Bab 9

E-mail

Meskipun telah muncul protokol dan tren yang lebih baru seperti Web dan IM (instant messaging), e-mail tetap dipandang banyak orang sebagai the real killer app of the Internet. Kesederhanaan konsepnya, fleksibilitasnya, dan kemudahan menerima dan mengirimkannya, membuat email tetap menjadi salah satu aplikasi terpenting bagi setiap orang. Dukungan PHP terhadap email memang sederhana dan hanya diwakili oleh sebuah fungsi, `mail()`. Tapi terdapat library dan tool yang bisa membantu programer PHP mengelola e-mail. Bab ini berisi resep-resep tugas umum yang sering Anda lakukan terhadap e-mail.

Resep 9-1: Mengirim E-mail

```
mail($to, $subject, $message[, $additional_headers[,
$additional_parameters]]);
```

Mengirim e-mail di PHP sangat mudah, cukup menggunakan satu fungsi saja. (Tapi terus terang, saya selalu lupa dengan urutan parameter fungsi yang satu ini, *penulis*). PHP otomatis dapat memilih mengirimkan e-mail menggunakan program eksternal seperti `/usr/sbin/sendmail` yang default di Linux dan Unix atau langsung ke port SMTP yang default di Windows.

`$additional_headers` adalah sebuah string yang terdiri dari satu atau lebih baris header tambahan yang ingin disebutkan. Biasanya yang paling sering disebutkan di sini adalah header `From` atau `Reply-To`. `$additional_parameters` berguna untuk mengirim opsi ke program `/usr/sbin/sendmail` atau yang setara. Biasanya di Windows tidak pernah kita pakai. Karena defaultnya di Windows `mail()` menggunakan SMTP, Anda perlu mengeset dulu di php.ini konfigurasi `SMTP` dan `SMTP_port`. Kalau Anda tidak memiliki/memasang SMTP sendiri, Anda perlu menggunakan SMTP ISP Anda. Contoh mengirim sebuah e-mail sederhana dalam sebuah form umpan balik:

```
 1|<?
  |
 3|//
 4|// feedback-form-sederhana.php
 5|//
  |
 7|$TO = "steven@steven.builder.localdomain";
  |
 9|echo "<h2>Feedback Form</h2>";
  |
11|if (isset($submit)) {
  |
13|  if (!$name) die("Mohon masukkan nama Anda!");
14|  if (!$e-mail) die("Mohon masukkan alamat e-mail Anda!");
15|  if (!$subject) die("Mohon isi subjek!");
16|  if (!$message) die("Mohon isi pesan!");
  |
18|  if (!mail($TO, $subject, $message,
19|            "From: \"$name\" <$e-mail>\n".
```



```
20|            "X-IP-Address: $_SERVER[REMOTE_ADDR]\n".
21|            "X-Referer: $_SERVER[HTTP_REFERER]\n"
22|           ))
23|    die("Maaf, gagal mengirim pesan. Mungkin bisa dicoba
   lagi?");
  |
25|  echo "<b>Terima kasih atas masukan Anda!</b>";
  |
27|} else {
  |
29|  echo "
  |
31|  <form method=POST>
32|    <table border=0 cellpadding=4>
33|      <tr><td>Nama Anda</td><td><input name=name></td></tr>
34|      <tr><td>Email Anda</td><td><input name=e-mail></td></tr>
35|      <tr><td>Subjek Pesan</td><td><input name=subject
   size=50></td></tr>
36|      <tr><td>Pesan</td>
37|          <td><textarea cols=50 rows=8 name=message></
   textarea></td></tr>
38|      <tr><td> </td><td><input type=submit
   name=submit></td></tr>
39|  </form>
  |
41|  ";
  |
43|}
  |
45|?>
```

Perhatikan format argumen keempat (additional headers) di baris 19-22. Setiap header harus dipisahkan dengan karakter newline (`"\n"` atau `"\r\n"`) Sedangkan di Windows kemungkinan Anda harus menggunakan `"\r\n"` agar e-mail dapat diterima SMTP server dengan baik. Kita mengirimkan tiga header tambahan. Header From adalah header standar, sementara header `X-IP-Address` dan `X-Referer` adalah buatan sendiri—standar Internet memang memberikan jatah prefiks X- sebagai tempat untuk header-header buatan user. Pada contoh resep di atas kita merekam alamat IP klien dan header HTTP `Referer`-nya. Ini lazim dilakukan untuk memudahkan pelacakan.

Resep 9-2: Mengirim E-mail dengan Prioritas Tertentu

Untuk mengirim e-mail yang berlabel prioritas bukan default (normal), misalnya penting, Anda dapat menambahkan dua header. Yakni `X-Priority` dan `X-MSMail-Priority` untuk program e-mail Microsoft. Contohnya:

```
<?
mail("steven@masterwebnet.com",
     "PENTING: server down!",
     "", // kosong, kita hanya menggunakan subjek saja, mirip SMS :-)
     "X-Priority: 2 (high)\n".
     "X-MSMail-Priority: High\n");
?>
```

Nilai yang dapat dipilih untuk `X-Priority`:

```
1 (highest)
2 (high)
3 (normal)
4 (low)
5 (lowest)
```

Nilai yang dapat dipilih untuk `X-MSMail-Priority`:

```
High
Normal
Low
```

Resep 9-3: Mengirim E-mail dengan Identitas Pengirim Lain

Jika Anda mengirim e-mail tidak hanya kepada mailbox seseorang, tapi juga kepada program seperti program MLM (mailing list manager) atau ke server dan alamat lain tertentu, maka mungkin Anda mengalami bahwa meskipun header sudah diset `From` di argumen keempat fungsi `mail()`, namun e-mail ditolak program MLM. Ini dikarenakan adanya dua macam "alamat From" yang berbeda. Yang diset dengan `"From: ..."` di argumen keempat `mail()` disebut dengan "RFC-822 From" atau

"RFC-822 Sender". Ini adalah From yang terlihat di tampilan program e-mail. Namun ada alamat From lain yang disebut "envelope sender" atau "envelope From" dan juga disebut sebagai "Return-Path" atau alamat tujuan bounce. Alamat From kedua ini biasanya dicatat oleh program pengirim e-mail (MTA, mail transfer agent) di header `X-Sender`.

Enveloper sender dipakai untuk routing e-mail dan untuk mengetahui ke mana sebuah e-mail yang mental dikembalikan.

