SISTEM MAKLUMAT DOKUMEN RAHSIA (SMDR)

MUNIRAH MAZLAN
ROSSILAWATI SULAIMAN

Fakulti Teknologi & Sains Maklumat, Universiti Kebangsaan Malaysia

ABSTRAK

Projek ini bertujuan untuk menyediakan satu platform bagi menguruskan pengumpulan kertas soalan dan memfokuskan untuk membina sistem yang selamat dari serangan suntikan SQL. Sistem web merupakan salah satu platform yang sangat terdedah kepada serangan SQL. Oleh kerana sistem yang bakal dibina ini merupakan sistem web, jadi inisiatif bagi memastikan keselamatan sistem ini adalah dengan memastikan tiada serangan suntikan SQL yang boleh dilakukan. Seperti kita sedia maklum kertas soalan merupakan sesuatu hal yang sulit. Oleh itu, sebagai alternatif lain bagi memastikan kesulitan kertas soalan ini tidak dicerobohi satu sistem yang lebih cekap dan teratur perlu dibina. Secure PHP merupakan bahasa pengatucaraan yang akan digunakan dalam proses pembinaan sistem ini. Segala maklumat tentang kertas soalan disimpan dalam pengkalan data yang hanya boleh diakses oleh orang tertentu sahaja.

1. PENGENALAN

Kertas soalan peperiksaan merupakan sesuatu yang sulit. Oleh itu, menjadi tanggungjawab para pensyarah dan juga pihak terbabit untuk memelihara kesulitan nya. Proses-proses yang terlibat di sepanjang menyediakan kertas soalan ini juga perlulah menjadi rahsia. Sistem yang bakal dibina ini akan menjadi medium storan yang lebih selamat untuk menyimpan kertas soalan peperiksaan.. Selain itu, proses pengurusan kertas soalan ini akan menjadi lebih cekap, teratur dan selamat daripada diceroboh pihak yang tidak bertanggungjawab.

Sistem yang akan dibina ini lebih menumpukan untuk menghalang daripada suntikan SQL (SQL Injection). Hal ini kerana sistem web yang mengggunkan bahasa pengatucaraan PHP amat sinonim dengan kelemahannya iaitu sangat terdedah dengan serangan suntikan SQL. Sebagai contoh serangan suntikan SQL ialah membolehkan penyerang untuk mengakses atau memadam data di pengakalan data. Jadi ciri-ciri keselamatan amat penting bagi melindungi kesulitan sesebuah dokumen. Antara ciri ciri keselamatan yang diterapkan dalam sistem ini ialah seperti Kerahsiaan (Confidentiality), Kesahihan (Authenticity) dan Integrity (Integrity). Ciri ciri ini dapat memastikan maklumat yang masuk atau keluar dari sistem ini adalah makumat yang asal dan tidak dipalsukan. Kesahihan sumber maklumat tersebut juga boleh

dipastikan. Medium storan bagi pengumpulan kertas soalan ini dibina bertujuan untuk memudahkan pihak staff pengurusan. Hal ini dapat memudahkan staff untuk mengenal pasti semua kertas soalan tersebut sudah dimuat naik oleh pensyarah ke dalam sistem sebelum tarikh yang sepatutnya. Bagi memastikan tiada masalah berlaku seperti keciciran kertas soalan, sistem ini diharapkan dapat memudahkan pihak yang terlibat dalam mengendalikan kertas soalan dengan lebih selamat dan efisyen.

