Министерство образования и науки, молодежи и спорта Украины
Донецкий национальный университет
Физический факультет

**Белоусов В.В., Бондаренко В.И.,
Данилов В.В., Каргин А.А.,
Кожемякин Ю.А.**

Обеспечение безопасности в UNIX-подобных операционных системах

Донецк 2011

Министерство образования и науки, молодежи и спорта Украины
Донецкий национальный университет
Физический факультет

Обеспечение безопасности в UNIX-подобных операционных системах

Лекции и практические работы

УТВЕРЖДЕНО
на заседании совета
физического факультета
Протокол № 2
от 21.11.2011 г.

Донецк ДонНУ 2011

УДК 519.682

Обеспечение безопасности в UNIX-подобных операционных системах Лекции и практические работы (для студентов дневной и заочной формы обучения, специальности 6.170101 «Безопасность информационных и коммуникационных систем») / сост. В.В. Белоусов, В.И. Бондаренко, В.В. Данилов, А.А. Каргин, Ю.А. Кожемякин – Донецк: ДонНУ. – 2011. – 48с.

Учебное пособие посвящено вопросам информационной безопасности. В лекциях дается общее понятие информационной безопасности, рассматривается краткая история ее развития. Анализируются современные стандарты обеспечения информационной безопасности в среде Unix. Определяются основные компоненты защиты информации. Анализируются средства технической безопасности.

Предназначено для студентов, магистров и аспирантов университетов, занимающихся проблемами информационных технологий.

Рецензенты:

Башков Е.А. – д-р техн. наук, проф., проректор по научной работе, зав.кафедрой Прикладной математики и информатики Донецкого национального технического университета.

Семко А.Н. – д-р техн. наук, проф. кафедры Общей физики и дидактики физики Донецкого национального университета

Составители: В.В. Белоусов, д-р техн. наук, проф.
В.И. Бондаренко, старший научный сотрудник
В.В. Данилов, д-р техн. наук, проф.
А.А. Каргин, д-р техн. наук, проф.
Ю.А. Кожемякин, старший научный сотрудник

Отв. за выпуск: А.А. Каргин, д-р техн. наук, проф.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В данном пособии рассматриваются основные положения операционной системы Unix и Unix-подобных систем. Сделан исторический обзор истории развития Unix, рассмотрены основные характеристики операционной системы, представлены условия обеспечения безопасности. В разделе «Шаг за шагом» представлены пути проверки системы Unix на ошибки в конфигурации или на наличие неизвестных процессов и учетных записей. Сформулированы контрольные вопросы.

В приложении даны практические рекомендации по обеспечению безопасности систем **Ubuntu Linux** и **Android.**

Название данного пособия объясняется тем, что Деннис Ритчи, один из создателей Unix, выразил своё мнение, что Unix-подобные системы, такие как Linux, являются де-факто Unix-системами. Эрик Рэймонд предложил разделить Unix-подобные системы на 3 типа:

1) Генетический Unix

Системы, исторически связанные с кодовой базой AT&T. Большинство, но не все коммерческие дистрибутивы Unix-систем попадают под эту категорию, так же, как и BSD-системы, которые являются результатом работы университета Беркли в поздних 1970-х и ранних 1980-х годах. В некоторых из этих систем отсутствует код AT&T, но до сих пор прослеживается происхождение от разработки AT&T.

2) Unix по товарному знаку, или бренду

Эти системы, в основном коммерческого характера, были определены The Open Group как соответствующие Единой спецификации Unix, и им разрешено носить имя Unix. Большинство этих систем – коммерческие производные кодовой базы Unix System V, в той или иной форме (например, Amiga Unix), хотя некоторые (например, z/OS компании IBM) получили торговую марку через слой совместимости с POSIX, не являясь по сути Unix. Многие старые Unix-системы не подходят под это определение.

3) Unix по функциональности

В целом, любая система, поведение которой примерно соответствует спецификации Unix. К таким системам можно отнести Linux и Minix, которые ведут себя подобно Unix-системе, но не имеют генетических связей с кодовой базой AT&T. Большинство свободных/открытых реализаций Unix, являясь генетическим Unix или нет, подпадают под ограниченное определение этой категории в связи с дороговизной сертификации The Open Group, которая стоит несколько тысяч долларов.

Cygwin, не являясь операционной системой, предоставляет Unix-подобную среду в Microsoft Windows; также существуют сервисы Microsoft Windows для Unix.

1. ИСТОРИЯ

В 1957 году в Bell Labs была начата работа по созданию операционной системы для собственных нужд. Под руководством Виктора Высоцкого была создана система BESYS. Впоследствии он возглавил проект Multics, а затем стал главой информационного подразделения Bell Labs.

В 1964 году появились компьютеры третьего поколения, которым возможности BESYS уже не соответствовали. Высоцкий и его коллеги приняли решение не разрабатывать новую собственную операционную систему, а подключиться к совместному проекту General Electric и Массачусетского технологического института Multics. Телекоммуникационный гигант AT&T, в состав которого входили Bell Labs, оказал проекту существенную поддержку, но в 1969 году вышел из проекта, поскольку он не приносил никаких финансовых выгод.

Первоначально Unix была разработана в конце 1960-х годов сотрудниками Bell Labs, в первую очередь Кеном Томпсоном, Деннисом Ритчи и Дугласом МакИлроем.

В 1969 году Кен Томпсон, стремясь реализовать идеи, которые были положены в основу MULTICS, но на более простом аппаратном обеспечении (DEC PDP-7), написал первую версию новой операционной системы, а Брайан Керниган придумал для неё название – UNICS (UNIplexed Information and Computing System) – в противовес MULTICS (MULTIplexed Information and Computing Service). Позже это название сократилось до Unix.

