

АТАКИ

на Веб
и
WordPress

Библиотека журнала



www.xakep.ru

АТАКИ

на Веб

и

WordPress

Санкт-Петербург

«БХВ-Петербург»

2021

УДК 004
ББК 32.973
В40

В40 Атаки на веб и WordPress. — СПб.: БХВ-Петербург, 2021. — 256 с.: ил. —
(Библиотека журнала «Хакер»)

ISBN 978-5-9775-6745-9

В сборнике избранных статей из журнала «Хакер» описаны методики поиска и эксплуатации уязвимостей в популярных CMS WordPress и Drupal, а также в веб-приложениях, таких как панель управления сервером Webmin, веб-сервер Ngnix, почтовый агент Exim, файрвол ModSecurity и утилита GitLab. Рассматривается инструментарий пентестеров, приводится подробный анализ кода и даны ссылки на видеоролики с наглядной демонстрацией работы уязвимостей. Приведен обзор методик пентеста веб-приложений OWASP Testing Guide v4 образца 2020 года.

Для читателей, интересующихся информационной безопасностью

УДК 004
ББК 32.973

Группа подготовки издания:

Руководитель проекта	<i>Павел Шалин</i>
Зав. редакцией	<i>Екатерина Сависте</i>
Компьютерная верстка	<i>Натали Смирновой</i>
Дизайн обложки	<i>Карины Соловьевой</i>

"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

ISBN 978-5-9775-6745-9

© ИП Югай А.О., 2021
© Оформление. ООО "БХВ-Петербург", ООО "БХВ", 2021

Содержание

Вместо предисловия.....	7
Об этой книге.....	8
Об авторах.....	9
 1. Хак в один клик.	
Сравниваем возможности автоматических сканеров уязвимостей	
<i>(Марк Бруцкий-Стемповский).....</i>	10
Категории и методы	10
Sn1per.....	11
Wapiti3	12
Nikto.....	13
OWASP ZAP	14
Sqlmap.....	15
Acunetix WVS	15
Vega	16
Nessus.....	18
Kube-hunter.....	19
Trivy	20
PVS-Studio.....	20
GitLeaks.....	21
QARK.....	22
Burp Suite.....	24
MobSF.....	24
Выводы.....	27
 2. Удаленное удаление.	
Как захватить контроль над WordPress, заставив его стереть файл	
<i>(Иван aLLy Комиссаров).....</i>	28
Стенд.....	28
Анализ уязвимости.....	30
Удаляем файлы. Версия 2.0.....	37
Выводы.....	40

3. Открытка для WordPress.

Захватываем контроль над сайтом, спрятав код в картинке

(Иван aLLy Комиссаров)	42
Стенд.....	42
Манипулируем метаданными, или CVE-2019-8942	44
Причины возникновения ошибки path traversal (CVE-2019-8943)	48
Решаем проблемы и эксплуатируем уязвимость path traversal	54
Эксплуатируем выполнение произвольного кода	59
Выводы	64

4. От XSS до RCE одним движением мыши.

Эксплуатируем уязвимость в WordPress

(Иван aLLy Комиссаров)	65
Стенд.....	65
Анализ уязвимости	67
Выводы	85

5. Опасный PHAR.

Эксплуатируем проблемы десериализации в PHP

на примере уязвимости в WordPress

(Иван aLLy Комиссаров)	86
Предыстория атаки	86
Стенд.....	87
Немного о PHAR	88
RCE в WordPress через Woocommerce	95
Выводы	106

6. Прессуем WordPress

(Иван aLLy Комиссаров)	107
Стенд.....	107
Детали уязвимости	108
Автоматизируй это	112
Выводы	113

7. Приготовься к инъекции.

Раскручиваем уязвимости в WordPress, препарировав метод prepare

(Иван aLLy Комиссаров)	114
Стенд.....	114
Работа с БД в WordPress	115
Первые проблемы	119
Проблемы продолжают	121
Другой вариант эксплуатации	125
Proof of concept	129
Патч.....	132
Выводы	133

8. Malware vs WordPress.

Проверяем защитные плагины в боевых условиях

<i>(Ex.Mi)</i>	134
Мифы и реальность	136
Подготовка CMS.....	137
Выбор плагинов.....	137
Внедрение вредоносного кода	139
Сканирование.....	142
Wordfence Security – Firewall & Malware Scan	142
Sucuri Security — Auditing, Malware Scanner and Security Hardening.....	145
Anti-Malware Security and Brute-Force Firewall.....	145
Cerber Security, Antispam & Malware Scan.....	146
Бонус: «Ай-Болит».....	147
Выводы.....	148
Взгляд со стороны атакующего	149
Взгляд со стороны защищающегося	149

9. Читай и выполняй.

Как работает эксплоит уязвимости в GitLab

<i>(Иван aLLy Комиссаров)</i>	150
Стенд.....	150
Чтение локальных файлов	151
От читалки к выполнению кода	163
Заключение	167

10. Досим ModSecurity.

Как работает критический баг в популярном WAF

<i>(Иван aLLy Комиссаров)</i>	168
Стенд.....	168
Детали.....	172
Заключение	178

11. Лазейка в Webmin.

Как работает бэкдор в панели управления сервером

<i>(Иван aLLy Комиссаров)</i>	179
Стенд.....	179
Детали.....	181
Заключение	191

12. Отправляем команды.

Как заставить популярный серверный почтовик выполнять произвольный

<i>(Иван aLLy Комиссаров)</i>	192
Стенд.....	192
Детали уязвимости и локальная эксплуатация	195
Удаленное выполнение команд и ограничения.....	202
RCE при использовании конфига по умолчанию	204
Заключение	209

13. Комбо для Drupal.

Как захватить сайт с Drupal, используя уязвимости

(Иван aLLy Комиссаров).....	210
Стенд.....	210
Путь к XSS	212
Как же это можно использовать?.....	217
Выполняем произвольный код.....	218
Объединяем атаки в цепочку	221
Выводы	222

14. Как подчинить конфиг.

Учимся эксплуатировать уязвимость в PHP-FPM и Nginx

(Иван aLLy Комиссаров).....	223
Стенд.....	224
Детали уязвимости	227
Заключение	241

15. Покоряем веб.

Как применять OWASP Testing Guide v4 в 2020 году

(v3l_v37).....	243
Не все то золото, что блестит	244
Testing Guide Introduction.....	244
Testing for Information Gathering	244
Conduct search engine discovery/reconnaissance for information leakage	244
Enumerate applications on webserver.....	245
Map execution paths through application	246
Configuration and Deployment Management Testing	246
Authentication testing / Authorization testing	246
Input validation testing	246
Business logic testing	247
Client side testing	247
Заключение	247

«Хакер»: безопасность, разработка, DevOps.....	249
--	-----

Предметный указатель	252
----------------------------	-----

Вместо предисловия

Тридцать восемь целых и четыре десятых процента — именно столько сайтов в Интернете работает в 2020 году под управлением WordPress согласно статистике **w3techs.com**. Сколько это в абсолютных цифрах, сложно даже представить. Более семисот миллионов. В силу своей высокой популярности, а также потому, что WordPress — это CMS с открытым исходным кодом, в этом движке регулярно находят уязвимости, которые не гнушаются использовать злоумышленники. Вот почему безопасность WordPress и веба в целом крайне важна для коммерческих предприятий и частных лиц — владельцев различных интернет-ресурсов.

Атаки на веб способны не только вывести из строя сайт на радость конкурентам. Они грозят нарушением бизнес-процессов, а также утечкой конфиденциальных данных, которые могут попасть в руки недоброжелателей и оказаться в публичном доступе. Сообщения о подобных случаях регулярно появляются на информационных порталах и в новостной ленте сайта **xakep.ru**. С учетом того, что современный бизнес все активнее уходит в онлайн (а в период коронавирусной пандемии эта тенденция лишь усилилась), безопасность веба становится критичной проблемой.

У любой проблемы, как известно, есть решение. Вопросы безопасности с успехом помогают решать пентестеры: специалисты по тестированию на проникновение. В их компетенцию входит в том числе исследование сетевого периметра и работающих на сайте веб-приложений, поиск уязвимостей и выявление слабых мест в защите IT-инфраструктуры, которые могут использоваться злоумышленниками для выполнения атак. Подобные услуги неплохо оплачиваются, к тому же это, пожалуй, единственный законный способ заниматься взломом, не опасаясь негативных последствий, связанных с нарушениями закона.

В этой книге собраны практические приемы поиска и анализа уязвимостей в CMS WordPress и других популярных веб-приложениях, они подробно описаны опытными специалистами по информационной безопасности — постоянными авторами журнала «Хакер». Книга представляет собой сборник лучших статей, опубликованных ранее на портале **xakep.ru** и посвященных эксплуатации уязвимостей в CMS WordPress, Drupal, в популярной панели управления веб-сервером Webmin, а также других сетевых приложениях, включая Ngnix. Прочитав эти материалы, читатель освоит методику исследования исходного кода веб-приложений, поиска уязвимостей и их эксплуатации на конкретных примерах. Он изучит практические приемы, используемые пентестерами в своей работе. В конце большинства разделов приве-

дены ссылки на видеоролики, наглядно демонстрирующие действие описанной в материале уязвимости.

Безусловно, в Интернете все меняется очень быстро. Разработчики ПО оперативно закрывают обнаруженные уязвимости патчами, после чего они утрачивают актуальность. Поэтому многие из описанных на страницах этой книги приемов не работают на современных версиях CMS и приложений. Но тем не менее среди сотен миллионов опубликованных в Сети сайтов насчитывается огромное множество ресурсов, администраторы которых не спешат вовремя устанавливать обновления, а значит, такие ресурсы по-прежнему остаются уязвимыми. Кроме того, в задачу этой книги входит показать общий принцип поиска брешей в защите, который используют специалисты по информационной безопасности и пентестеры. А для этого как нельзя лучше подходят устаревшие версии веб-приложений с известными, тщательно проанализированными и документированными уязвимостями. Освоив навыки анализа кода и эксплуатации известных недочетов в ПО, можно двигаться дальше и приступать к тестированию актуальных и современных версий программ, что повысит их безопасность и защищенность.

Об этой книге

Под обложкой этой книги собраны лучшие статьи из журнала «Хакер», объединенные общей темой — поиск уязвимостей, атаки на WordPress и веб-приложения. Авторы этих статей — опытные эксперты в сфере информационной безопасности, истинные профессионалы в своей области и неутомимые исследователи.

Помимо текста, фрагментов кода и иллюстраций некоторые разделы книги содержат специальные врезки — в точности те же, что используются на портале **hacker.ru**. Так, во врезке **INFO** содержится дополнительная информация, переключаясь с темой текущего раздела. Врезка **WWW** содержит ссылки на интернет-ресурсы, где читатель сможет отыскать полезные информационные материалы.