2. PENYATAAN MASALAH

Tugas-tugas seharian yang melibatkan penggunaan internet kian meningkat dari hari ke hari. Medium untuk perkongsian data seperti emel, dropbox dan sebagainya tidak dijamin akan isu keselamatannya. Segala maklumat yang dikongsi melaui internet mempunyai kebarangkalian untuk dicuri atau digodam. Menurut Ketua Pegawai Eksekutif Cyber Security Malaysia, Leftenan Kolonel (B) Husin Haji Jazri, “Dari Januari hingga September 2008, sebanyak 1346 insiden dilaporkan termasuk 471 insiden pencerobohan siber dan 55 insiden ancaman (Salmiah A Hamid 2008). Sistem yang akan dibina ini merupakan medium storan bagi menyimpan segala dokumen kertas soalan, maka segala data yang terlibat akan disimpan di dalam Sistem Pengurusan Pengkalan Data Relational (RDBMS). Bahasa pengatucaraan SQL terlibat dalam menguruskan data dalam pengkalan data ini boleh digynakan untuk mengakses, mengubah dan memadam data. Bagi serangan suntikan SQL, penyeerang boleh menembusi sistem log masuk sesuatu sistem tersebut. Mereka juga mampu memanipulasi data yang tedapat dalam pengkalan data. Beberapa teknik dan kaedah yang sesuai akan digunakan bagi membangunkan sistem yang selamat dari serangan suntikan SQL.Kerahsiaan adalah kunci utama kepada kredibiliti sesebuah kertas peperiksaan dan jika perkara tersebut tidak dapat dipertahankan, maka apa lagi harapan yang boleh disandarkan oleh rakyat terhadap sistem pendidikan negara untuk anak-anak pada masa akan datang. (Bartholomew Chan 2014).

3. OBJEKTIF

Objektif pelaksanaan projek ini adalah untuk membangunkan satu sistem storan bagi menyimpan kertas soalan yang memudahkan para pensyarah dan pihak pengurusan fakulti.

Selain itu juga, sistem ini juda diberi penekanan terhadap serangan suntikan SQL dan bagaimana cara untuk memastikan sistem ini selamat dari sebarang serangan suntikan SQL.

1. Mengenalpasti ciri ciri keselamatan yang sesuai bagi melaksanakan sistem yang selamat dengan menggunakan secure PHP.
2. Membangun sistem pengumpulan kertas soalan berdasarkan ciri-ciri keselamatan dalam (1).
3. Menguji sistem dengan menggunakan ujian penembusan (Penetration Testing) daripada serangan suntikan SQL.

4. METODOLOGI KAJIAN

Setiap projek memerlukan langkah langkah atau kaedah yang efisyen untuk memastikan projek ini dapat dibangunkan mengikut aturannya. Model air terjun digunakan bagi membangunkan sistem penyimpanan kertas peperiksaan ini. Kaedah air terjun ini mempunyai beberapa fasa penting iaitu fasa perancangan, fasa analisis, fasa rekabentuk, fasa pembangunan, fasa pengujian dan akhir sekali fasa implementasi. Model air terjun ini sesuai bagi pembangunan sistem ini kerana fasa-fasa yang terlibat menerangkan dengan lebih jelas dan terperinci akan proses pembangunan sistem ini. Metodologi ini juga membantu bagi memahami dengan lebih jelas bagaimana sistem ini bakal dibangunkan.

4.1 Fasa Perancangan

Proses yang terlibat dalam fasa perancangan adalah seperti mengenal pasti masalah, objektif untuk membangunkan sistem, dan juga skop pengguna sistem. Fasa perancangan merupakan fasa yang utama dalam memulakan pembangunan sistem. Carta Gantt dibina sebagai pelan perancangan agar sistem ini dapat disiapkan dalam jangka masa yang ditetapkan. Pelbagai rujukan dan maklumat digunakan bagi memastikan perancangan awal yang dibuat dapat dilaksanakan dengan jayanya.

4.2 Fasa Analisis

Setelah melalui fasa perancangan, fasa analisis dilakukan untuk mentafsir segala maklumat yang dicari. Analisis dilakukan untuk mengenalpasti kesesuaian teknik, perkakasan, perisian yang akan digunakan. Selain itu, kajian dan tafsiran maklumat dilakukan dengan lebih terperinci agar semua objektif dapat dicapai. Kajian terhadap sistem sedia ada juga dilakukan sebagai inisiatif untuk melakukan inovasi terhadap sistem yang akan dibangunkan ini.