В ноябре 1971 года вышла версия для PDP-11, наиболее успешного семейства миникомпьютеров 1970-х (в СССР его аналоги, выпускавшиеся Министерством электронной промышленности, были известны как СМ ЭВМ и «Электроника», позже – ДВК, производились в Киеве, Воронеже, Зеленограде). Эта версия получила название «первая редакция» (Edition 1) и была первой официальной версией. Системное время все реализации Unix отсчитывают с 1 января 1970 года.

Первые версии Unix были написаны на ассемблере и не имели встроенного компилятора с языком высокого уровня. В 1969 году Кен Томпсон при содействии Денниса Ритчи разработал и реализовал язык Би (B), представлявший собой упрощённый (для реализации на миникомпьютерах) вариант разработанного в 1966 языка BCPL. Би, как и BCPL, был интерпретируемым языком. В 1972 году была выпущена вторая редакция Unix, переписанная на языке Би. В 1969 - 1973 годах на основе Би был разработан компилируемый язык, получивший название Си (C).

В 1973 году вышла третья редакция Unix, со встроенным компилятором языка Си. 15 октября того же года появилась четвёртая редакция, с переписанным на Си системным ядром (в духе системы Multics, также написанной на языке высокого уровня ПЛ/1), а в 1975 – пятая редакция, полностью переписанная на Си.

С 1974 года Unix стала бесплатно распространяться в университетах и академических учреждениях. С 1975 года началось появление новых версий, разработанных за пределами Bell Labs, и рост популярности системы. В том же 1975 году Bell Labs выпустила шестую редакцию, известную благодаря широко разошедшимся комментариям Джона Лайонса.

К 1978 году система была установлена более чем на 600 машинах, прежде всего в университетах. Седьмая редакция была последней единой версией Unix. Именно в ней появился близкий к современному интерпретатор командной строки Bourne shell.

С 1978 года начинает свою историю BSD Unix, созданная в университете Беркли. Ее первая версия была основана на шестой редакции. В 1979 году выпущена новая версия, названная 3BSD, основанная на седьмой редакции. BSD поддерживала такие полезные свойства, как виртуальную память и замещение страниц по требованию. Автором BSD был Билл Джой.

В начале 1980-х годов компания AT&T, которой принадлежали Bell Labs, осознала ценность Unix и начала создание коммерческой версии Unix. Эта версия, поступившая в продажу в 1982 году, носила название Unix System III и была основана на седьмой версии системы.

Важной причиной раскола Unix стала реализация в 1980 году стека протоколов TCP/IP. До этого межмашинное взаимодействие в Unix пребывало в зачаточном состоянии – наиболее существенным способом связи был UUCP (средство копирования файлов из одной Unix-системы в другую, изначально работавшее по телефонным сетям с помощью модемов).

Было предложено два интерфейса программирования сетевых приложений: Berkley sockets (сокет Беркли) и интерфейс транспортного уровня TLI (англ. *Transport Layer Interface*). Интерфейс Berkley sockets был разработан в университете Беркли и использовал стек протоколов TCP/IP, разработанный там же. TLI был создан AT&T в соответствии с определением транспортного уровня модели OSI и впервые появился в системе System V версии 3. Хотя эта версия содержала TLI и потоки, первоначально в ней не было реализации TCP/IP или других сетевых протоколов, но подобные реализации предоставлялись сторонними фирмами. Реализация TCP/IP официально и окончательно была включена в базовую поставку System V версии 4. Это, как и другие соображения (по большей части, рыночные), вызвало окончательное размежевание между двумя ветвями Unix – BSD (университета Беркли) и System V (коммерческая версия от AT&T). Впоследствии многие компании, лицензировав System V у AT&T, разработали собственные коммерческие разновидности Unix, такие как AIX, CLIX, HP-UX, IRIX, Solaris.

В середине 1983 года была выпущена версия BSD 4.2, поддерживающая работу в сетях Ethernet и Arpanet. Система стала весьма популярной. Между 1983 и 1990 годом в BSD было добавлено много новых возможностей, таких как отладчик ядра, сетевая файловая система NFS, виртуальная файловая система VFS, и существенно улучшены возможности работы с файловыми сетями.

В это же время AT&T выпускала новые версии своей системы, названной System V. В 1983 году была выпущена версия 1 (SVR1 – System V Release 1), включавшая полноэкранный текстовый редактор vi, библиотеку curses, буферизацию ввода-вывода, кеширование inode. Версия 2 (SVR2), выпущенная в 1984 году, реализовывала монопольный доступ к файлам (file locking), доступ к страницам по требованию (demand paging), копирование при записи (copy-on-write). Версия 3 вышла в 1987 году и включала, среди прочего, TLI, а также систему поддержки удалённых файловых систем RFS. Версия 4 (SVR4), разработанная в сотрудничестве с фирмой Sun и вышедшая 18 октября 1988, поддерживала многие возможности BSD, в частности, TCP/IP, сокеты, новый командный интерпретатор csh. Кроме того, в нее были введены: символические ссылки, командный интерпретатор ksh, сетевая файловая система NFS (заимствованная у SunOS) и т. д.

Современные реализации Unix, как правило, не являются системами V или BSD в чистом виде. Они реализуют возможности как System V, так и BSD.

В начале 80-х годов прошлого века Ричард Столлман (Richard Matthew Stallman) решил создать клон популярной в то время в промышленных и академических сетях ОС Unix. Это было вызвано тем, что, по мнению Столлмана, Unix стал слишком коммерциализирован, а его исходный код стал закрытым. Он разработал концепцию Свободного программного обеспечения (Free Software), которая гласила, что пользователи должны всегда иметь возможность создавать, модифицировать и обмениваться программами без всяких ограничений. Этот принцип лег в основу так называемого Открытого лицензионного соглашения GNU (GNU GPL – GNU General Public License). Свою версию операционной системы Столлман назвал GNU.