В журнале «Хакер» царит своя неповторимая атмосфера. Во-первых, здесь принято общаться с читателем на «ты». Во-вторых, на страницах журнала допустим компьютерный сленг, а манеру изложения в статьях можно назвать ироничной и шутиливой. Все эти добрые традиции в полной мере распространились и на книгу. Потому не удивляйся, встретив на страницах издания слово «фича» вместо «функциональная особенность приложения» или «пофиксить» вместо «исправить выявленную ошибку в коде». У нас так заведено.

Поскольку тема этой книги весьма специфичная, мы не можем не опубликовать в предисловии несколько важных предупреждений. Вот они:

ВНИМАНИЕ!

Вся приведенная на страницах этой книги информация, код и примеры публикуются исключительно в ознакомительных целях. Ни издательство «БХВ», ни редакция журнала «Хакер», ни авторы не несут никакой ответственности за любые последствия использования информации, полученной в результате прочтения книги, а также за любой возможный вред, причиненный информацией из этого издания.

Помните, что несанкционированный доступ к компьютерным системам и распространение вредоносного ПО преследуются по закону. Все рассмотренные в книге методы представлены в ознакомительных целях. Каким-либо образом используя представленную в книге информацию, вы действуете исключительно на собственный страх и риск.

Об авторах

Авторы «Хакера» — опытные эксперты в области ИБ, профессиональные исследователи, пенестеры, аналитики и настоящие «этичные хакеры». Их объединяет любовь к компьютерным технологиям и новым знаниям, а также стремление поделиться своими знаниями с нашими читателями.

Иван aLLy Комиссаров — постоянный резидент «Хакера», автор большинства увлекательных статей, собранных в этой книге. Иван трудится специалистом по информационной безопасности в компании ONsec. В сферу его интересов входит безопасность веб-приложений, мобильных приложений и сетевой инфраструктуры предприятий. Публикации aLLy отличает грамотный систематизированный подход, внимание к деталям и высочайший уровень экспертности. Его статьи неизменно получают самую высокую оценку со стороны постоянных читателей «Хакера», прежде всего благодаря тому, что автор является высококвалифицированным экспертом в сфере анализа защищенности. В этом ты можешь самостоятельно убедиться, ознакомившись с собранными в этой книге статьями.

v31_v37 — постоянный автор портала **xakep.ru**, известный нашим читателям публикациями на тему сетевой безопасности и безопасности Microsoft Windows. Занимается тестированием на проникновение и интересуется различными областями в сфере ИБ.

Ех.Mi — читатель «Хакера» с выпуска № 3, а также автор **xakep.ru**, специализирующийся в области разработки ПО, информационной безопасности, защиты веб-приложений и сайтов, работающих под управлением CMS WordPress.

Марк Бруцкий-Стемпковский — родился в 2003 году в Минске, в настоящее время учится в Белорусском государственном университете информатики и радиоэлектроники. Пишет в «Хакер» с 10-го класса, увлекается программированием и информационной безопасностью. Регулярно участвует в разных конкурсах и хакатонах. Открыт к новым знакомствам: автору можно смело писать в «телегу»: **@HackcatDev**.

Кстати, и ты, дорогой читатель, тоже можешь пополнить ряды авторов «Хакера» и стать частью нашей дружной команды. Или даже написать собственную книгу об интересующей тебя области IT-технологий. Если у тебя есть желание творить и поделиться своим опытом с читателями **xakep.ru** и издательства «БХВ» — отправь мне письмо на адрес **valentin@holmogorov.ru** или сообщение в Телеграм **@holmogorov**. Присоединяйся!

Валентин Холмогоров, редактор рубрики «Взлом» журнала «Хакер»

<http://xakep.ru>
<http://holmogorov.ru>

1. Хак в один клик.

Сравниваем возможности автоматических сканеров уязвимостей

Марк Бруцкий-Стемпковский

Для поиска уязвимостей вручную требуются особые знания, богатый опыт и редкое чутье. Но как быть новичкам? Как набраться опыта, если не знаешь, с чего начинать? На помощь приходят автоматические сканеры уязвимостей. В этом разделе мы посмотрим, какие они бывают и как ими пользоваться.

Айтишники, как известно, стремятся все автоматизировать, и хакеры в этом не отстают. Существуют автоматические сканеры уязвимостей — чтобы можно было запустить, откинуться на спинку кресла и потягивать кофе (или пиво), пока они сделают целую гору работы. Поиск уязвимостей с их использованием сводится к тому, чтобы отдать сканеру адрес цели и нажать большую кнопку **Start**, ну или **Enter**, если ты любитель терминала.

При этом понятно, что сканер найдет только типовые уязвимости и, чтобы пойти дальше, нужно уметь не только нажимать на кнопку. Но почему бы не сэкономить немного сил? Во многих случаях это вполне оправданно.

Категории и методы

Универсальных инструментов не существует, и сканеры уязвимостей не стали исключением из этого правила. Они обычно нацелены на уязвимости какого-то определенного рода. Мы рассмотрим следующие виды сканеров.

- ❑ WVS (Web Vulnerability Scanner) — сканеры веб-уязвимостей. У меня это самая многочисленная категория. Сюда входят как общеизвестные OWASP ZAP и sqlmap, так и менее известные, но не менее полезные, вроде Vega.
- ❑ Анализаторы мобильных приложений. Тут очень мало достойных продуктов, и мы остановимся на самых ярких из них.

- Полууниверсальные сканеры для локальной сети предприятия или дома. Это уже не просто сканеры, а целые комбайны для анализа и учета оборудования в сети. Многие из них заодно ищут уязвимости.
- Всекие узкоспециализированные сканеры типа анализа исходного кода, Git/SVN-репозиторий и других сложных для ручной обработки массивов данных.

Сканеры бывают со свободной лицензией и коммерческие. Если с опенсорсом все понятно, то для использования коммерческих придется выложить весьма приличную сумму. К сожалению, ни редакция «Хакера», ни автор не настолько богаты, чтобы покупать их для обзора. Поэтому для всех коммерческих сканеров была использована официальная пробная версия, если не оговорено иное.

ВНИМАНИЕ!

Вся информация в этой главе предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами этой главы. Помни, что доступ к данным без предварительного письменного соглашения с их владельцем преследуется по закону.

Само тестирование тоже бывает разным: Black Box либо White Box. При первом типе пентестер или его инструмент должны работать с сервисом через те же интерфейсы, через которые с ним взаимодействуют пользователи. Например, если для тестирования методом Black Box тебе дан сайт, то ты можешь проверять его только как посетитель, без какого-либо специального доступа к исходному коду или привилегированным аккаунтам. Если это приложение, то подразумевается, что у тебя нет доступа к исходникам: ковыряй сам, если сможешь. В общем, Black Box значит, что у тебя нет ничего, чего бы не было у всех.

При тестировании методом White Box пентестер (или хакер) имеет доступ ко всем потрохам целевого объекта. Если это сайт — у тебя есть его код. Если это сервер — у тебя есть доступ к его внутренностям вроде версии ОС и установленного софта или к некоторым файлам. В этом случае возможности куда шире и ты можешь найти проблему, которую способен эксплуатировать только продвинутый злоумышленник.

В сегодняшней подборке представлены сканеры обеих категорий, так что этот раздел можно считать универсальной отмычкой почти к любому замку.

Sn1per

Цена: Community edition — бесплатно, Professional edition — от 150 долларов

Sn1per — мощный фреймворк для автоматического анализа безопасности цели. Разработан небезызвестным 1N3, основателем компании XeroSecurity. Из других его известных инструментов — Findsploit (<https://github.com/1N3/Findsploit>, средство для быстрого поиска эксплоитов к уязвимости) и PrivEsc (<https://github.com/1N3/PrivEsc>) — для поиска локальных багов EoP.

Sn1per поставляется в двух вариантах. Есть версия Community «для всех и даром» и Sn1per Professional, лицензия на который стоит от 150 зеленых американских рублей.

В бесплатном варианте сканер умеет собирать базовую информацию (IP цели, ping, whois, DNS); запускает Nmap для поиска открытых портов и определения сервисов, в том числе и с помощью NSE; ищет часто встречающиеся уязвимости и автоматически эксплуатирует их; пробует получить доступ ко всем файловым шарам (FTP, NFS, Samba); запускает Nikto, WPScan и Arachni для всех найденных веб-приложений и многое другое. Поддерживает интеграцию с Hunter.io, OpenVAS, Burp Suite, Shodan, Censys и Metasploit.

Установка довольно проста и поддерживает Docker, что сводит ее к двум командам:

```
docker pull xerosecurity/sn1per
docker run -it xerosecurity/sn1per /bin/bash
```

Для сканирования выполни

```
sniper -t [TARGET]
```

Чтобы задействовать все возможности Sn1per, понадобятся дополнительные ключи:

- `-o` — использует движок OSINT;
- `-re` — разведка;
- `-fp` — полностью проработать все порты;
- `-m stealth` — старательно скрывает сканер, чтобы цель не поняла, что ее сканируют;
- `-m webscan` — Sn1per будет работать как обычный сканер WVS;
- `-b` — использовать брутфорс при необходимости;
- `-f [FILE]` — сканировать сразу несколько целей, которые перечислены в файле [FILE];
- `-m nuke` — «ядерный» режим сканирования. Включает в себя брутфорс, обработку всех портов, OSINT, разведку и сохранение всех находок (loot);
- `-m massvulnscan` — очень мощная функция в сочетании с `-f`. Массово сканирует на многие известные уязвимости все заданные цели. Если в тестируемой компании много хостов, эта опция будет весьма полезна;
- `-m discover` — опция поиска всех хостов в заданной подсети и запуск сканирования на каждый из найденных. Если ты даже не знаешь всех возможных целей, это будет очень полезно.

Пример репорта после сканирования очень большой, но он есть в репозитории автора: <https://gist.github.com/1N3/8214ec2da2c91691bcbc>.

Wapiti3

Цена: бесплатно

Wapiti — полностью бесплатный сканер веб-уязвимостей. На момент написания этого материала последняя версия была 3.0.3, выпущенная 20 февраля 2020 года, то есть проект живой. Несмотря на скромные размеры сканера (всего 2,3 Мбайт в рас-

пакованном виде), набор функций у него довольно обширный. По официальному заявлению, сканер умеет обнаруживать следующие баги:

- раскрытие содержимого файла (local file inclusion), в том числе бэкапов и исходного кода сайта;
- SQL-инъекции и внедрение кода PHP/ASP/JSP;
- отраженные и хранимые XSS;
- инъекции команд ОС;
- XXE Injection;
- неудачные конфигурации .htaccess;
- Open Redirect.

Wapiti3 поддерживает прокси, аутентификацию на целевом сайте, умеет не кричать на самопальные сертификаты SSL и может вставлять в запросы любые заголовки (в том числе кастомный User-Agent).

