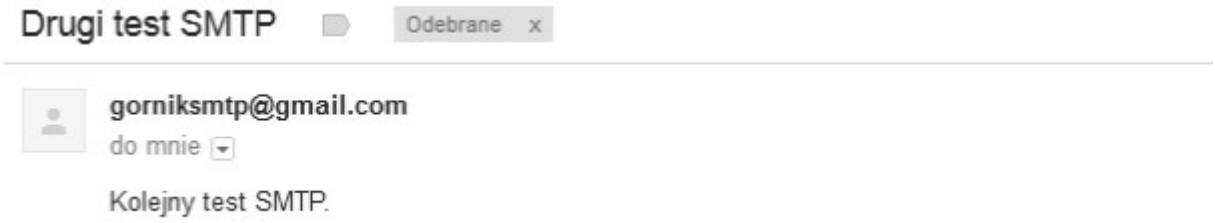
W tym rozdziale zostanie zaprezentowany przykład skryptu w języku PHP, który ma za zadanie wysłać e-maila poprzez protokół SMTP. Wykorzystane zostaną funkcje: fsockopen, fwrite oraz fclose. Więcej o nich można dowiedzieć się na oficjalnej stronie języka PHP.

1
```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=UTF-8" />
</head>
<body>
<?php
// Sprawdzanie odpowiedzi serwera
function ChechResponse($conn, $expected_response) {
    $response = '';
    while (substr($response, 3, 1) != ' ') {
        if (!($response = fgets($conn, 256))) {
        echo('Nie można pobrac odpowiedzi.');
        }
    }
    if (!(substr($response, 0, 3) == $expected_response)) {
    echo('Nie można wysłać wiadomości, błąd: '.$response.'.');
    }
}
// Kody informacyjne
define('READY', '220');
define('SUCCESS', '250');
define('WAITING_FOR_LOGIN', '334');
define('LOGIN_SUCCESS', '235');
define('WAITING_FOR_DATA', '354');
// Dane logowania
$host = 'ssl://smtp.gmail.com';
$port = 465;
$user = 'gorniksmtp@gmail.com';
$pass = 'gorniksmtp';
```

2
```
// Adres nadawcy i odbiorcy
$from = 'gorniksmtp@gmail.com';
$to = 'gorniksmtp@gmail.com';
// Właściwa część skryptu
$conn = fsockopen($host, $port, $errno, $errstr, 15);
ChechResponse($conn, READY);
fwrite($conn, 'EHLO '.$host."\r\n");
ChechResponse($conn, SUCCESS);
fwrite($conn, 'AUTH LOGIN'."\r\n");
ChechResponse($conn, WAITING_FOR_LOGIN);
fwrite($conn, base64_encode($user)."\r\n");
ChechResponse($conn, WAITING_FOR_LOGIN);
fwrite($conn, base64_encode($pass)."\r\n");
ChechResponse($conn, LOGIN_SUCCESS);
fwrite($conn, 'MAIL FROM: <'.$from.'>'."\r\n");
ChechResponse($conn, SUCCESS);
fwrite($conn, 'RCPT TO: <'.$to.'>'."\r\n");
ChechResponse($conn, SUCCESS);
fwrite($conn, 'DATA'."\r\n");
ChechResponse($conn, WAITING_FOR_DATA);
fwrite($conn, 'SUBJECT: Drugi test SMTP'."\r\n");
fwrite($conn, 'TO: <'.$to.'>'."\r\n");
fwrite($conn, 'Content-type: text/html; charset=utf-8'."\r\n");
fwrite($conn, 'Kolejny test SMTP.'."\r\n");
fwrite($conn, '.'."\r\n");
ChechResponse($conn, SUCCESS);
fwrite($conn, 'QUIT'."\r\n");
fclose($conn);
?>
</body>
</html>
```

Rysunek 12 - Przykładowy skrypt w PHP służący do wysyłania wiadomości

Jak widać, skrypt do komunikacji używa poleceń SMTP. Po każdym poleceniu sprawdzany jest kod odpowiedzi. Dzięki temu można sprawdzić, czy podczas obsługi SMTP nie pojawiły się żadne błędy. Poniżej znajduje się efekt wysłania wiadomości tym skryptem:



Rysunek 13 - Zrzut ekranu z odebraną wiadomością

SMTP to protokół, który charakteryzuje się tym, że można go obsłużyć wieloma sposobami. Dzięki temu SMTP może być stosowany w najróżniejszych projektach, które tworzą programiści.

4. ZAKOŃCZENIE

SMTP to protokół, który doskonale nadaje się do transportowania poczty. SMTP jest bardzo prosty w użyciu, dzięki prostym komendom i odpowiedziom. Serwer można uruchomić praktycznie wszędzie, co wpływa na rozwinięcie struktury poczty w Internecie. Ponadto SMTP po odpowiedniej konfiguracji staje się bezpiecznym protokołem, którego podsłuchanie nie jest możliwe.

BIBLIOGRAFIA

[1]*Poczta elektroniczna*, Wikipedia.org, 2012
[2]*SMTP*, itpedia.pl, 2008.
[3]*SMTP-AUTH*, Wikipedia.org, 2012
[4]*Wzory kodów zwracanych przez SMTP, uw-team.org.*
[5]SMTP Replies, Greenend.org.uk

ELECTRONIC MAIL, SMTP PROCOL – EXAMPLES OF COMMUNICATION

Summary

The paper contains information about communication with e-mail server. The paper describes SMTP (Simple Mail Transfer Protocol), which is used to transfer electronic mail on the Internet. It's presents ways to communicate using the SMTP protocol, and is an example to send a message. In addition, the work provides information about safety when using the SMTP protocol.