4.3 Fasa Reka Bentuk

Fasa reka bentuk dilakukan bagi mengenal pasti antara muka yang sesuai bagi sistem yang akan dibangunkan. Reka bentuk antara muka sistem ini haruslah mesra pengguna dan tidak kompleks. Bahasa pengatucaraan seperti CSS dan Javascript digunakan bagi memastikan antara muka sistem ini nampak lebih menarik.

4.4 Fasa Pembangunan

Dalam fasa pembangunan, segala perancangan awal untuk membangunkan sistem perlu dilaksanakan. Masa yang agak lama diambil pada fasa ini untuk memastikan semua fungsi sistem berjaya dibangunkan. Pembagunan sistem ini uga perlu mengambik kira spesifikasi keperluan sistem seperti keperluan fungsian dan keperluan bukan fungsian.

4.5 Fasa Pengujian

Fasa pengujian merupakan fasa yang amat penting dalam membangunkan sistem ini. Hal ini kerana, pengujian terhadap keselamatan sistem haruslah dilakukan supaya selamat daripada serangan suntikan SQL. Oleh itu, perisian acunetix telah digunakan bagi melaksanakan ujian penembusan. Ujian penembusan dilakukan berulang kali bagi memastikan tiada sebarang serangan SQL terhadap sistem ini. Sekiranya terdapat serangan pengekodan perlu diubah sehingga mencapai hasil yang memuaskan.

4.6 Fasa Implementasi

Bagi fasa implementasi perkakasan dan perisian yang digunakan untuk membangunkan projek haruslah sesuai. Hal ini bagi memastikan pembangunan projek berjalan lancar dan semua fungsi sistem dapat dibangunkan. Perkakasan yang digunakan bagi membangunkan sistem ini ialah komputer riba mamakala perisian yang digunakan ialah, beberapa bahsa pengaticaraan seperti HTML, MySQL, PHP. Selain itu perisian acunetix digunakan bagi

melakukan ujian penembusan dan ingin mencari kelemahan sistem dari segi serangan suntikan SQL.

5. HASIL KAJIAN

Oleh kerana sistem yang dibina ini mefokuskan kepada menghalang serangan suntikan SQL pendekatan yang digunakan adalah dengan menguji keselamatan sistem ini. Perisian yang digunakan bagi menguji keselamatan sistem adalah Acunetix. Acunetix merupakan perisian yang boleh digunakan untuk mencari kelemahan dalam sesebuah sistem web. Acunetix akan memaparkan sebarang kelemahan yang terdapat dalam sistem web. Jenis imbasan yang digunakan ialah *SQL Injection Vulnerabilities*. Namun terdapat dua jenis kelemahan yang dipaparkan iaitu *SQL Injection* dan juga *Blind Sql Injection*. Namun apakah perbezaan kedua-dua serangan ini? Bagi serangan suntikan SQL, serangan berlaku apabila penyerang menyuntik pertanyaan sql yang berbahaya (*malicious sql query*) atas aplikasi web dan menyebabkan penyerang mengetahui data dalam pangkalan data.

Bagi suntikan SQL yang berjenis *Blind* serangan berlaku apabila penyerang menyuntik pertanyaan sql query dan melakukan cubaan *true/false* ke atas pangkalan data. Namun penyerang tidak akan mengetahui data yang terdapat dalam pangkalan data. Secara amnya, bagi suntikan SQL penyerang perlu mengetahui respon mesej ralat dari pangkalan data bagi melakukan serangan. Penyerang juga boleh mengubah struktur dalam pangkalan data. Bagi suntikan SQL berjenis Blind, penyerang tidak perlu menunggu maklum balas dari pangkalan data kerana penyerang menggunakan kaedah cuba jaya dengan menguji soalan *true/false* ke atas pangkalan data. Selain itu, penyerang tidak boleh mengubah struktur di pangkalan data seperti suntikan SQL.

