PENERAPAN REGULAR EXPRESSION DALAM MELINDUNGI ALAMAT EMAIL DARI SPAM ROBOT PADA KONTEN WORDPRESS

Agus Muliantara
Program Studi Teknik Informatika, Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Udayana
Email : muliantara@ilkom.unud.ac.id

ABSTRAK

Email sebagai sebuah kemajuan teknologi informasi dan telekomunikasi telah dapat menghubungkan jutaan orang di seluruh dunia. Namun seiring perkembangan tersebut, terdapat juga perkembangan yang mungkin dapat dikatakan merugikan yaitu munculnya spam robot. Dimana spam robot akan mencari setiap alamat email yang ada di seluruh situs untuk kemudian dikirimi iklan yang lebih sering tidak bermanfaat. Ada beberapa cara untuk melindungi alamat email pada suatu situs. Salah satunya yaitu dengan cara mengganti alamat email yang berbasis teks tersebut menjadi gambar/image. Sehingga mencegah spam robot melakukan parsing terhadap alamat email.

Penelitian yang dilakukan dengan mengambil studi kasus pada situs yang menggunakan Content Management System (CMS) wordpress ini bertujuan untuk melakukan pengembangan pada konten situs wordpress sehingga setiap alamat email yang ditulis secara standar nantinya secara otomatis akan diubah menjadi gambar.

Dari pengujian yang dilakukan terhadap pengembangan didapat hasil pengembangan dapat berjalan dengan baik. Pengembangan aplikasi mampu merubah setiap alamat email yang ditulis secara standar secara otomatis menjadi gambar.

Kata Kunci : *Regular expression, Email, Spam Robot, Konten, Wordpress*

SUMMARY

Email as a progress of information technology and telecommunications has connected millions of people around the world. However, over the development, there are developments that may harm called a spam robot. Where is the spam robots will explore every email address in the entire website and then sending alot of ads to those email address. There are several ways to protect email addresses on a site that is to change the text-based email address into image. So to prevent spam robots parsing of email addresses.

This research carried out by taking a case study on the site using the Content Management System (CMS) Wordpress, with aims to make development on the wordpress site content so that each email address that is written by default will automatically be converted into images.

From the testing carried out the results that the development can run better. The application able to change the email address written in the standard form into image automatically

Keyword : *Regular expression, Email, Spam Robot, Content, Wordpress*

PENDAHULUAN

Seiring dengan perkembangan teknologi informasi dan telekomunikasi, email atau surat elektronik tidak dipungkiri dapat menghubungkan jutaan orang di seluruh dunia untuk tetap saling berhubungan tanpa dibatasi oleh jarak dan waktu. Perkembangan teknologi yang pesat ini selain memudahkan orang ternyata memacu orang-orang "cerdas" untuk memanfaatkan email sebagai media iklan/promosi suatu produk.

Tanpa persetujuan pemilik email bersangkutan, email-email iklan membanjiri inbox mereka. Bagaimana cara orang "cerdas" tersebut mengetahui alamat email tersebut, yang sebelumnya tidak pernah berhubungan dengan pemilik iklan?

Terdapat beberapa cara untuk mendapatkan alamat email diantaranya adalah dengan menggunakan *engine*/program yang biasa disebut sebagai *spam robot* yang mencari alamat email di seluruh konten situs web di dunia. Saat alamat email ditemukan, maka sejak saat itulah iklan akan mulai dikirimkan.

Dalam penelitian ini akan dikembangkan sebuah cara/metode untuk melindungi alamat email dari *spam robot* pada konten Wordpress dengan menggunakan *regular expression.*

TINJAUAN PUSTAKA

Wordpress

WordPress adalah sebuah perangkat lunak blog yang ditulis dalam PHP dan mendukung sistem basis data MySQL (Wordpress, 2009). WordPress adalah penerus resmi dari cafelog yang dikembangkan oleh Michel Valdrighi. Nama WordPress diusulkan oleh Christine Selleck, teman dari ketua developer, Matthew Charles Mullenweg.

