BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi dan pengujian terhadap sistem yang baru. Tahapan ini dilakukan setelah perancangan selesai dilakukan dan selanjutnya akan diimplementasikan pada bahasa pemrograman. Setelah implementasi maka dilakukan pengujian terhadap sistem yang baru dan akan dilihat kekurangan-kekurangan pada aplikasi yang baru untuk pengembangan sistem selanjutnya.

4.1 Implementasi

Implementasi merupakan penerapan dari seluruh rancangan yang sudah dibuat sebelumnya dengan aplikasi program yang sudah ditentukan dalam mengaplikasikan sistem keamanan website.

4.1.1 Batasan Implementasi

Sistem keamanan website ini adalah hanya untuk memblokir bot spam dan untuk mendeteksi komentar pada website.

4.1.2 Kebutuhan Perangkat Keras

Perangkat keras yang digunakan selama membuat sistem keamanan website ini memiliki spesifikasi sebagai berikut :

a. *Processor* Intel core 2 duo 1.50 GHz

b. RAM 1 GB

c. *Harddisk* 120 GB

d. *Color* monitor 15”

e. CD ROM 52x

f. *Keyboard* dan *mouse*

g. Modem, RJ 45 dan *Switch Hub*

4.1.3 Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan dalam pembuatan sistem keamanan website yaitu sebagai berikut :

a. Windows XP Professional SP 2

b. PHP 5

c. PHPMyAdmin – 3.0.0-dev

d. MySQL client version 5.0.32

e. Server version 5.0.32-Debian_7etch6-log

f. GD Library

g. Microsoft frontpage

h. Webhosting + Domain : skripsi.charleysakul.net

4.1.4 Implementasi Aplikasi

4.1.4.1 Tampilan Halaman Pengecekan Bot Spam

Bot atau robot akan diblokir pada bagian pertama ini, dimana semua yang melakukan pemostingan harus memasukkan dua (2) kode keamanan dengan benar. Bila pada tahap pertama ini kode yang dimasukkan salah, maka tidak akan

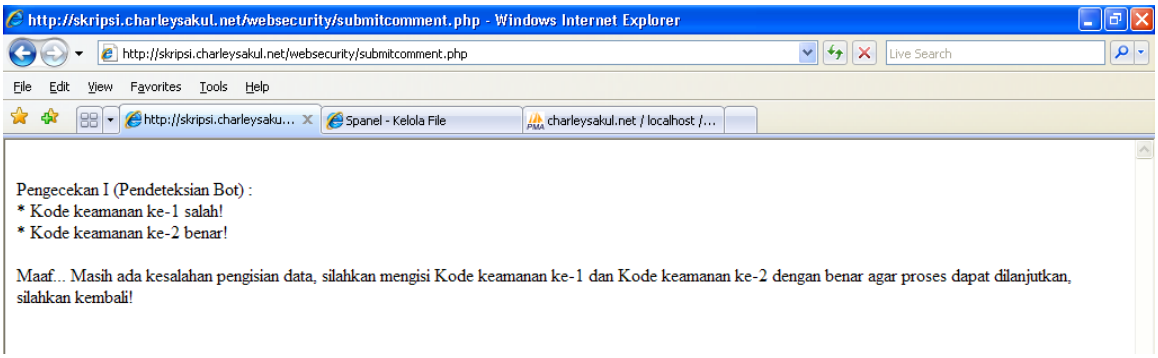
dilakukan pengecekan selanjutnya terhadap komentar dan otomatis pengiriman komentar gagal.



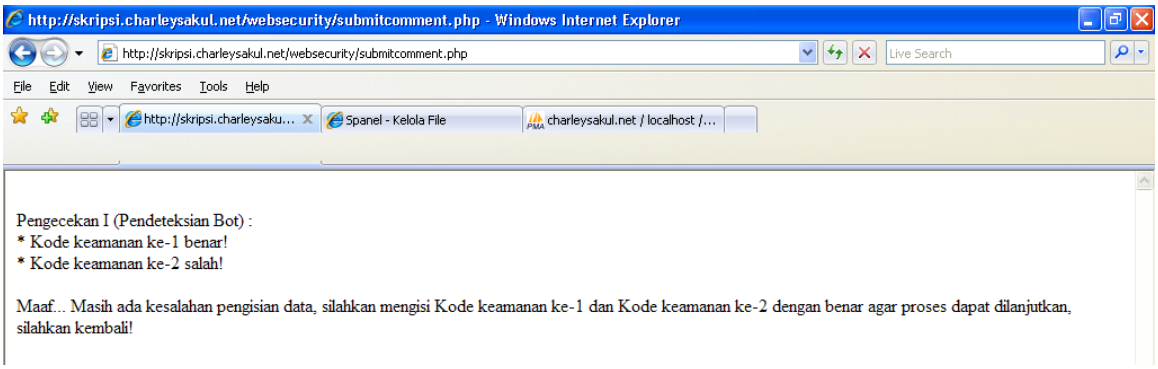
Gambar 4.1 Tampilan Kode keamanan ke-1 dan kode keamanan ke-2

