Rootkits и их обнаружение .

Rootkit - это комбинация из двух слов , “root” и “kit” . “Root” - имя суперпользователя UNIX , который имеет полный доступ к UNIX системе , “kit” – набор утилит . Поэтому Rootkit можно назвать так же “набор суперпользователя” , то есть это набор утилит , которые позволяют атакующему удерживать права суперпользователя на атакуемой системе и одновременно скрыть своё присутствие от системного администратора .

В UNIX окружении атакующий устанавливает rootkit в системе после получения доступа либо на пользовательском , либо на административном уровне . Административный уровень необходим для установки большей части “набора суперпользователя” . Его можно получить посредством атаки , используя уязвимости в программном обеспечении . Если атакующий обладает только пользовательским уровнем доступа , подбор пароля суперпользователя или локальная атака позволяет получить права суперпользователя и тем самым rootkit может быть успешно внедрён в систему .

Поскольку цель атакующего - скрыть своё присутствие в системе , утилиты rootkit должны обладать возможностью “маскировки” . Rootkit может состоять из нескольких утилит , таких как :

- “Back door”
  Backdoor это несанкционированный путь получения доступа к программе , сервису или ко всей системе . Утилита устанавливается в систему для того , чтобы позволить атакующему входить в систему , не осуществляя взлом заново . Существуют различные типы backdoor такие как :
  1. Login Backdoor – модифицирует login.c . Атакующий может входить в систему под любой пользовательской записью , используя пароль backdoor .
  2. Telnetd Backdoor - использует in.telnetd , позволяя атакующему получать доступ к системе используя backdoor пароль .
  3. Service Backdoor - заменяет и управляет сервисами такими как ftp , rlogin и даже inetd для получения доступа .
  4. Cronjob backdoor - позволяет получение доступа в определённые периоды времени .
  5. Library Backdoor - затрагивает разделяемые библиотеки UNIX для получения административного уровня доступа .
  6. Kernel Backdoor - затрагивает ядро операционной системы для обеспечения эффективности и для возможности скрытия rootkit .
  7. Network traffic Backdoor - обычно использует TCP , UDP и ICMP , управляя сетевым трафиком . Даёт возможность скрывать сетевые соединения и избегать действия firewall и netstat .
- “Packet sniffers”
  Sniffer это программа и/или устройство , которое осуществляет мониторинг данных , передаваемых по сети . Многие сервисы , такие как ftp и telnet передают нешифрованные пароли и они легко могут быть захвачены и прочитаны .
- “Log-wiping utilities”
  В лог-файлы программы (сервисы) помещают информацию о своих действиях . Например для операционных систем UNIX wtmp - log это файл

, в который помещаются время и дата входа пользователя в систему . Лог-файлы позволяют администратору осуществлять обзор функционирования операционной системы , производительности , а так же зафиксировать попытки неавторизованного доступа . Удалением записи доступа в лог-файле атакующий может скрывать своё присутствие в системе .

- “Miscellaneous programs”
  Rootkits также содержат дополнительные утилиты (обычно определяется составляющим пакетом) как например :
  IRC and bot - программа , устанавливаемая атакующим в скомпрометированную систему . Обычно осуществляет коннект к какому-либо серверу и ожидает от атакующего команды на выполнение .
  Log editor - полезен для редактирования лог-файлов на атакуемой системе .
  System patches - после успешной установки rootkit атакующий осуществляет изменение кода системы , для того чтобы предотвратить возможность других атак .

В 80-х годах UNIX был преобладающей операционной системой . UNIX система содержит набор утилит для мониторинга процессов и доступа , например ls - для просмотра и получения списка файлов , who - определяющая , кто в данный момент находится в системе , ifconfig - для определения состояния сетевых интерфейсов а так же их настройки .

Поскольку UNIX содержит системные утилиты , которые осуществляют просмотр процессов , запущенных в системе , хакеры постоянно пытались найти обходной путь , чтобы скрыть своё присутствие в системе .

Первый “Троянский конь” был написан для SunOS 4 и позже для Linux .
В 90-х годах было обнаружено большое число серверов , на которых были установлены rootkits . Поскольку исходные тексты ядра Linux открыты , это дало хакерам новую возможность осуществления атак .

