Введение в Web. SQL-Injection

План лекции

CTF

URL

COOKIE

SQL

SQL-Injetion

Проект в лаборатории Parallels

Арыков Никита, nikita.arykov@gmail.com

Соревнования Capture the Flag

SQL-Injection

Blind SQL over JSON

Dom based XSS – Ajax

XSS with Flash

DoS-атака(Denial of Service)

Buffer Overflow

Реверс инженерия

Стеганография/Криптография

Взаимодействие в Web

Клиент(Браузер, Opera, Firefox, telnet, etc.) ↔
Сервер(Веб-сайт, Apache, IIS, Nginx) ↔
СУБД(MySQL, OracleDB, MSSQL)

URL(RFC3986)

- foo://example.com:8042/over/there?name=ferret#nose
- Scheme := foo(http, https, ftp)
- Authority := example.com:8042(домен, порт)
- Path := /over/there
- Query := name=ferret(после ?)
- Fragment := nose(после #)

Cookie

- Используются для авторизации на сайте, хранения персональных данных.
- Авторизация:
  - Вводим user_email, password
  - Браузер отправляет их на сервер
  - Сервер проверяет существует ли такой пользователь
  - Если существует, то в браузере(и в каком-то виде на сервере) сохраняется значения user_email и «секрет».

Cookie

- Параметры
  - Ключ-значения(user_email=vasia@mail.ru)
  - Время жизни(минута, час, год)
  - Путь(http://ya.ru/mail/show/show_mail.php)
  - Домен(http://ya.ru)
- Cookie посылается обратно серверу только в том случае, если имя хоста, с которого запрашиваются страницы, заканчивается строкой с именем указанного до мена.

Клиет/Браузер

- Хотим отрыть URL http://blogsphere.ru/show_mail.php?user_id=5&mail_dir=inbox
- Получаем IP-адресс blogsphere.ru 10.7.22.9
- Протокол HTTP — port=80(обычно)
- Создаем socket
- Делаем connect

Клиент/Браузер

- Хотим отрыть URL http://blogsphere.ru/show_mail.php?user_id=5&mail_dir=inbox
- Браузер пишет в socket(с помощью send, write) следующий текст

GET http://blogsphere.ru/show_mail.php?user_id=5&mail_dir=inbox HTTP/1.1

User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: image/png,image/*;q=0.8,*/*;q=0.5

Cookie: user_email=vasia@mail.ru; signature=0adb5668b52ad56861d43a682f16f1de

Сервер

Сидит в accept(ждет клиента)

- Пришел клиент(socket). Вычитывает запрос клиента(с помощью recv, read)
- Формируются некоторые массивы(php)
- $_GET['user_id'] = 5
- $_GET['mail_dir'] = 'inbox'
- $_COOKIE['user_email']='vasia@mail.ru'
- $_COOKIE['signature']='0adb5668b52ad56861d43a682f16f1de'

GET http://blogsphere.ru/show_mail.php?user_id=5&mail_dir=inbox HTTP/1.1

User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:7.0.1) Gecko/20100101 Firefox/7.0.1

Accept: image/png,image/*;q=0.8,*/*;q=0.5

Cookie: user_email=vasia@mail.ru;
signature=0adb5668b52ad56861d43a682f16f1de

Основы СУБД/SQL

- СУБД — набор программ для работы с БД.
- БД — набор таблиц
- Таблица — поля, набор значений

mail_id	user_id	text_mail	datetime	deliver
1	3	Hello!	2011-08-18 14:46:26	1
2	1	How are You?	2011-08-18 14:46:42	0

СУБД/SQL

- Получение данных(в результате получим первую строчку)
  - SELECT text_mail, datetime FROM tb_name WHERE user_id=3 AND mail_id=1;
- Вставка данных
  - INSERT INTO tb_name VALUES(2, 3, 'Hi!', '2011-02-03 12:12:03', 5)
- Удаление таблицы
  - DROP tb_name;

mail_id	user_id	text_mail	datetime	deliver
1	3	Hello!	2011-08-18 14:46:26	1
2	1	How are You?	2011-08-18 14:46:42	0