В соответствии с идеей построения свободного программного обеспечения, GNU вырос в масштабный проект со многими участниками. Однако GNU не хватало ядра (kernel).

В начале девяностых годов молодой финский программист Линус Торвальдс (Linus Torvalds) заинтересовался одной из версий Unix, которая называлась ОС Minix. Это произошло благодаря прочитанной им книге создателя Minix Эндрю Таненбаума (Andrew Stuart Tanenbaum) «Операционные системы: разработка и реализация» [5]. Установив Minix на своем компьютере, Линус обнаружил ошибки и стал их исправлять, написал собственный эмулятор терминала, реализовывавший переключение задач. Добавляя все новые и новые функции, он, фактически, создал новую операционную систему. После опубликования ее исходного кода в 1991 система сразу же вызвала большой интерес. Сочетание имени Линус (Linus) и название ОС Unix дало наименование нового ядра, которое мы знаем и сейчас – Linux. Однако наиболее правильным наименованием все же является GNU/ Linux, в силу их совместимости и открытости.

В настоящее время существует множество различных версий Linux, построенных на основе единого ядра. Эти версии принято называть дистрибути-

вами (distributions, distros). Одними из самых известных дистрибутивов являются Mandriva, Xandros, Red Hat, SUSE. Но еще сотни версий уже существуют и появляются каждый день. Такое разнообразие возможно только благодаря идее свободного программного обеспечения. Созданием своих версий занимаются как энтузиасты одиночки, так и крупные компании, спонсирующие написание и поддержание новых дистрибутивов. Одним из наиболее распространенных дистрибутивов является Ubuntu, который сочетает в себе оба подхода. Его разработка спонсируется компанией Canonical Ltd., основанной в 2004 году Марком Шаттлвортом (Mark Shuttleworth), и поддерживается большим сообществом разработчиков и пользователей. Сам Ubuntu основан на дистрибутиве Debian (www.debian.org), созданном коллективными усилиями.

2. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ UNIX

Идеи, заложенные в основу Unix, оказали большое влияние на развитие компьютерных операционных систем (ОС). В настоящее время Unix-системы признаны одними из самых исторически важных ОС.

Как и Multics, Unix была написана на языке высокого уровня, а не на ассемблере (доминировавшем в то время). Она содержала значительно упрощённую, по сравнению с современными ей операционными системами, файловую модель. Файловая система включала как службы, так и устройства (такие как принтеры, терминалы и жёсткие диски) и предоставляла внешне единообразный интерфейс к ним, но дополнительные механизмы работы с устройствами (такие как IOCTL и биты доступа) не вписывались в простую модель «поток байтов».

Unix популяризовала предложенную в Multics идею иерархической файловой системы с произвольной глубиной вложенности. Другие операционные системы того времени позволяли разбивать дисковое пространство на каталоги или разделы, но число уровней вложенности было фиксировано и, зачастую, уровень вложенности был только один. Позднее все основные фирменные операционные системы обрели возможность создания рекурсивных подкаталогов, также заимствованных из Multics.

То что интерпретатор команд стал просто одной из пользовательских программ, а в качестве дополнительных команд выступают отдельные программы, является ещё одной инновацией Multics, популяризированной Unix. Язык командной оболочки Unix используется пользователем, как для интерактивной работы, так и для написания скриптов, то есть не существует отдельного языка описания заданий, как, например, в системе JCL фирмы IBM. Так как оболочка и команды операционной системы являются обычными программами, пользователь может выбирать их в соответствии со своими предпочтениями, или даже написать собственную оболочку. Наконец, новые команды можно добавлять к системе без перекомпиляции ядра. Новый, предложенный в командной строке Unix, способ создания цепочек программ, по-

следовательно обрабатывающих данные, способствовал использованию параллельной обработки данных.

Существенными особенностями Unix были полная ориентация на текстовый ввод-вывод и предположение, что размер машинного слова кратен восьми битам. Первоначально в Unix не было даже редакторов двоичных файлов – система полностью конфигурировалась с помощью текстовых команд. Наибольшей и наименьшей единицей ввода-вывода служил текстовый байт, что полностью отличало ввод-вывод Unix от ввода-вывода других операционных систем, ориентированных на работу с записями. Ориентация на использование текста для представления всего, что только можно, сделала полезными так называемые *конвейеры* (англ. *pipelines*). Ориентация на текстовый восьмибитный байт сделала Unix более масштабируемой и переносимой, чем другие операционные системы. Со временем текстовые приложения одержали победу и в других областях, например, на уровне сетевых протоколов, таких как Telnet, FTP, SMTP, HTTP и других.

Unix способствовала широкому распространению регулярных выражений, которые были впервые реализованы в текстовом редакторе ed для Unix. Возможности, предоставляемые Unix-программам, стали основой стандартных интерфейсов операционных систем (POSIX).

Широко используемый в системном программировании язык Си, созданный изначально для разработки Unix, превзошёл Unix по популярности. Язык Си был первым «веротерпимым» языком, который не пытался навязать программисту тот или иной стиль программирования. Он был первым высокоуровневым языком, предоставляющим доступ ко всем возможностям процессора, таким как ссылки, таблицы, битовые сдвиги, приращения и т. п. С другой стороны, свобода языка Си приводила к ошибкам переполнения буфера в таких функциях стандартной библиотеки Си, как **gets** и **scanf**. Результатом стали многие печально известные уязвимости, например, та, что эксплуатировалась в знаменитом черве Морриса.