Rilis terbaru WordPress adalah versi 2.7. WordPress didistribusikan dengan lisensi GNU General Public License.

PHP Image

Bahasa pemrograman PHP tidak hanya terbatas pada pembuatan halaman HTML, tetapi PHP juga dapat digunakan untuk membuat dan memanipulasi gambar dengan menggunakan berbagai macam format. Kemampuan PHP ini didukung oleh *library* yang disebut dengan GD (PHP,2009) .

Beberapa fungsi yang mungkin sering digunakan dalam pembuatan image adalah sebagai berikut : *imagestring()* dan *imagegif(),imagejpeg(),imagepng().*

Regular expression

Regular expression atau yang sering disebut sebagai *Regex* adalah sebuah formula untuk pencarian pola suatu kalimat/*string* (Regular,2009). Sering kali orang beranggapan bahwa *regex* susah dan membingungkan. Namun sebenarnya *regex* sangatlah membantu dalam menemukan pola-pola kalimat. Sehingga percobaan terhadap semua kemungkinan pola kalimat tidak perlu dilakukan.

Regular expression umumnya digunakan oleh banyak pengolah kata/*text editor* dan peralatan lainnya untuk mencari dan memanipulasi kalimat dengan berdasrkan kepada suatu pola tertentu. Banyak bahasa pemrograman yang mendukung *regular expression* seperti misalnya PHP, perl, VB dan Tcl.

Sebuah alasan yang sangat bagus untuk menggunakan *regex* adalah karena *regex* sangatlah *powerfull*. Pada level rendah *regex* dapat mencari sebuah penggalan kata. Pada level tinggi *regex* mampu melakukan kontrol terhadap data. Baik mencari, menghapus dan merubah.

Mari kita pikirkan bagaimana cara untuk mencari sebuah file di hard disk. Seringkali digunakan karakter "?" dan "*". Penggunaan karakter "?" mengandung arti bahwa sedang dicari sebuah sebuah file yang mengandung sebuah karakter tertentu dan karakter "*" mengandung arti sedang dicari nol atau lebih karakter. Sebagai contoh : pencarian dengan menggunakan pola "*file?.dat***"** akan menghasilkan beberapa contoh sebagai berikut :

- file1.dat
- file2.dat
- file3.dat
- file.dat
- fileN.dat

Menggunakan karakter "*" akan menghasilkan lebih banyak hasil pencarian. "*file*.dat*" akan menghasilkan :

- file1.dat
- file2.dat
- file12.dat
- filex.dat
- fileXYZ.dat

Kedua contoh di atas adalah salah satu penggunaan *regular expression.* Bayangkan saja jika pencarian file hanya menggunakan metode nama file tanpa menyertakan *regular expression*. Tentunya saat proses pencarian menggunakan pola "*file*.dat*" hasilnya adalah **harus** "*file*.dat*". Hal inilah yang menyebabkan *regex* menjadi sangat *powerfull*. Dengan menggunakan sebuah pola didapatkan hasil yang luas.

Beberapa pola yang umum digunakan dalam *regex* tampak pada tabel 1 berikut.

Tabel 1. pola umum pada regex

Pola	Penjelasan
[ ]	ekspresi kurung. cocok dengan satu karakter yang berada dalam kurung, misal: pattern "a[bcd]i" cocok dengan string "abi", "aci", dan "adi". penggunaan range huruf dalam kurung diperbolehkan, misal : pattern "[a-z]" cocok dengan salah satu karakter diantara string "a" sampai "z". pattern [0-9] cocok dengan salah satu angka. jika ingin mencari karakter "-" juga, karakter tersebut harus diletakkan di depan atau di belakang kelompok, misal: "[abc-]".
[^]	cocok dengan sebuah karakter yang tidak ada dalam kurung, berlawanan dengan yang diatas. misal: pattern "[^abc]" cocok dengan satu karakter apa saja kecuali "a", "b", "c".
?	cocok dengan nol atau satu karakter sebelumnya. misal: pattern "died?" cocok dengan string "die" dan "died".
+	cocok dengan satu atau lebih karakter sebelumnya. misal: "yu+k" cocok dengan "yuk", "yuuk", "yuuuk", dan seterusnya.