SQL-Injection

- Незапланированное внедрение кода
- В коде есть запрос к БД
  - $query = “SELECT * FROM table WHERE password = '$_GET[“password”]’ AND user_id = $_GET['user_id']“
- Пусть от клиента пришли следующие данные
  - $_GET['password'] = 1' OR 1=1; /*!DROP TABLE table*/ --
  - $_GET['user_id'] = 100500
- После подстановки параметров получим
  - $query = “SELECT * FROM table WHERE password = ‘1’ OR 1=1 ; /*!DROP TABLE table*/ -- 'AND user_id=100500“

Защита на уровне приложения

- Проверка ввода(RegExp)
- Параметризованные запросы
- Пример:
  - SELECT * FROM table_name WHERE user_id = ? AND user_password = ?
  - При вызове запроса передаем параметры и СУБД сама выполняет безопасную подстановку

Нормализация/Обфускация

Раскодирование символов(стандарт RFC 3986):

Символы (%41 - 5A% и %61 -%7A), цифры (%30 -%39), дефис (%2D), точка (% 2E), подчеркивания (%5F) или тильды (%7E)

http://example.com/%7Eusername/ --> http://.example.com/~username/

Раскодирование hex последовательностей:

http://example.com/show?param=foo%2Fbar%2Bbaz#.D0.A1.D1.81.D1.8B.D0.BB.D0.BA.D0.B8 ---> http://example.com/show?param=foo/bar+baz#

Пример обфускации:

http://www.modsecurity.org/testphp.vulnweb.com/artists.php?artist=0+div+1+union%23foo*%2F*bar%0D%0Aselect%23foo%0D%0A1%2C2%2Ccurrent_user

Раскодируется:

http://www.modsecurity.org/testphp.vulnweb.com/artists.php?artist=0+div+1+union#foo*/*bar select#foo 1,2,current_user

Защита числовых параметров

Атака

SELECT password FROM tb_users WHERE user_id = 10 OR 1=1

Экранирование не защищает!

Существующие способы(на уровне приложения)

Приведение типа(intval)

Защита строковых параметров

Использовать экранирование(добавляем \ перед ', и т.д.)

Атака

SELECT * FROM table WHERE name = 'Д'Артаньян ‘ DROP TABLE --‘

SELECT * FROM table WHERE name = 'Д\'Артаньян \’ DROP TABLE --‘

Summary

- Чтобы проверить сайт на уязвимости
  - Добавить ' OR 1=1 в URL
  - http://site.ru/index.php?id=1 ' OR 1=1
  - Возможно в ответе будет текст ошибки, что говорит о небезопасном коде на сервере
- Если не понятно происходит ли атака
  - Более интересный вариант добавить к запросу SLEEP(5) — если страница будет грузится на 5 секунд дольше, то можно провести атаку.

Как потренироваться

- Установить Apache+MySQL+PHP
- Для Windows всё это ставится одним пакетом denwer
- http://localhost/tools/phpmyadmin
- Заходим в папку Z:/home/test1.ru/www/index.php
- В нем пишем код
- И тестируем через браузер адрес http://test1.ru
- http://zaic101.ru/daredevil/sql_inj/ - войти не зная пароля, путем ввода в поля SQL-Injection

Литература

Безопасность

- Ховард М., Лебланк Д. Защищенный код
- Syngress - SQL Injection Attacks and Defense
- Скляров И.С. – Головоломки для хакера
- Джек Козиол - Искусство взлома и защиты систем
- Сайт OWASP

PHP

- Гутманс Э. и др. - PHP5. Профессиональное программирование
- Дэвид Скляр – PHP сборник рецептов.