Первые разработчики Unix способствовали внедрению принципов модульного программирования и повторного использования в инженерную практику.

Unix предоставляла возможность использования протоколов TCP/IP на сравнительно недорогих компьютерах, что привело к быстрому росту Интернета. Это, в свою очередь, способствовало быстрому обнаружению нескольких крупных уязвимостей в системе безопасности, архитектуре и системных утилитах.

Unix используется как в качестве сервера, так и рабочей станции. В номинации серверов с ней конкурируют MS WindowsNT, Novell Netware, DEC VMS и операционные системы мэйнфреймов. Каждая система имеет свою область применения, в которой она лучше других.

• WindowsNT – для администраторов, которые предпочитают удобный интерфейс экономному расходованию ресурсов и высокой производительности.

• Netware – для сетей, где нужна высокая производительность файлового и принтерного сервиса и не столь важны остальные сервисы. Главный недостаток – на сервере Netware трудно запускать приложения.

• VMS – мощный, ничем не уступающий Unix (а во многом и превосходящий ее) сервер приложений, но только для платформ VAX и Alpha фирмы DEC.

• Мэйнфреймы – для обслуживания очень большого количества пользователей (порядка нескольких тысяч). Но работа этих пользователей, как правило, организована в виде не клиент-серверного взаимодействия, а в виде хост-терминального. Терминал же в этой паре скорее не клиент, а сервер. К преимуществам мэйнфреймов необходимо отнести более высокую защищенность и устойчивость к сбоям, а к недостаткам – соответствующую этим качествам цену.

Unix хороша для квалифицированного (или желающего стать таковым) администратора, т.к. требует знания принципов функционирования происходящих в нем процессов. Реальная многозадачность и жесткое разделение памяти обеспечивают высокую надежность функционирования системы, хотя в производительности файл- и принт-сервисов Unix'ы уступают Netware.

Недостаточная гибкость предоставления прав доступа пользователей к файлам, по сравнению с Windows NT, затрудняет организацию на уровне файловой системы группового доступа к данным (точнее, к файлам), что компенсируется простотой реализации, а значит меньшими требованиями к аппаратуре. Впрочем, такие приложения, как SQL-сервер решают проблему группового доступа к данным своими силами, так что отсутствующая в Unix возможность запретить доступ к файлу конкретному пользователю является явно избыточной.

Практически все протоколы, на которых основан Internet, были разработаны под Unix, в частности стек протоколов TCP/IP придуман в университете Berkeley.

Защищенность Unix при правильном администрировании ни в чем не уступает ни Novell, ни Windows NT.

Важным свойством Unix, которое приближает ее к мэйнфреймам, является многотерминальность – много пользователей могут одновременно запускать программы на одной Unix-машине. Если не требуется использовать графику, можно обойтись дешевыми текстовыми терминалами (специализированными на базе дешевых PC), подключенными по медленным линиям. В этом с ним конкурирует только VMS. Можно использовать и графические X-терминалы, когда на одном экране присутствуют окна процессов, выполняющихся на разных машинах.

В номинации рабочих станций с Unix конкурируют MS Windows, Mac OS X (бывший Macintosh) и Acorn RISC-OS.

• Windows – для тех, кто ценит совместимость больше эффективности; для тех, кто готов купить большое количество памяти, дискового пространства и мегагерц; для тех, кто любит, не вникая в суть, щелкать мышкой по кнопочкам в окошке. Однако, рано или поздно все равно придется изучить

принципы работы системы и протоколов, но тогда уже будет поздно – выбор сделан.

• Apple Mac OS X – для графических, издательских и музыкальных работ, а также для тех, кто любит понятный, красивый интерфейс и не хочет (не может) разбираться в подробностях функционирования системы.

• RISC-OS, прошитая в ПЗУ, позволяет не тратить время на инсталляцию операционной системы и восстановление ее после сбоев. Кроме того, практически все программы под ней очень экономно расходуют ресурсы, благодаря чему не нуждаются в свопинге и работают очень быстро.

Unix функционирует как на PC, так и на мощных рабочих станциях с RISC-процессорами, под нее написаны действительно мощные САПР и геоинформационные системы. Своей масштабируемостью она из-за своей многоплатформенности на порядок превосходит любую другую операционную систему.

Операционная система Unix базируется на двух основных понятиях: «процесс» и «файл». Процессы являют собой динамическую сторону системы, это субъекты; а файлы – статическую, это объекты действия процессов. Почти весь интерфейс взаимодействия процессов с ядром и друг с другом выглядит как запись/чтение файлов. Хотя надо добавить такие вещи, как сигналы, разделяемая память и семафоры.

Процессы нельзя путать с программами – одна программа (как правило, с различными данными) может выполняться в разных процессах. Процессы можно, весьма условно, разделить на два типа – задачи и демоны. Задача – это процесс, который выполняет свою работу, стремясь побыстрее закончить ее и завершиться. Демон ждет событий, которые он должен обработать, обрабатывает произошедшие события и снова ждет; завершается он, как правило, по приказу другого процесса, чаще всего его убивает пользователь, дав команду «kill номер_процесса». В этом смысле получается, что интерактивная задача, обрабатывающая ввод пользователя, скорее похожа на демона, чем на задачу.

В старых Unix'ах отводилось 14 букв на имя, в новых это ограничение снято. В директории кроме имени файла находится его идентификатор **inode** – целое число, определяющее номер блока, в котором записаны атрибуты файла. Среди них: номер пользователя – хозяина файла; номер группы; количество ссылок на файл (см. далее), даты и время создания, последней модификации и последнего обращения к файлу; атрибуты доступа. Атрибуты доступа содержат тип файла, атрибуты смены прав при запуске и права доступа к нему для хозяина, одногруппника и остальных, на чтение, запись и выполнение. Право на стирание файла определяется правом записи в вышележащую директорию.

