Эссе

по “Защите информации”

на тему:

“ Скрытый модуль для удалённого доступа к Линукс-машинам”

Выполнил: Кондрашов Антон (116 группа)

Если ваш компьютер работает под управлением Linux, подключен к Internet и на нем хранится важная информация, наверное, вы бы не хотели, чтобы к ней имел доступ кто-то еще. Абсолютно защищённый компьютер – это недостижимый идеал, поэтому существует опасность, что к вашей системе получит доступ (и некоторые права) хакер. Если это произошло (а это возможно, так как в любой системе есть ошибки), то следущим шагом злоумышленника будет попытка скрыть своё присутствие, а также обеспечить возможность попадания в вашу систему в будущем. Если ваш компьютер подключён к сети, то хакер может попытыться поперехватывать информацию и собрать пароли от других компьютеров. Существует коллекция инструментов (Rootkit) как раз и созданная, чтобы помочь ему в этом.

В состав Rootkit'а может входить множество различных программ, но их можно разбить на следущие группы: файлы анализатора пакетов, скрипты для "подчистки" логов и программы-троянцы, заменяющие системные утилиты.

Демоны.

Основная цель rootkit состоит в том, чтобы позволить злоумышленникам возвращаться в систему позже и получать к ней доступ без угрозы быть обнаруженным. rootkit делает это достаточно просто, устанавливая программу-демона удаленного доступа, как например измененные версии telnetd или sshd. Такие демоны как правило работают с портами, отличными от используемых по умолчанию. Системные демоны обычно запускаются при старте системы из **/etc/rc?.d/** файлов или при помощи мета-демона **inetd**. В эти файлы можно добавить старт своего демона. Для затруднения обнаружения посторонних соединений путем прослушивания сети или обмана firewalla чужая программа может использоваться **UDP** протокол или **ICMP** пакеты.

Модифицированные файлы.

В ядре системы есть много бинарных файлов, например, как w, who, find, ps. Приполучении доступа к системе хакер постарается их заменить своими. Именно для этого в rootkit'е собраны модифицированные версии этих файлов. В результате замены бинарных файлов в ядре, используемых для мониторинга серверной активности, процессы, запускаемые хакерами, становятся невидимыми для системных администраторов.

Также изменениям подвержены и другие системные файлы. Вот несколько примеров конфигурационных файлов подверженных модификациям со стороны хакеров:

/etc/passwd, /etc/shadow - в этих файлах хранится информация о аккаунтах. Злоумышленник может добавить в них свои записи или изменить аттрибуты этих файлов для внесения в произвольный момент своей информации.

/var/spool/cron/crontabs/ Cron создает заказанные процессы в установленное время. Взломщик может добавить строку, создающую, например, .rhosts файл в полночь и уничтожающую его утром в файл заданий root.

```
/var/spool/cron/crontabs/root:
0 22 * * 6 "echo '+ +' > /.rhosts"
0 6 * * 1 "rm -rf /.rhosts"
```

Rootkit'ы при замене системных бинарных файлов заражёнными, как правило, стараются изменять дату создания и размер, так чтобы они были неотличимы от исходных, поэтому ведение учета статистических данных файла не достаточно.

/var/sadm/install/contents - этот файл хранит информацию о проинсталлированных в системе файлах, их размеры, атрибуты и контрольную сумму. Этот файл может быть изменен для сокрытия модификации программ.

Таким образом, наилучший способ обнаружения подобных действий хакера – это создание списка информации файлов системы. С помощью такого списка можно идентифицировать подозрительные действия на сервере. Для этого нужно вычислить котрольные суммы всех системных файлов и хранить эту информацию в безопасном месте, таком как, например, компакт-диск. А дальше при малейших подозрениях проверять контрольный суммы.

Естественно, существуют вспомогательные инструменты для системных администраторов. Например, Tripwire или AIDE делают этот процесс намного легче, автоматизируя вычисление подписей файла.

Так как в процессе работы вы устанавливаете новое программное обеспечение в систему, то вам придёт повторять процесс подсчёта контрольных сумм. Для это цели также можно воспользоваться **RPM** или **R**ed **H**at **P**ackage **M**anager (Менеджер пакетов RedHat). Она позволяет пользователям брать исходный код для нового программного обеспечения и создавать контрольные суммы с помошью алгоритма **MD5**. Далее вы сможете сравнивать текущие контрольные суммы ваших файлов с контрольными суммами в инсталяционной базе данных **RPM**. К сожалению, **RPM** приложению и локальной базе данных **RPM** при обеспечении точной информации нельзя доверять наверняка, потому что существует потенциальная опасность, что злоумышленники могут заразить и их.

Sniffer'ы и keylogger'ы.

Некоторый rootkits может также содержать анализаторы сетевых пакетов или приложения для записи клавиатурных логов, которые используются для сбора паролей других систем и слежения движения аналогичной информации. Чтобы сделать это, rootkit'ы переводят сетевые карты в состояние **PROMISCIOUS**. В результате чего карта начинает прослушивать всю информацию в сети, а не только ту, которая предназначена для неё.

