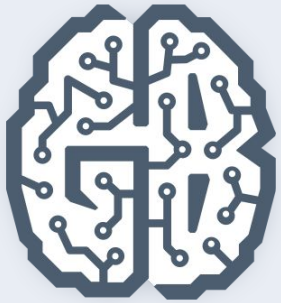
Client-side уязвимости

Урок 1



Введение в Cross Site Scripting (XSS)

Как возникают и чем опасны уязвимости XSS.

Регламент курса

- 8 уроков по 2 часа.
- Домашние задания.
- Виртуальная машина для практики.
- Видеозаписи уроков будут выкладываться.
- Задавайте вопросы.

Что будем изучать на курсе?

- Мы будем рассматривать уязвимости клиентской части веб-приложений, которые приводят к XSS. Такие уязвимости несут угрозу непосредственно клиентам веб-приложения и их данным.
- Мы научимся находить уязвимости XSS и настраивать защиту от них.

Что получим по окончании курса?

1. Изучим признаки наличия уязвимостей, которые приводят к атакам на клиентов.
2. Изучим на практике способы реализации злоумышленником XSS-атак.
3. Научимся находить уязвимости клиентской части веб-приложения и настраивать защиту от атак, использующих эти уязвимости.

Чем опасны уязвимости клиентской части веб-приложения?

1. Кража пользовательских данных.
2. Возможность подделки запросов пользователей.
3. Взятие под контроль браузера пользователя.

План урока

1. Что понимают под XSS.
2. Как возникают уязвимости XSS.
3. Виды XSS.
4. Практика.
5. Вопросы участников.

Что понимают под XSS

Что понимают под XSS

Уязвимость межсайтового скриптинга (англ. Cross Site Scripting или XSS) заключается в том, что существует возможность внедрения JavaScript кода в веб-страницу. Вводимые данные при этом не проверяются.

То есть данные вида:
<script>alert(document.cookie)</script>
так и вставляются на возвращаемую пользователю страницу.

Switch to SOAP Web Service Version of this Page

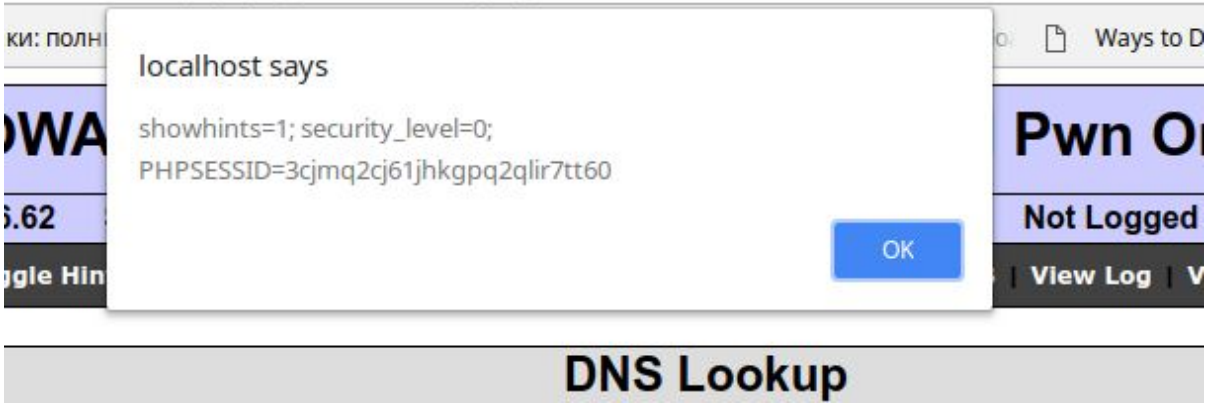
Who would you like to do a DNS lookup on?

Enter IP or hostname

Hostname/IP <script>alert(document.cookie)</script>

Lookup DNS

```
<div class="report-header" ReflectedXSSExecutionPoint="1">Results for
<script>alert(document.cookie)</script></div><pre class="report-header"
style="text-align:left;"></pre>
        <!-- I think the database password is set to blank or perhaps sa
```


localhost says
showhints=1; security_level=0;
PHPSESSID=3cjmq2cj61jhkgpq2qlir7tt60
OK
Ways to D
Pwn O
Not Logged
View Log
DNS Lookup

Опасность для жертвы:

Злоумышленник может составить сценарий, позволяющий, используя XSS, получить данные, к которым должен иметь доступ только пользователь веб-приложения (например, session id) и использовать их в своих целях (например, «угнать» сессию пользователя).