*****	cocok dengan nol atau lebih karakter sebelumnya. misal: pattern "hu*p" cocok dengan string "hp", "hup", "huup" dan seterusnya.
{x}	cocok dengan karakter sebelumnya sejumlah x karakter. misal: pattern "[0-9]{3}" cocok dengan bilangan berapa saja yang berukuran 3 digit.
{x,y}	cocok dengan karakter sebelumnya sejumlah x hingga y karakter. misal: pattern "[a-z]{3,5}" cocok dengan semua susunan huruf kecil yang terdiri dari 3 sampai 5 huruf
!	jika diletakkan di depan pattern, maka berarti "bukan". misal pattern "!a.u" cocok dengan string apa saja kecuali string "alu", "anu", "abu", "asu", "aiu", dan seterusnya
^	jika diletakkan di depan pattern, akan cocok dengan awal sebuah string.
$	jika diletakkan di belakang pattern, akan cocok dengan akhir sebuah string
( )	gruping. digunakan untuk mengelompokkan karakter-karakter menjadi single unit. string yang cocok dalan pattern yang berada dalam tanda kurung dapat digunakan pada operasi berikutnya. semacam variable.
\	escape character. mengembalikan fungsi metacharacter menjadi karakter biasa. pada beberapa system dapat berarti sebaliknya, yaitu metacharacter menggunakan escape character didepannya

METODE

Penelitian ini dilaksanakan melalui beberapa tahap yaitu :

1. Tahap Pengumpulan Kebutuhan

   Pada tahap ini dikumpulkan apa saja yang dibutuhkan dalam membangun sitem.

2. Tahap Studi Literatur

   Pada tahap ini dilaksanakan pengumpulan teori-teori yang mendukung penelitian. Seperti penggunaan *regex* dan php image.

3. Tahap Perancangan Sistem

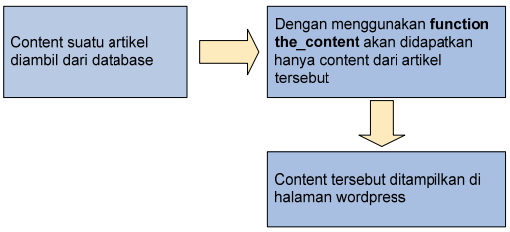
   Pada tahapan ini dilakukan perancangan sistem dngan mengacu pada hasil yang telah didapatkan pada tahapan sebelumnya. Kemudian dituangkan dalam bentuk diagram alir sehingga lebih mudah dipahami

4. Tahap Implementasi

   Pada tahapan ini, sistem diimplementasikan menggunakan PHP image, wordpress' function serta dijalankan pada server berbasis linux.

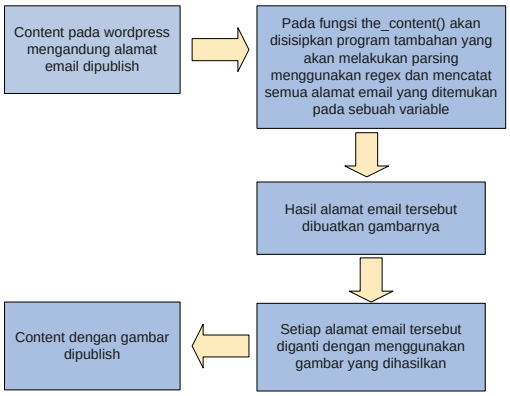
HASIL DAN PEMBAHASAN

Dari hasil analisa pada masing-masing tahapan pada metodologi, didapatkan skema bagaimana wordpress menampilkan sebuah konten dari database ke halaman muka yaitu menggunakan fungsi **the_content()** yang terdapat pada file *post-tempalte.php* pada folder *wp-includes* seperti ditunjukkan pada gambar 1.