Каждый файл (но не директория) может быть известен под несколькими именами, но обязательно лежащими на одном разделе. Все ссылки на файл равноправны; файл стирается, когда удаляется последняя ссылка на файл. Если файл открыт (для чтения и/или записи), то число ссылок на него

увеличивается еще на единицу; так многие программы, открывающие временный файл, сразу удаляют его, чтобы при аварийном завершении, когда операционная система закрывает открытые процессом файлы, этот временный файл был удален операционной системой.

Есть еще одна интересная особенность файловой системы: если после создания файла запись в него шла не подряд, а с большими интервалами, то для этих интервалов место на диске не выделяется. Таким образом, суммарный объем файлов в разделе может быть больше объема раздела, а при удалении такого файла освобождается меньше места, чем его размер.

Файлы бывают следующих типов:

- обычный файл прямого доступа;
- директория (файл, содержащий имена и идентификаторы других файлов);
- символьный линк (строка с именем другого файла);
- блочное устройство (диск или магнитная лента);
- последовательное устройство (терминалы, последовательные и параллельные порты; диски и магнитные ленты тоже имеют интерфейс последовательного устройства);
- поименованный канал.

Специальные файлы, предназначенные для работы с устройствами, как правило, сосредоточены в директории «**/dev**». Вот некоторые из них (в номинации FreeBSD):

- tty* – терминалы, в т.ч.:
  - ttyv<цифра> – виртуальная консоль;
  - ttyd<цифра> – DialIn терминал (обычно последовательный порт);
  - cuaa<цифра> – DialOut линия;
  - ttyp<цифра> – сетевой псевдо-терминал;
  - tty – терминал, с которым ассоциирована задача;
- wd* – жесткие диски и их подразделы, в т.ч.:
  - wd<цифра> – жесткий диск;
  - wd<цифра>s<цифра> – партиция этого диска (именуемая здесь «slice»);
  - wd<цифра>s<цифра><буква> – раздел партиции;
- fd<цифра>[<буква>] – floppy-диск;
- rwd*, rfd* – то же, что wd* и fd*, но с последовательным доступом.

Иногда требуется, чтобы программа, запущенная пользователем, имела не права запустившего ее пользователя, а какие-то другие. В этом случае устанавливается атрибут смены прав на права пользователя – хозяина программы (в качестве примера можно привести программу, которая читает файл с вопросами и ответами и на основании прочитанного тестирует запустившего эту программу студента. Программа должна иметь право читать файл с ответами, а запустивший ее студент – нет). Так, например, работает программа passwd, с помощью которой юзер может изменить свой пароль. Юзер может запустить программу passwd, она может произвести изменения в системной базе данных – а пользователь не может.

В отличие от DOS, в которой полное имя файла выглядит как «диск:\путь\имя», и RISC-OS, в которой оно выглядит «-файловая_система-диск:$.путь.имя» (что имеет свои преимущества), Unix использует прозрачную нотацию в виде «/путь/имя». Корень отсчитывается от раздела, с которого было загружено ядро Unix. Если мы собираемся использовать другой раздел (а на загрузочном разделе, как правило, находится только самое необходимое для загрузки), используется команда «mount /dev/файл_раздела директория». При этом файлы и поддиректории, ранее находившиеся в этой директории, становятся недоступными, пока раздел не будет размонтирован (естественно, все используют для монтирования разделов пустые директории). Производить монтирование и размонтирование имеет право только супервизор.

При запуске каждый процесс может рассчитывать, что для него уже открыты три файла, которые ему известны как стандартный ввод stdin по дескриптору 0; стандартный вывод stdout по дескриптору 1; и стандартный вывод stderr по дескриптору 2. При регистрации в системе, когда пользователь вводит имя и пароль, а ему запускается shell, все три направлены на /dev/tty; позже любой из них может быть перенаправлен в любой файл.

3. ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ СИСТЕМЫ UNIX

На протяжении большей части истории Интернета, системы Unix обеспечивали наивысший уровень функционирования служб в сети. Когда хакеры стали серьезной проблемой для всемирной сети, системам Unix начали уделять все больше внимания. На сегодняшний день большая часть систем в интернете работает под управлением ОС Unix, и для надежной защиты от хакеров эти системы должны быть правильно настроены.

В данной работе приводятся некоторые базовые положения безопасности, связанные с построением и защитой системы Unix. Ввиду большого числа доступных на рынке Unix-систем, точные местоположения файлов и команды не являются абсолютно правильными для всех версий Unix. Где это представляется возможным, авторы указывают корректировочную информацию для систем Sun Solaris и Linux.

После построения системы Unix в ней, как правило, присутствует ряд уязвимостей. Большую их часть можно устранить посредством обновления системы или внесения изменений в конфигурационные файлы. В следующих разделах выделяются наиболее распространенные проблемы безопасности и способы их устранения.

3.1. Файлы загрузки

Системы Unix настраиваются при загрузке с использованием соответствующих загрузочных файлов. В зависимости от версии Unix, файлы загрузки могут располагаться в различных местах. В системе Solaris файлы загрузки находятся в каталоге /etc/rc2.d, в системе Linux – в каталоге /etc/rc.d/rc2.d. В различных версиях Unix файлы могут располагаться в различных местах, это расположение действительно для Red Hat.