Использование инструмента весьма тривиально. После установки выполни в терминале (да, это консольное приложение) такую команду:

```
wapiti -u [URL]
```

Wapiti просканирует весь сайт и выдаст соответствующий отчет. Чтобы исключить ненужные адреса (например, logout), добавь параметр `-x [URL]`, а для авторизованного сканирования требуются куки. Для их использования сначала сгенерируй JSON-файл с помощью специального скрипта. Он лежит в `bin/wapiti-getcookie` и запускается следующим образом:

```
wapiti-getcookie -u [LOGIN_URL] -c cookies.json -d  
"username=[USER] &password=[PASS]"
```

`[LOGIN_URL]` — это адрес страницы логина, а `[USER]` и `[PASS]` — логин и пароль соответственно. Затем подключаем готовый файл к сканеру:

```
wapiti -u [URL] -x [EXCLUDE] -c cookies.json
```

Вот и все. Отчет генерируется в HTML и сохраняется в `/home/[USER]/.wapiti/generated_report/[TARGET_HOST]_[DATE]_[ID].html`, где `[USER]` — твой логин, `[TARGET_HOST]` — целевой сайт, `[DATE]` — дата сканирования и `[ID]` — четыре цифры. Можно открыть в браузере и посмотреть.

Nikto

Цена: бесплатно

Nikto — весьма популярный сканер веб-приложений, изначально встроенный в Kali Linux. Он крайне простой, даже не прячется от WAF прочего зловредного ПО на сайте, но довольно точен. Умеет находить:

- странные и необычные заголовки;
- утечки inode через заголовок ETag;

- использование WAF;
- множество интересных файлов, к которым не стоило бы открывать доступ.

Как видишь, не очень много. Но зато он быстро работает и не требует настольного справочника для запуска сканирования.

Имеет кучу параметров. Самый главный из них — `-h [HOST]`, задающий цель. Если цель умеет в SSL, стоит указать параметр `-ssl`. Также есть формат вывода (`-Format`) и возможность работать с Metasploit. Инструмент немного устарел, но по-прежнему годится для разведки и взлома совсем уж безнадёжных целей.

OWASP ZAP

Цена: бесплатно

Сканер той самой организации OWASP, которая призвана сделать наш с тобой Интернет безопаснее. Впрочем, пока не сильно успешно. А еще, кстати, есть список OWASP Top 10, где собраны десять самых распространенных багов в веб-приложениях. ZAP (Zed Attack Proxy) — бесплатный инструмент для тестирования на проникновение и поиска уязвимостей в веб-приложениях. Его главные фишки:

- MITM-прокси для захвата трафика браузера;
- пассивный и активный сканеры уязвимостей;
- паук-краулер, который может работать даже с AJAX;
- фаззер параметров;
- поддержка плагинов;
- поддержка WebSocket.

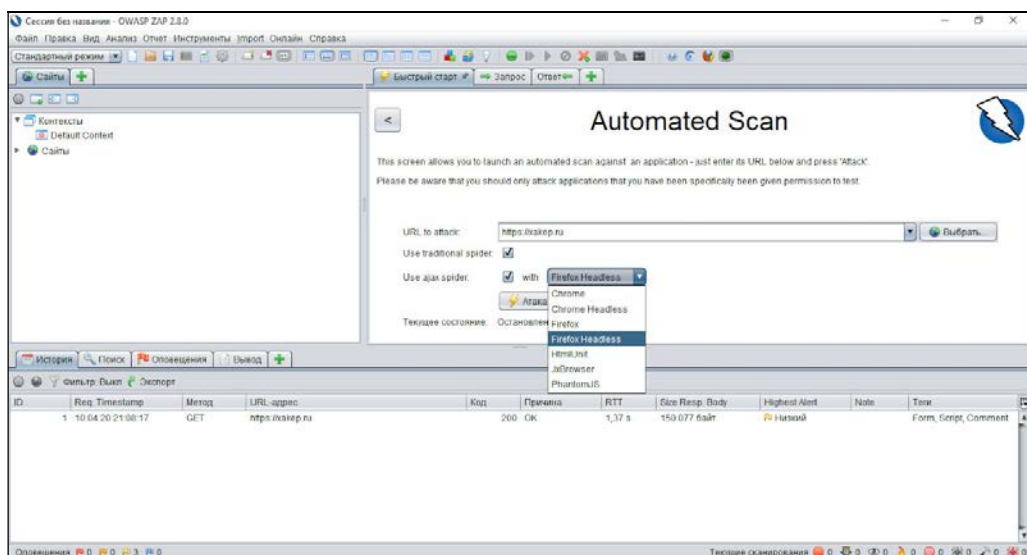


Рис. 1.1. Press to Hack

У программы есть русский интерфейс (частично), неплохой GUI и инструкция по пользованию для новичков. При запуске показывает советы.

Сканирование требует только указать адрес сайта. Воистину «нажал и взломал» (рис. 1.1)!

По эффективности обнаружения багов ZAP очень хорош, использую его параллельно с Vega и Acunetix. Однозначно рекомендую.

Sqlmap

Цена: бесплатно

Sqlmap (<http://sqlmapgui.blogspot.com/>) — это, наверное, самый известный сканер для поиска SQL-инъекций. Его разработкой занимаются Мирослав Штампар (Хорватия) и Бернардо Дамеле (Италия). Особенность этого сканера в том, что он может не только найти ошибку, но и сразу эксплуатировать ее, причем в полностью автоматическом режиме. Умеет работать с БД MySQL, MS SQL, PostgreSQL и Oracle.

Недавно у sqlmap наконец появился GUI, что еще сильнее понизило порог вхождения (рис. 1.2).

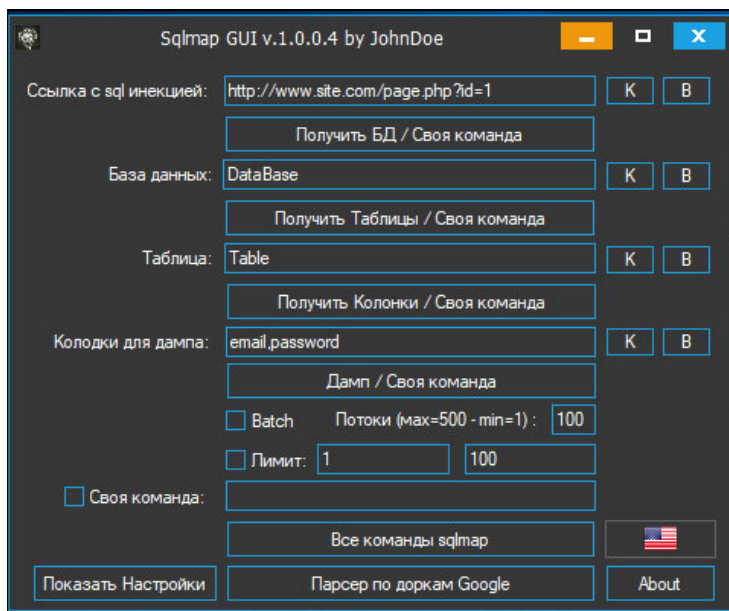


Рис. 1.2. Тот самый GUI

Acunetix WVS

Цена: базовая версия — от 4495 долларов за один сайт (и растет по мере количества сайтов, которые ты планируешь сканировать). Расширенная версия — от 6995 долларов за один сайт (но умеет намного больше).

Было бы странно, если бы эта подборка обошлась без коммерческих сканеров. Именно Acunetix WVS недавно нашел баг у Google (<https://www.acunetix.com/blog/web-security-zone/xss-google-acunetix/>), а одноименная компания-разработчик — один из лидеров рынка.

Сам сканер — это веб-приложение, и его можно ставить на «безголовый» сервер (то есть вообще без графической оболочки). Есть поддержка и Windows, и Linux. К сожалению, сам сканер кому попало не продается, так что «кто попало» выкручивается с помощью тематических сайтов. К последней, 13-й версии крика нет, поэтому сейчас у хакеров в ходу 12-я версия.

Установка — классическая для Windows-приложений. Там задается логин и пароль к веб-интерфейсу, также есть возможность открыть удаленный доступ к сканеру (удобно поставить его на VPS).

После установки видим главную страницу интерфейса. Интерфейс простой, разберется любой школьник. Есть цветовое определение тяжести найденного бага и готовый рейтинг CVE. Пользоваться сканером действительно удобно и приятно.

Сканирование требует только указать адрес цели (на вкладке **Targets**) и нажать кнопку **Scan**, опционально задав время начала. Сканер имеет несколько профилей сканирования, может сканировать только в рабочее или нерабочее время и, по словам производителя, умеет находить почти все виды багов. В этот список входят:

- XSS, в том числе DOM;
- SQL-инъекции, кроме слепых (blind);
- CSRF;
- обход директории;
- XXE Injection;
- небезопасная сериализация;
- проблемы с SSL-сертификатами (скорое истечение срока годности, слабые шифры);
- проблемы с CORS.

Сканер действительно быстрый и качественный, для участия в bug bounty подходит идеально. Жаль, цена кусается. Но я просто обязан его порекомендовать, это один из лучших инструментов.

Vega

Цена: бесплатно

Еще один сканер с открытым исходным кодом, разработан в компании Subgraph. Да, той самой Subgraph, которая сделала клиент Tog на чистой Java. Удивительно, но Vega бесплатный, а по возможностям ничуть не хуже Acunetix.

По заверениям производителя и собственным наблюдениям, сканер хорошо ищет следующие баги:

- SQL-инъекции;
- XSS;

- XXE Injection;
- Integer Overflow/Underflow (кстати, единственный сканер, который их нормально ищет);
- раскрытие содержимого файла (local file inclusion);
- внедрение кода;
- path traversal;
- внедрение HTTP-заголовков;
- плохие настройки CORS.

Сканер написан на Java, а значит, работает везде, где есть Java VM, включая, конечно, Windows и Linux. Недостатки: нужна та самая JVM, к тому же здесь нет веб-интерфейса.