Gambar 1. cara menampilkan konten pada wordpress

Triknya adalah dengan memanfaatkan fungsi **the_content()** tersebut. Kode dari fungsi **the_content()** akan dimodifikasi agar mampu melakukan *parsing*, pembuatan gambar sesuai dengan alamat email dan penggantian alamat email menggunakan gambar yang sebelumnya sudah dihasilkan. Dengan melakukan sedikit manipulasi terhadap file tersebut maka diharapkan setiap alamat email dapat digantikan dengan gambar. Rancangan sistem yang akan dibangun tampak pada gambar 2.



Gambar 2. skema rancangan sistem

Beberapa hal yang harus disiapkan sebelum memulai adalah :

1. *Regular expression* untuk mendapatkan alamat email
2. fungsi untuk merubah *string*/teks menjadi gambar
3. Akses File *post-template.php* Wordpress

1. *Regular expression* untuk mendapatkan alamat email.

Untuk mendapatkan alamat email, dibithkan sebuah pola umum. Pola umum yang biasanya digunakan pada alamat email adalah username@domaniname. Contoh:

- `agus@yahoo.com`
- `puslitbudpar@unud.ac.id`
- `president@whitehouse.gov`

Untuk mendapatkan semua alamat email seperti tersebut di atas maka yang harus dilakukan adalah memverifikasi username dan domain name.

Aturan Username yang diperlukan adalah:

a. boleh huruf → **`A-Z`**
b. boleh angka → **`0-9`**
c. boleh gabungan antara huruf dan angka → **`A-Z0-9`**
d. boleh menggunakan karakter tambahan : “. _ % + - “ → **`._%+-`**
e. boleh terdiri atas 1 kata atau lebih → **`[A-Z0-9._%+-]+`**

sehingga jika digabungkan maka *regex* untuk username adalah sebagai berikut “[A-Z0-9._%+-]+” dengan arti boleh gabungan antara huruf, angka dan karakter khusus sebanyak sekali atau lebih. Kemudian jika karakter “@” langsung dimasukkan maka hasilnya adalah sebagai berikut : **“[A-Z0-9._%+-]+(@)”**.

Sebagai contoh berikut ini adalah beberapa *string*/alamat email yang cocok dengan *regex* di atas.

- foodsafety[at]unud.ac.id
- foodsafety[at]unud.ac.id
- **puslitbudpar@**unud.ac.id
- **president@**whitehouse.gov
- john.doe+*regex*buddy[at]gmail.com

Aturan domain name adalah :

a. terdiri atas satu kata → **(?:[A-Z0-9-])**
b. terdiri atas minimal satu kata yang dapat berupa gabungan antara angka atau huruf yang diakhiri oleh karakter titik → **(?:[A-Z0-9-]+\.)+**
c. kata paling akhir harus terdiri atas huruf saja dengan jumlah karakter antara 2-4 buah. → **([A-Z]{2,4})**

Sehingga jika digabungkan maka aturannya adalah **(?:[A-Z0-9-]+\.)+([A-Z]{2,4})** yang berarti terdiri atas minimal 2 kata, dimana *set* kata pertama boleh terdiri atas minimal satu kata yang merupakan gabungan antara huruf dan angka dengan diakhiri tanda titik dan kata selanjutnya haruslah kata dengan jumlah karakter antara 2 - 4 buah.

Sebagai contoh, berikut ini adalah beberapa *string*/alamat email yang cocok dengan *regex* di atas.

- foodsafety[at]**unud.ac.id**
- foodsafety[at]**unud.ac.id**
- puslitbudpar@**unud.ac.id**
- **president**@whitehouse.gov
- **john.doe**+*regex* buddy[at]**gmail.com**

Aturan username dan aturan domain name sudah didapatkan. Kemudian langkah selanjutnya adalah menggabungkan kedua aturan tersebut, seperti pada tabel 2.

