Программно-аппаратные средства Web

Безопасность веб-приложений

Сергей Геннадьевич Синица

КубГУ, 2020

sin@kubsu.ru

Зачем взламывают сайты?

Слабое администрирование

- Ботнеты
- Спам-сети
- Ad-ware и редиректы
- Ссылки с бирж
- Трояны и вирусы

Слабый код

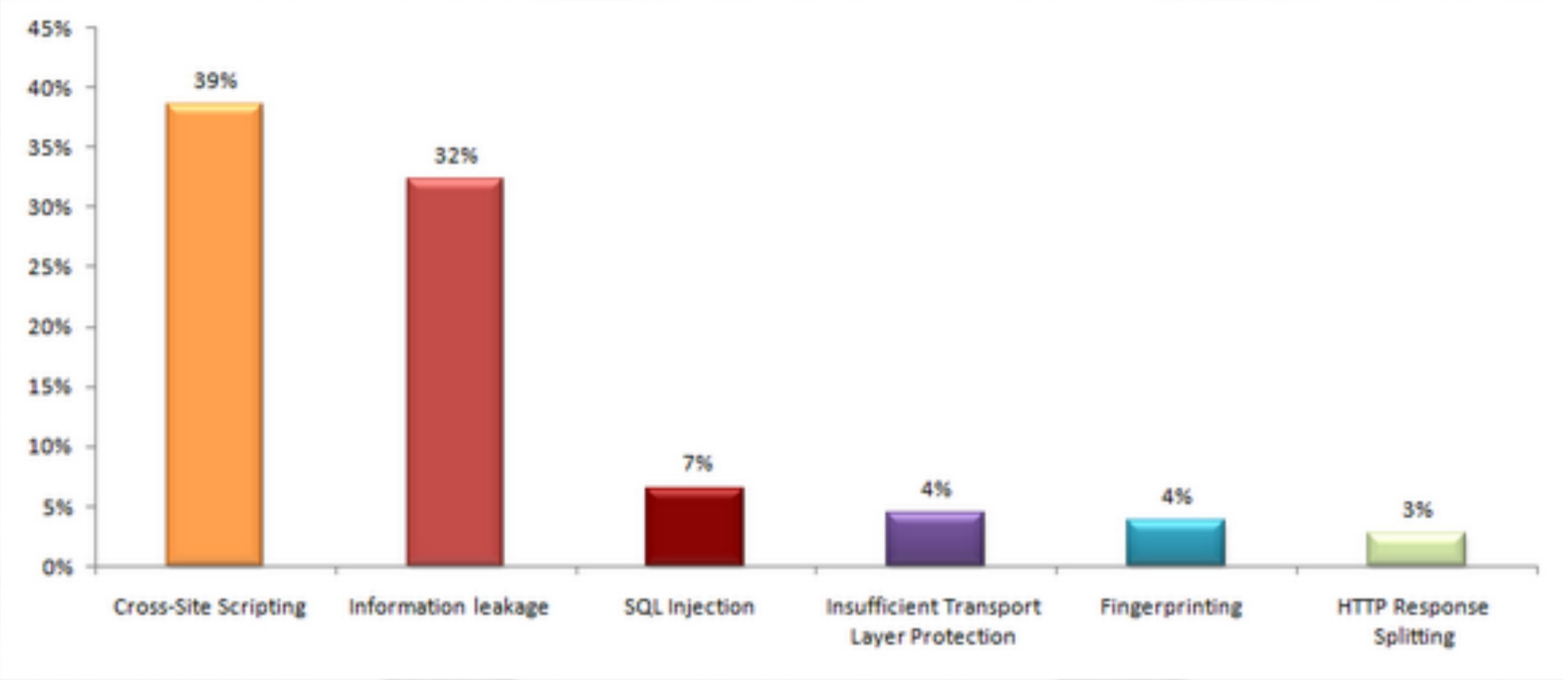
- Кража персональных данных
- Политические акции
- Заказ конкурентов
- Кража денег
- Потому что могу

Примеры? Как обнаружить? Что делать?