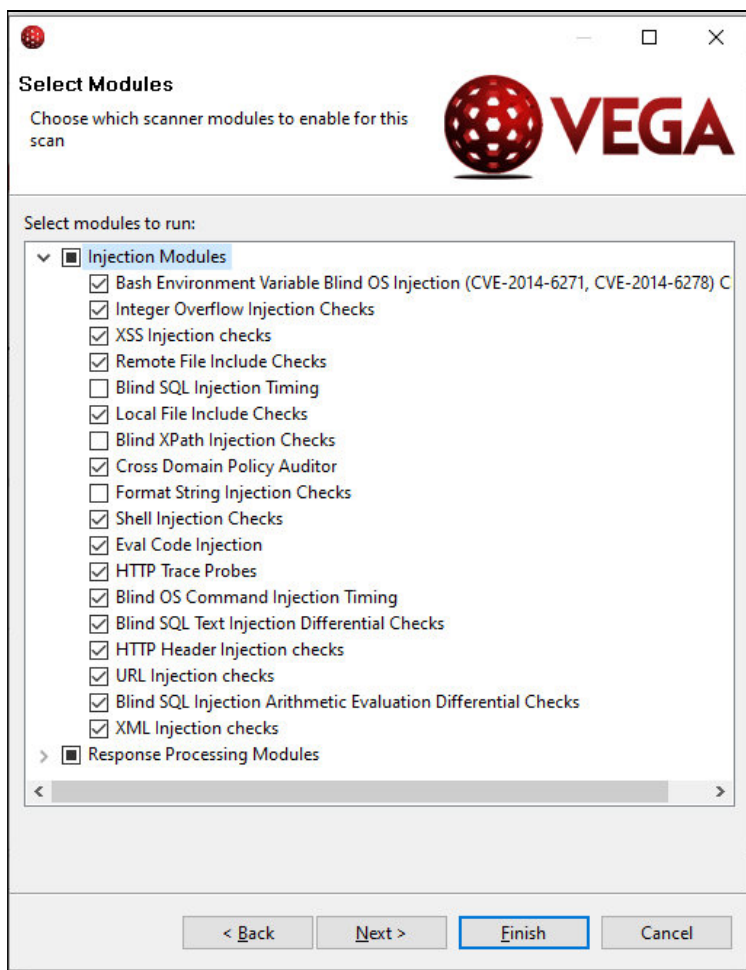


Рис. 1.3. Настройки Vega

Запуск сканирования тоже тривиален, но, в отличие от других сканеров, у Vega много настроек. А скрыты эти настройки за кнопкой **Next** (рис. 1.3).

Но и это еще не все! Есть поддержка авторизованного сканирования, причем без необходимости добавлять cookies из консоли.

Короче, благодаря удобному GUI, качественной работе и куче возможностей это сейчас лучший выбор для пользователя Windows. Если же чего-то не хватит, всегда можно написать свой модуль на JavaScript.

INFO

Было бы неправильно не упомянуть в этой подборке Nmap. Сам по себе он на звание сканера уязвимостей не тянет, но у него есть скриптовый движок. Даже «из коробки» он умеет проверять популярные баги, но ты легко можешь сделать этот глаз еще зорче с помощью своих (или чужих) скриптов. Как их создавать, журнал «Хакер» уже писал: <https://xakep.ru/2016/02/25/pimp-my-nmap/>, а найти скрипт на любой вкус можно на GitHub: <https://github.com/topics/nmap-scripts>.

Nessus

Цена: бесплатно / 3120 долларов

Nessus — коммерческий сканер безопасности американской компании Tenable. Также есть облачный сканер Tenable.io, на котором я не буду останавливаться подробно.

У сканера три редакции: Essentials (для всех и бесплатно), Professional (3120 долларов в год) и Tenable.io, который, по сути, представляет собой отдельный продукт со своей ценой. При этом разница между редакциями Essentials и Professional лишь в количестве доступных для сканирования адресов и наличии поддержки по email.

Сам сканер довольно увесистый (установщик больше 120 Мбайт, а после активации скачивает еще много дополнений), и скачать его можно только после регистрации, во время которой на почту придет код активации.

Инициализация весьма продолжительная, у меня заняла порядка двадцати минут. Когда с этим будет покончено, можно начинать пользоваться веб-панелью. Nessus ищет следующие проблемы:

- раскрытие версий ПО на хостах;
- активная малварь;
- уязвимость к брутфорсу;
- слабые методы авторизации;
- открытые данные на целях (возможность перечислить учетные записи и группы, удаленный реестр и сетевые папки);
- некорректные разрешения и политики безопасности.

Может работать как краулер. Лично мне не очень понравился, хотя бы жесткими рамками бесплатной версии.

Kube-hunter

Цена: бесплатно

Специализированный сканер для анализа безопасности кластеров Kubernetes. Распространяется по свободной лицензии Apache.

Этот охотник ищет косяки в удаленных кластерах, а затем одним метким выстрелом пробивает защиту. Пользоваться им следует с осторожностью, потому что в погоне за добычей он может что-нибудь испортить. Впрочем, об этом можно не беспокоиться, пока ты работаешь с ним в «обыкновенном» режиме. В таком режиме он будет находить дыры, но не полезет в них сам. Если же ты решил доверить все в руки Kube-hunter, есть режим «активной» охоты.

Скачать сканер можно на GitHub (<https://github.com/aquasecurity/kube-hunter>), он написан на Python и нормально работает почти в любой ОС. Впрочем, можно использовать и Docker:

```
docker pull aquasec/kube-hunter
docker run --rm aquasec/kube-hunter [ARGUMENTS]
```

После скачивания и установки зависимостей запускаем. Весь набор функций можно увидеть только с помощью дополнительных ключей запуска. Вот они:

- `--remote [ADDRESS]` — сканировать кластер по адресу;
- `--cidr [CIDR]` — найти и атаковать все кластеры в диапазоне адресов;
- `--active` — тот самый режим активной охоты. Используй, если в разрешении на пентест указано, что ты не несешь ответственности за сохранность инфраструктуры заказчика. Короче, я предупредил;
- `--mapping` — вывести все найденные узлы Kubernetes. Полезно с опцией `--cidr`;
- `--log [LEVEL]` — выводить сообщения по уровню важности. `[LEVEL]` может быть `DEBUG`, `INFO` (по умолчанию) или `WARNING`;
- `--report [TYPE]` — указывает формат вывода отчета. Может быть `json`, `yaml` или `plain`. Эту опцию можно сочетать со следующей;
- `--dispatch [MODE]` — указывает, куда нужно отправить отчет после завершения сканирования. По умолчанию `stdout`, но можно отправить и по HTTP, тогда параметром нужно передать `--dispatch http`. А чтобы Kube-hunter знал, куда именно отправлять результаты, объясни ему это в переменных среды:

```
KUBEHUNTER_HTTP_DISPATCH_URL (по умолчанию — https://localhost)
KUBEHUNTER_HTTP_DISPATCH_METHOD (по умолчанию — POST)
```

Очень удобная функция, которой нет в других сканерах, — возможность просматривать результаты работы на онлайн-ом дашборде, даже если сканер работает за NAT или еще как-то отгорожен от сети. Для использования фичи нужно зарегистрироваться на сайте компании: <https://kube-hunter.aquasec.com/>.

Скажу еще, что Kube-hunter можно использовать не только против удаленной цели. Еще его можно установить как `rod` и сканировать изнутри. Об этом подробнее написано в файле `readme`.

Trivy

Цена: бесплатно

Еще один сканер безопасности контейнеров того же разработчика (Aquasec). Для bug bounty пригоден куда меньше предыдущего, но довольно точен и быстр. Специализируется конкретно на Docker. Устанавливается чуть сложнее, чем Kube-hunter. Для установки в Debian, Ubuntu и Kali можно использовать следующий скрипт:

```
sudo apt-get install wget apt-transport-https gnupg lsb-release
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo apt-key add -
echo deb https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main | sudo tee -a /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy
```

Затем можно сканировать. Для этого просто выполни `trivy [IMAGE_NAME]:`

```
trivy python:3.4-alpine
```

Можно сканировать образы в виде файлов:

```
trivy --input image.tar
```

За формат вывода отвечает ключ `-f`, который можно выставить в json. Есть также поддержка вывода по кастомному шаблону.

Чтобы показать только определенные типы найденных уязвимостей, нужно указать ключ `--severity` и через запятую перечислить категории для отображения (UNKNOWN, LOW, MEDIUM, HIGH, CRITICAL).

PVS-Studio

Цена: ~5250 евро с возможностью получить бесплатно (и легально)

Бывает, что в рамках bug bounty для исследования предлагают исходный код продукта. Однако хороших инструментов для автоматизации такой работы очень мало. Вручную же искать баги среди миллионов строк кода — занятие весьма неблагодарное, тем более что кодовая база в активном проекте регулярно обновляется.

Исследователь устает, да и в принципе не способен выследить большинство косяков сам. Другое дело — софт! И тут нам помогли наши соотечественники, создав PVS-Studio. Это весьма годный коммерческий статический анализатор.

PVS-Studio умеет находить баги в коде на C, C++, C# и Java. Для использования его нужно скачать с сайта разработчика (<https://www.viva64.com/ru/pvs-studio-download/>) и установить. Есть версии для Windows (в виде расширения для Visual Studio), Linux (пакеты deb и rpm) и macOS.

Сканер требует регистрации — ввести ключ, полученный от разработчика. И тут я встретил самую быструю и дружелюбную реакцию среди всех производителей коммерческих сканеров.

Пользоваться PVS-Studio можно разработчикам открытых проектов, публичным специалистам в ИБ и обладателям статуса Microsoft MVP. Я получил лицензию по

второму способу как автор «Хакера». Если ты тоже занимаешься ИБ — попробуй получить такую лицензию.

Если ты используешь плагин для Visual Studio, то все найденные ошибки будут выведены в лог при компиляции или принудительной проверке (рис. 1.4).

В общем, сканер действительно хороший и быстрый, а пользоваться им при определенных обстоятельствах можно и бесплатно. Жаль, для остальных случаев цена кусается.

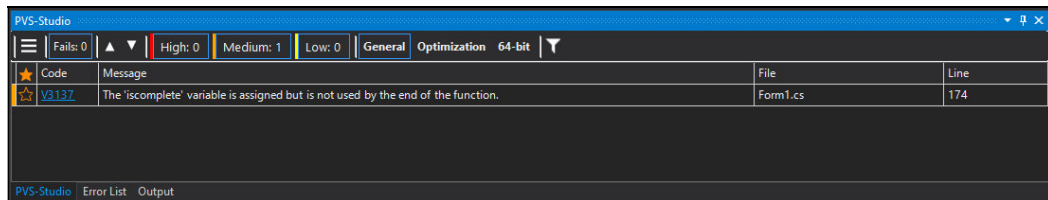


Рис. 1.4. Пример работы PVS-Studio

Gitleaks

Цена: бесплатно

Как известно, чем крупнее продукт, тем больше для его разработки привлекается подрядчиков и тем больше рядовых сотрудников взаимодействуют с его кодом. И пока все действия выполняются на виртуальных машинах, не подключенных к сети, а всех входящих и выходящих проверяют на рентгеновском сканере, все выглядит хорошо. Но когда для разработки и тестирования приглашают подрядчиков, которые работают из дома, начинаются проблемы.