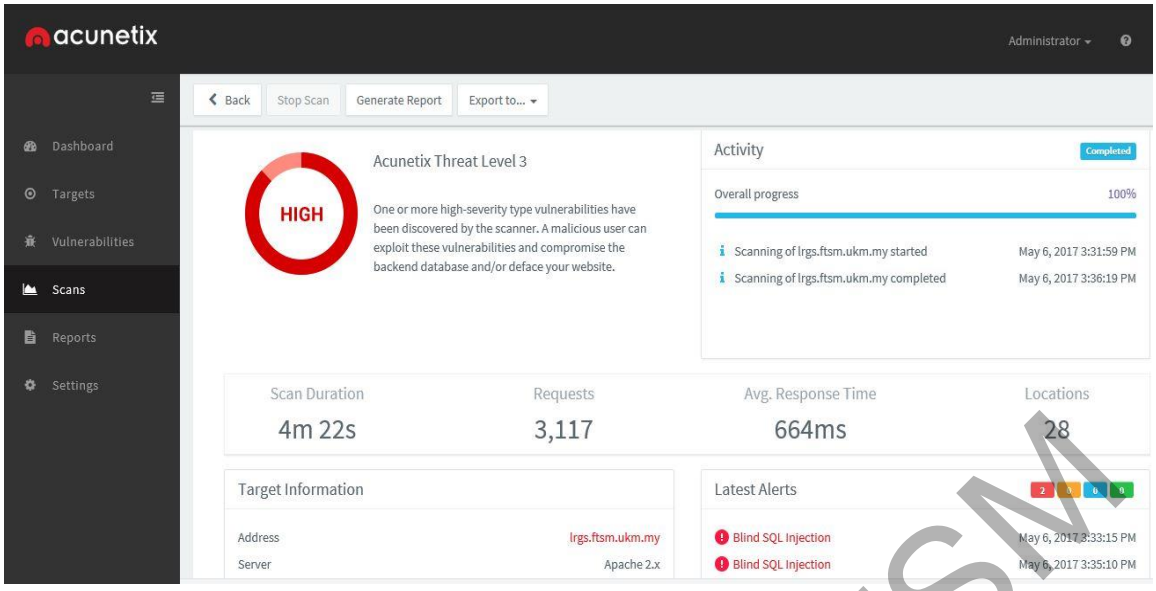
Setelah imbasan dilakukan, terdapat banyak kelemahan berkaitan suntikan SQL. Rajah 5.1 menunjukkan laporan imbasan oleh Acunetix. Imbasan ini dilakukan sebelum penambahan kod PHP yang selamat.

a) Log Masuk Pensyarah/Staff Pengurusan

Rajah 5.1 Imbasan oleh perisian Acunetix untuk log masuk

Perbandingan dua jenis kod dilakukan bagi mengenal pasti kod yang sesuai digunakan bagi menghalang serangan suntikan SQL. Bagi cubaan pertama untuk mengaplikasikan kod PHP selamat, MySQLi digunakan bagi menggantikan penggunaan MySQL. MySQLi merupakan lanjutan daripada MySQL dan merupakan versi yang lebih baik. Terdapat beberapa kelebihan penggunaan MySQLi berbanding MySQL. Salah satunya adalah ia menyokong *prepared statement* dan membantu untuk menghantar data ke pangkalan data dengan cara yang selamat dan melindungi dari suntikan SQL. Hal ini kerana, *prepared statement* tidak menggabungkan pembolehubah yang digunakan dengan rentetetan SQL. Pembolehubah dihantar secara berasingan dan bukan sebahagian daripada penyata SQL. Jadi, penyerang tidak mungkin dapat mengubah suai kenyataan SQL jika *prepared statements* digunakan di dalam kod. Setelah kod diubah, bilangan kelemahan dari segi suntikan SQL telah berkurang selepas diimbas menggunakan perisian Acunetix.

Rajah 5.2 menunjukkan paparan setelah pautan bagi log masuk pengguna diimbas. Manakala rajah 5.3 (a) dan (b) menunjukkan perbezaan kod bagi log masuk pengguna.