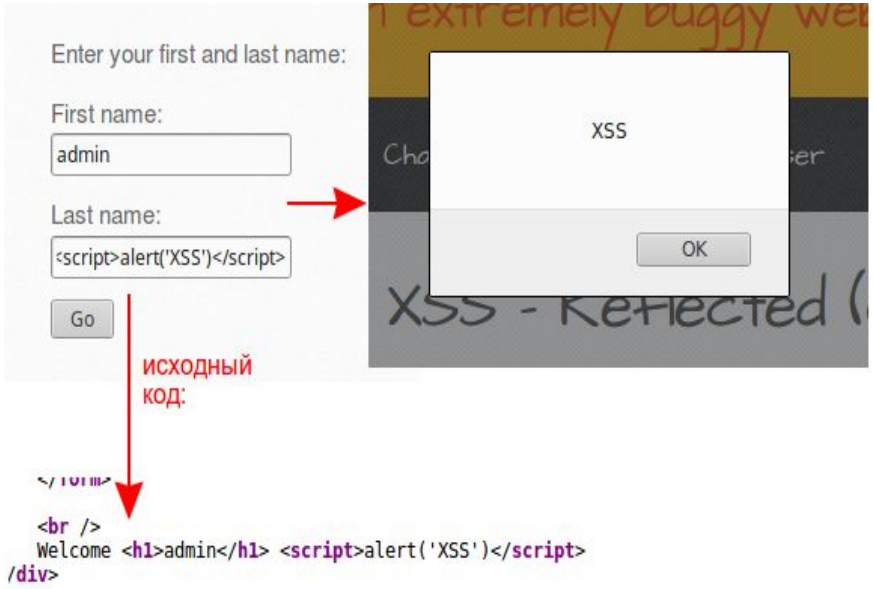
Используя XSS, злоумышленник:

- Получает доступ к пользовательским данным.
- Может вносить любые изменения во внешний вид страницы.
- Может использовать для атак программы на JavaScript, например кейлоггеры.
- Может управлять браузером пользователя.

Причины возникновения XSS

Причины возникновения XSS

Введенные пользователем данные не фильтруются, а вставляются на страницу «как есть», с сохранением форматирования.



Причины возникновения XSS

Данные пользователя, полученные из внешних источников, передаются и сохраняются с сохранением форматирования. В исходном коде серверной части это выглядит так:

```
<?php
$name=$_POST["login"];
?>
<html>
<h1>Login is:</h1>
<p><?php echo $name?><p>
</html>
```

Причины возникновения XSS

На странице присутствует объект (тег, код на JS), который динамически изменяет исходный код страницы. Например:

```
<select><script>

document.write("<OPTION
value=1>"+document.location
.href.substring(document.lo
cation.href.indexOf("defaul
t=")+8)+"</OPTION>");

</script></select>
```

Основные заблуждения относительно XSS

Ожидание:

XSS – это проблема сервера или серверной части.

Реальность:

При наличии XSS атаке подвергнутся в первую очередь клиенты, а на сервере не останется об этом информации в логах.

File Edit View Terminal Tabs Help

```
shodin@shodinpc:~$ cat /var/log/apache2/access.log|grep alert
shodin@shodinpc:~$ cat /var/log/apache2/access.log|grep document.cookie
shodin@shodinpc:~$
```