Masing-masing kode keamanan ini selalu berganti secara random dan akan berubah setiap kali halaman ini diakses.

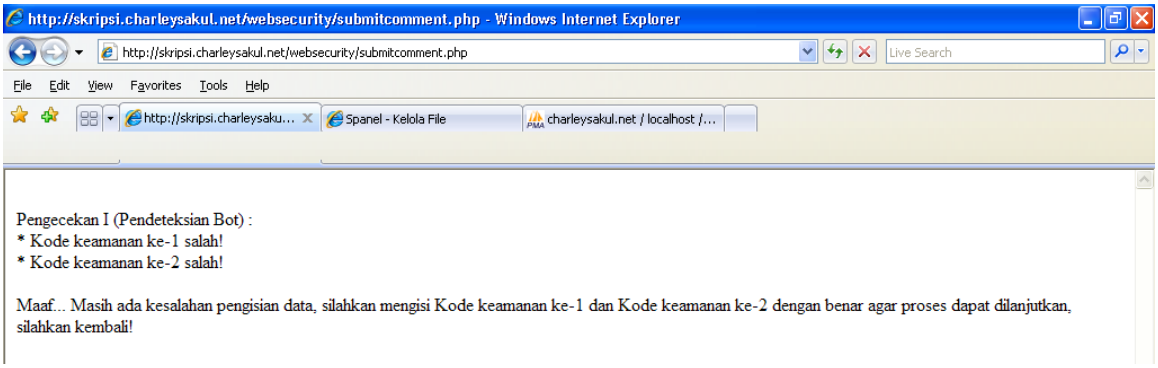
Berikut ini adalah tampilan halaman pengecekan 2 kode keamanan di atas, yang lebih ditujukan untuk pemblokiran bot (robot).



Gambar 4.2 Tampilan halaman bila kode keamanan 1 salah



Gambar 4.3 Tampilan halaman bila kode keamanan 2 salah

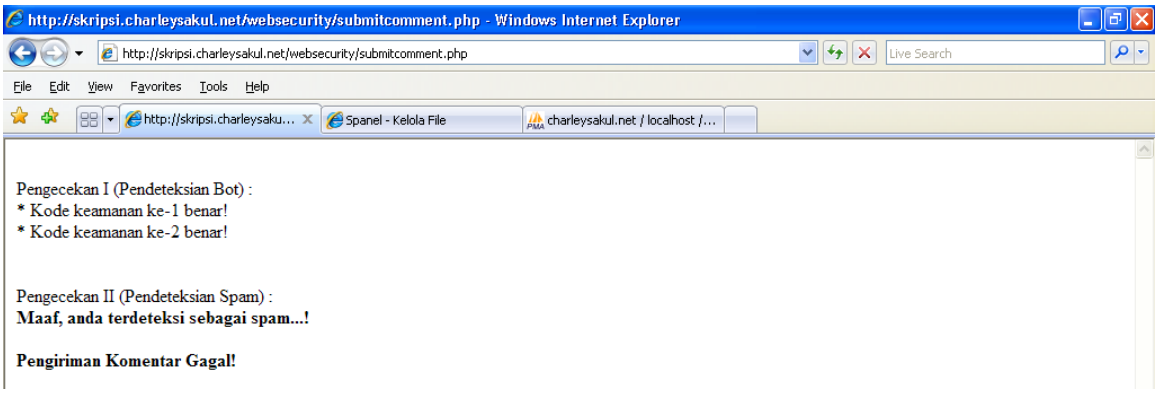


Gambar 4.4 Tampilan halaman bila kode keamanan 1 dan 2 salah

4.1.4.2 Tampilan Pengecekan Komentar

Sampai di tahap ini berarti user telah lulus melewati pengecekan I dan sekarang memasuki pengecekan II.