Часто в коде тестов QA-специалистов попадают привилегированные ключи для доступа к инфраструктуре, а Security-инженеры забывают приватную информацию в своих публичных репозиториях для бэкапов. Если найти такой ключик, то взлом может закончиться, даже не начавшись, так что нельзя забывать про этот вектор атаки. Подобные данные можно найти по запросам вида `company.com pass` и `company.com private` на GitHub, а можно и с помощью автоматических скриптов.

Один из таких инструментов — это Gitleaks, который быстро и качественно найдет почти что угодно в любом открытом репозитории. Он умеет следующее:

- проверять локальные изменения до коммита, чтобы избежать утечек данных еще на стадии разработки;
- проверять любые репозитории GitHub/GitLab, в том числе приватные репозитории, если есть ключ доступа;
- проверять все репозитории заданного пользователя или организации;
- выдавать отчет в JSON, что удобно для последующего автоматического анализа;
- интегрироваться с Git, чтобы предотвратить непреднамеренную утечку.

Установить его можно с помощью Docker.

```
docker pull zricethezav/gitleaks
```

```
docker run --rm --name=gitleaks zricethezav/gitleaks -v -r [REPO_URL]
```

Использование инструмента и все его ключи запуска описаны в справке к программе, но некоторые из них я продублирую здесь.

- `-v` — детальный вывод сообщений;
- `--repo=[REPO]` — ссылка на репозиторий для проверки;
- `--disk` — клонировать репозиторий локально, чтобы не исчерпать всю память сразу;
- `--username=[USER]` и `--password=[PASS]` — указывает логин и пароль для доступа к приватному репозиторию;
- `--access-token=[TOKEN]` — альтернатива авторизации по логину и паролю;
- `--commit=[COMMIT]` — SHA коммита для анализа, если хочется проверить репозиторий в какой-то момент. По умолчанию проверяется текущее состояние;
- `--repo-path=[PATH]` — анализировать локальный репозиторий по заданному пути;
- `--branch=[BRANCH]` — анализировать только конкретный branch;
- `--depth=[NUM]` — анализировать только [NUM] последних коммитов. Альтернатива — `--commit-from=[COMMIT]` и `--commit-to=[COMMIT]`, проверяющие коммиты между заданными включительно;
- `--threads=[NUM]` — сканировать в несколько потоков.

Чтобы выпотрошить все репозитории конкретной организации или пользователя, используйте следующие параметры:

- `--org=[ORG]` — искать и анализировать все репозитории организации [ORG];
- `--user=[USER]` — то же, но для пользователя;
- `--exclude-forks` — исключить из анализа форки репозитория. В них редко можно найти что-либо полезное, а вот сканирование эта опция ускорит значительно.

Для использования этой функции надо указать утилите, какой сервис мы хотим задействовать. Пока поддерживаются GitHub и GitLab. Задать сервис можно опцией `--host=[SERVICE]`, где [SERVICE] — GitHub или GitLab.

В целом инструмент очень хороший и часто незаменимый. Рекомендую для использования в bug bounty и при пентестах.

QARK

Цена: бесплатно

Бывает, что нужно протестировать приложения для Android и iOS. И если с проверкой на секретные значения в коде все понятно, да и с анализом данных, сохраняемых в небезопасных местах, тоже, то некоторые баги выявить ой как непросто. Сюда попадает плохо реализованное шифрование (с помощью XOR, к примеру),

некорректная обработка внешних ссылок, открытые activity, которые раскрывают приватную информацию, флаг `android:debuggable=true` и так далее.

QARK (Quick App Review Kit) — бесплатный инструмент, созданный в компании LinkedIn, для быстрого анализа пакета APK на некоторые уязвимости. Их список, как говорит разработчик, следующий:

- ❑ некорректно экспортируемые элементы или неправильные права доступа к экспортируемым объектам;
- ❑ уязвимые интенты;
- ❑ неправильная работа с сертификатами X.509;
- ❑ создание файлов, которые доступны другим приложениям, и работа с такими файлами;
- ❑ дырявые activity;
- ❑ использование захардкоженных приватных ключей;
- ❑ слабые шифры;
- ❑ tapjacking;
- ❑ приложение разрешает бэкап своей приватной папки или имеет флаг `android:debuggable=true`.

Утилита написана на Python и способна работать как на Linux, так и в Windows. Установить можно через pip или путем сборки исходников самостоятельно. Первый способ:

```
pip install --user qark
```

Второй способ:

```
git clone https://github.com/linkedin/qark
cd qark
pip install -r requirements.txt
pip install . --user
```

После установки можно выполнить `qark --help`, чтобы прочитать справку. А можно и не выполнять, сейчас я кратко перескажу главное.

Для анализа целого APK используется аргумент `--apk FILE.APK`:

```
qark --apk ./my_app.apk
```

Если ты решил тестировать с помощью этого инструмента свой (или декомпилированный чем-то другим) код, то используй аргумент `--java`:

```
qark --java ./my_app_src
```

Можно прогнать через сканер только какой-то отдельный файл:

```
qark --java ./my_app_src/Main.java
```

Особенность сканера в том, что к некоторым багам он умеет сразу сделать эксплоит, но эта функция слегка нестабильна и работает далеко не для всех багов, так что особенно надеяться на нее не стоит.

Burp Suite

Цена: бесплатно / 400 долларов

Было бы странно, если бы я не упомянул этот швейцарский нож из мира сканеров. Это целый комбайн, который умеет работать как прокси, сканер уязвимостей, паук-краулер, репитер запросов или платформа для множества плагинов.

WWW

Про полезные плагины для Burp читай в статье «Прокачай свой Burp! 11 наиболее полезных плагинов к Burp Suite»: <https://xakep.ru/2018/08/23/burp-suite-plugins/>.

Сканер разработала компания PortSwigger (<https://portswigger.net/burp>) и выпустила в двух редакциях: Community (бесплатно) и Professional (400 долларов). Последняя отличается наличием в комплекте большого количества плагинов для Burp Intruder, наличием автоматического сканера и отсутствием ограничений в App Store.

Имеется неплохой GUI с удобными вкладками, автоматические модули для подбора паролей, идентификаторов, фаззинга, кодировщики и раскодировщики данных в разных форматах. Обзоров Burp не делал только ленивый, так что я только оставляю ссылку на такой обзор, чтобы не повторяться: <https://habr.com/ru/post/328382/>.

MobSF

Цена: бесплатно

Последний в этом разделе, но не последний по ценности и наворотам сканер, который будет отлично смотреться в арсенале пентестера. Это еще один статический анализатор мобильных приложений, тоже написанный на Python, но работающий только в Windows. MobSF запускается даже на «безголовом» сервере и открывает наружу порт 8000. Если зайти туда браузером, видим типичное веб-приложение (рис. 1.5).



Рис. 1.5. Главная страница MobSF

Загружаем туда APK или IPA (да, приложения для iOS тоже поддерживаются) и ждем окончания анализа. После этого видим весьма обширный отчет о найденных багах с указанием возможности эксплуатации. Цветовая индикация серьезности тоже присутствует, и в целом интерфейс весьма дружелюбный, не хватает только поддержки русского языка (рис. 1.6).

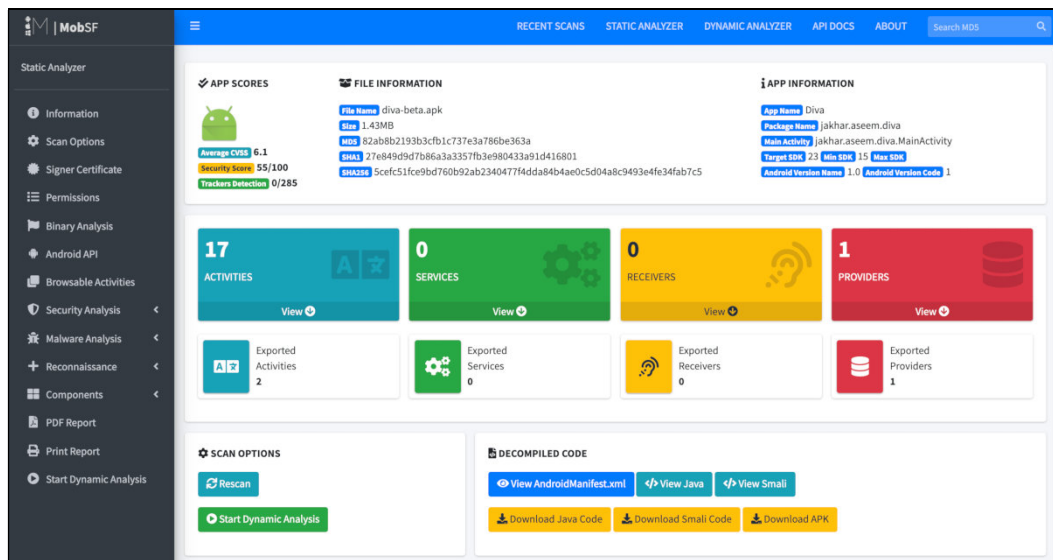


Рис. 1.6. Работа MobSF (иллюстрация взята из репозитория проекта)

Сканер умеет анализировать код, сертификат, которым подписано приложение, его манифест (AndroidManifest.xml) и позволяет выгрузить декомпилированный код для последующего анализа в других программах. Анализ выполняется как статически (декомпиляция и анализ полученного кода), так и динамически (запуск в виртуальном окружении) — пример работы динамического анализатора показан на рис. 1.7.

Можно скачать инструмент на GitHub (<https://github.com/MobSF/Mobile-Security-Framework-MobSF>) или установить по инструкции ниже. Для начала нам нужен Python с pip (<https://www.python.org/downloads/>). Далее установи `rsa` следующей командой:

```
python -m pip install rsa
```

Скачай скрипт установки (<https://raw.githubusercontent.com/MobSF/Mobile-Security-Framework-MobSF/master/install/windows/setup.py>) и выполни. Отвечай на вопросы (установщик интерактивный) — и пользуйся на здоровье.

Можно также основную часть инструмента запустить на одной машине (это может быть Linux), а сервер для статического анализа — на другой (тут нужна Windows). Чтобы выключить эту возможность, поправь `MobSF/settings.py`, указав в `WINDOWS_VM_IP` адрес твоей виртуальной машины с RPC-сервером.

В целом инструмент очень хороший и удобный. Правда, лично у меня установка вызвала некоторые проблемы — из-за битых зависимостей.