Tabel 2. regex alamat email

Username	Domain name
[A-Z0-9._%+-]+(@)	(?:[A-Z0-9-]+\.)+([A-Z]{2,4})

Alamat email
[A-Z0-9._%+-]+(@)(?:[A-Z0-9-]+\.)+([A-Z]{2,4})

Jika diterapkan pada contoh di atas maka hasilnya adalah sebagai berikut :

- foodsafety[at]unud.ac.id
- foodsafety[at]unud.ac.id
- **puslitbudpar@unud.ac.id**
- **president@whitehouse.gov**
- john.doe+*regex*buddy[at]gmail.com

sehingga hasilnya sudah sesuai dengan yang diinginkan. Artinya *regex* sudah mampu melakukan parsing terhadap alamat email.

Setelah *regex* didapatkan, maka langkah selanjutnya adalah membuat fungsi tambahan dengan *parameter content* dan memberikan nilai balik/*return value* sebuah *content yang sudah berubah*.

```
function do_reg($text, $regex ){
preg_match_all($regex , $text, $result, PREG_PATTERN_ORDER);
  for ($i = 0; $i < count($result[0]); $i++) {
  $namaImage=str_replace("@","DI",$result[0][$i]);
  $namaImage = str_replace(".","DOT",$namaImage);
  $hasil= to_image($text,$result[0][$i],$namaImage);
  $text= str_replace($result[0][$i],$hasil,$text);
 }
return $text;
```

```
}
```

2. Menyiapkan fungsi untuk merubah *string*/teks menjadi gambar

Sebuah fungsi untuk merubah text menjadi gambar ditunjukkan oleh penggalan program di bawah ini. Cara yang dilakukan adalah dengan menggunakan parameter *$string* sebagai inputan alamat email yang akan diubah menjadi gambar. Kemudian dengan menggunakan fungsi *imagegif()* dihasilkanlah sebuah gambar dengan format gif dan diletakkan di folder "*wp-content/uploads/*". Setelah gambar terbentuk kemudian dilanjutkan dengan memberikan *return value* sebuah tag HTML berupa tag "*<img>*" dimana source gambarnya adalah sesuai dengan path tempat gambar tersebut disimpan.

```
$font = 3;
$width = ImageFontWidth($font)* strlen($string);
$height = ImageFontHeight($font);
$im = ImageCreate($width,$height);
$x=imagesx($im)-$width ;
$y=imagesy($im)-$height;
$background_color=imagecolorallocate($im,242,242,242
);
$text_color=imagecolorallocate ($im, 100, 100,234);
$trans_color = $background_color;
imagecolortransparent($im, $trans_color);
imagestring($im,$font,$x, $y, $string, $text_color);
imagegif($im,"wp-content/uploads/$namaImage.gif");
ImageDestroy($im);
$hasil ="<img src=\"".get_settings('siteurl')."/wp-
content/uploads/$namaImage.gif\"title=\"$namaImage\"
>";
```

3. Akses File post-template.php Wordpress

Melakukan perubahan pada pada fungsi *the_content()* yang terdapat pada file *post-template.php* yang terletak pada folder *wp-includes*.

Fungsi asli sebelum perubahan:

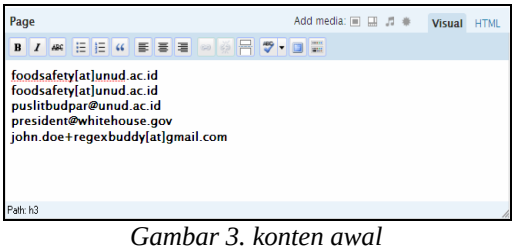
```
function the_content($more_link_text = '(more...)',
$stripteaser = 0, $more_file = '') {
$content=get_the_content($more_link_text,$striptease
r, $more_file);
  $content = apply_filters('the_content', $content);
  $content = str_replace(']]>', ']]&gt;', $content);
  echo $content ;
}
```