В файлах загрузки запускается ряд служб. Некоторые из них (сеть, монтировка файловых систем и журнал запуска) необходимы для функционирования системы и ничто не должно препятствовать их работе. Другие службы не являются столь критичными и запускаются в зависимости от того, каким образом используется система. Чтобы предотвратить запуск службы, просто измените имя файла. Убедитесь, что новое имя файла не начинается с буквы S или K. Рекомендуется размещать в качестве первого символа точку (<.>) в имени файла (это скрывает файл от просмотра, поэтому его нельзя будет перепутать с функционирующим файлом). Если служба не понадобится в будущем, файл можно удалить.

Службы, обычно запускаемые при помощи файлов загрузки, включают в себя следующие сервисы:

- Inetd;
- NFS;
- NTP;
- Routed;
- RPC;
- Sendmail;
- Web servers.

Необходимо обязательно просмотреть файлы загрузки и определить не запускаются ли необязательные службы (в следующем разделе рассказывается о том как выявлять необязательные службы).

3.2. Службы, работу которых следует разрешить

Набор служб, выбранных для систем Unix, зависит от того, каким образом они будут использоваться. Некоторые из этих служб будут запускаться с помощью файлов загрузки; ряд служб контролируется через сервис inetd и настраивается в файле /etc/inetd.conf. Приведенный ниже текст представляет собой часть файла inetd.conf системы Solaris. Строки, начинающиеся с символа решетки <#> – комментарии.

```
#ident "@(#)inetd.conf 1.27 96/09/24 SMI"
/*SVr4.0 1.5 */
# Ftp and telnet are standard Internet services.
ftp stream tcp nowait root
/usr/sbin/in.ftpd in.ftpd
#telnet stream tcp nowait root /usr/sbin/in.telnetd
```

```
in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#shell stream tcp nowait root
/usr/sbin/in.rshd in.rshd
#login stream tcp nowait root /usr/sbin/in.rlogind
in.rlogind
#exec stream tcp nowait root
/usr/sbin/in.rexecd in.rexecd
#comsat dgram udp wait root
/usr/sbin/in.comsat in.comsat
#talk dgram udp wait root
/usr/sbin/in.talkd in.talkd
#
# Solstice system and network administration class agent server
#100232/10 tli rpc/udp wait root /usr/sbin/sadmind sadmind
```

Файл inetd.conf не только контролирует службы типа FTP и telnet, но и некоторые службы RPC. Файл inetd.conf необходимо внимательно проверять сконфигурированы ли в нем только необходимые службы. После правильной настройки файла необходимо перезапустить службу inetd посредством следующей команды:

```
#kill -HUP <номер процесса inetd>
```

Команда -HUP вызывает повторное считывание службой inetd ее конфигурационного файла.

Многие службы, настраиваемые по умолчанию на системах Unix, необходимо отключить. Ниже приведен перечень этих служб.

```
Chargen rexd    Systat
Discard Routed  Tftp
Echo    Rquotad Uucp
Finger  Rusersd Walld
netstat sprayd
```

Кроме того, можно отключить службы Daytime, Time и SNMPD, если они не используются. Служба Time может использоваться некоторыми системами синхронизации, а служба SNMPD – для управления системой.

Как видно из приведенного выше фрагмента содержимого файла inetd.conf, службы telnet и FTP, как правило, настроены на рабочее состояние. Эти два протокола позволяют передавать идентификаторы пользователей и пароли через сеть в открытом виде. Возможно использование шифрующих версий этих протоколов для защиты паролей. Во время использования telnet рекомендуется применять Secure Shell (SSH). Некоторые версии SSH входят в программу Secure Copy (SCP) для передачи файлов.

Внутри организации может потребоваться использование файловой системы Network File System (NFS). Если это не так, отключите NFS на любой системе, на которой не требуется ее использование. NFS предназначена для монтирования файловой системы с одной системы на другую. Если NFS настроена неправильно, то велика вероятность того, что кто-то получит доступ к секретным файлам. Чтобы правильно настроить NFS, следует соответствующим образом изменить файл /etc/dfs/dfstab.

Системы Unix, используемые в DMZ как веб-серверы, почтовые серверы или серверы DNS, должны настраиваться еще более тщательно, с точки зрения безопасности, чем системы, используемые исключительно внутри сети. Такие системы, как правило, не требуют использования служб RPC и NFS. Их можно удалить посредством внесения изменений в файлы загрузки.

4. СЕРВЕРЫ И РАБОЧИЕ СТАНЦИИ

В некоторых организациях операционная система Unix используется как на серверах, так и на рабочих станциях. При использовании на рабочей станции система обычно настраивается на функционирование системы X Window System. На системах Solaris в этом случае используется программа ToolTalk (RPC-программа, предназначенная для связи между приложениями)[1].

Эти службы не нужны на серверах, а службы DNS и routed не требуются на рабочих станциях. Необходимо разработать руководство по настройке серверов и руководство для настройки рабочих станций, если система Unix используется описанным выше образом.

4.1. Использование программ TCP Wrappers

Программы TCP Wrappers используются для обеспечения дополнительного уровня защиты (доступны по адресу ftp://ftp.porcupine.org/ pub/security) в случае применения служб telnet или FTP. Как видно из названия, программы TCP Wrappers (wrap – оболочка) создают «оболочку» для служб telnet и FTP с целью обеспечения дополнительного контроля доступа и ведения журналов. Для использования программы TCP Wrappers необходимо настроить файл inetd.conf так, чтобы строки telnet и FTP выглядели следующим образом:

```
ftp stream tcp nowait root /usr/local/bin/tcpd /usr/sbin/in.ftpd
telnet stream tcp nowait root /usr/local/bin/tcpd /usr/sbin/in.telnetd
```

---

[1] Программа ToolTalk контролируется посредством inetd.conf на системах Solaris. Чтобы отключить эту программу, необходимо закомментировать следующую строку:

```
100083/1 tli rpc/tcp wait root
 /usr/dt/bin/rpc.ttdbserverd/usr/dt/bin/rpc.ttdbserverd
```

Эти строки вызывают запуск TCP Wrappers (tcpd)[2] службой inetd, когда кто-либо пытается установить с системой сеанс связи через telnet или FTP.