Beberapa program seperti ezmlm membutuhkan envelope sender yang benar, bukan RFC-822 From. Bahkan RFC-822 From tidak diperlukan sama sekali olehnya. Jadi jika membuat form untuk subscribe ke milis berbasis ezmlm, caranya adalah:

```
 1|<?
  |
 3|//
 4|// form-subscribe.php
 5|//
  |
 7|$TO = "antarkita-subscribe@yahoogroups.com";
  |
 9|echo "<h2>Subscribe ke milis antarkita, milisnya kita-kita!</
  h2>";
  |
11|if (isset($submit)) {
  |
13|  if (!$e-mail) die("Mohon masukkan alamat e-mail Anda!");
  |
15|  if (!mail($TO, "susbcribe", "", "Return-Path: <$e-mail>\n"))
16|    die("Maaf, gagal mengirim e-mail. Mungkin bisa dicoba
  lagi?");
  |
18|  echo "<b>Email konfirmasi akan segera terkirim ke e-mail
  Anda!</b>";
  |
20|} else {
  |
22|  echo "
  |
24|  <form method=POST>
25|    Email Anda: <input name=e-mail>
```

```
26|    <input type=submit name=submit>
27|  </form>
  |
29|  ";
  |
31|}
  |
33|?>
```

Perhatikan di baris 15, kita mengirimkan `Return-Path`, bukan header `From`. Header `Return-Path` nanti akan dipakai sebagai envelope sender. Anda tentu bisa saja mengeset kedua-duanya; tapi sebetulnya antara kedua From ini tidak berhubungan dan tidak perlu sama.

Resep 9-4: Mengirim E-mail HTML

Email HTML sering disalahkan sebagai biang kerok penyebar worm; dan di milis-milis tertentu kehadirannya sama sekali tidak diharapkan. Tapi sebetulnya e-mail HTML berguna dalam beberapa kasus, misalnya bagi orang-orang awam yang menginginkan agar bisa menyertakan berbagai warna/tampilan teks, ketimbang harus belajar smiley dan notasi-notasi seperti `*cetak tebal*` atau `_cetak miring_`. Atau bagi departemen marketing saat ingin mengirim e-mail yang mengandung link gambar, logo perusahaan, dan lain sebagainya.
Untuk mengirim e-mail HTML atau attachment menggunakan `mail()`, Anda perlu mengerti tentang struktur dokumen MIME (Multipurpose Internet Mail Extensions). Untung saja, dengan paket PEAR seperti `Mail_Mime` Anda tidak perlu repot memahami hal ini. Resep berikut menggunakan paket dari PEAR yang cara instalasinya telah dijelaskan di Bab Pendahuluan.

```
 1|<?
  |
 3|//
 4|// e-mail-html.php
 5|//
  |
 7|include('Mail.php');
 8|include('Mail/mime.php');
  |
```



```
10|$TO = "antarkita@citramas.co.id";
11|$FROM = "santi@citramas.co.id";
  |
13|// versi teks pesan, bagi yang tidak bisa membaca
14|$message_text =
  |
16|  "Jangan sampai pada lupa ya? Acaranya *Kamis* tanggal 20,
   bukan Rabu.".
17|  "Jam 20.00 - selesai. Sampai ketemu di lantai 7.";
  |
19|// versi teks HTML
20|$message_html =
  |
22|  "<html><body><p>Jangan sampai pada lupa ya? Acaranya <font
23|  color=red><b>Kamis</b></font> tanggal 20, bukan Rabu. Jam
   20.00 - selesai.
24|  Sampai ketemu di lantai 7.</body></html>";
  |
26|$headers = array(
27|  'From'    => $FROM,
28|  'Subject' => 'Rapat panitia seminar'
29|);
  |
31|// pertama bentuk MIME
32|$mime = new Mail_mime("\n");
  |
34|$mime->setTXTBody($message_text);
35|$mime->setHTMLBody($message_html);
  |
37|$body = $mime->get();
38|$header = $mime->headers($headers);
  |
40|$mail =& Mail::factory('mail');
41|$mail->send($TO, $header, $body);
  |
43|?>
```

MIME adalah kumpulan satu atau lebih bagian atau part. Tiap bagian dapat berupa teks, gambar, atau attachment file apa saja yang di-encoding. Pada resep di atas kita mula-mula membuat objek yang mewakili dokumen MIME di baris 32. Setelah itu menambahkan tiap bagian. Mula-mula dokumen teks polos (baris 34), lalu pesan

versi HTML-nya (baris 35). Terakhir kita meminta hasil dokumen MIME-nya di baris 37. Deretan header pun kita minta dari objek MIME. Selain header-header di baris 26-29, objek $mime juga akan mencetakkan header yang diperlukan untuk menandai bahwa e-mail kita berupa MIME. Terakhir, kirim e-mail ini.

Resep 9-5: Mengirim Attachment

Sama dengan di resep sebelumnya, attachment pun dikirimkan dalam bentuk satu atau lebih part MIME. Resep di bawah adalah form yang mengizinkan seorang pengunjung meng-upload file via browser untuk dikirimkan ke pembuat skrip lewat attachment. Salah penerapannya ketika seorang dosen kuliah menggunakan form seperti ini untuk mengumpulkan tugas para mahasiswanya.

```
 1|<?
  |
 3|//
 4|// e-mail-attachment.php
 5|//
  |
 7|include('Mail.php');
 8|include('Mail/mime.php');
  |
10|$TO = "dosen@kampus.com";
  |
12|echo "<h1>Form Upload tugas mata kuliah XY-123</h1>";
  |
14|if (isset($submit)) {
  |
16|  if (!$name) die("Masukkan nama Anda!");
17|  if (!$e-mail) die("Masukkan alamat e-mail Anda!");
  |
19|  $headers = array(
20|    'From'    => "\"$name\" <$e-mail>",
21|    'Subject' => "Tugas mata kuliah XY-123 (NIM=$nim)",
22|    'X-IP'    => $_ENV['REMOTE_ADDR']
23|  );
  |
25|  $path = $_FILES['file1']['tmp_name'];
26|  $fp = fopen($path, "rb");
27|  $content = fread($fp, filesize($path));
```



```
  |
29|  $mime = new Mail_mime("\n");
30|  $mime->setTXTBody("Komentar mahasiswa:\n\n" . $comment);
31|  $mime->addAttachment($content,
32|                       $_FILES['file1']['type'],
33|                       $_FILES['file1']['name'],
34|                       0);
35|  $body = $mime->get();
36|  $header = $mime->headers($headers);
37|  $mail =& Mail::factory('mail');
38|  $mail->send($TO, $header, $body);
  |
40|  echo "Terima kasih telah mengumpulkan tugas.";
  |
42|} else {
  |
44|  echo '
  |
46|  <form enctype="multipart/form-data" method="POST">
47|  <p>Nama Lengkap:<br><input name=name>
48|  <p>Email:<br><input name=e-mail><br>
49|  <p>NIM:<br><input name=nim><br>
50|  <p>File (sebaiknya dalam format .ZIP):<br>
51|     <input name="file1" type="file">
52|  <p>Keterangan/catatan/komentar Anda:<br>
53|     <textarea cols=50 rows=4 name=comment></textarea>
  |
55|  <p><input type="submit" name="submit" value="Kirim">
56|  </form>
  |
58|  ';
  |
60|}
  |
62|?>
```

Perhatikan baris 31 yaitu saat menambahkan sebuah part attachment ke dalam objek MIME. Metode `addAttachment` menerima empat argumen:

```
$mime->addAttachment($filename_or_content, $type, $name,
$is_filename);
```

Di argumen pertama, dapat dipilih sebutan path ke sebuah file di mana nanti metode ini yang membaca filenya. Atau bisa juga menyebutkan isi file itu sendiri di mana nanti di argumen akan disebutkan nama file yang diinginkan. Pada resep di atas kita menggunakan pilihan kedua. Argumen keempat kita set 0, yang berarti argumen pertama kita bukanlah nama file melainkan isi file.