Pengecekan II ini adalah untuk mendeteksi semua komentar yang diisi oleh user. Pengecekan II ini menggunakan Algoritma Bayesian untuk memblokir spam.

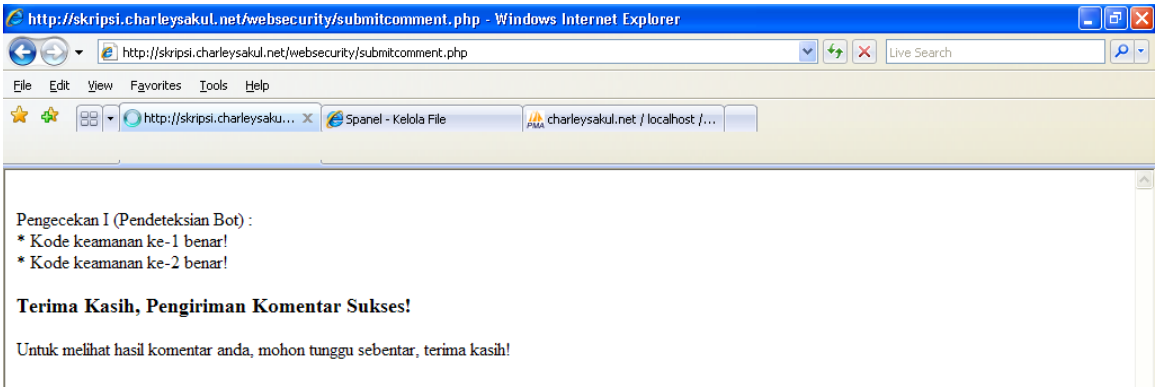


Gambar 4.5 Tampilan halaman pengecekan komentar

4.1.4.3 Tampilan Halaman Sukses

Bila user memasukkan kode keamanan ke-1 dan 2 dengan benar, dan isi komentar user tersebut tidak terdeteksi sebagai spam, berarti user telah berhasil dan proses pemostingan akan dilakukan.

Berikut ini adalah perancangan halaman sukses, yaitu bila user sukses melakukan posting.



Gambar 4.6 Tampilan halaman sukses

Setelah halaman sukses ini ditampilkan maka dalam waktu 2 detik, user akan di-*redirect* ke halaman utama, dimana semua hasil pemostingan ditampilkan di halaman utama.

4.2 Implementasi Class

4.2.1 Class Trainer

```
class trainer {
   var $examples;
   var $ngram;
   var $knowledge;
      function trainer() {
      $this->ngram = new ngram;
   }
      function add_example($text, $clasification) {
      $this->examples[$clasification][] = $text;
   }
      function setPreviousLearn($f) {
      $this->previous = $f;
   }
      function extractPatterns() {
      $previous = & $this->previous;
      $examples = & $this->examples;
      $ngram = & $this->ngram;
      $knowledge = & $this->knowledge;

      foreach($examples as $tipo => $texts) {
         $params[$tipo] = 0;
         $ngram->setInitialNgram( isSet($previous[$tipo]) ? $previous[$tipo] :
array() );
         foreach ($texts as $text) {
            $ngram->setText($text);
            for($i=3; $i <= 5;$i++) {
               $ngram->setLength($i);
               $ngram->extract();
            }
         }

         $actual = & $knowledge[$tipo];
         foreach( $ngram->getnGrams() as $k => $v) {
            $actual[$k]['cant'] = $v;
            $params[$tipo] += $v;
         }
      }
      $this->computeBayesianFiltering($params);
   }

   function computeBayesianFiltering($param) {
      $knowledge = & $this->knowledge;
```

```
        //print_r($param);
        //
        foreach($knowledge as $tipo => $caracterist) {
            foreach($caracterist as $k => $v) {
                $t = ($v['cant']/$param[$tipo]);
                $f = 0;
                foreach($param as $k1 => $v1)
                    if ( $k1 != $tipo) {

                        $f += isset($knowledge[$k1][$k]['cant']) ?
$knowledge[$k1][$k]['cant'] / $v1 : 0;
                    }
                $knowledge[$tipo][$k]['bayesian'] = $t / ($t + $f);
            }
        }
    }
}
```