Rajah 5.2 Imbasan oleh perisian Acunetix untuk log masuk

```php
$result = mysql_query($sql);
if ($result){
//get the number of results based n the sql statement
$numrows = mysql_num_rows($result);
//check the number of result, if equal to one
//IF theres a result
if ($numrows == 1) {
//store the result to a array and passed to variable found_user
$found_user = mysql_fetch_array($result);
```

(a) Dengan kod dengan PHP selamat

```php
$result = mysqli_query(  $conn, $sql);
if ($result){
//get the number of results based n the sql statement
$numrows = mysqli_num_rows($result);
//check the number of result, if equal to one
//IF theres a result
if ($numrows == 1) {
//store the result to a array and passed to variable found_user
$found_user = mysqli_fetch_array($result);
```

(b) Tanpa kod PHP selamat

Rajah 5.3 Perbandingan kod bagi log masuk pengguna

Setelah diimbas, ternyata penggunaan lanjutan MySQLi membantu dari segi menghalang suntikan SQL. Namun, kod masih belum selamat bagi log masuk pengguna dan perlu dibaiki dengan mengaplikasi teknik yang lain untuk fasa yang kedua.

Bagi fasa kedua, kod *Htmlentities.* digunakan bagi membantu untuk menghalang suntikan SQL. Namun, *Htmlentities.* bukanlah teknik untuk menghalang serangan SQL, akan tetapi mampu menghalang dari serangan SQL berjenis Blind. Sebagai contoh bagi log masuk, sudah pasti pengguna perlu memasukkan nilai (*user input*) seperti id pengguna dan kata laluan. *Htmlentities* akan mengubah aksara kepada *Htmlentities* dan mengelakkan input pengguna diekploitasi oleh penyerang. Setelah itu, ujian penembusan terakhir dilakukan dan

hasilnya, pautan bagi log masuk telah bebas dari serangan suntikan SQL berjenis *Blind.* Rajah 5.4 (a) dan (b) menunjukkan perbezaan bagi kod yang sudah ditambah *Htmlentities.*

```
<input class="form-control" name="user_id" type="text" autofocus
value="<?php echo htmlentities($ousername, ENT_QUOTES,
'UTF-8'); ?>" >
```

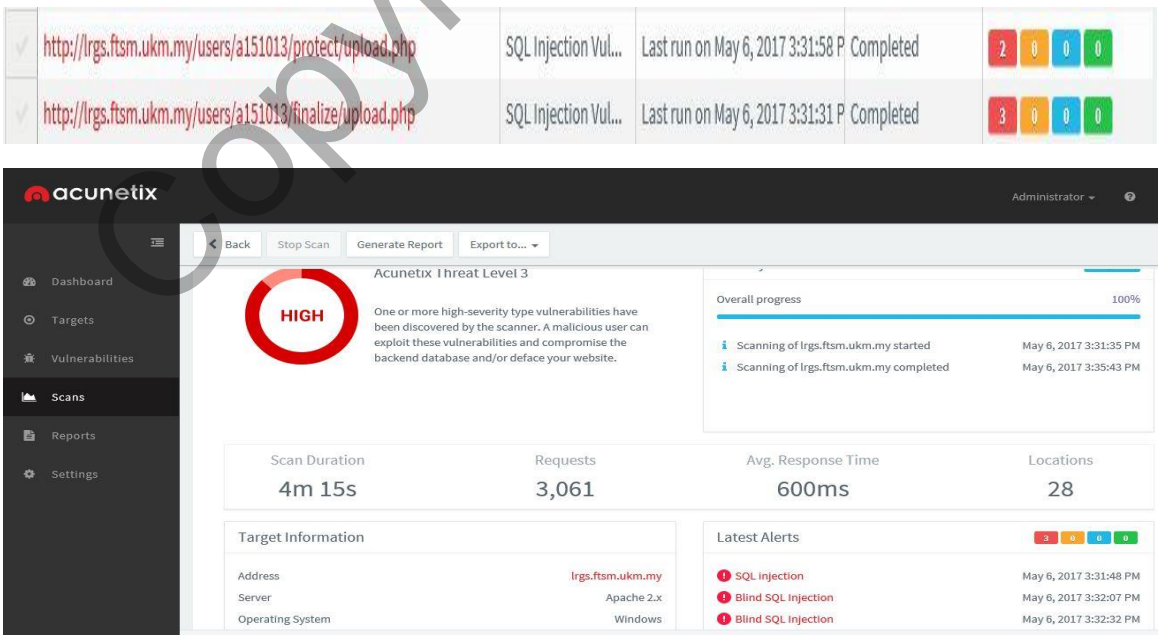
```
<input class="form-control" placeholder="User id" name="user_id"
type="text" autofocus>
```