Tentu saja jika resep ini digunakan untuk tujuan nyata, perlu ditambahkan beberapa hal penting seperti otentikasi (tidak boleh sembarangan orang mengirimkan tugas atas nama orang lain), pembatasan ukuran file (kalau tidak, mailbox si dosen bisa penuh!), pembatasan lainnya (misalnya, mahasiswa yang sama dilarang mengupload tugas yang sama hingga lebih dari 2 kali, dan sebagainya.

Resep 9-6: Mengirim E-mail yang Dienkripsi

Sering terjadi sebuah situs ditaruh di alamat `https://` yang aman. Pengunjung cukup yakin berbelanja atau mengirimkan informasi yang sensitif, seperti kartu kredit, karena berpikir ada enkripsi SSL. Tapi ternyata apa yang terjadi? Setelah diterima oleh skrip di server, informasi sensitif ini dikirim ke webmaster atau pemilik situs dengan e-mail biasa lewat SMTP tanpa dienkripsi dulu. Dengan kata lain, enkripsi SSL untuk situs bisa menjadi sia-sia karena saat mengalir dari server ke mailbox pemilik situs, informasi tetap bisa disadap atau di-sniff pihak ketiga. Meskipun mailbox berada di server yang sama, bisa saja pemilik server hosting yang nakal mengintip isinya sebelum mail diambil pemilik situs.

Salah satu cara melindungi data adalah dengan enkripsi public key. Sebagai pemilik situs, Andi pertama membuat sepasang kunci, yang satu disebut kunci publik dan satu lagi kunci privat. Kunci publik hanya dapat dipakai untuk mengenkripsi (mengunci) data, dan Anda taruh kunci ini di server. Kunci privat hanya dapat dipakai untuk mendekripsi (membuka kunci) data yang sebelumnya dienkripsi oleh kunci publik pasangannya. Kunci yang satu ini Anda taruh baik-baik di komputer pribadi dan diproteksi password.

Jadi, jika data yang diterima dari form langsung Anda enkripsi, maka selama berkelana di mail server dan bertengger di mailbox server, data tidak bisa dibaca. Data baru bisa dibaca setelah sampai ke komputer pribadi Anda sendiri dan dibuka dengan kunci privat. Tentunya kunci privat harus dijaga baik-baik dan tidak boleh diberikan pada siapapun. Sebaliknya kunci publik boleh-boleh saja diintip, diambil, dikopi oleh orang lain.

GPG atau **GnuPG** (Gnu Privacy Guard) adalah salah satu program gratis yang bisa dipakai untuk melakukan enkripsi public key. Anda dapat meng-install GPG di Windows maupun Unix. Programnya bia diambil dari CD buku ini atau dari www.gnupg.com. Di CD disediakan source code dalam bentuk RPM, Anda dapat membuat RPM binernya dengan perintah:

```
rpmbuild –rebuild <nama-source-rpm>
```

Beberapa subsubbab ke depan akan membahas dulu dasar-dasar penggunaan GPG.

Membuat kunci

Setelah instalasi GPG selesai, langkah pertama menggunakannya adalah membuat sepasang kunci. Lakukan ini di komputer pribadi Anda, jangan di server:

```
$ gpg –gen-key

gpg (GnuPG) 1.0.7; Copyright (C) 2002 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

 Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) ElGamal (sign and encrypt)
   (5) RSA (sign only)
 Your selection? 1
DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
              minimum keysize is  768 bits
              default keysize is 1024 bits
    highest suggested keysize is 2048 bits
What keysize do you want? (1024) 2048
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 0
```

```
Key does not expire at all
Is this correct (y/n)? y

You need a User-ID to identify your key; the software constructs
the user id
from Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Steven Haryanto
Email address: steven@masterwebnet.com
Comment: kunci contoh
You selected this USER-ID:
    "Steven Haryanto (kunci contoh) <steven@masterwebnet.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
You need a Passphrase to protect your secret key.
Enter passphrase:
Repeat passphrase:

We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
++++++++++++++++++++++++++++++++++++++++.++++++++++++++++++++..+++++++++
++++++++++++++++.++++++++++++++++..++++++++++..++++++++++++++++++++..++++
+.....>+++++..>++++++<.+++++.........................+++++
We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
..++++++++++.+++++.+++++..++++++++++.+++++.++++++++++++++++++++.+++++.+++
++++++++++++++++.+++++.++++++++++.++++++++++++++++.++++++++++.++++++++++++
++++++++++++++>.++++++++++..>.+++++......>.+++++.......................
.......+++++^^^
gpg: /home/steven/.gnupg/trustdb.gpg: trustdb created
public and secret key created and signed.
key marked as ultimately trusted.

pub  1024D/B3EC3C85 2003-12-06 Steven Haryanto (kunci contoh)
<steven@masterwebnet.com>
     Key fingerprint = F1F5 4CE5 88A9 B7EF A42O  4333 68E4 648C
B3EC 3C85
sub  2048g/E4193A54 2003-12-06
```


Kunci yang baru dibuat akan disimpan di file `$HOME/.gnupg/pubring.gpg` (untuk kunci publik) dan `$HOME/.gnupg/secring.gpg` (untuk kunci privat). Di Windows lokasi defaultnya adalah di `c:\gnupg\pubring.gpg` dan `c:\gnupg\secring.gpg`. Kedua file ini adalah semacam "gantungan kunci" tempat menyimpan kunci-kunci.

Menaruh kunci publik di server

Langkah selanjutnya adalah mengekstrak atau mengambil kunci publik untuk ditaruh di server. Di komputer pribadi:

```
$ gpg –output steven.gpg –export steven@masterwebnet.com
```

Lalu kopikan `steven.gpg` ke server. Di server yang juga harus telah di-install GPG:

```
$ gpg –import steven.gpg
```

Untuk mengecek hasil impor kunci publik ke pubring.gpg di server, ketikkan:

```
$ gpg –list-keys
/home/steven/.gnupg/pubring.gpg
————————————
pub  1024D/B3EC3C85 2003-12-06 Steven Haryanto (kunci contoh)
<steven@master
webnet.com>
sub  2048g/E4193A54 2003-12-06

$ gpg –list-keys –with-colons
/home/steven/.gnupg/pubring.gpg
————————————
pub:u:1024:17:68E4648CB3EC3C85:2003-12-06:::u:Steven  Haryanto
(kunci contoh)
<steven@masterwebnet.com>::scESC:
sub:u:2048:16:28C41CODE4193A54:2003-12-06::::::e:
```

Perhatikan output perintah pada list-keys kedua yang menggunakan opsi with-colons. Catat keyID untuk digunakan nanti. Pada contoh di atas, keyID adalah `68E4648CB3EC3C85`.