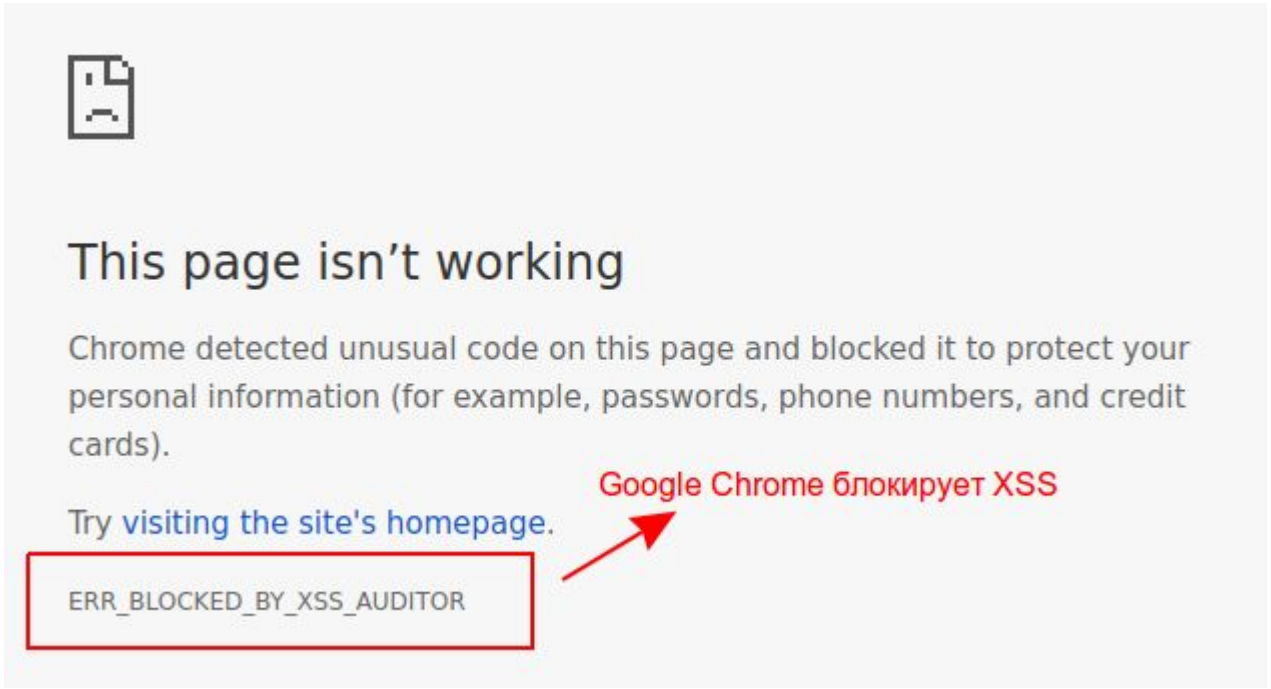
Основные заблуждения относительно XSS

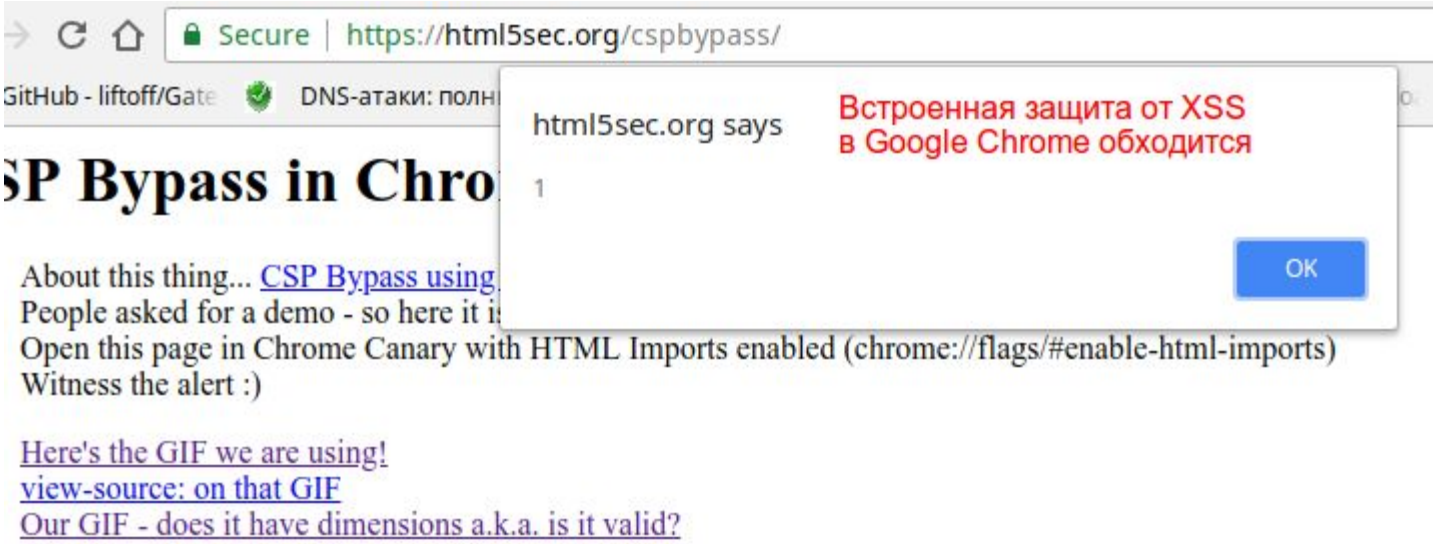
Ожидание:

В современных браузерах существует защита от XSS

Реальность:

В Firefox защиты нет, если не установить специальный плагин no-script.
В Google Chrome защита есть, но ее можно обойти.
В остальных браузерах ситуация аналогичная.


This page isn’t working
Chrome detected unusual code on this page and blocked it to protect your personal information (for example, passwords, phone numbers, and credit cards).
Try visiting the site's homepage.
ERR_BLOCKED_BY_XSS_AUDITOR
Google Chrome блокирует XSS


Secure | https://html5sec.org/cspbypass/
GitHub - liftoff/Gate
DNS-атаки: полн
html5sec.org says
Встроенная защита от XSS
в Google Chrome обходится
1
OK
SP Bypass in Chro
About this thing... CSP Bypass using
People asked for a demo - so here it i
Open this page in Chrome Canary with HTML Imports enabled (chrome://flags/#enable-html-imports)
Witness the alert :)
Here's the GIF we are using!
view-source: on that GIF
Our GIF - does it have dimensions a.k.a. is it valid?

Основные заблуждения относительно XSS

Ожидание:

От XSS спасут фильтры, правила и антивирусы

Реальность:

Далеко не все пользователи умеют устанавливать дополнительные фильтры в браузер.
Фильтрацию почти всегда можно обойти.

Основные заблуждения относительно XSS

Главное заблуждение:

XSS лично меня не коснется, у меня красть нечего.

Основные заблуждения относительно XSS

Реальность:

- Можно использовать XSS, чтобы помещать на сайт рекламу, фальшивые окна с уведомлениями.
- Можно использовать XSS как точку входа в корпоративную сеть, минуя средства защиты.
- Можно использовать сайт с XSS, чтобы похитить данные пользователя, а для усиления атаки используется социальная инженерия.

Виды XSS

Виды XSS

- Reflected XSS (Отраженная XSS).
- Stored XSS (Хранимая XSS).
- Blind XSS (Слепая XSS).
- DOM-based XSS (XSS в параметрах DOM).
- Self XSS.

Reflected XSS

Reflected XSS

Непостоянная (англ. Reflected – отраженный) XSS:

- Образуется, когда данные, предоставленные клиентом, выполняются непосредственно серверными скриптами.
- Данные передаются без надлежащей обработки.
- Данные возвращаются сразу и не проверяются.
- Данные **не сохраняются** на сервере.

Reflected XSS

Выводы:

- Payload злоумышленника не сохраняется на сервере, а сразу отражается пользователю.
- Жертва должна запустить payload, чтобы атака сработала.

Stored XSS

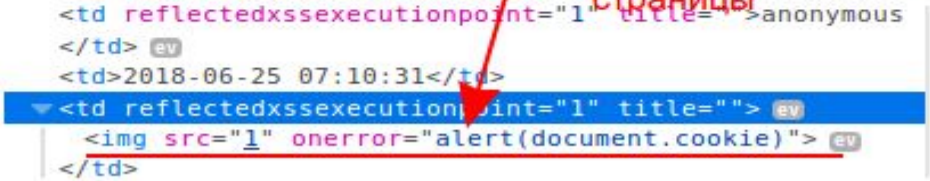
Stored XSS

Хранимая (англ. Stored – хранимый) XSS:

- Образуется, когда данные, предоставленные клиентом, выполняются непосредственно серверными скриптами.
- Данные передаются без надлежащей обработки.
- Данные возвращаются сразу и не проверяются.
- Данные **сохраняются** на сервере.
- Payload будет доступен на сервере, атака будет на каждого пользователя, который зайдет на страницу, содержащую payload.