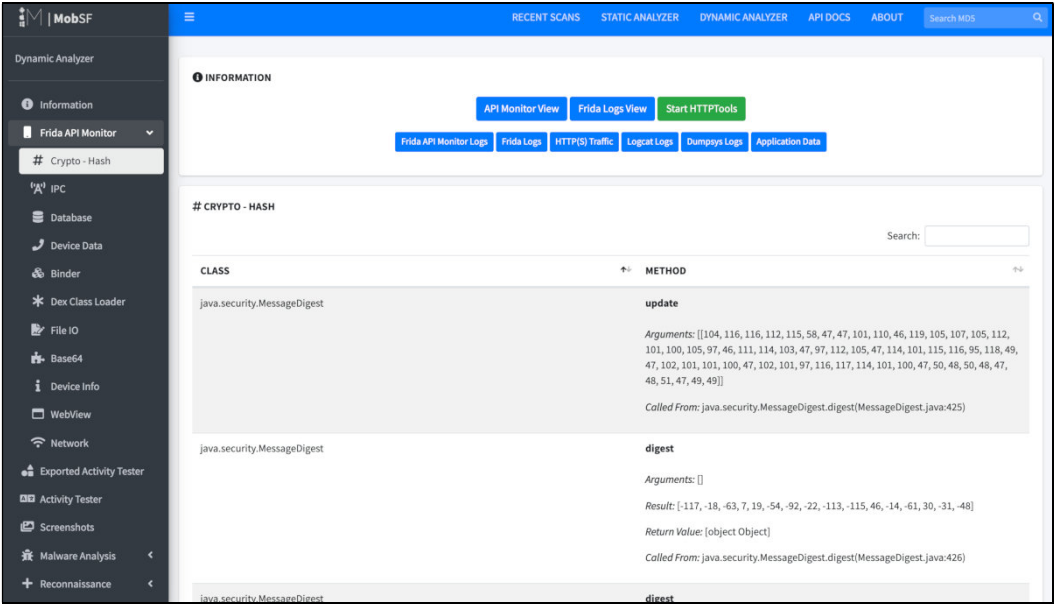


Рис. 1.7. Работа динамического анализатора

Вся информация о рассмотренных в этом разделе сканерах сведена в табл. 1.1.

Таблица 1.1. Сводные данные о сканерах

Сканер	Категория	Цена	Интерфейс	Комментарии
Sn1per	Web/Recon	Бесплатно/ 150+ долларов	Консольный	Универсальный ком- байн
Wapiti3	Web	Бесплатно	Консольный	
Nikto	Web	Бесплатно	Консольный	Немного устарел
OWASP ZAP	Web	Бесплатно	GUI	Быстрый и удобный
Sqlmap	Web	Бесплатно	Консольный+ GUI	Только SQL-инъекции
Acunetix	Web	4500+ долларов	Web-приложение	
Vega	Web	Бесплатно	GUI	Хорошая замена Acunetix
Kube-hunter	Kubernetes	Бесплатно	Консольный	
Trivy	Docker	Бесплатно	Консольный	
PVS-Studio	Анализ кода	Бесплатно/ ~5250 евро	Дополнение к VS	Умеет C/C++/C#/Java

Таблица 1.1 (окончание)

Сканер	Категория	Цена	Интерфейс	Комментарии
Gitleaks	Git	Бесплатно	Консольный	
QARK	Анализ кода	Бесплатно	Консольный	Слабоват по сравнению с MobSF
Burp Suite	Web	Бесплатно / 400+ долларов	GUI	
MobSF	Анализ кода	Бесплатно	Web-приложение	
Nessus	Анализ сети	Бесплатно / 3120+ долларов	Web-приложение	

Выводы

Как видишь, вовсе не обязательно ковырять сложные веб-приложения с тысячами страниц и прочий софт вручную. Вместо этого можно пойти по своим делам, сделать что-нибудь по дому, ну или написать еще одну статью в «Хакер». Огромная часть работы багхантера уже автоматизирована, так что не упusti свой шанс заработать немного денег. Или хотя бы на платную лицензию к сканеру!

15. Покоряем веб.

Как применять OWASP Testing Guide v4 в 2020 году

v31_v37

Безопасность веба — очень широкое понятие. Это и недостатки старых протоколов, и использование каких-то опасных вещей, и просто человеческие ошибки, допущенные при разработке софта. Очень непросто тестировать продукты в такой широкой области: нужно придерживаться какого-то плана. И организация OWASP облегчила жизнь специалистам в области ИБ, создав OWASP Testing Guide.

Существует несколько методик пентеста (<https://www.apriorit.com/dev-blog/524-web-application-security-testing>), но конкретно для веба создана только одна. О соответствии стандартам типа PCI DSS сейчас речь не идет, поскольку это узкоспециализированное направление. А мы поговорим об универсальной методологии тестирования. Что предлагает нам OWASP Testing Guide? Давай пробежимся по этому объемному документу и отметим его части, в которых тебя может ждать больше всего подводных камней.

WWW

Если у тебя не получается освоить английскую версию гайда, воспользуйся краудсорсинговым переводом (правда, он не доделан до конца), который можно найти по адресу <https://crowdin.com/project/owasp-testing-guide-40>. Кстати, у OWASP также есть методика для ревью исходного кода (https://www.owasp.org/images/5/53/OWASP_Code_Review_Guide_v2.pdf) и для тестирования мобильных приложений (<https://owasp.org/www-project-mobile-security-testing-guide/>). А с полным списком проектов ты можешь ознакомиться на owasp.org.

Не так давно в «Хакере» вышла статья (<https://xakep.ru/2020/02/27/hack-the-web/>), раскрывающая основы тестирования сайтов на безопасность, в ней мельком говорится и о методологии OWASP и области ее применения. Текущая версия OWASP Testing Guide (https://owasp.org/www-project-web-security-testing-guide/assets/archive/OWASP_Testing_Guide_v4.pdf) имеет номер четыре, пятая версия находится в стадии разработки (кстати, ты можешь делать коммиты в их публичном репозитории на GitHub: <https://github.com/OWASP/wstg>). Но хоть руководство по тестированию довольно большое и на первый взгляд всеобъемлющее, его надо воспринимать как основу, а не как рецепт на все случаи жизни. В этом материале тебя ждет краткая инструкция по использованию OWASP Testing Guide.

Не все то золото, что блестит

Как говорил Эйнштейн, «порядок необходим глупцам, а гений властвует над хаосом». Но в тестировании четкое планирование — это синоним успеха. Тем не менее план обычно описывает лишь приблизительную последовательность действий, даже если он очень детализирован. Предусмотреть все возможные нюансы зачастую нереально.

И дело не только в том, что новые технологии появляются с гораздо большей скоростью, чем обновляется методика, но и в том, что веб-приложения могут использоваться для чего угодно: от создания простого сайта-визитки до панели администратора, с помощью которой можно управлять физическими устройствами. Поэтому подобные методологии стоит использовать только в качестве фундамента и думать своей головой, при этом не забывая дополнять существующий план практическим опытом.

Следует использовать все доступные инструменты. Во-первых, во время тестирования по одному разделу инструменты могут давать разные результаты, а во-вторых, наложение части разделов на другие поможет закрыть потенциальные недочеты, ранее не выявленные тестировщиком или автоматическим инструментарием.

Также может сложиться впечатление, что методика больше предназначена для black box тестирования (несмотря на gray box и white box в самом тексте), но в принципе ее можно распространить на любой вид тестирования, добавив соответствующие методы (<https://www.ptsecurity.com/ww-en/analytics/knowledge-base/sast-dast-iaast-and-rasp-how-to-choose/>) и связанные с ними инструменты.

Testing Guide Introduction

В начале руководства по тестированию от OWASP есть небольшое предисловие, гласящее, что автоматизированное black box тестирование имеет недостатки и его надо дополнять ручным тестированием. Это так, однако в самом тексте гайда встречаются примеры использования сканера Nessus, но нет ни слова про сканер OpenVAS, который, в принципе, не сильно хуже (<https://www.capterra.com/vulnerability-management-software/compare/130577-171380/Nessus-vs-OpenVAS>).

Имеет смысл использовать все имеющиеся сканеры и другие фишки платных продуктов (например, Burp Pro — <https://medium.com/@svyatoslavlogyn/scanning-web-application-with-burp-suite-199abf6513a4>), поскольку разные инструменты могут дать разные результаты. Не пренебрегай и ложноположительными срабатываниями, поскольку такие результаты иногда внезапно оказываются истинными (<https://www.acunetix.com/blog/web-security-zone/xss-google-acunetix/>).

Testing for Information Gathering

Conduct search engine discovery/reconnaissance for information leakage

Сбор информации из открытых источников (OSINT) — первый этап любого пентеста, и пентеста веб-приложения тоже. Этот этап проводится еще до начала работ,

чтобы проверить, действительно ли тестируемые объекты принадлежат заказчику, или чтобы оценить примерный объем работ для оценки трудозатрат.

В методологии этот этап в основном строится на использовании поисковых движков (причем разных, чтобы скомпенсировать ограничения одного преимуществами другого). Здесь тебе на помощь придет статья, описывающая возможности DuckDuckGo (<https://xakep.ru/2017/03/06/duckduckgo-cli-options/>), заметка про операторы поиска (<https://xakep.ru/2017/09/21/google-dorks/>) и Google Dorks или материал о скрытых возможностях Google (<https://xakep.ru/2015/07/08/google-hidden-functions/>).

Но, разумеется, OSINT не ограничивается только использованием поиска, как минимум из-за наличия неиндексируемых форумов, в том числе в даркнете, или персонализированной выдачи. Поиграть с ней поможет сайт <http://isearchfrom.com/>. И на любом этапе тестирования не стоит забывать о персонализированной выдаче самого тестируемого ресурса. К слову, недавно появился форк Sherlock'a для поиска по СНГ (<https://habr.com/ru/post/488432/>). Можно использовать разные техники пассивного сбора информации (<https://xakep.ru/2018/11/21/recon-guide/>), например такой инструмент, как FOCA, для получения метаданных из документов, которые, скорее всего, присутствуют на тестируемых ресурсах.

Не стоит забывать про сайты, прямо или косвенно связанные с IT, а также тематические (связанные с тематикой тестируемого объекта) ресурсы. В общем, про OSINT можно говорить долго, суть в том, что надо использовать руководство по тестированию как основу, а не как пошаговую инструкцию.

Enumerate applications on webserver

В этом разделе речь идет о различных веб-приложениях, доступных либо по секретным субдоменам, либо по относительным путям, к которым имеется доступ извне. Имеются в виду некие ресурсы сайта, куда может проникнуть лишь тот, кто знает их URL, по понятным причинам нигде не афишируемый. Раздел можно дополнить следующими трюками:

- сайты могут быть похожи друг на друга (например, их делал один подрядчик). Можно использовать инструмент для поиска идентичных фрагментов кода (комментарии, разные идентификаторы в JS-библиотеках, имена авторов в комментариях и прочее) наподобие **publicwww.com** в надежде, что эти сайты были проиндексированы;
- можно использовать разные инструменты (<https://pentest-tools.com/information-gathering/find-subdomains-of-domain>) для поиска субдоменов или искать их самому вручную, используя поиск по сертификатам (<https://medium.com/@catalyst256/osint-certificate-transparency-lists-a603c9d2b776>) или DNS-запросы (<https://dnsdumpster.com/>). Можно использовать свои или общедоступные словари для перебора (<https://github.com/rbsec/dnscan>), ну и в целом привлекать разные инструменты (<https://medium.com/@dalvikbytecode/top-10-osint-tools-to-help-you-do-recon-a-domain-53d3af8b1ad2>), поскольку они постоянно обновляются и совершенствуются.