Mengenkripsi pesan di command line

Misalnya kita memiliki sebuah pesan teks di file `pesan.txt` yang ingin kita enkripsi. Di prompt shell, lakukan sebagai berikut:

```
$ gpg –output pesan.gpg –encrypt –recipient
steven@masterwebnet.com pesan.txt
```

Perintah di atas akan menghasilkan file biner `pesan.gpg` yang dienkrip. Pesan hanya dapat dibuka oleh penerima (recipient), yaitu steven@masterwebnet.com. Atau setiap orang yang memiliki kunci privat steven@masterwebnet.com.

Mengenkripsi pesan e-mail di PHP

Resep di bawah sama-sama menggunakan program `gpg`, tapi kita memberikan opsi `–trusted-keys` agar gpg tidak perlu bertanya soal identitas kunci. Serta opsi `–armor` agar output hasil enkripsi tidak berupa biner tapi di-encode sebagai teks yang dapat dibaca. Ini agar hasil enkripsi tidak perlu ditaruh sebagai attachment, tapi cukup di body e-mail.

```
 1|<?
  |
 3|//
 4|// e-mail-gpg.php
 5|//
  |
 7|$USER = "steven"; # skrip berjalan sebagai user siapa?
 8|$HOME = "/home/steven"; # di mana home directory account
  $USER?
 9|$RECIPIENT = "steven@masterwebnet.com";
  |
11|echo "<h2>Form Pesan Buku 'Resep PHP'</h2>";
  |
13|if (isset($submit)) {
  |
15|  if (!$name) die("Mohon masukkan nama Anda!");
16|  if (!$e-mail) die("Mohon masukkan alamat e-mail Anda!");
17|  if (!$address) die("Mohon masukkan alamat Anda!");
18|  if (!$city) die("Mohon masukkan kota Anda!");
19|  if (!$zip) die("Mohon masukkan nomor kodepos Anda!");
20|  if (!$amount) die("Mohon masukkan jumlah yang dipesan!");
  |
22|  // susun pesan yang ingin dienkripsi
```



```
23|  $message =
24|    "Nama: $name\n".
25|    "Email: $e-mail\n".
26|    "Alamat: $address\n".
27|    "Kota/kodepos: $city/$zip\n".
28|    "Jumlah: $amount\n";
  |
30|  // panggil gpg untuk mengenkripsi
31|  $cmd = "echo ".escapeshellarg($message)." | USER=$USER
  HOME=$HOME ".
32|    "gpg –quiet –no-secmem-warning –encrypt –armor ".
33|    "–recipient $RECIPIENT";
34|  $message_e = '$cmd';
  |
36|  if (!mail($RECIPIENT, 'Pesanan buku', $message_e,
37|            "From: \"$name\" <$e-mail>\n".
38|            "X-IP-Address: $_SERVER[REMOTE_ADDR]\n".
39|            "X-Referer: $_SERVER[HTTP_REFERER]\n"
40|           ))
41|    die("Maaf, gagal mengirim pesan. Mungkin bisa dicoba
  lagi?");
  |
43|  echo "<b>Terima kasih atas pesanan Anda!</b>";
  |
45|} else {
  |
47|  echo "
  |
49|  Harga per buku Rp 40.000,-
  |
51|  <form method=POST>
52|    <table border=0 cellpadding=4>
53|      <tr><td>Nama Anda</td><td><input name=name></td></tr>
54|      <tr><td>Email Anda</td><td><input name=e-mail></td></tr>
55|      <tr><td>Alamat</td><td><input name=address size=50></
  td><     /tr>
56|      <tr><td>Kota</td><td><input name=city></td></tr>
57|      <tr><td>Kodepos</td><td><input name=zip size=6></td></tr>
58|      <tr><td>Jumlah</td><td><input name=amount size=3
  value=1></td></tr>
59|      <tr><td> </td><td><input type=submit
```

```
    name=submit></td></tr>
 60|  </form>
   |
 62|  ";
   |
 64|}
   |
 66|?>
```

Hasil e-mail yang terkirim adalah sebagai berikut:

```
Date: 6 Dec 2003 20:43:15 -0000
Message-ID: <20031206204315.19733.qmail@builder.localdomain>
To: steven@masterwebnet.com
Subject: Pesanan buku
From: "Budi Budiman" <budi@budiman.com>
X-IP-Address: 192.168.0.3
X-Referer: http://steven.builder.localdomain/test/php/e-mail-
gpg.php


--BEGIN PGP MESSAGE--
Version: GnuPG v1.0.7 (GNU/Linux)

hQIOAyjEHA3kGTpUEAgA54ryJmaJaopnCY3STqdKbLxPmuoIMvoSMVEtgz/HPyVd
5f/CKIBXAxnd1Oywx4VHTC3UpJ1m6854AIPveFxSGu7DdxoXk7O3DxApdWwl2Wna
HNzAfMI661Lz3tckU8a47o97byEyjlHlv25ZgO2CiJCjkpaud3S26cvC5FWWXLCx
fsgBQQ7Xlh+/ONXCgAl3RXxO6cWzlt8Dog+1tk7BnzXp1yl8/YdOszO9NOWHfhOa
c7XNWIrxwOFswOWHmCIj6BRuMiz9/H3DcluZMF+t8sXpspHu16Tr//Xpu5yq+PXN
Bo/fT5BK8++cLOr3zrTpJvyBnZCKtbRJPJ7wwYusewf8DwBiKYgs2htIXM2J1a3K
i2e3TdaLHDJya+ml2tMhMEYyNLwqTS7KvGYvcgul1pXx8AUgyXBWn1owziYUtFDe
gitSIluf7++aSd7Z4EB/C6rMaMSKdrpQDJMQ4tpNGC88G/KehkXtfB7cNwj1xxut
8KJ7oBtHh3nm4l3Hd+QcLIBUP+DSIXFLIwRKUpFjJ6TOKIXEIwY5TtbYyiDBoGm6
XLSZyXgpm/b/Yfl2KNlKCPxGivHtsqODWlmhZjmnl5hO5cQLaPmMwXXS1+1uOcYU
zgkcqbg6dOrFqqIylhdsVkLW6seqvEfYrctw3+G9OwWOKKjXSiYreveWNQfvWBbT
WdKOAZgPi8BYYHK4BNieIfM+6LVNL89TEJM2xsFeL+Z/upBcA+NTosryyXnlmDm/
7ysghmrFkfrfcCxogfmMPA3r192JgZb7sj+lY84Vh+r8PqNQqkG5CX+H/GGB2Iex
S3vgklfYzSUO+ZinRBA4Zm6U+cZmIoZydDEGci6JpP3maiL/Suyd8akXuJeNpA7T
OQ==
=6lMG
--END PGP MESSAGE--
```


Sehingga kini e-mail aman untuk dikirimkan dengan SMTP biasa, di-forward ke server lain, disimpan di server yang tidak aman. Atau bahkan diambil lewat POP biasa yang bukan APOP atau POP+SSL.

