Handout – Session Hijacking

Dieses Dokument entstand als Handout zur Präsentation bezüglich des Themas «Session Hijacking» von Silas Näf. In diesem Dokument wird das Thema Session Hijacking näher erklärt und auch ein Codebeispiel dazu gezeigt.

Bei fast allen Webapplikationen werden heutzutage Sessions verwendet um Benutzerdaten zwischen zu speichern. Diese helfen die Benutzerfreundlichkeit der Applikation zu verbessern, da der Benutzer sich nicht immer neu Einloggen muss. Zudem werden Sessions benutzt um die Benutzerdaten bei der Navigation auf eine neue Seite präsent zu halten.

Was ist Session Hijacking?

Unter Session Hijacking versteht man das Übernehmen von Sessions in einer Webapplikation. Die Gefahr einer solchen Übernahme besteht darin, dass der Angreifer vollen Zugriff auf die Applikation des geklauten Users hat. Besonders gefährlich wird es dann, wenn zum Beispiel ein Administrator Account übernommen wird. Der Angreifer hat dann nämlich kompletten Zugriff auf die gesamte Applikation und kann sogar Daten ändern oder auslesen. Besonders gefährlich ist Session Hijacking deshalb, weil der User dessen Session übernommen wurde, gar nicht mitkriegt das seine Session übernommen wurde.

Das Ziel von Session Hijacking ist normalerweise das Auslesen/Manipulieren von Daten. Im schlimmsten Fall, können dem User sogar manipulierte Daten angezeigt werden. Da auch heute noch viele Webseiten die Daten unverschlüsselt übertragen(HTTP), ist es recht einfach eine Session zu übernehmen.

Wie funktioniert Session Hijacking?

Die meisten Applikationen sind grundsätzlich relativ sicher und gut geschützt gegen Angriffe. Eine wirkliche Gefahr besteht nur, wenn die Applikation auf PHP & MySQL basiert. Und auch da nur wenn die Übertragung ohne Verschlüsselung(HTTP) stattfindet. Das Übernehmen einer Session ist nur möglich, wenn der Angreifer im gleichen Netzwerk ist oder durch einen Trojaner den Netzwerkverkehr abhören kann.

Wie verhindert man Session Hijacking?

Heutzutage kann mich sich recht einfach gegen jegliche Session Hijacking Attacken schützen. Wenn die Daten verschlüsselt übertragen(HTTPS) werden, ist man praktisch immun gegen jegliche Session Hijacking versuche. Auch kann bei der Programmierung auf Sicherheit geachtet werden, dies in Kombination mit verschlüsselter Übertragung sichert die Applikation gegen praktisch alle Angriffe.

Beispiel

Um zu demonstrieren wie eine Session geklaut werden kann, ist im folgenden Teil der gesamte benötigte Programmcode sowie der Vorgang niedergeschrieben.

StartSession.php

```php
<?php
// Start the session
  session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
  $_SESSION["favcolor"] = "green";
  $_SESSION["favanimal"] = "cat";
  echo "Session variables are set.";
?>

</body>
</html>
```

DestroySession.php

```php
<?php
  session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// remove all session variables
  session_unset();

// destroy the session
  session_destroy();
?>

</body>
</html>
```

ModifySession.php

```php
<?php
  session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// to change a session variable, just overwrite it
  $_SESSION["favcolor"] = "yellow";
  echo "Favorite color set to " . $_SESSION["favcolor"] . ".<br>";
?>

</body>
</html>
```

Schritt 1: Im ersten Browser die Seite **StartSession.php** aufrufen.

Schritt 2: Im ersten Browser die Seite **GetSession.php** aufrufen, Werte prüfen.

Schritt 3: Mit dem ersten Browser über die Browserkonsole auf die SessionID zugreifen, die SessionID kopieren.

Schritt 4: Mit einen anderen Browser **StartSession.php** aufrufen und in der Browserkonsole die SessionID mit der aus dem anderen Browser austauschen.

GetSession.php

```php
<?php
  session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Echo session variables that were set on previous page
  echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
  echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>

</body>
</html>
```

Schritt 5: Auf dem ersten Browser die Seite **ModifySession.php** aufrufen. Die Farbe wird auf «yellow» gesetzt.

Schritt 6: Im zweiten Browser die Seite **GetSession.php** aufrufen, die Farbe wird auf «yellow» angezeigt. Dies beweist das im zweiten Browser die Session des ersten Browsers verwendet wird.