Map execution paths through application

Здесь говорится о составлении «карты» веб-приложения, то есть об отображении в текстовом или графическом виде всех или почти всех разделов сайта. Если этот процесс автоматизировать с помощью соответствующих инструментов (<https://github.com/BruceDone/awesome-crawler>), то ты получишь схему веб-приложения или сайта, на которую можно опираться при тестировании. Например, такая схема поможет классифицировать рубрики сайта по разделам методологии. К тому же автоматизированные утилиты могут обнаружить то, что ты упустил на этапе сбора информации.

Configuration and Deployment Management Testing

В этом разделе описано тестирование инфраструктуры веб-приложения. В гайде речь идет в основном о веб-сервере и СУБД. И хоть это фундамент любого веб-приложения, не стоит забывать про CI/CD-системы (https://github.com/ggquere/pwn_jenkins), шины сообщений (<https://quentinkaiser.be/security/tool/2017/08/28/cottontail-release/>) и прочие компоненты инфраструктуры. Конечно, если они входят в намеченный план работ.

Authentication testing / Authorization testing

При тестировании аутентификации и авторизации не стоит забывать про такие вещи, как OAuth(<https://www.sans.org/reading-room/whitepapers/application/paper/33644>), SSO(<https://blog.netspi.com/attacking-ss0-common-saml-vulnerabilities-ways-find/>), OpenID(<https://blog.compass-security.com/2019/07/practical-openid-connect-pentesting/>). Тебе даже может встретиться аутентификация по сертификатам (<https://habr.com/ru/post/349720/>).

В общем, не теряйся, когда на горизонте появится что-то подобное, ведь схем аутентификации и авторизации существует множество (<https://habr.com/ru/company/dataart/blog/262817/>). В один присест все не изучить, и практика их эксплуатации на реальных проектах появится не сразу: главное — понять, к какому разделу тестирования это относится.

Input validation testing

Первые два пункта этого раздела связаны с reflected/stored XSS-уязвимостями. Но XSS-уязвимости представляют собой подкласс более общих уязвимостей — reflected/stored HTML injection. Может случиться так, что XSS нет, а вот HTML injection есть. Кстати, помимо XSS/HTML-инъекций, также не забывай про инъекции в шаблоны (<https://portswigger.net/research/server-side-template-injection>) — довольно серьезный подкласс атак, который может привести к удаленному испол-

нению кода. Еще один подвид атаки удаленного исполнения кода — атака SSRF (<https://portswigger.net/web-security/ssrf>).

Также не стоит забывать о том, в каком окружении работает веб-приложение. Нужно подумать, как потенциальный злоумышленник может использовать это для своей выгоды. Вот пример утечки хешей с Windows-сервера из-за одной лишь уязвимости, связанной с недостаточно хорошей фильтрацией вводимых данных: <http://www.mannulinux.org/2020/03/abusing-file-system-functions-in-web.html>.

В этом разделе также говорится про бинарные уязвимости Overflow и Format String: сюда вообще стоит включить весь спектр бинарных уязвимостей со всевозможными ухищрениями и атаками, которые можно выполнить удаленно. Это тема для отдельной статьи или даже книги и еще одно подтверждение тому, что в области ИБ надо развиваться всесторонне.

Business logic testing

Вообще, в раздел, посвященный тестированию бизнес-логики, можно включить все что угодно. Проблемы в этой сфере могут привести к возможности DDOS-атаки, нарушению целостности, конфиденциальности и доступности информации. Вариантов можно придумать уйму, тут суть не в том, чтобы предусмотреть все возможные, а в том, чтобы научиться действовать по ситуации. Поэтому раздел носит в основном теоретический характер: практические приемы тестирования зависят от архитектуры и внутреннего устройства конкретного исследуемого объекта.

Client side testing

В первых двух пунктах этой части руководства речь снова заходит об XSS-уязвимостях, но на стороне клиента. Тут следует обратить внимание на два обстоятельства. Во-первых, не стоит забывать про HTML injection (ее тоже можно осуществить на стороне клиента). Во-вторых, XSS-уязвимости делятся на четыре типа: server-side reflected, server-side stored, client-side reflected и client-side stored. В последнем случае в качестве хранилища для XSS-нагрузки используется хранилище браузера (в пределах сессии или же на более долгий срок). На мой взгляд, деление должно быть именно по client-side reflected и client-side stored, ибо атаки DOM-based XSS и arbitrary JS injections могут быть выполнены в контексте обеих вышеупомянутых уязвимостей.

Аналогично родственником атаки SSTI (server side template injection) является CSTI (https://portswigger.net/kb/issues/00200308_client-side-template-injection). Принцип ее тот же самый, но выполняется она на стороне клиента.

Заключение

Веб-технологии имеют разные (иногда неочевидные) нюансы, и держать все это в голове просто невозможно, даже при наличии опыта. Главное оружие пентестера — это поисковые системы и свой личный набор инструментов.

А опыт приходит с практикой. Теоретические же знания можно почерпнуть в книгах, форумах, статьях, репортах на багбаунти-площадках. Можно даже вести свою собственную базу знаний — это особенно удобно, если ты посещаешь конференции вроде PHDays, ZeroNights, RuCTF, OffensiveCon или просматриваешь видеолекции. А методология тестирования в каждом новом проекте — это лишь отправная точка для дальнейшей работы.

Вот лишь небольшой список ресурсов, с которых можно начать составлять собственную базу знаний:

- Блог, посвященный пентестингу и ИБ: <https://bo0om.ru/>
- База данных веб-уязвимостей от Acunetix: <https://www.acunetix.com/vulnerabilities/web/>
- База уязвимостей, зафиксированных Portswigger Burp Scanner: <https://portswigger.net/kb/issues>
- Раздел про веб-безопасность на DEFCON: <https://defcon.ru/web-security/>
- Хаброблог OWASP: <https://habr.com/ru/company/owasp/>
- Гитхаб GoSecure: <https://github.com/GoSecure/presentations>
- Блог Corben Leo: <https://www.corben.io/>
- Блог Geekboy: <https://www.geekboy.ninja/blog/>
- Блог Detectify Labs: <https://labs.detectify.com/>
- Блог Wallarm: <https://lab.wallarm.com/>
- Блог Cisecurity: <https://www.cisecurity.org/cis-benchmarks/>

Стоит гуглить все, что связано с безопасностью того компонента, который используется в веб-приложении. Например, JWT(<https://habr.com/en/post/450054/>) по отношению к JAVA, Web Services (<https://resources.infosecinstitute.com/web-services-penetration-testing-part-1/>), если ты встретился с SOAP, или XXE (<https://habr.com/ru/company/owasp/blog/325270/>) и XSLT (<https://cryptoworld.su/4055-2-2/>), если нужно разобраться с XML-документами. Также надо быть готовым к ситуациям, не описанным в методологии (<https://habr.com/ru/company/vdsina/blog/481496/>), в том числе и к тому, что заказчик захочет протестировать свои системы защиты (https://www.owasp.org/images/b/bf/OWASP_Stammtisch_Frankfurt_WAF_Profiling_and_Evasion.pdf).

Подходи к процессу творчески: даже очень подробный гайд все равно не предусмотрит всех возможных случаев. Дополнительно для систематизации собственных знаний можно изучить разные классификации угроз безопасности (<https://safe-surf.ru/specialists/article/5210/595970/>).

«Хакер»: безопасность, разработка, DevOps

История журнала «Хакер» началась задолго до февраля 1999 года, когда увидел свет первый номер издания. Еще в ноябре 1998-го в сети DALnet появился русскоязычный IRC-канал #хакер, где активно обсуждались компьютерные игры и приемы их взлома, а также прочие связанные с высокими технологиями вещи. Тогда же в недрах основанной Дмитрием Агаруновым компании Gameland зародилась идея выпускать одноименный журнал; правда, изначально он задумывался как геймерский. Новое издание должно было подхватить выпавшее знамя нескольких закрывшихся компьютерных журналов, не переживших кризис 1998 года. В отличие от популярного «глянца» первой половины «нулевых», идея «Хакера» не была заимствована у какого-либо известного западного издания, а изначально являлась полностью оригинальной и самобытной.

Читатели приняли журнал более чем благосклонно: первый номер «Хакера» был полностью раскуплен в Москве за несколько часов, даже несмотря на то, что он поступил в продажу в 6 вечера. Журнал быстро набрал вирусную популярность, а одной из самых читаемых рубрик «Хакера» стал раздел «западлостроение», в котором авторы щедро делились с аудиторией практическими рецептами и проверенными способами напасть на ближнего своему при помощи различных технических средств разной степени изощренности.

Вскоре под влиянием читательских откликов тематика журнала стала меняться, постепенно смещаясь от игровой индустрии в сторону технологий взлома и защиты информации, что в общем-то вполне логично для издания с таким названием. Один из отцов-основателей «Хакера», Денис Давыдов, посвятивший свое творчество компьютерным играм, вскоре покинул редакционный коллектив, чтобы стать во главе собственного журнала: так появилась на свет легендарная «Игромания». Ну а «Хакер» с тех пор сосредоточился на вопросах, изначально заложенных в его ДНК: хакерство, взлом и защита данных. В марте 1999 года был запущен сайт журнала, на котором публиковались анонсы свежих номеров, — этот сайт и по сей день можно найти по адресу **xakep.ru**.

Уже в 2001 году тираж «Хакера» составил 50 тыс. экземпляров. Вскоре после своего появления на свет журнал уверенно завоевал звание одного самых популярных компьютерных изданий в молодежной среде — по крайней мере, именно так считает русскоязычная «Википедия». «Хакер» регулярно взрывал читательские массы

веселыми статьями о методах взлома домофонов, почтовых серверов и веб-сайтов, временами вызывая фрустрацию у производителей программного обеспечения и прочих представителей крупного бизнеса. На «Хакер» писали жалобы, а благодарные читатели приносили в редакцию пиво. Его сотрудников приглашали на телевидение и радио, а само издание в то же самое время называли «вестником криминальной субкультуры». В общем, и авторы, и читатели развлекались как могли.

«Хакер» развивался и рос, продолжая публиковать интересные статьи об операционных системах, программах, сетях, гаджетах и компьютерном «железе». Очень скоро все присылаемые авторами материалы перестали помещаться под одну обложку, и некоторые сугубо технические тексты постепенно перекочевали в отдельное тематическое приложение под названием «Хакер Спец».