TCP Wrappers можно настроить на блокировку или разрешение определенным узлам или сетям доступа к службам telnet и FTP. Файлы, используемые для этих действий по настройке, – это файлы /etc/hosts.allow и /etc/hosts.deny. Синтаксис для работы с этими файлами выглядит следующим образом:

```
<имя программы-оболочки>: <ip-адрес>/<маска сети>
```

Следующие файлы представляют собой примеры файлов конфигурации TCP Wrapper.

```
hosts.allow:
#Allow telnets from my internal network (10.1.1.x)
in.telnet: 10.1.1.0/255.255.255.0
#Allow ftp from the world
in.ftpd: 0.0.0.0/0.0.0.0
hosts.deny:
#Deny telnets from anywhere else
in.telnetd: 0.0.0.0/0.0.0.0
```

Файл hosts.allow оценивается в первую очередь, после чего обрабатывается файл hosts.deny. Следовательно, можно сначала настроить все системы, которым разрешено работать с различными службами, после чего запретить все остальное в файле hosts.deny. Кроме того, следует внести изменение в настройку журнала, чтобы разрешить TCP Wrappers заносить данные в журнал системы. Это изменение далее описано в разделе «Файлы журнала».

Существует ряд изменений, которые можно внести в файлы конфигурации системы Unix, чтобы увеличить общий уровень безопасности системы[3]. Это могут быть как предупреждающие сообщения, так и защита от переполнения буфера на некоторых системах. Любые изменения должны вноситься в конфигурацию в соответствии с политикой безопасности организации.

Приветственные сообщения могут использоваться для заявления о правах собственности перед входом пользователя в систему. Сообщение должно быть написано на языке, разрешенном для использования юридическим отделом организации.

Приветственное сообщение хранится в /etc/motd (сокр. от «message of the day» – сообщение дня). Однако это сообщение отображается не перед входом пользователя в систему, а после него. Большинство уведомлений,

---

[2]TCP Wrappers можно использовать и для других служб, таких как POP и IMAP. Нужно просто внести соответствующие изменения в строки конфигурации, представленные выше

[3]В различных версиях систем Unix файлы конфигурации располагаются в различных местах. Обратитесь к руководствам или инструкциям конкретной используемой версии Unix, чтобы удостовериться в корректности вносимых изменений в отношении рассматриваемой версии системы

связанных с юридическими вопросами, необходимо отображать перед входом пользователя в систему.

Чтобы сообщение отображалось перед входом пользователя в систему, используйте следующий способ. В ОС Solaris предварительное уведомление хранится в каталоге /etc/default/telnetd. Можно создать сообщения входа для FTP посредством редактирования файла /etc/default/ftpd. Для создания сообщения добавьте в файл строку, аналогичную следующей:

```
BANNER="\n\n<Enter Your Legal Message Here\n\n"
```

Параметр \n означает новую строку. Поэкспериментируйте с символами новой строки, чтобы сообщение приняло нужный вам вид.

В системах Linux для сообщений telnet используются два файла: /etc/issue и /etc/issue.net. Файл issue применяется для терминалов, подключенных напрямую, а issue.net используется в том случае, когда кто-либо устанавливает по сети соединение через telnet с рассматриваемой системой. К сожалению, только на изменении этих файлов создание сообщения не закончится, так как они создаются заново при каждой загрузке системы. Однако можно изменить сценарий загрузки, создающий эти файлы.

Файлы создаются в сценарии загрузки /etc/rc.d/rc.local. Чтобы предотвратить автоматическое создание /etc/issue и /etc/issue.net, закомментируйте следующие строки /etc/rc.d/rc.local:

```
# This will overwrite /etc/issue at every boot. So, make any changes you
# want to make to /etc/issue here or you will lose then when you reboot.
echo "" > /etc/issue
echo "$R" > /etc/issue
echo "Kernel $(uname -r) on $a $SMP$(uname -m)" >> /etc/issue
```

После этого можно изменить /etc/issue и /etc/issue.net, введя в них соответствующий текст с заявлением о правах.

4.2. Настройки паролей

Существует три этапа процедуры управления паролями в системе Unix:

- настройка требований к паролям;
- запрет на вход без пароля;
- указание требований к содержимому паролей.

Настройка требований к паролю. В системах Unix требования к возрасту паролей и их длине устанавливаются посредством изменения файла конфигурации. В системе Solaris этим файлом является /etc/default/passwd. Файл содержит приведенные ниже строки, которые следует редактировать для соответствия политике безопасности организации.

```
#ident  "@(#)passwd.dfl       1.3     92/07/14 SMI"
MAXWEEKS=7
MINWEEKS=1
PASSLENGTH=8
```

Где системный администратор может узнать о том, как следует настроить систему? Определение требований к конфигурации систем всегда начинается с политики безопасности организации. В каждой организации должны быть разработаны процедуры конфигурации, специфичные для конкретной используемой системы; при этом необходимо руководствоваться политикой безопасности. Эти процедуры должны определять, каким образом следует настраивать систему с использованием конкретной операционной системы, чтобы обеспечить соответствие ОС требованиям политики безопасности.