(a) Dengan kod dengan PHP selamat (b) Tanpa kod PHP selamat

Rajah 5.4 Perbandingan kod *Htmlentities.*

b) Muat Naik Dokumen

Imbasan pertama dilakukan dan didapati terdapat tiga kelemahan iaitu satu suntikan SQL dan dua suntikan SQL berjenis Blind seperti dalam rajah 5.5. Manakala imbasan kedua dilakukan setelah kod diubah dengan menggunakan lanjutan MySQLi bagi menggantikan lanjutan MySQL. Seperti dinyatakan sebelum ini MySQLi dalam membantu menghalang serangan suntikan SQL. Rajah 5.6 (a) dan (b) menunjukkan perbezaan kod penggunaan lanjutan MySQL dan juga MySQLi.



Rajah 5.5 Imbasan oleh perisian Acunetix bagi muat naik dokumen

```
 $conn =
mysqli_connect("lrgs.ftsm.ukm.my","a151013","a151013");
        if (!$conn)
 echo('Could not connect: ' . mysqli_error());
 else{
 if (file_exists("uploads/" . $_FILES["file"]["name"]))
         {
echo '<script language="javascript">alert(" Sorry!!
Filename Already Exists...")</script>';}
          else{
move_uploaded_file($_FILES["file"]["tmp_name"],
           "uploads/" . $_FILES["file"]["name"]) ;
           mysqli_select_db($conn, "a151013" );
```

(a) Dengan kod dengan PHP selamat

```
$con =
mysql_connect("lrgs.ftsm.ukm.my","a151013","a151013");
if (!$con)
echo('Could not connect: ' . mysql_error());
else{
if (file_exists("uploads/" . $_FILES["file"]["name"]))
                    {
echo '<script language="javascript">alert(" Sorry!!
Filename Already Exists...")</script>';}
else{
     move_uploaded_file($_FILES["file"]["tmp_name"],
     "uploads/" . $_FILES["file"]["name"]) ;
     mysql_select_db("a151013", $con);
```

(b) Tanpa kod PHP selamat

Rajah 5.6 Perbandingan kod sebelum dan selepas lanjutan MySQLi digunakan

Kemudian bagi fasa kedua teknik yang berlainan digunakan iaitu dengan menggunakan *prepared statement*. Bagi memanggil data dari pangkalan data, penggunaan *prepared statement* merupakan praktik yang terbaik bagi menghalang serangan suntikan SQL dengan *sanitize input* dari pangkalan data. Modul PHP PDO bersama dengan *prepared statement* mampu mengelakkan serangan suntikan SQL. Bagi memanggil data dari pangkalan data adalah bagus sekiranya kita menggunakan kaedah selamat seperti membersihkan atau mengelakkan *(sanitize and escape*) dari menggunakan teknik yang boleh mengundang serangan suntikan SQL. Rajah 5.7 (a) dan (b) menunjukkan cara penggunaan *prepared statement* bagi memanggil data dari pangkalan data.

```
$sql = $conn->prepare("INSERT INTO
listofdoc_a151013(fld_namadokumen,
     fld_kodsubjek,  fld_namasubjek, fld_file) VALUES ('" .
$_POST["namadokumen"] ."','" . $_POST["kodsubjek"] ."', '" .
$_POST["namasubjek" ] . "','" .
          $_FILES["file"]["name"] ."');");
```