Типы Rootkits :

Rootkits можно разделить на два типа :

1. Application rootkit - устанавливается на уровне приложений .
2. Kernel rootkit - устанавливается на уровне ядра .

Application rootkit заменяет системные утилиты модифицированными аналогами . Модифицированные утилиты скрывают присутствие backdoor и также не помещают информацию об активности атакующего в лог-файлы . Утилиты , которые обычно заменяются : ls , find , du - скрывают файлы и директории атакующего ; ps , top , pidof - скрывают процессы атакующего ; netstat - скрывает сетевые коннекты атакующего ; killall - не способен убивать процессы атакующего ; crontab - скрывает активность процессов атакующего , которые прописываются в crontab ; tcpd , syslogd – в лог-файлы не будет помещена информация о каких либо сетевых соединениях , осуществляемых атакующим с или на скомпрометированную систему .
Другим методом сокрытия является хранение данных в скрытых директориях или файлах . Файл или директория , начинающаяся с “.” - наиболее простой метод скрытия . Они не будут видны командой ls при условии что не установлен флаг –a . Обычно атакующий создаёт скрытые файлы или директории в таких каталогах как

/dev , /var , /tmp , которые редко проверяются системными администраторами .

Kernel Rootkit обнаружить намного сложнее . Манипулируя и используя возможности ядра ОС kernel rootkits могут обходить проверку на уровне приложений . Хотя первый kernel rootkit был написан для Linux , он может быть модифицирован для других операционных систем . Ядро - главный модуль операционной системы . Это часть ОС которая первой загружается в память и остаётся в ней . Вследствие этого ядро должно быть настолько малым насколько это возможно , одновременно обеспечивая и выполняя нужные функции , требуемые остальной частью операционной системы . Обычно ядро ответственно за операции ввода-вывода , драйверы устройств , обслуживание центрального процессора , управление процессами и задачами и операции с диском .

Kernel rootkit главным образом использует полезное свойство LKM (linux kernel modules) Модуль ядра – это объектный код , который способен динамически подгружаться в ядро . LKM используются для загрузки драйверов и других модулей по требованию . Каждая операционная система имеет встроенные в ядро функции , которые используются в каждой операции . Эти функции известны так же как системные вызовы . Например открытие файла влечёт за собой системный вызов sys_open . Управляя системными вызовами , атакующий получает широкие возможности для действий . Например команды такие как ls , du могут быть использованы для скрытия файлов и директорий . В linux это может быть сделано посредством использования системного вызова sys_getdents() . Информация о процессах , находящаяся в /proc так же может быть скрыта .
Подобным образом также осуществляется :

- сокрытие сетевых соединений (/proc/net/tcp , /proc/net/udp)
- сокрытие LKM
- перенаправление исполнения файла

и многие другие .

Некоторые примеры kernel rootkits .

Knark

Knark устанавливает модуль ядра sysmod.o , который осуществляет сокрытие сокетов , файлов и директорий .Этот модуль изменяет семь системных вызовов : fork , read , execve , kill , ioctl , settimeofday , clone . Knark включает в себя так же набор утилит и несколько эксплоитов для установления контроля на других системах .

Adore

Adore осуществляет сокрытие файлов , процессов и выполнение привилегированных Команд . Текстовая утилита ava используется для контроля и управления модулем ядра , определяя , какой файл или процесс должен быть скрыт . Пароль встроен в модуль и саму утилиту . Подобно Knark Adore изменяет 8 системных вызовов .

Rkit

Rkit один из наиболее простых из kernel rootkits . Он просто изменяет системный вызов setuid для обеспечения прав суперпользователя определённому пользователю в системе .

Итак , как было показано выше , rootkits представляют собой серьёзную угрозу безопасности . Каким образом можно обнаружить присутствие rootkit в системе а также по возможности избежать его установки ?