Mendekripsi pesan

Untuk membuka pesan di komputer pribadi, Anda bisa menggunakan program command line `gpg` lagi dengan opsi `–decrypt`. Atau untuk lebih praktisnya, gunakan plugin GnuPG yang tersedia untuk beberapa program klien e-mail, seperti Outlook Express dan Mozilla.

Resep 9-7: Membuat Autoresponder

Autoresponder adalah kebalikan dari mengirim e-mail, yaitu program yang akan dijalankan ketika menerima sebuah e-mail. Autoresponder bisa dibuat dalam berbagai bahasa program, tidak perlu dengan PHP (bahkan bisa dikatakan mungkin bahasa lain lebih cocok dalam hal ini). Kunci membuat autoresponder bukanlah PHP-nya, melainkan memahami bagaimana e-mail diterima di server Anda.

Setiap program pengirim e-mail (MTA, mail transfer agent) seperti **Qmail**, **Postfix**, **Sendmail**, atau **Courier** memiliki mekanisme yang berbeda dalam menyimpan e-mail ke mailbox. Tapi rata-rata mengizinkan setiap user di server untuk menaruh skrip yang akan dijalankan ketika sebuah e-mail hendak dimasukkan ke dalam account user tersebut.

Di Sendmail, Anda dapat membuat sebuah file di `~/.foward` (file bernama "dot-forward" di home directory) yang isinya:

```
|/usr/bin/php -q /home/steven/scripts/autoresponder.php
```

di mana autoresponder.php adalah sebuah skrip PHP biasa, atau:

```
|/home/steven/scripts/autoresponder.php
```

di mana autoresponder.php adalah sebuah skrip yang telah di-chmod 755 dan diberi baris `#!/usr/bin/php -q` di baris pertama skrip.

Tanda garis vertikal ("|") berarti eksekusi program.

Di Qmail, padanan untuk `.forward` adalah `.qmail-default` (file bernama "dot-qmail-default"). Sintaks isinya untuk mengeksekusi program sama dengan sintaks `.forward`. Di Courier MTA juga ada padanannya bernama `.courier` ("dot-courier").

Program yang dieksekusi akan mendapatkan e-mail dari `stdin` dan informasi seperti envelope sender berada di variabel-variabel environment tertentu.

Resep 9-7 adalah contoh `autoresponder.php` yang dirancang untuk bekerja dengan Qmail. Isi .qmail-default:

```
./Maildir/
|/home/steven/scripts/autoresponder.php
```

Baris pertama adalah perintah untuk menaruh dulu e-mail yang diterima di mailbox (di `/home/steven/Maildir/`). Sementara perintah kedua untuk menjalankan skrip yang membalas pengirim dengan ucapan terima kasih dan konfirmasi bahwa pesan telah diterima serta disimpan di mailbox. Perhatikan bahwa di server perlu terpasang PHP dalam mode biner (`/usr/bin/php`) dan bukan mode modul Apache, karena tidak mengeksekusi skrip PHP via webserver.

```
 1|#!/usr/bin/php -q
 2|<?
  |
 4|//
 5|// autoresponder.php
 6|//
  |
 8|$FROM = "steven@masterwebnet.com";
 9|$MEMORY_PATH = "/home/steven/var/autoresponder.serialize";
  |
11|$now = time();
  |
13|// siapa pengirim e-mail
14|$sender = $_ENV['SENDER'];
15|if ($sender == "" || $sender == "<>") exit;
  |
17|// baca e-mail
18|$input = "";
19|if (!isset($stdin)) $stdin = fopen('php://stdin', 'r');
20|while (!feof($stdin)) $input .= fgets($stdin, 8192);
```



```
  |
22|// parse e-mail
23|require_once "Mail/mimeDecode.php";
24|$structure = Mail_mimeDecode::decode(array("input"=>$input),
  "\r\n");
  |
26|// jangan kirim autoresponse jika ini e-mail dari milis
27|if (isset($structure->headers['mailing-list']) exit;
28|if (isset($structure->headers['precedence']) &&
29|    preg_match("/bulk/i", $structure->headers['precedence']))
  exit;
  |
31|// jangan kirim jika sudah pernah dikirim
32|if (file_exists($MEMORY_PATH)) {
33|  $fp = fopen($MEMORY_PATH, "rb");
34|  $memory = unserialize(fread($fp, filesize($MEMORY_PATH)));
35|  // hapus memori lama
36|  foreach (array_keys($memory) as $m) {
37|    if ($memory[$m] < $now - 48*3600) unset($memory[$m]);
38|  }
39|} else {
40|  $memory = array();
41|}
42|if (isset($memory[$sender])) exit;
  |
44|// kirim autoresponse
45|mail(
46|$sender,
  |
48|  // subjek
49|"Terima kasih telah menghubungi saya",
  |
51|  // isi pesan
52|"Pesan Anda telah saya terima dan jika perlu dibalas akan
  dibalas dalam 3x24
53|jam.
  |
55|Salam,
56|Steven
57|",
  |
```

```
59|  // header tambahan
60|"From: $FROM\n".
61|"Reply-To: <>\n".
62|"Return-Path: <>\n"
63|);
  |
65|// ingat di memori
66|$memory[$sender] = $now;
67|$fp = fopen($MEMORY_PATH, "w");
68|fputs($fp, serialize($memory));
69|fclose($fp);
  |
71|?>
```

Pembahasan

Ada beberapa kriteria dalam menulis sebuah autoresponder yang baik. Pertama, tidak merespon balik ke milis hingga semua anggota milis lain terganggu. Kedua, tidak merespon berkali-kali kepada orang yang sama. Ketiga, sebaiknya merespon kepada envelope sender (Return-Path) dan bukan kepada RFC-822 From.

Mula-mula skrip mengambil nilai envelope sender (Return-Path) dari variabel environment `SENDER` yang disediakan oleh Qmail (baris 14-15). Jika `SENDER` kosong, maka kita tidak bisa membalas e-mail ini, jadi skrip langsung berhenti. Lagipula, e-mail dengan Return-Path kosong biasanya berupa pesan kesalahan atau bounce yang tidak perlu dibalas dengan autoresponse.