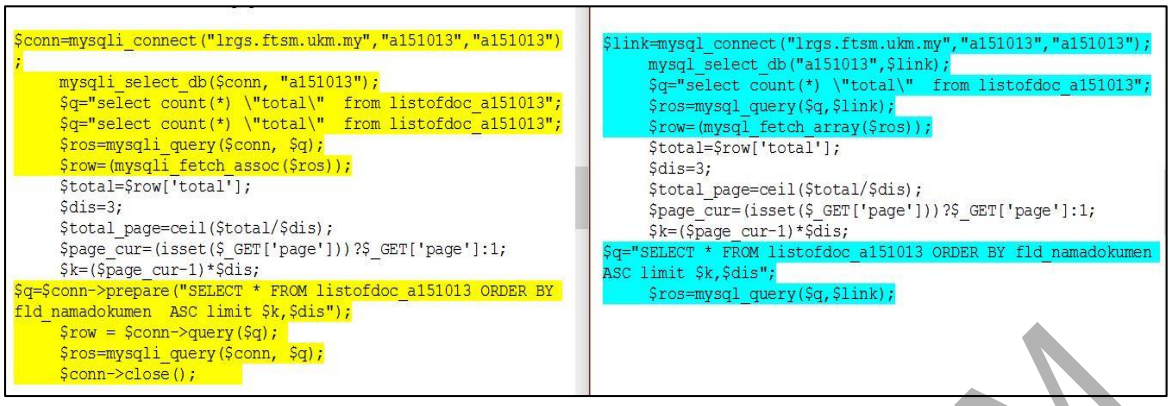
(a) Dengan kod dengan PHP selamat

```
    $sql = "INSERT INTO listofdoc_a151013(fld_namadokumen,
     fld_kodsubjek,  fld_namasubjek, fld_file) VALUES ('" .
$_POST["namadokumen"] ."','" . $_POST["kodsubjek"] ."', '" .
$_POST["namasubjek" ] . "','" .
$_FILES["file"]["name"] ."');";
```

(b) Tanpa kod PHP selamat

Rajah 5.7 Perbandingan kod bagi pre*pared statement*

c) Muat turun dokumen



```
$conn=mysqli_connect("lrgs.ftsm.ukm.my","a151013","a151013")
;
     mysqli_select_db($conn, "a151013");
     $q="select count(*) \"total\"  from listofdoc_a151013";
     $q="select count(*) \"total\"  from listofdoc_a151013";
     $ros=mysqli_query($conn, $q);
     $row=(mysqli_fetch_assoc($ros));
     $total=$row['total'];
     $dis=3;
     $total_page=ceil($total/$dis);
     $page_cur=(isset($_GET['page']))?$_GET['page']:1;
     $k=($page_cur-1)*$dis;
$q=$conn->prepare("SELECT * FROM listofdoc_a151013 ORDER BY
fld_namadokumen  ASC limit $k,$dis");
     $row = $conn->query($q);
     $ros=mysqli_query($conn, $q);
     $conn->close();
```