13 Current Blog Entries			
	Name	**Date**	**Comment**
1	anonymous	2018-06-25 07:10:31	
2	admin	2009-03-01 22:31:13	Fear me, for I am ROOT!
		2009-03-01	Social Engineering is

запись в блоге сохраняется на сервере и вызывается при каждом отображении страницы



Stored XSS

Выводы:

- Payload злоумышленника **сохраняется** на сервере и доступен для всех пользователей, которые перейдут на зараженную страницу.
- Атака не требует участия жертвы, достаточно перейти на страницу, содержащую payload.

Blind XSS

Blind XSS

Слепая (англ. Blind – слепой) XSS:

- Образуется, когда данные, предоставленные клиентом, выполняются непосредственно серверными скриптами.
- Данные передаются без надлежащей обработки.
- Данные возвращаются сразу и не проверяются.
- Данные **сохраняются** на сервере.
- Инъекция сработает в другой части приложения или вообще в другом приложении, отличном от того, в котором была выполнена.

В случае Blind XSS это не обязательно та же страница, где ввели payload

Где встречается Blind XSS

- Contact/Feedback страницы – срабатывание после просмотра страницы модератором.
- Просмотрщики логов – срабатывание после просмотра страницы логов админом.
- Сервис для работы с тикетами – срабатывание после просмотра страницы админом.

Blind XSS

Выводы:

- Blind XSS срабатывает не там, где вводится payload.
- Blind XSS может быть очень опасной, если страницу с инъекцией откроет привилегированный пользователь.
- Сценарий эксплуатации Blind XSS требует выждать, иногда очень долго.

DOM-Based XSS

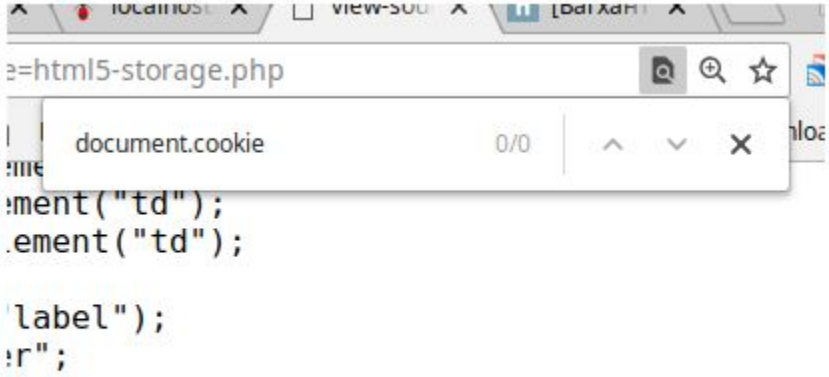
DOM-Based XSS

DOM-based XSS (XSS в параметрах DOM):

- Происходит эксплуатация параметров DOM – нельзя отследить наличие кода инъекции в отображаемой HTML-странице.
- Модифицируется страница, которую сервер отдает пользователю.

<img src=1 onerror=alert(document.cookie)>

Added key canary to Session storage



Инъекция происходит

Но в исходном коде payload не встречается

Где встречается DOM-Based XSS

- document.write(…)
- document.writeln(…)
- document.body.innerHtml=…
- Прочие сущности, которые динамически меняют структуру документа.