Если в вашей сети используются некоммутирующие HUB'ы, то все приходящие извне пакеты пересылаются всем компьютерам в сети. Но если адаптор сети находится в нормальном состоянии, то он не глядя отсеивает все пакеты, которые предназначены не ему. Если же сетевая карта находится в состоянии **PROMISCIOUS**, то она просматривает всю информацию.Если злоумышленник просматривает эту информацию в достаточно большой сети, результаты могут быть катастрофическими. Защитить всю сеть, если одна из машин взломана и используются прямые кабельные соединения и простые HUB'ы, не получится. Switch'и и другое более интеллектуальное сетевое оборудование не распространяет пакеты на все машины в сети, а лишь посылают их машине, которой они предназначены, эффективно защищая все другие машины в сети.

Утилиты, для модификации лог-файлов.

Деятельность всех процессов в системе фиксируется в лог-файлах. Поэтому, первое, что делает системный администратор для контроля системы, просматривает и анализирует лог-файлы. Чтобы скрыть следы своей деятельности хакер должен модифицировать эту информацию. Для этой цели во все rootkit'ы включена утилита, с помощью которой легко затирается нужная информация логах системы. В некоторых

чрезвычайных случаях, rootkit'ы вообще отключают ведение лог-файлов и стирают все существующие логи. Но обычно, если злоумышленники намереваются использовать сервер длительный промежуток времени, то они удаляют только те части лог-файлов, которые могут указать на их присутствие.

Некоторые администраторы установают в системе программы, которые обнаруживают аномалии в системе и присылают электронные письма и/или сами лог-файлы. Однако, методу анализа состояния сети и периодической пересылке лог-файлов нельзя доверять после того, как хакер получил доступ к системе.

LKM-трояны.

На некоторых Линукс-машинах разрешено использовать механизм добавления новых фунциональных возможностей ядру операционной системы (**Loadable Kernel Module** или Загружаемый Модуль Ядра) во время работы этой системы, не требуя отдельной перекомпиляции ядра. Конечно, выгоды использования этого механизма очевидны, но прииспользовании этого механизма появляется дополнительная опасность исходящая от **LKM**-троянов встроеных в rootkit'ы. Дело в том, что если под видом какой-то полезной функции в ядро системы был добавлен **LKM**-троян, то теперь он будет в строен в систему и будет запускаться вместе с запуском системы как и другие модули. Загружаемые Модули Ядра используются многими операционными системами, включая Linux, Solaris, и FreeBSD.

Примером может послужить следующий кусок кода, помогающий взломщику перехватывать SYS_setuid:

```
SYS_setuid Solaris 2.6
....
proc_t  p;
user_t  ut;
int my_setuid(uid_t uid)
{
        int rval;

        p = ttoproc(curthread);
        mutex_enter(&p->p_lock);
        up = prumap(p);
        rval = bcmp(up->u_comm, "devil", 5);
        (void)prunmap(p);
        mutex_exit(&p->p_lock);
        if( rval) {
                rval = setuid(uid);
        } else {
                ...
        }
        return 0;
}
....
```

Аналогичным образом можно перехватывать системные вызовы обращения к файлам, получения информации о системе и сознательно искажать получаемую информацию. Например, при открытии файла **"/etc/inetd.conf"** будет происходить открытие резервной копии этого файла спрятанной в системе. Таким образом скрывается изменение системных файлов.

```
SYS_open   Solaris 2.6
...
static char     new_path = "/var/tmp/.locks/.idx/inetd.conf";
int my_open(char *fname, int fmode, int cmode)
{
        int rval;

        rval = bcmp(fname, "/etc/inetd.conf", 16);
        if( rval) {
                return(copen(fname, (int)(fmode-FOPEN), cmode));
        } else {
                return(copen(new_path, (int)(fmode-FOPEN), cmode));
        }
}


            ...
```

Вообще-то можно криптографически распознать модули ядра. Но всё же лучше не рисковать. Естественно, если можно обойтись без **LKM** возможностей, то лучше вообще запретить их использование и компилировать новые функции статически. Например, в системе сервера, где вряд ли понадобятся дополнительные функциональные возможности ядра длительное время, использование LKM-функции – необоснованный риск.

Единственный стопроцентно работающий способ борьбы с rootkit'ами в вашей системе состоит в том, чтобы недопустить их попадание в вашу систему. А для этого нужно не дать возможности изначального проникновения хакера на ваш компьютер. Помните, что rootkit разработан не для того, чтобы помочь злоумышленнику получить доступ к системе. rootkit разработан, чтобы хакеры чувствовали себя как дома у вас в системе и тихо работали на вашем компьютере не беспокоясь ни о чём. Чтобы устанавить rootkit, злоумышленник сначала должен получить неправомочный доступ к вашему серверу, используя традиционные методы (например использование известной уязвимости или социальная инженерия).

Ссылки, использованные в эссе:

http://www.linuxdevcenter.com/pub/a/linux/2001/12/14/rootkit.html - Understanding Rootkits, by Oktay Altunergil

http://www.linuxdevcenter.com/pub/a/linux/2002/02/07/rootkits.html - Scanning for Rootkits, by Oktay Altunergil

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.impb.psn.ru/DOC/security/doors/doors.html - Backdoors

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