JavaScript + Ajax

- Дэвид Флэнаган – JavaScript Подробное руководство
- Бринзаре, Дари, Черчез - AJAX и PHP. Разработка динамических веб-приложений

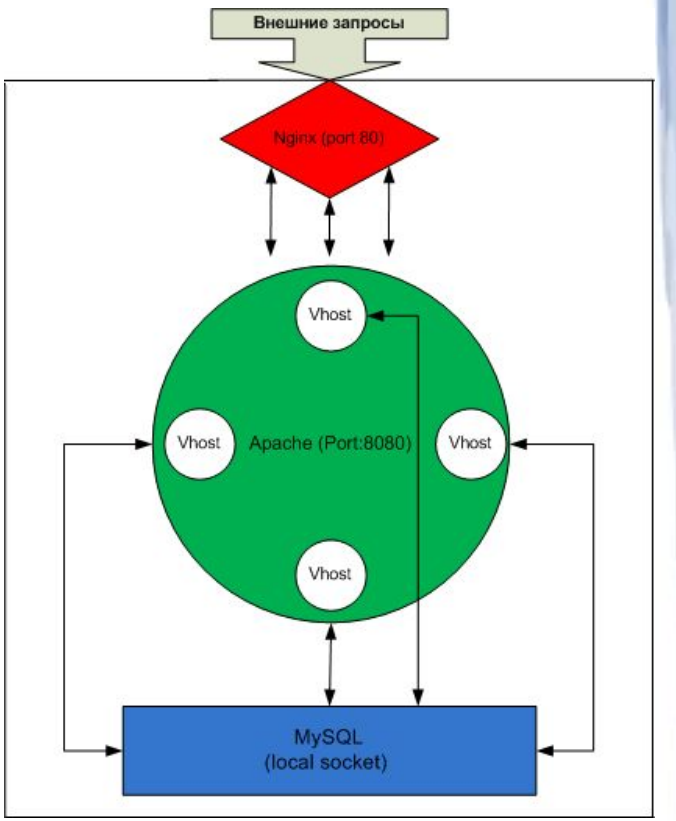
SQL

- Tutorial с сайта MySQL refman-4.0-ru.html-chapter
- Глава 3. Учебное пособие по MySQL