Пример кода с DOM-Based XSS

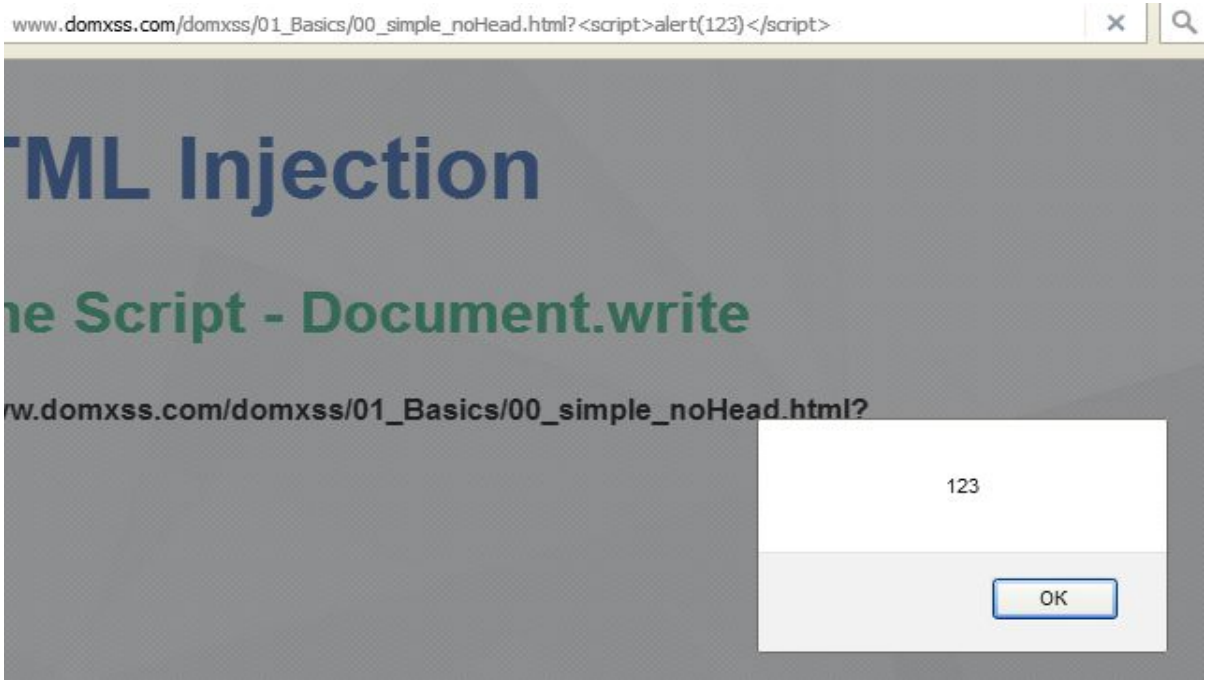
```
<script>
var
pos=document.URL.indexOf(
"name=")+5;
var username =
unescape(document.URL.su
bstring(pos,document.URL.le
ngth));
var r='<b>'+username+'</b>'
document.write(r);
</script>
```

Если отправить параметр:

```
<script>alert(123)</script>
```

получим

```
var
r='<b>'+<script>alert(123)</s
cript>+'</b>'
document.write(r);
```


www.domxss.com/domxss/01_Basics/00_simple_noHead.html?<script>alert(123)</script>
TML Injection
ne Script - Document.write
w.domxss.com/domxss/01_Basics/00_simple_noHead.html?
123
OK

DOM-Based XSS

Выводы:

- Payload должен передаваться «как есть» и попадать на страницу в неизменном виде.
- Контекст, в который попадает payload, должен позволять выполнять переданные данные (например, document.write(…)).
- Без тестирования не обойтись – анализ кода может быть затруднен.

Self XSS

Self XSS

Что будет, если XSS есть, но у злоумышленника не воспроизводится?

Только сам пользователь сайта может выполнить код, который приводит к XSS?

Может ли пользователь атаковать себя сам?

Пример:

Пользователю приходит сообщение, где ему предлагают в консоли разработчика ввести некий код, который якобы приводит к взлому аккаунта другого пользователя.

Особенности Self XSS

- Многие bug-bounty программы не считают Self XSS уязвимостью.
- Злоумышленнику может не хватать какого-то параметра, чтобы реализовать XSS самому, но этот параметр есть у жертвы.
- Задача злоумышленника – заставить жертву, которая вошла в аккаунт, выполнить вредоносный код.

Практическая часть. Демонстрация эксплуатации некоторых видов XSS

Практическое задание

1. Внимательно изучите все рассмотренные примеры и ответьте на вопрос: можно ли использовать поисковые системы для поиска уязвимостей XSS? Насколько будет эффективен данный поиск?

1. * Попробуйте самостоятельно установить OWASP Mutillidae и проделать рассматриваемые задания.

Вопросы участников...