4.2.2 Class Spam

```
class spam {
    var $_source;

    function spam($callback='') {
        if ( !is_callable($callback) ) {
            trigger_error("$callback is not a valid funciton",E_USER_ERROR);
        }
        $this->_source = $callback;
    }

    function isItSpam($text,$type) {
        $ngram = new ngram;
        $ngram->setText($text);

        for($i=3; $i <= 5;$i++) {
            $ngram->setLength($i);
            $ngram->extract();
        }

        $fnc = $this->_source;
        $ngrams =  $ngram->getnGrams();
        $knowledge =  $fnc( $ngrams,$type );
        $total=0;
        $acc=0;
        foreach($ngrams as $k => $v) {
```

```
        if ( isset($knowledge[$k]) ) {
            $acc += $knowledge[$k] * $v;
            $total++;
        }
    }
    $percent = ($acc/$total);
    $percent = $percent > 1.0 ? 1.0 : $percent;
    return $percent * 100;
}
function isItSpam_v2($text,$type) {
    $ngram = new ngram;
    $ngram->setText($text);

    for($i=3; $i <= 5;$i++) {
        $ngram->setLength($i);
        $ngram->extract();
    }

    $fnc = $this->_source;
    $ngrams =  $ngram->getnGrams();
    $knowledge =  $fnc( $ngrams,$type );
    $total=0;
    $acc=0;

    $N = 0;
    $H = $S = 1;

    foreach($ngrams as $k => $v) {
        if ( !isset($knowledge[$k]) ) continue;
        $N++;
        $value = $knowledge[$k] * $v;
        $H *= $value;
        $S *= (float)( 1 - ( ($value>=1) ? 0.99 : $value) );
    }

    $H = $this->chi2Q( -2 * log( $N *  $H), 2 * $N);
    $S = (float)$this->chi2Q( -2 * log( $N *  $S), 2 * $N);
    $percent = (( 1 + $H - $S ) / 2) * 100;
    return is_finite($percent) ? $percent : 100;
}
    function chi2Q( $x,  $v) {
    $m = (double)$x / 2.0;
    $s = exp(-$m);
    $t = $s;

    for($i=1; $i < ($v/2);$i++) {
```

```
            $t *= $m/$i;
            $s += $t;
        }
        return ( $s < 1.0) ? $s : 1.0;
    }
}
```

4.2.3 Class Ngram

```
class ngram {
    var $text;
    var $length;
    var $ngrams;
    function ngram($letter=1) {
        $this->setLength($letter);
    }
    function setLength($length=1){
        $this->length=$length;
    }

    function setText($text) {
        $this->text =" ".$text."  ";
    }

    function setInitialNgram($arg) {
        $this->ngrams = $arg;
    }

    function getnGrams() {
        return $this->ngrams;
    }

    function extract() {
        $txt = & $this->text;
        $len = strlen($txt);
        $length = & $this->length;
        $ngrams = & $this->ngrams;
        $buf='';
        $ultimo='';
        for($i=0; $i < $len; $i++) {
            if ( strlen($buf) < $length) {
                if ( !useful($txt[$i]) )
                    continue;

                if (is_space($txt[$i]) && is_space($ultimo))
```

```
                continue;

            $buf .= is_space($txt[$i]) ? '_' : $txt[$i];
            $ultimo = $txt[$i];
        } else {
            $buf = strtolower($buf);
            $buf = str_replace(" ","_",$buf);
            $ngrams[$buf] = isset($ngrams[$buf]) ? $ngrams[$buf] + 1 : 1;
            $ultimo='';
            $buf = '';
            /**
             *    The last caracter weren't accumulated, so decrement
             *    the counter and in the next itineration it will be
             *    hanled.
             */
            $i--;
        }
    }
  }
}

function is_space($f) {
  return $f==' ' || $f=="\n" || $f=="\r" || $f=="\t";
}
function useful($f) {
  $f = strtolower($f);
  return ($f >= 'a' && $f <= 'z') || is_space($f);
}
```