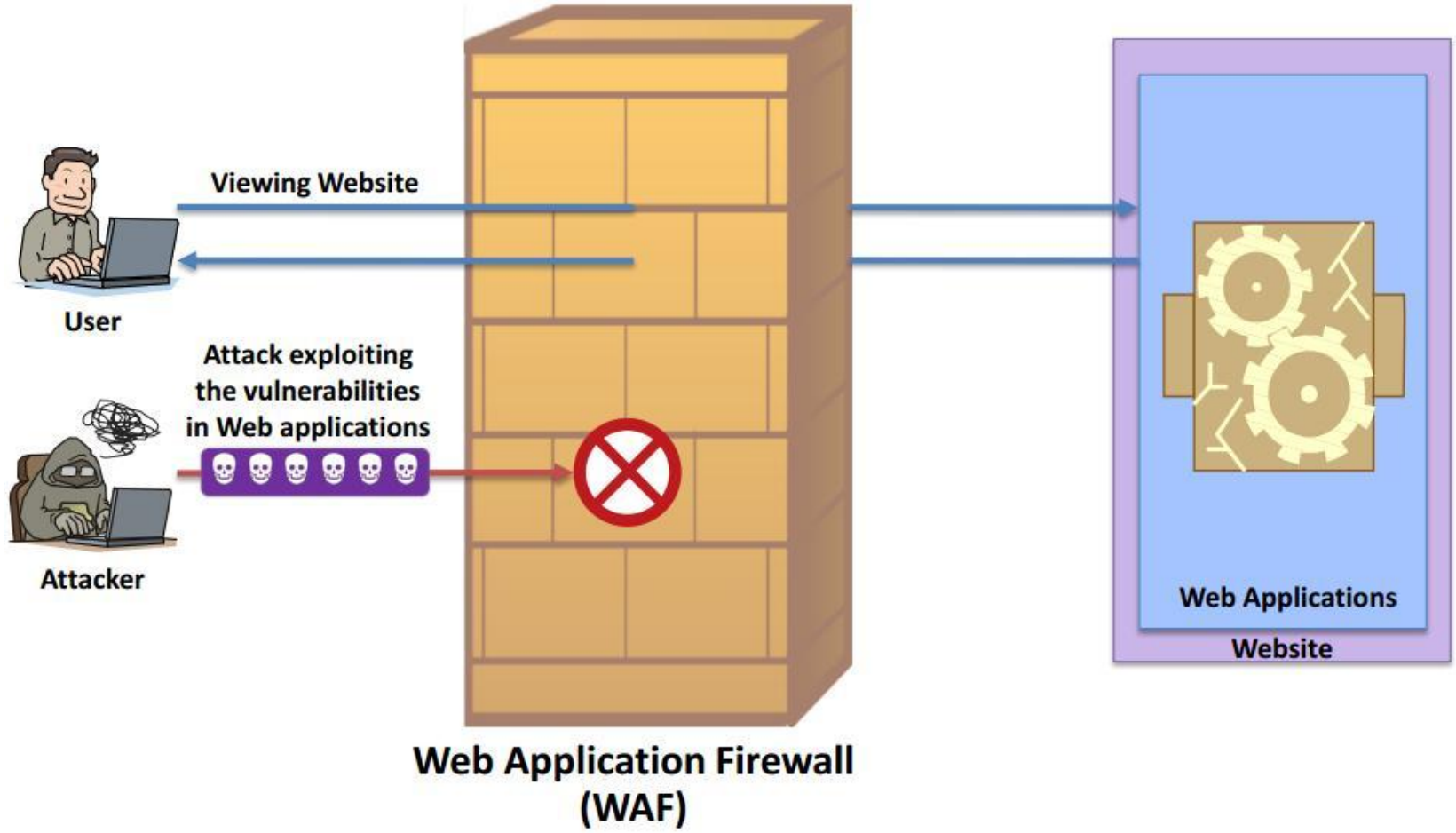
Проект в лаборатории Parallels - «Защита от SQL-Injection»

Постановка задачи

- Имеется архитектура
- Есть сервер на котором работает много сайтов. Клиент посылает запрос к сайту. Сначала запрос приходит на proxy — nginx Последний направляет запрос на нужный сайт.



Проект в лаборатории Parallels - «Защита от SQL-Injection»



Проект в лаборатории Parallels - «Защита от SQL-Injection»

Постановка задачи

Необходимо организовать фильтрацию трафика, на предмет атак. И если пользователь послал запрос с атакой, то блокировать его. Необходимо реализовать аналог WAF(Web Application Firewall)

Предлагаемое решение

Просканировать сайт выявив уязвимые места, на основе данных сканирования реализовать фильтрацию/защиту.

Задачи

Получить все ссылки на сайте

На вход подается адрес сайта(например http://fenster.name)

В результате должен сформироваться фаил

http://fenster.name/group.php?id=8201

http://fenster.name/blog.php?note=10

И т.д.(при этом http://fenster.name/group.php?id=8201 и http://fenster.name/group.php?id=9201 считается идентичными, и необходимо вывести только одну)

Реализация: Небольшой пример есть в книге Гутманс и др.PHP5 Профессиональное программирование, конец главы 11

Задачи

Найти уязвимые параметры

На вход подается список ссылок

http://fenster.name/group.php?id=8201

http://fenster.name/blog.php?note=10

Нужно определить параметры на которые может быть совершена атака(например node_id)

Реализация: Можно использовать существующие сканеры(например sqlmap) запустить его и распарсить его лог. Так же можно попробывать написать собственный сканер основываясь на SLEEP, и других методах. Чем больше будет использовано сканеров тем лучше.

Задачи

Фильтр

В результате будут получены правила, что например note_id уязвим при вводе кавычки или %27 или \' и т. д.

Анализировать запрос на наличие таких символов(последовательностей) и либо блокировать его либо пропускать дальше. Необходимо разработать формат правил.