Уязвимости веб-приложений

- Cross Site Scripting (XSS)
- SQL Injection
- Cross Site Request Forgery (CSRF)
- Upload
- Include
- Access bypass
- Information disclosure

Уязвимости веб-приложений



http://projects.webappsec.org/w/page/13246989/Web%20Application%20Security%20Statistics

XSS

Произвольный JavaScript выполняется в контексте сессии пользователя.

Чем это грозит?

Бывает активная и пассивная.

Как найти уязвимость?

Пример?

Защита от XSS

Проверять все данные, поступившие извне, на соответствие формату и санитизовать эти данные непосредственно перед выдачей клиентам.

Что такое сантизация?
- фильтрация (preg_match, strip_tags)
- приведение к типу (is_numeric, intval)
- экранирование (htmlspecialchars, библиотеки)

Что надо санитизовать?

SQL Injection

Изменение SQL запроса параметрами, поступившими от пользователя.

К чему приводит?

Примеры?

Защита от SQL Injection

- применять библиотеки абстракций и ORM

- использовать подготовленные запросы (Prepared Statements, PDO)

- проверять данные на соответствие формату и экранировать кавычки и спецсимволы в поступающих от пользователя данных перед использованием их в SQL-запросе (mysql_real_escape_string)

CSRF

Выполнение запроса на атакуемый сайт в контексте сессии пользователя при открытии им страницы с вредоносным кодом, на стороннем сайте.

Чем грозит?

Может использовать метод POST и GET.

Пример?

Защита от CSRF

- Использовать POST и GET по назначению
- Реализовать устаревание сессии
- Использовать подтверждения
- Защищать формы и XHR токенами

Upload-уязвимости

Загрузка на сервер и выполнение произвольного кода.

Чем грозит?

Примеры?

Уязвимости на Shared-хостинге?

Настройка прав доступа в Linux?

Защита от Upload

Настроить права доступа

Проверять расширения загружаемых файлов по белому списку

Запретить выполнение скриптов в каталоге с загрузками

Include-уязвимости

Включение в код приложения сторонних файлов.

Чем грозит?

Бывают локальные и удаленные.

Примеры?

Защита от Include

В коде приложения иметь белый список модулей.

Явно сопоставлять URL модулям (роутинг).

Правильно использовать фреймворки.

Утечки информации

Сообщения при обработке ошибок

Сообщения об ошибках на сервере

Заголовки ответов

Комментарии в коде и верстке

Золотое правило веб-разработчика

– не доверять любым внешним данным, включая параметры GET и POST, Cookies, заголовки запросов;

– проверять формат перед использованием и сохранением на сервере;

– на сервере сохранять в исходном виде без изменений;

– санитизовать перед отображением пользователю.

Защита клиента

1) использование лицензионного программного обеспечения либо популярных программ с открытым исходным кодом;

2) установка обновлений безопасности для операционной системы;

3) установка обновлений безопасности для браузера;

4) установка обновлений безопасности для плагинов браузера,таких как Flash-плеер;

5) регулярное обновление антивирусной программы;

6) использование политики безопасности с ограниченными правами (не работать под администратором);

7) мониторинг сетевой активности компьютера специалистами.

Защита сервера

1) проверка всех поступающих от клиента данных на соответствие формату перед любым использованием на сервере;
2) экранирование данных, поступивших от клиента, при выдаче их другим клиентам;
3) использование политики безопасности с ограниченными правами (не запускать программы под root, разграничить права доступа, корректно сконфигурировать все программы на сервере);
4) использование фреймворков создания веб-приложений;
5) установка обновлений безопасности для операционной системы и сторонних веб-приложений;
6) резервное копирование;
7) мониторинг запросов к веб-приложению, логов, ошибок;
8) использование стойких к перебору паролей, периодическая смена паролей;
9) хранение паролей в зашифрованном виде или хранение хэшей паролей;
10) доступ к защищённым паролем ресурсам с гарантированно чистых от вредоносных программ клиентов.

Литература

Веб-программирование и веб-сервисы / С.Г. Синица. Краснодар: КубГУ, 2013.