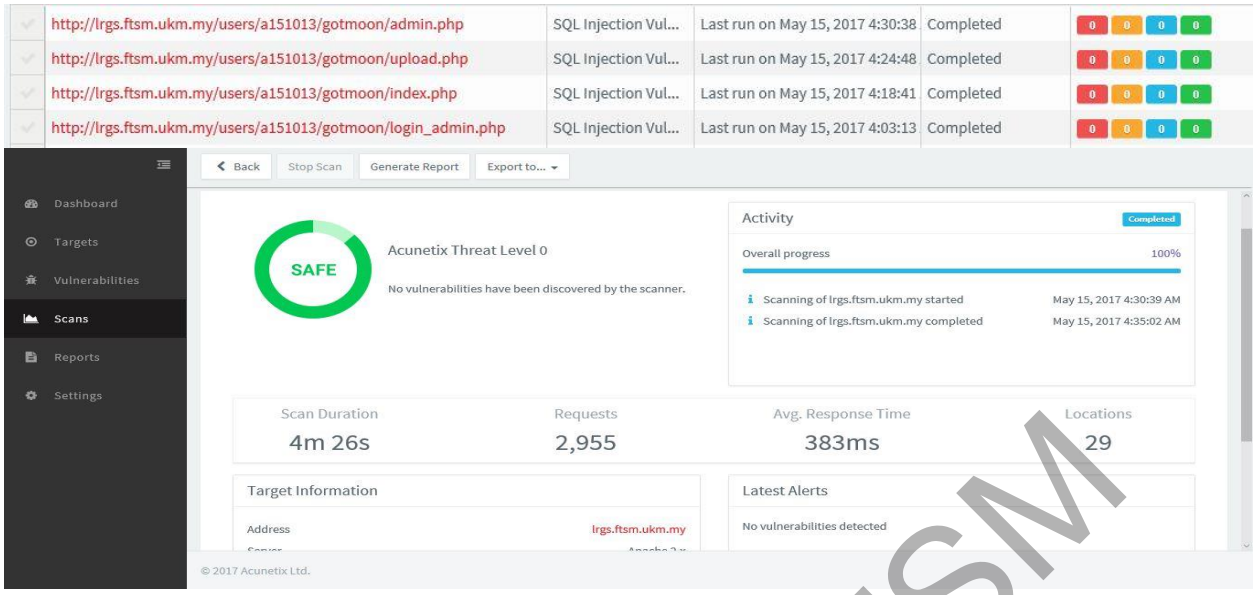
```
$link=mysql_connect("lrgs.ftsm.ukm.my","a151013","a151013");
     mysql_select_db("a151013",$link);
     $q="select count(*) \"total\"  from listofdoc_a151013";
     $ros=mysql_query($q,$link);
     $row=(mysql_fetch_array($ros));
     $total=$row['total'];
     $dis=3;
     $total_page=ceil($total/$dis);
     $page_cur=(isset($_GET['page']))?$_GET['page']:1;
     $k=($page_cur-1)*$dis;
$q="SELECT * FROM listofdoc_a151013 ORDER BY fld_namadokumen
ASC limit $k,$dis";
     $ros=mysql_query($q,$link);
```

(a) Dengan kod dengan PHP selamat (b) Tanpa kod PHP selamat

Rajah 5.8 Perbandingan kod bagi MySQLi dan pre*pared statement*

Rajah 5.8 (a) dan (b) menunjukkan perbezaan bagi kod dengan PHP selamat dan tanpa kod PHP selamat. Kaedah yang sama digunakan iaitu menggantikan lanjutan MySQL dengan lanjutan MySQLi dan juga kaedah *prepared statement.* Sama seperti sebelumnya, dengan menggunakan *prepared statement* dalam PHP dapt mengelakkan sistem dari suntikan SQL. Apa yang penting setiap data yang dihantar ke pangkalan data haruslah di *sanitize* dan cara yang paling mudah untuk melaksanakan adalah dengan menggunakan *prepared statement.*

Pautan sekali lagi diimbas bagi mengenalpasti masih adakah lagi kelemahan dalam sistem. Sekiranya tiada kelemahan atau potensi untuk sistem untuk diserang serangan suntikan SQL, paparan seperti gambar rajah 5.9 akan dipaparkan bermakna selamat dari serangan suntikan SQL.



Rajah 5.9 Imbasan Acunetix pautan yang selamat dari suntikan SQL

5. KESIMPULAN

Bagi memastikan sistem ini berjalan dengan lancar dan yang paling penting selamat untuk digunakan fasa pengujian memainkan peranan yang amat penting. Pelbagai kaedah yangdigunakan untuk mengenal pasti kelemahan sistem agar selamat dari serangan SQL. Antarapendekatan yang digunakan bagi mengenal pasti kelemahan sistem adalah dengan melakukan ujian penembusan dengan menggunakan perisian Acunetix. Bagi menangani serangan suntikan SQL kaedah secure PHP digunakan iaitu dengan mengimplementasi lanjutan MySQLi dan prepared statement Akhir sekali, diharapkan segala isu yang berkaitan serangan suntikan SQL dapat diselesaikan dan sistem ini selamat dari serangan suntikan SQL.