4.2.4 Class CaptchaSecurityImages

```
class CaptchaSecurityImages {

        var $font = 'arial';

        function generateCode($characters) {
                /* list all possible characters, similar looking characters and vowels
have been removed */
                $possible = '23456789bcdfghjkmnpqrstvwxyz';
                $code = '';
                $i = 0;
                while ($i < $characters) {
                        $code .= substr($possible, mt_rand(0, strlen($possible)-1),
1);
                        $i++;
```

```
		}
		return $code;
	}

	function
CaptchaSecurityImages($width='100',$height='40',$characters='6') {
		$code = $this->generateCode($characters);
		/* font size will be 75% of the image height */
		$font_size = $height * 0.50;
		$image = @imagecreate($width, $height) or die('Cannot initialize
new GD image stream');
		/* set the colours */
		$background_color = imagecolorallocate($image, 255, 255, 255);
		$text_color = imagecolorallocate($image, 20, 40, 100);
		$noise_color = imagecolorallocate($image, 100, 120, 180);
		/* generate random dots in background */
		for( $i=0; $i<($width*$height)/3; $i++ ) {
			imagefilledellipse($image, mt_rand(0,$width),
mt_rand(0,$height), 1, 1, $noise_color);
		}
		/* generate random lines in background */
		for( $i=0; $i<($width*$height)/150; $i++ ) {
			imageline($image, mt_rand(0,$width), mt_rand(0,$height),
mt_rand(0,$width), mt_rand(0,$height), $noise_color);
		}
		/* create textbox and add text */
		$textbox = imagettfbbox($font_size, 0, $this->font, $code) or
die('Error in imagettfbbox function');
		$x = ($width - $textbox[4])/2;
		$y = ($height - $textbox[5])/2;
		imagettftext($image, $font_size, 0, $x, $y, $text_color, $this->font
, $code) or die('Error in imagettftext function');
		/* output captcha image to browser */
		header('Content-Type: image/jpeg');
		imagejpeg($image);
		imagedestroy($image);
		$_SESSION['security_code'] = $code;
	}
}
$width = isset($_GET['width']) ? $_GET['width'] : '50';
$height = isset($_GET['height']) ? $_GET['height'] : '10';
$characters = isset($_GET['characters']) && $_GET['characters'] > 1 ?
$_GET['characters'] : '6';
$captcha = new CaptchaSecurityImages($width,$height,$characters);
```

4.3 Pengujian Sistem

Pengujian sistem merupakan hal terpenting yang bertujuan untuk menemukan kesalahan-kesalahan atau kekurangan-kekurangan pada perangkat lunak yang diuji. Pengujian bermaksud untuk mengetahui perangkat lunak yang dibuat sudah memenuhi kriteria yang sesuai dengan tujuan perancangan perangkat lunak tersebut.

Pengujian perangkat lunak ini menggunakan metode pengujian *white box*. Pengujian *black box* berfokus pada persyaratan fungsional perangkat lunak.

4.3.1 Pengujian Alpha

4.3.1.1 Skenario Pengujian

Skenario pengujian sistem keamanan website adalah sebagai berikut:

Tabel 4.1 Skenario Pengujian Alpha

KASUS DAN HASIL UJI					
Kriteria Pengecekan dan Data Input			Yang Diharapkan	Pengamatan	Kesimpulan
Yang Tampil:	Yang Diisi User:	Pendeteksian Sistem Keamanan Website:	Komentar Lolos	Komentar Lolos	Diterima
-Kode Keamanan 1: whs5x	-Kode Keamanan 1: whs5x	-Kode Keamanan 1: whs5x (Valid)			
-Kode Keamanan 2: 8 x 8	-Kode Keamanan 2: 64	-Kode Keamanan 2: 64 (Valid)			
-Komentar: (masih kosong, dan harus diisi user)	-Komentar: hello	- Komentar: hello (Bukan Spam)			

Yang Tampil: -Kode Keamanan 1: 3rdsw -Kode Keamanan 2: 9 + 9 -Komentar: (masih kosong, dan harus diisi user)	Yang Diisi User: -Kode Keamanan 1: 3rdsw -Kode Keamanan 2: 11 -Komentar: hello world	Pendeteksian Sistem Keamanan Website: -Kode Keamanan 1: 3rdsw (Valid) -Kode Keamanan 2: 18 (Tidak Valid) - Komentar: hello world (Bukan Spam)	Komentar Tidak Lolos	Komentar Tidak Lolos	Diterima
Yang Tampil: -Kode Keamanan 1: qmpj6 -Kode Keamanan 2: 2 + 2 -Komentar: (masih kosong, dan harus diisi user)	Yang Diisi User: -Kode Keamanan 1: qmpj6 -Kode Keamanan 2: 4 -Komentar: all free and free and free!!!	Pendeteksian Sistem Keamanan Website: -Kode Keamanan 1: qmpj6 (Valid) -Kode Keamanan 2: 4 (Valid) -Komentar: all free and free and free!!! (Spam)	Komentar Tidak Lolos	Komentar Lolos	Tidak Diterima
Yang Tampil: -Kode Keamanan 1: 8kpqb -Kode Keamanan 2: 6 x 8	Yang Diisi User: -Kode Keamanan 1: cnscc -Kode Keamanan 2: 52	Pendeteksian Sistem Keamanan Website: -Kode Keamanan 1: 8kpqb (Tidak Valid) -Kode Keamanan 2: 48 (Tidak Valid)	Komentar Tidak Lolos	Komentar Tidak Lolos	Diterima