Skrip lalu membaca dan mem-parse e-mail (baris 18-24). Untuk mem-parse e-mail, kita minta bantuan pada kelas `Mail_mimeDecode` yang masih merupakan bagian paket PEAR `Mail_Mime`. Hasil parse berupa array `$structure`. Anda bisa mencoba mem-`print_r()` struktur ini jika ingin melihat hasilnya. Contohnya, header-header disimpan di `$structure['headers'][<nama-header>]`, di mana `<nama-header>` adalah nama header yang telah dikonversi menjadi huruf kecil semua. Selanjutnya, di baris 27-29 kita mengecek apakah e-mail ini berasal dari milis. Sebuah cara untuk mengetahuinya adalah mengecek keberadaan header `Mailing-List` (ini merupakan header standar RFC, dan dikirimkan antara lain oleh **Mailman** dan **ezmlm** versi yang cukup baru). Tidak semua menyertakan header ini, karena itu kita mencari tanda pengenal lain yaitu header `Precedence: bulk`, yang juga biasanya diset oleh pengirim milis atau newsletter.

Baris 32-42 dan 66-69 adalah implementasi "memori". Setiap kali skrip ini dieksekusi, di akhir skrip (baris 66-69) kita menyimpan alamat e-mail pengirim

dalam sebuah array `$memory`. Array ini kita simpan ke file (dalam format serialisasi PHP). Di baris 33-34 kita membaca lagi file ini untuk memperoleh kembali array `$memory`. Nilai elemen array adalah timestamp di mana terakhir kali kita mengirim pesan autoresponse kepada alamat tersebut.

Jika timestamp sudah cukup tua (di resep diset 2x24 jam), maka kita melupakan memori dan menganggap ini alamat e-mail baru. Jadi, dengan kata lain, kalau ada orang yang mengirim e-mail lebih dari satu kali dalam rentang waktu 2 hari, kita hanya akan mengirimkan autoresponse sekali saja kepada orang tersebut.

Baris 45-63 adalah inti program, yaitu pemanggilan fungsi `mail()`. Perhatikan header-header yang kita kirimkan, yaitu: `Return-Path: <>` agar jika e-mail kita mental, tidak usah menerima bouncenya. `Reply-To: <>` agar pesan autoresponse ini tidak bisa dibalas oleh pengirim (karena kita memang tidak mengharapkan balasan atas pesan autoresponse kita).

Resep 9-8: Membuat Skrip yang Diperintah Lewat E-mail

Selain membuat autoresponder, mekanisme `.forward/.qmail` juga bisa dipakai untuk membuat program yang merespon perintah via e-mail Misalnya untuk men-download URL dan mengirimkan balik via attachment, atau mengirimkan SMS via e-mail, dan sebagainya. Resep di bawah adalah contoh sederhana program PHP yang dapat menerima perintah lewat subjek e-mail.

```
 1|#!/usr/bin/php -q
 2|<?
  |
 4|//
 5|// server-status.php
 6|//
  |
 8|$FROM = "server-status@masterwebnet.com";
 9|$AUTHORIZED_SENDERS = array(
10|  "steven@masterwebnet.com",
11|  "dhiar@masterwebnet.com",
12|  "steven@steven.builder.localdomain");
  |
14|function send_reply($to, $subject, $message) {
```

```
15|  global $FROM;
  |
17|  mail($to, $subject, $message,
18|       "From: $FROM\n".
19|       "Reply-To: <>\n".
20|       "Return-Path: <>\n"
21|  );
22|}
  |
24|function send_error($to, $error_message, $orig_mail) {
25|  send_reply($to, "ERROR",
  |
27|  "Perintah Anda tidak dapat dilaksanakan karena:\n\n".
28|  "  $error_message\n\n".
29|  "Email asli Anda:\n\n".
30|  $orig_mail);
  |
32|  exit;
33|}
  |
35|// baca e-mail
36|$input = "";
37|if (!isset($stdin)) $stdin = fopen('php://stdin', 'r');
38|while (!feof($stdin)) $input .= fgets($stdin, 8192);
  |
40|// parse e-mail
41|require_once "Mail/mimeDecode.php";
42|$structure = Mail_mimeDecode::decode(array("input"=>$input),
   "\r\n");
  |
44|// siapa pengirim e-mail
45|$sender = $_ENV['SENDER'];
46|if ($sender == "" || $sender == "<>") exit;
47|if (!in_array($sender, $AUTHORIZED_SENDERS))
48|  send_error($sender, "Permission denied", $input);
  |
50|// cek dan laksanakan perintah
51|$subject = rtrim($structure->headers['subject']);
52|if     ($subject == "ps ax") $result = `ps ax`;
53|elseif ($subject == "w") $result = `w`;
54|elseif ($subject == "df") $result = `df`;
```



```
55|else send_error($sender, "Unknown command: '$subject'",
   $input);
  |
57|// kirim hasil
58|send_reply($sender, "Re: $subject", $result);
  |
60|?>
```

Agar program ini berjalan dengan semestinya, Anda perlu memilih atau menaruh lokasi file .forward atau apa nama file .qmail yang sesuai. Misalnya saja, jika ingin program ini merespon di alamat e-mail server-status@masterwebnet.com, maka Anda perlu memasangnya di `/home/server-status/.qmail-default` atau bisa saja di `/home/mwn/.qmail-server-status`. Tergantung konfigurasi di tempat Anda.

Perintah yang diterima program ini yaitu "ps ax" (untuk melihat daftar proses di server), "w" (untuk mengecek uptime dan daftar user yang sedang login di server), dan "df" (untuk mengecek ruang sisa disk). Di luar ketiga perintah yang dikenali, skrip akan mengembalikan pesan kesalahan.

Demi "keamanan", skrip juga tidak mau mengirimkan hasil perintah kepada sembarang orang. Namun hanya pada alamat yang terdaftar di array `$AUTHORIZED_SENDERS`, baris 9-12 dan 47-48. Kata "keamanan" memang relatis, karena tidak betul-betul aman. Siapa saja di Internet bisa mengubah From-nya menjadi nilai sembarang. Anda juga bisa memodifikasi skrip ini dengan menambahkan perintah-perintah lainnya, memparse bodi jika ingin perintah dispesifikasikan di bodi, bukan subjek. Atau memberlakukan limit jumlah perintah per hari dengan mekanisme yang mirip-mirip seperti "memori" di resep autoresponder 9-7. Atau bisa juga dengan menambahkan perintah serta mekanisme password atau konfirmasi, dan lain sebagainya.