В 2006 году объем «Хакера» едва не стал рекордным — 192 полосы. Выпустить номер такой толщины не получилось исключительно по техническим причинам. Со временем редакционная политика стала меняться: в журнале появлялось все меньше хулиганских статей, посвященных всевозможным компьютерным безобразиям, и все больше — аналитических материалов о секретах программирования, администрирования, информационной безопасности и защите данных. Но взлому компьютерных систем на страницах «Хакера» по-прежнему уделялось самое пристальное внимание.

Ключевым для истории журнала стал 2013 год, когда параллельно с традиционной бумажной версией стала выходить электронная, которую можно было скачать в виде PDF-файла. А последний бумажный номер журнала увидел свет летом 2015 года. С той поры «Хакер» издается исключительно в режиме онлайн и доступен читателям по подписке.

Сегодняшний «Хакер» — это популярное электронное издание, посвященное вопросам информационной безопасности, программированию и администрированию компьютерных сетей. Основу аудитории **xakep.ru** составляют эксперты по кибербезопасности и IT-специалисты. Мы пишем как о трендах и технологиях, так и о конкретных темах, связанных с защитой информации. На страницах «Хакера» публикуются подробные HOWTO, практические материалы по разработке и администрированию, интервью с выдающимися людьми, создавшими технологические продукты и известные IT-компании, и, конечно, экспертные статьи об информационной безопасности. С подборкой таких статей ты имел возможность ознакомиться на страницах этой книги. Аудитория сайта **xakep.ru** составляет 2 500 000 просмотров в месяц, еще несколько сотен тысяч подписчиков следят за новинками журнала в социальных сетях.

Современный «Хакер» отличается непринужденная, веселая атмосфера. Участники сообщества «Хакер.ru» получают несколько материалов каждый день: мануалы по кодингу и взлому, гайды по новым возможностям и новым эксплойтам, подборки хакерского софта и обзоры веб-сервисов. На сайте «Хакера» ежедневно публикуются знаковые новости из мира компьютерных технологий, рассказывающие о самых интересных событиях в сфере IT. Мы еженедельно готовим дайджесты, делаем подборки советов и полезных программ, изучаем свежие уязвимости.

В рубрике «Взлом» выходят интересные статьи о хакерских технологиях и утилитах, раздел «Кодинг» посвящен хитростям программирования, в рубрике «Приватность» собраны советы и мануалы по сетевой безопасности и сохранению своего инкогнито в Интернете. Статьи из раздела «Трюки» расскажут о недокументированных возможностях софта и нестандартных аппаратных решениях, системные администраторы найдут массу полезных рекомендаций по настройке ОС и прикладного ПО в разделе «Админ», а любители гаджетов и новомодного «железа» смогут насладиться рубрикой «Geek».

Присоединяйся к сообществу «Хакера» прямо сейчас! Материалы журнала выходят в нескольких форматах на выбор. Ты можешь подписаться в приложении на iOS или Android и читать ежемесячные выпуски, либо оформить подписку на сайте и получать статьи каждый будний день — сразу, как только они выходят. Подписка на сайте также дает возможность скачивать ежемесячный PDF и читать на любом удобном устройстве.

Когда «Хакер» только создавался, мы сказали себе: «Наша цель — чтобы среди наших ребят программирование стало самой популярной профессией». Мы использовали для этого все, что могли придумать, — развлекались, дурачились, как могли популяризировали ИБ, нашу субкультуру и тягу к IT в любых ее проявлениях. И мы считаем, что во многом достигли своей цели.

Присоединяйся, мы будем рады видеть тебя в нашей тусовке!

С самыми теплыми пожеланиями
редакция журнала «Хакер»

Предметный указатель

A

Acunetix WVS, 16
AJAX, 14, 37
Apache, 43, 66, 168
APK, 23, 25
Arachni, 12
arbitrary JS injections, 247
ASCII, 125

B

Base64, 84
black box, 244
Black Box, 11
Black SEO, 135
bounce message, 204
buffer over-read, 86
buffer underflow, 223, 230, 234
Burp Suite, 12

C

Censys, 12
Checkmarx, 129
CI/CD, 246
CMS, 210
Contao, 106
Content-Type, 217
cookie, 150
cookie-стиллер, 142
CORS, 16, 17, 83
CSRF, 16, 34, 65, 74, 84
CSRF-пейлоад, 84
CSRF-токен, 74, 101, 131
CSTI, 247

CVE-2018-6389, 107
CVE-2019-10149, 192
CVE-2019-11043, 223
CVE-2019-15107, 179
CVE-2019-6339, 210
CVE-2019-6341, 210
CxxQL, 129

D

DDOS, 247
Docker, 20, 65
DOM, 16
DOM-based XSS, 247
DoS, 107
Drupal, 210

E

Embedded Ruby (ERB), 163
EoP, 11
ETag, 13
Events, 196
EXIF, 59
Exim, 192, 193
Exim Monitor, 194
exim4-config, 194
expanded strings, 192

F

FastCGI, 223
Findsploit, 11
FOCA, 245
Format String, 247
FPM, 223

G

GD, 43, 62
 GitHub, 22
 GitLab, 22, 150
 Gitleaks, 21
 gray box, 244
 Guzzle, 219

H

HTML injection, 246, 247
 Hunter.io, 12

I

IIS, 168
 ImageMagick, 43, 58
 Integer Overflow/Underflow, 17
 IPA, 25

J

JavaScript, 108
 JSON, 13, 21
 JWT, 248

K

Kali Linux, 13
 Kubernetes, 19

L

local file inclusion, 17

M

Markdown, 151
 Marshal, 163
 Metasploit, 12, 14
 MITM-прокси, 14
 MobSF, 24
 ModSecurity, 168
 MS SQL, 15
 MySQL, 15, 42, 65, 115, 123
 MySQLi, 115

N

NAT, 19

Nessus, 244
 netcat, 197
 nginx, 168, 223, 225
 Nikto, 12
 Nmap, 12
 Nonce, 74
 nonce-токен, 74
 NSE, 12

O

OAuth, 246
 Open Redirect, 13
 OpenID, 246
 OpenVAS, 12, 244
 Oracle, 15
 OSINT, 244
 Overflow, 247
 OWASP, 14
 OWASP Core Rule Set, 170
 OWASP Testing Guide, 243
 OWASP ZAP, 10

P

path traversal, 17, 58
 PCI DSS, 243
 PCRE-модификатор, 213
 Perl, 179
 PHAR, 86, 88, 210, 221
 Phar::setMetadata, 91
 PHAR-десериализация, 218
 PHP, 86, 210, 223
 PHP Archive, 88
 PHP Object Injection, 94, 95
 phpBB, 210
 PHP-FPM, 223, 224, 228, 231
 phpggc, 219
 PHPGGC, 106
 PhpStorm, 211
 PHPStorm, 44
 PoC, 42, 84, 114
 PostgreSQL, 15
 POST-запрос, 54
 prepared statements, 116
 PrivEsc, 11
 PVS-Studio, 20

Q

QARK (Quick App Review Kit), 23

R

RCE, 36, 62, 121, 150, 192, 221, 223
 reflected/stored HTML injection, 246
 reflected/stored XSS, 246
 relay, 203
 RFC 3629, 213
 RFC 6265, 176

S

serialize, 91
 SHA, 22
 Shodan, 12
 Sn1per, 11
 SOAP, 248
 sqlmap, 10
 SQL-инъекция, 13, 15, 16, 114, 117
 SSL, 13, 14
 SSO, 246
 SSRF, 86, 106, 247
 SSTI, 247
 string expansion, 200
 String Expansion, 195
 stub, 90
 suid, 202

T

tapjacking, 23
 TCPDF, 106
 Twenty Nineteen, 62
 TYPO3, 106

U

unserialize, 210

user enumeration, 203
 User-Agent, 13
 UTF-8, 210, 213

V

Vega, 10, 16
 vulhub, 224

W

WAF, 14, 168
 Wapiti, 12
 Webmin, 179
 WebSocket, 14
 white box, 244
 White Box, 11
 Woocommerce, 95
 WPScan, 12
 WVS (Web Vulnerability Scanner), 10

X

X.509, 23
 xdebug, 65
 Xdebug, 44, 211
 Xdebug helper, 211
 XML-RPC, 99
 XML-RPC API, 130
 XSLT, 248
 XSS, 13, 16, 70, 82, 175, 210, 212, 221, 246
 XXE, 248
 XXE Injection, 13, 16, 17

Z

ZAP, 15
 ZAP (Zed Attack Proxy), 14

В

веб-шелл, 136

Д

десериализация, 86

динамическая типизация, 123

М

манифест, 25

маркер безопасности, 74

модификатор типа, 128

Н

небезопасная сериализация, 16

О

обфускация, 134

описатели преобразований, 121

П

пингбэк (pingback), 74

плейсхолдер, 121

С

санитизация, 68, 79

сериализованный объект, 45

скриптинг межсайтовый, 70

спам-страницы, 147

Т

трекбэк (trackback), 74

Ш

шелл, 58

АТАКИ

на Веб
и
WordPress

Перед вами сборник лучших, тщательно отобранных статей из легендарного журнала «Хакер», посвященных взлому веб-приложений и безопасности популярной CMS WordPress, под управлением которой работает более 700 млн сайтов в Интернете. Статьи рассказывают о практических методах поиска и анализа уязвимостей, раскрывают способы их эксплуатации. Подробно рассматриваются конкретные примеры уязвимостей, даны ссылки на видеоролики, наглядно демонстрирующие принцип их работы, приводится анализ кода, а также представлен обзор самых полезных хакерских инструментов.

Вы узнаете:

- как искать и эксплуатировать уязвимости в WordPress и веб-приложениях;
- какие инструменты используют профессиональные хакеры и пентестеры;
- как настроить стенд для самостоятельного анализа кода;
- как эксплуатировать брешы в Ngnix, Webmin, Drupal, Exim и GitLab.

Иван aLLy Комиссаров
Марк Бруцкий-Стемпковский
v31_v37
Ex.Mi

«Хакер» — легендарный журнал об информационной безопасности, издающийся с 1999 года. На протяжении 20 лет на страницах «Хакера» публикуются интересные статьи об операционных системах, программах, сетях, гаджетах и компьютерном «железе». На сайте «Хакера» ежедневно появляются знаковые новости из мира компьютерных технологий, мануалы по кодигу и взлому, гайды по новым эксплойтам, подборки хакерского софта и обзоры веб-сервисов. Среди авторов журнала — авторитетные эксперты по кибербезопасности и IT-специалисты.



191036, Санкт-Петербург,
Гончарная ул., 20
Тел.: (812) 717-10-50,
339-54-17, 339-54-28
E-mail: mail@bhv.ru
Internet: www.bhv.ru

ISBN 978-5-9775-6745-9



9 785977 567459