-Komentar: (masih kosong, dan harus diisi user)	-Komentar: hai semua	-Komentar: hai semua (Bukan Spam)			
Yang Tampil: -Kode Keamanan 1: qdx6c -Kode Keamanan 2: 1 x 7 -Komentar: (masih kosong, dan harus diisi user)	Yang Diisi User: -Kode Keamanan 1: qdx6c -Kode Keamanan 2: 9 -Komentar: free PTC, free PTC, get money here!!	Pendeteksian Sistem Keamanan Website: -Kode Keamanan 1: qdx6c (Valid) -Kode Keamanan 2: 7 (Tidak Valid) -Komentar: free PTC, free PTC, get money here!! (Spam)	Komentar Tidak Lolos	Komentar Tidak Lolos	Diterima
Yang Tampil: -Kode Keamanan 1: rkvhd -Kode Keamanan 2: 9 + 5 -Komentar: (masih kosong, dan harus diisi user)	Yang Diisi User: -Kode Keamanan 1: rkvhc -Kode Keamanan 2: 10 -Komentar: duit gratis, duit gratis, klik disini	Pendeteksian Sistem Keamanan Website: -Kode Keamanan 1: rkvhd (Tidak Valid) -Kode Keamanan 2: 14 (Tidak Valid) -Komentar: duit gratis, duit gratis, klik disini (Spam)	Komentar Tidak Lolos	Komentar Tidak Lolos	Diterima

4.3.1.2 Kesimpulan Hasil Pengujian Alpha

Berdasarkan hasil pengujian alpha (fungsional) dengan kasus uji diatas dapat ditarik kesimpulan bahwa sistem keamanan website ini sudah bisa mendapatkan hasil yang diharapkan.

4.3.2 Pengujian Beta

Pengujian ini merupakan pengujian yang dilakukan pada beberapa website, yaitu, forum diskusi PHPBB versi 2x, buku tamu (*guest book*), dan pada form pengisian komentar. Pada forum diskusi PHPBB versi 2x dan pada buku tamu (*guet book*) benar-benar teruji, karena sebelumnya kedua website ini benar-benar terserang bot-bot spam yang sangat mengganggu dimana dalam sehari pasti terdapat 5-15 bot yang melakukan pengiriman komentar.

Pengujian ini meliputi 2 bagian yaitu :

1. Pemblokiran penyerangan bot yang akan mengirimkan komentar pada website.
2. Pendeteksian komentar yang dikirim pada website. Komentar yang akan dikirim dideteksi terlebih dahulu, apakah termasuk dalam kategori spam atau bukan.

Berikut ini adalah tabel hasil pengujian:

Tabel 4.2. Hasil Pengujian Beta

No	Pertanyaan	Jawaban			Presentase		
		Ya	Cukup	Kurang	Ya	Cukup	Kurang
1	Apakah sistem keamanan website ini dapat memblokir penyerangan bot spam	8	2	0	80%	20%	0%
2	Apakah sistem keamanan website ini dapat mendeteksi komentar spam yang akan dikirim	6	4	0	60%	40%	0%

4.3.3 Kesimpulan Hasil Pengujian Beta

Berdasarkan hasil pengujian beta sistem keamanan website ini dengan kasus uji diatas, dapat ditarik kesimpulan sebagai berikut:

1. Pada pemblokiran bot, sistem ini dapat berfungsi dengan baik, dimana sistem ini benar-benar dapat menangkal penyerangan bot yang akan mengirimkan komentar.

2. Pada pendeteksian komentar spam, sistem ini masih terus melakukan pembelajaran dan pengembangan, mengingat *spammer* yang terus melakukan segala cara dimana kata-kata yang akan dikirim dirangkai sedemikian rupa agar tidak terdeteksi oleh sistem.