Resep 9-9: Mengecek Validitas Alamat E-mail

```
<?
if (preg_match("/^[^@\s]+@([-a-z0-9]+\.)+[a-z]{2,}$/i", $e-mail))
  echo "Valid!";
else
  echo "Tidak valid!";
?>
```

Resep di atas mengecek validitas sintaks alamat e-mail dengan regex. Sebetulnya regex yang benar-benar presisi sesuai standar Internet jauh lebih kompleks dari ini, tapi untuk sebagian besar kebutuhan, regex di atas sudah cukup.

```
<?
$default_host = "citramas.co.id";

$parsed = imap_rfc822_parse_adrlist($e-mail, $default_host);
if ($parsed['mailbox'] == 'INVALID ADDRESS') {
  echo "Tidak valid!";
else
  echo "Valid!";
?>
```

Resep di atas ini menggunakan fungsi yang telah disediakan oleh ekstensi IMAP. Sederhana karena tidak perlu mengingat regex dan lebih komplian terhadap standar, tapi membutuhkan ekstensi IMAP diaktifkan di server atau komputer pribadi Anda.

Resep 9-10: Mengecek E-mail POP3

Email biasanya disimpan dalam bentuk mailbox berformat mbox atau Maildir di server, dan diakses/diambil lewat webmail, POP3, atau IMAP. Protokol POP3, meskipun sederhana dan terbatas, tetap merupakan cara yang terpopular di Internet untuk mengakses mailbox di server.

POP3 terbatas karena hanya mendukung mengambil (dan menghapus e-mail yang telah diambil), tapi tidak dapat menyimpan, membuat folder, memindahkan e-mail, dan operasi lainnya.

Resep di bawah mendemonstrasikan penggunaan paling sederhana paket PEAR Net_POP3, yaitu untuk mengecek jumlah e-mail di server. Di balik layar, kelas Net_POP3 mengirimkan perintah login lalu perintah "STAT", di mana server POP3 mengembalikan jawaban berupa `"<jumlah-e-mail> <total-ukuran>"`.

Sebetulnya extension IMAP PHP (fungsi `imap_open()` dkk) pun mampu menangani protokol POP3, namun kita akan menggunakan interface OO Net_POP3 pada contoh ini.


```
 1|<?
  |
 3|//
 4|// net_pop3.php
 5|//
  |
 7|$POP_HOST = "pop.cbn.net.id";
 8|$POP_PORT = 110; // port default POP3
 9|$POP_USER = "steven";
10|$POP_PASS = "steven123";
11|$POP_APOP = false;
  |
13|require_once 'Net/POP3.php';
  |
15|$pop3 =& new Net_POP3();
  |
17|if (!$pop3->connect($POP_HOST, $POP_PORT))
18|  die("Gagal konek ke server '$POP_HOST:$POP_PORT'");
  |
20|if ($pop3->login($POP_USER, $POP_PASS, $POP_APOP))
21|  die("Gagal login");
  |
23|echo "Jumlah e-mail di mailbox server saat ini: ".
24|  $pop3->numMsg();
  |
26|?>
```

`$POP_APOP` dapat Anda set menjadi true jika server POP3 yang ingin Anda gunakan mendukung APOP.

Resep di bawah berfungsi sebagai POP checker, yaitu mengecek apakah ada e-mail yang baru di server, dan jika ada, menampilkan subjeknya. Karena dari panggilan yang satu ke yang lain membutuhkan memori atau state, kita gunakan sesi untuk menyimpan daftar UIDL (ID) e-mail sekaligus username dan password POP3.

```
 1|<?
  |
 3|//
 4|// net_pop3.php
 5|//
  |
```

```
require_once 'Net/POP3.php';
require_once 'Mail/mimeDecode.php';

ob_start();
session_register('pop_host');
session_register('pop_port');
session_register('pop_user');
session_register('pop_pass');
session_register('pop_apop');
session_register('uidls');

echo "<h2>POP3 checker</h2>";

if (!isset($action)) $action = "";

if ($action == 'login') {

  if (isset($submit)) {
    // tes dulu login ke POP
    $form_pop_apop = isset($form_pop_apop) ? true:false;
    $pop3 =& new Net_POP3();

    if (!$pop3->connect($form_pop_host, $form_pop_port))
      die("Gagal konek ke server '
$form_pop_host:$form_pop_port',
           pastikan informasi POP Anda benar");

    if (!$pop3->login($form_pop_user, $form_pop_pass,
$form_pop_apop))
      die("Gagal login POP, pastikan informasi POP Anda
benar");

    $pop_host = $form_pop_host;
    $pop_port = $form_pop_port;
    $pop_user = $form_pop_user;
    $pop_pass = $form_pop_pass;
    $pop_apop = $form_pop_apop;
    $uidls = array();

    // lempar ke menu
    header("Location: $PHP_SELF");
```



```
  } else {
    echo "
      <form method=POST>
      <input type=hidden name=action value=login>
      <table border=0>
      <tr><td>Host</td>
        <td><input name=form_pop_host size=24></td></tr>
      <tr><td>Port</td>
        <td><input name=form_pop_port size=4 value=110></
td></tr>
      <tr><td>User</td>
        <td><input name=form_pop_user size=15></td></tr>
      <tr><td>Password</td>
        <td><input name=form_pop_pass size=15></td></tr>
      <tr><td>Gunakan APOP?</td>
        <td><input type=checkbox name=form_pop_apop>Ya</td></
tr>
      <tr><td><input type=submit name=submit></td><td></td></
tr>
      </table>
      </form>
    ";
  }

} elseif ($action == 'check') {

  if (!isset($pop_host))
    die("Anda belum <a href=$PHP_SELF?action=login>login</
a>");

  $pop3 =& new Net_POP3();

  if (!$pop3->connect($pop_host, $pop_port))
    die("Gagal konek ke server '$pop_host:$pop_port'");

  if (!$pop3->login($pop_user, $pop_pass, $pop_apop))
    die("Gagal login POP");

  $new = array(); $nmsg = 0;
  foreach ($pop3->getListing() as $msg) {
    if (!$msg['msg_id']) continue;
```

```
 82|    $nmsg++;
 83|    if (!isset($uidls[ $msg['uidl'] ])) {
 84|      // e-mail baru, ambil headernya...
 85|      $headers = $pop3->getRawHeaders($msg['msg_id']);
   |
 87|      // ... lalu parse untuk ambil Subject dan From
 88|      $structure = Mail_mimeDecode::decode(
 89|                      array("input"=>$headers), "\r\n");
 90|      $new[] = array(
 91|        'subject' => $structure->headers['subject'],
 92|        'from' => $structure->headers['from'],
 93|      );
   |
 95|      // ingat UIDL ini
 96|      $uidls[ $msg['uidl'] ] = 1;
 97|    }
 98|  }
   |
100|  $pop3->disconnect();
   |
102|  // tampilkan hasilnya
103|  echo "<p><b>$nmsg</b> total mail di mailbox.";
104|  echo "<br><b>",count($new),"</b> mail baru.";
   |
106|  if (count($new)) {
107|    $i=1;
108|    echo "<table border=1>
109|          <tr><td>No</td><td>From</td><td>Subject</td></tr>";
110|    foreach ($new as $msg) {
111|      echo "<tr><td>$i</td>
112|                <td>",htmlentities($msg['from']),"</td>
113|                <td>",htmlentities($msg['subject']),"</td></
    tr>";
114|      $i++;
115|    }
116|    echo "</table>";
117|  }
   |
119|  echo "<p><a href=$PHP_SELF?action=check>Refresh</a><br>";
120|  echo "<a href=$PHP_SELF?action=logout>Logout</a>";
   |
```