Fungsi setelah perubahan :

```
function the_content($more_link_text = '(more...)',
$stripteaser = 0, $more_file = '') {
  $content=get_the_content($more_link_text,
$stripteaser, $more_file);
  //mengganti semua email dengan image
  $regex  nya='/\b[A-Z0-9._%+-]+(\[at\]|@)(?:[A-Z0-9-
]+\.)+[A-Z]{2,4}\b/i';
  $content = do_reg($content,$regex nya);
  $content = apply_filters('the_content', $content);
  $content = str_replace(']]>', ']]&gt;', $content);
  echo $content ;
}
```

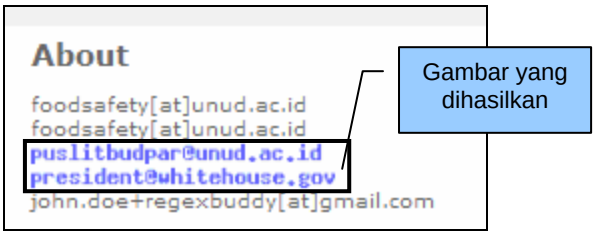
Tujuan dari perubahan ini adalah melakukan parsing alamat email menggunakan *regex* terhadap isi dari variable *$content*. Jika ditemukan pola seperti yang dimaksudkan, maka setiap alamat email tersebut akan diubah menjadi gambar.

Dengan menggunakan *content editor* milik wordpress tampak kodenya seperti gambar 3.



Gambar 3. konten awal

Kemudian saat konten tersebut *dipublish*, akan didapatkan hasil seperti pada gambar 4.



Gambar 4. konten setelah perubahan

Tampak pada gambat 4, bahwa alamat email yang cocok dengan pola *regex* berubah menjadi gambar. Sedangkan alamat email yang lainnya masih tetap dalam bentuk *string*/teks.

Php image yang digunakan pada penelitian ini, masih standar Mungkin dengan menggunakan tambahan program *Optical Character Recognized* (OCR) gambar alamat email yang dihasilkan oleh program ini dapat dibobol oleh *spam robot*. Namun hal tersebut tentu membutuhkan usaha yang tidak sedikit. Untuk itu penggunaan *Php image* yang lebih kompleks mungkin akan dapat meningkatkan kualitas perlindungan alamat email dari *spam robot*.

SIMPULAN

Dari hasil pengujian yang dilakukan pada pengembangan aplikasi konten wordpress, dapat disimpulkan bahwa :

1. fungsi *regex* yang dikemukakan oleh penulis berhasil menangani alamat email pada konten wordpress dan merubahnya menjadi gambar.
2. *regex* dapat digunakan sebagai salah satu tool untuk melindungi alamat email dari *spam robot* pada CMS Wordpress

DAFTAR PUSTAKA

Kuchling, A.M , Regular expression HOWTO, 2002

PHP : Introduction – Manual http://id2.php.net/manual/en/intro.image.php diakses pada tanggal 10 maret 2009, pk 21.14 wita

Regex - Henry Spencer's regular expression libraries http://arglist.com/regex diakses pada tanggal 10 maret 2009, pk 21.14 wita

Regular Expression – wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Regular_expression diakses pada tanggal 3 Februari 2009, pk 15.00 wita

Regular-Expressions.info - Regex Tutorial, Examples and Reference - Regexp Patterns http://www.regular-expressions.info diakses pada tanggal 10 maret 2009, pk 21.14 wita

Spamboot – wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Spambot diakses pada tanggal 24 maret 2009, pk 11.30 wita

STIKOM Bali Forum Melihat topik - Regular Expression (Regex), http://forum.stikom-bali.ac.id/viewtopic.php?f=28&t=313 diakses pada tanggal 24 maret 2009, pk 11.30 wita

Wordpress – Wikipedia Bahasa Indonesia, Ensiklopedia bebas http://id.wikipedia.org/wiki/WordPress diakses pada tanggal 4 Januari 2009, pk 09.20 wita