Прежде всего , атакующий должен обладать привилегиями суперпользователя для установки rootkit . Поэтому необходим постоянный контроль защищённости системы , установка необходимых критических обновлений , закрытие неиспользуемых портов , использование сложных паролей , установка различных средств защиты таких как grsecurity , RSBAC .
К сожалению , даже если уровень системы поддерживается на должном уровне , хакеры всё же могут обнаружить дыры в системе и получить привилегии суперпользователя , поскольку 100 % защищённость не может быть достигнута .
Один из возможных методов обнаружения rootkit - использование команды echo * Для просмотра содержимого каталога . В то время как команда ls очень часто подвергается модификации , команда echo практически не затрагивается . Поэтому , echo * покажет содержимое каталога полностью . Однако , использование такой команды удобно только в некоторых случаях . Более разумно использовать другие методы . Сейчас доступны утилиты , которые осуществляют проверку /bin/login программ , проверяя на присутствие известных rootkits , причём агент устанавливается на защищённой машине . Ещё один , пожалуй самый лучший метод защиты – вычисление контрольных сумм важных системных файлов . MD5 - очень подходящий алгоритм для таких целей . Он поддерживает создание уникальных последовательностей бит , основанных на содержимом данного файла . Поскольку MD5 - “one-way” функция , атакующий не сможет определить каким образом нужно модифицировать файл , чтобы значение функции не изменилось . Следовательно , системный администратор должен создать базу значений MD5 для всех системных файлов , сохранить базу в защищённом месте на устройствах доступных только на чтение (например CDROM) и периодически сравнивать значения функции текущих системных файлов с данными из базы . Конечно , такую базу нужно создавать до того как система будет скомпрометирована , иначе обнаружить какие-то изменения не удастся . Поэтому , некоторые компании , разрабатывающие ОС , хранят базу данных значений функций всех критических исполняемых системных файлов , и эти базы доступны через Web (http://sunsolve.sun.com/pub-cgi/show.pl?target=content/content7.) Поскольку заранее известно , что эти суммы верны , системные администраторы могут сравнить свои базы и убедиться в идентичности .

Обнаружение kernel rootkit значительно сложнее . Для того чтобы получить список модулей , присутствующих в ядре на данный момент , используется методы такие как : lsmod ; cat /proc/modules . Кроме того , возможно посмотреть список символов в /proc/ksyms для каждого модуля . К несчастью , поскольку kernel rootkit это модуль , попытки определения rootkit такими методами можно легко свести на нет . Для того чтобы заменить системные вызовы ядра своими , LKM rootkit изменяет таблицу адресов системных вызовов , указывающих на заменяемые модулем функции. Во время компиляции ядра создаётся так называемая карта символов ядра (map of kernel symbols) . Эта карта называется System.map и обычно устанавливается в то же место что и ядро . С её помощью возможно определить изменённые системные вызовы , сравнивая их адреса текущими .

Другим методом защиты от kernel rootkits является отключение возможностей LKM .

Поскольку динамическая загрузка модулей становится невозможной , можно не беспокоиться о том , что вредоносный код будет помещён в ядро . К сожалению , это не единственная возможность . Даже без поддержки LKM , код может быть внедрён посредством прямой записи в /dev/kmem . И всё же , отключение LKM принесёт серьёзные проблемы атакующему .

4.4BSD представила новые уровни защиты ядра и флагов файлов . Некоторые флаги не могут быть установлены на файл . Для поддержки этой возможности , ядро работает на определённом уровне защиты . Уровень защиты может быть повышен любым процессом с привилегиями суперпользователя , он он может быть снижен только процессом init (pid 1) . Существуют 4 уровня защиты :

-1 : Постоянный незащищённый режим . Ядро работает на уровне защиты 0 и он не может быть повышен .

0 : Незащищённый режим . Флаги могут быть установлены и сброшены и устройства могут быть доступны для чтения или записи в соответствии с установленными правами .

1 : Защищённый режим . Флаги , установленные суперпользователем , не могут быть сброшены . Файлы устройств /dev/mem , /dev/kmem и для примонтированных файловых систем в режиме только чтение .

2 : Очень защищённый режим . Файлы устройств для файловых систем всегда в режиме только чтение , монтированы они или нет . Правила межсетевого экрана не могут быть изменены .

Для изменения флагов файлов , создания новых файловых систем , либо изменения правил межсетевого экрана система должна быть перезапущена в более низкий уровень защиты . Однако это гарантирует , что администратирование осуществляется только проверенным лицом , что обеспечивается не только программно но и физически .

Существует ряд утилит , которые позволяют определить изменения в ядре на работающей системе :

chkrootkit - http://www.chkrootkit.org
rkscan - http://www.hsc.fr/ressources/outils/rkscan
carbonite - http://www.foundstone.com/rdlabs/termsofuse.php?filename=carbonite.tar.gz
rkdet - http://www.vancouver-webpages.com/rkdet/
LSM - http://freshmeat.net/projects/lsm/

Используемые статьи :

http://infosecwriters.com/texts.php?op=display&id=156 - The Art of Rootkits .

http://la-samhna.de/library/rootkits/detect.html - Detecting Kernel Rootkits .

http://www.phptr.com/articles/article.asp?p=23463&seqNum=3 - Even Nastier: Traditional RootKits.

http://www.l0t3k.org/biblio/rootkit/english - Saliman_Manap_GSEC.

Дополнительные ссылки :

http://www.rsbac.org/
http://www.grsecurity.net/