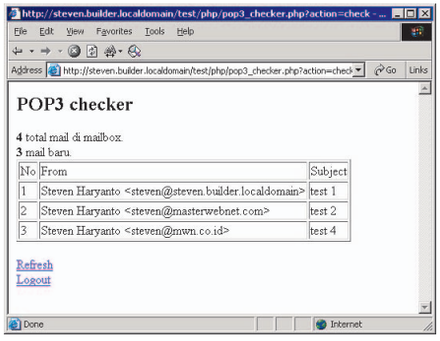


```
122|} elseif ($action == 'logout') {
   |
124|  session_destroy();
   |
126|  // lempar ke menu
127|  header("Location: $PHP_SELF");
   |
129|// menu
130|} else {
   |
132|  if (isset($pop_host)) {
133|    // sedang login
134|    echo "<a href=$PHP_SELF?action=check>Cek</a><br>";
135|    echo "<a href=$PHP_SELF?action=logout>Logout</a><br>";
   |
137|  } else {
138|    // tidak sedang login
139|    echo "<a href=$PHP_SELF?action=login>Login</a><br>";
140|  }
   |
142|}
   |
144|?>
```

Pembahasan

Mula-mula seseorang yang ingin mengecek account POP3-nya kita minta dulu untuk "login" (baris 137-140). "Login" di sini bukan berarti login ke account POP3 melainkan login untuk mengeset variabel sesi kita (baris 36-41). Baru setelah login, kita menyediakan link untuk mengecek account POP3 karena informasi account POP3 telah dimasukkan sebelumnya.

Baris 66-121 merupakan bagian utama program. Di sini kita login ke account POP3 dan mengambil daftar e-mail di mailbox dengan metode `getListing()`. Di balik layar, metode ini mengirimkan perintah "UIDL" kepada server POP3, dan jawaban server POP3 adalah baris "<msg-id> <UIDL>", setiap baris untuk setiap e-mail. *<msg-id>* adalah angka seperti 1, 2, 3 yang merupakan urutan e-mail di dalam mailbox (angka ini tidak merujuk ke satu pesan tertentu secara tetap, melainkan hanya urutan pesan yang berlaku pada sesi POP itu saja). Sementara *<UIDL>* adalah sebuah string yang merupakan pengenal unik untuk sebuah pesan. UIDL yang sama berarti pesan e-mail yang sama, tak peduli apakah urutannya berbeda di mailbox. Karena itu, untuk mengingat e-mail yang ada di mailbox, kita menggunakan UIDL (baris 96).



Gambar 9-1. Tampilan POP3 Checker.

Baris 81 mengecek apakah sebuah e-mail telah kita ingat UIDL-nya di dalam variabel sesi `$uidls` (yang merupakan array dengan key = UIDL). Jika belum, berarti e-mail ini adalah e-mail baru dan kita ambil subjek dan From-nya (baris 84-89). Protokol POP3 hanya memiliki perintah TOP untuk mengambil header (dan sebagian isi e-mail), karena itu untuk memparse header ini kita kembali meminta bantuan kepada kelas `mimeDecode` dari paket PEAR `Mail_Mime`.

Baris 113-117 menampilkan hasil. Perhatikan kita menggunakan `htmlentities()` untuk memastikan kalau ada tag-tag HTML tidak akan terinterpretasi oleh browser. Ini penting karena header From biasanya mengandung "<" dan ">". Jika kita menekan *Refresh* pada halaman Cek, maka e-mail-e-mail baru yang sebelumnya tampil akan sudah tercatat di `$uidls` sehingga tidak dianggap e-mail baru lagi.

Jika kita menekan logout, maka informasi login POP3 beserta array `$uidls` akan dikosongkan (baris 124), sehingga jika kita masuk kembali e-mail-e-mail di mailbox akan dianggap baru kembali ketika pertama kali dicek. Contoh keluaran program ini bisa dilihat di Gambar 9-1.

Resep 9-11: Mengecek E-mail IMAP

IMAP adalah protokol yang lebih baru dari POP3 dan memang dibuat untuk mengatasi keterbatasan POP3. IMAP menyediakan spektrum akses yang lebih luas dalam memanipulasi mailbox dan pesan e-mail. Berbeda dengan POP3, sebuah account tunggal IMAP dapat memiliki lebih dari satu mailbox (meskipun tiap mailbox harus diakses dengan handle berbeda). IMAP juga memiliki fitur lain seperti mailbox publik dan subscription ala newsgroup atau mailing list. Bahkan karena fasilitas manipulasinya lengkap, Anda bisa selalu menyimpan dan mengakses mailbox

Anda di server tanpa harus membuat kopinya di komputer lokal Anda (kecuali untuk tujuan cache). Namun tidak semua server hosting atau ISP menyediakan fasilitas ini karena berpotensi memberatkan server (misalnya: sebuah mesin dapat mendukung ribuan bahkan puluhan ribu account POP3 dan ratusan sesi POP simultan, tapi hanya dapat mendukung puluhan sesi IMAP simultan). Untuk mengakses IMAP, Anda memerlukan extension imap pada instalasi PHP Anda. Resep di bawah ini memperlihatkan contoh login ke account IMAP dan menampilkan daftar mailbox.

```
 1|<?
  |
 3|//
 4|// coba-imap.php
 5|//
  |
 7|$IMAP_HOST = "localhost";
 8|$IMAP_PORT = 143; // port default IMAP. port default IMAP SSL
  = 993.
 9|$IMAP_USER = "steven@steven.builder.localdomain";
10|$IMAP_PASS = "123";
11|$IMAP_SSL  = false;
  |
13|$ref = "{" . "$IMAP_HOST:$IMAP_PORT" .
14|       "/imap" .
15|       ($IMAP_SSL ? "/ssl" : "") .
16|       "/novalidate-cert}";
  |
18|// buka koneksi ke mailbox default IMAP.
19|if (($mbox = imap_open($ref."INBOX", $IMAP_USER,
  $IMAP_PASS))===false)
20|  die("Gagal konek ke INBOX");
  |
22|echo "<p>Daftar folder mailbox:</p>";
23|echo "<ul>";
  |
25|foreach (imap_list($mbox, $ref, "*") as $m) {
26|  echo "<li>$m";
27|}
  |
29|echo "</ul>";
  |
31|?>
```