В системах Linux требования к паролям находятся в файле /etc/login.defs. Следующие строки файла /etc/login.defs представляют собой настраиваемые параметры:

```
# Password aging controls:
#
# PASS_MAX_DAYS Maximum number of days a password may be used.
# PASS_MIN_DAYS Minimum number of days allowed between password changes.
# PASS_MIN_LEN Minimum acceptable password length.
# PASS_WARN_AGE Number of days warning given before a password expires.
#
PASS_MAX_DAYS 45
PASS_MIN_DAYS 1
PASS_MIN_LEN 8
PASS_WARN_AGE 7
```

Linux также позволяет предупреждать пользователей о том, что до окончания срока действия пароля осталось несколько дней.

Запрет на вход без пароля. Программы rlogin, rsh и rexec позволяют пользователям осуществлять вход в систему с определенных систем без указания пароля вручную. Этого делать не рекомендуется, так как злоумышленник, проникший в одну из систем, может таким образом получить доступ к остальным компьютерам. Помимо удаления служб rlogin, rsh и rexec из /etc/inetd.conf следует удостовериться в том, что файл /etc/host.equiv и любые файлы .rhost, имеющиеся в системе, найдены и удалены. Не забудьте также проверить домашние каталоги всех пользователей.

Указание требований к содержимому паролей. Запрет пользователям использования ненадежных паролей является одним из лучших способов повышения уровня безопасности системы. Программы типа passwd+ и npasswd имеются для Linux, но не для Solaris. Обе эти программы позволяют указывать требования к надежности паролей и вынуждают пользователей выбирать пароли, соответствующие установленным правилам.

С выходом Solaris 2.6 и более поздних реализаций Linux появилось более совершенное средство отслеживания надежности паролей пользователей – это Pluggable Authentication Modules (PAM). Более подробная информация о PAM и о том как создать фильтры паролей находится по адресу

http://www.sun.com/solaris/pam/; для системы Linux – по адресу ftp://ftp.kernel.org/pub/linux/libs/ pam/index.html.

4.3. Контроль доступа к файлам

В системе Unix доступ к файлам контролируется посредством набора разрешений. Для владельца файла, группы, которой принадлежит файл, и для всех остальных лиц можно присваивать привилегии чтения, записи и выполнения. Файловые разрешения изменяются посредством команды chmod. Как правило, не рекомендуется разрешать пользователям создавать файлы, доступные для чтения или записи любыми лицами. Такие файлы могут считываться или записываться любым пользователем системы. Если злоумышленник получит доступ к идентификатору пользователя, он сможет считать или изменить любые из таких файлов.

Так как достаточно трудно убедить всех пользователей в необходимости изменять разрешения доступа к файлу при его создании, разумно создать механизм, используемый по умолчанию, предназначенный для настройки соответствующих разрешений при автоматическом создании файла. Это можно осуществить с помощью параметра unmask. В системах Solaris этот параметр располагается в файле /etc/default/login, в системах Linux – в /etc/profile. Команда выполняется следующим образом: unmask 077

Цифры, указываемые после команды, определяют разрешения, которые не будут присвоены по умолчанию вновь создаваемому файлу. Первая цифра определяет разрешения относительно владельца файла, вторая цифра указывает разрешения для группы, а третья – для всех остальных пользователей. В случае, рассмотренном выше, все новые файлы присваивают разрешения чтения, записи и выполнения владельцу того или иного файла, а группе и всем остальным пользователям не предоставляется никаких разрешений.

Разрешения определяются числами следующим образом:

- 4 – разрешение на чтение;
- 2 – разрешение на запись;
- 1 – разрешение на выполнение.

Следовательно, если требуется разрешить группе иметь по умолчанию разрешение на чтение, но запретить запись и выполнение, нужно указать команду unmask 037. Если требуется запретить группе запись, следует указать команду unmask 027.

Доступ через корневую учетную запись. Как правило, рекомендуется ограничивать прямой доступ с использованием корневой учетной записи. При таком подходе даже администраторам необходимо сначала выполнить вход в систему с использованием их аутентификационных данных и только после этого с помощью команды su получить доступ к корневой учетной записи. Это также обеспечивает создание записей в журнале, отображающих идентификаторы пользователей используемых для получения доступа к корневой

учетной записи. В качестве альтернативы вместо команды su можно использовать команду sudo. Команда sudo обеспечивает дополнительные возможности по ведению журналов, заключающиеся в фиксировании команд, выполняемых пользователями, работающими в корневой учетной записи.

Существует возможность ограничить вход под корневой учетной записью таким образом, чтобы его можно было осуществлять только из консоли Solaris или Linux. В системе Solaris следует изменить файл /etc/default/login и убедиться в том, что следующая строка не закомментирована

```
# If CONSOLE is set, root can only login on that device.
# Comment this line out to allow remote login by root.
#
CONSOLE=/dev/console
```

Посредством этого система разрешит прямой вход в корневую учетную запись только через консоль. В системе Linux можно реализовать аналогичную конфигурацию, редактируя файл /etc/securetty. Этот файл представляет собой список TTY, который используется для входа в корневую учетную запись. Содержимым этого файла должно быть /dev/tty1. Если для управления системой используется последовательный канал связи, файл должен содержать /dev/ttyS0. Сетевые TTY – это, как правило, /dev/ttyp1 и выше.

Если требуется контролировать корневой доступ к системе, рекомендуется осуществлять контроль корневого доступа к FTP. Файл /etc/ftpusers и в системах Solaris, и в системах Linux представляет перечень учетных записей, которым не разрешено осуществлять доступ к системе через FTP. Убедитесь, что в данном списке присутствует корневая учетная запись.

4.4. Защита от переполнения буфера

Переполнение буфера – одна из наиболее серьезных опасностей, угрожающих системе. Solaris предоставляет способ предотвращения выполнения команд вне стека при проявлении атак на переполнение буфера. Для этого необходимо добавить следующие строки в файл /etc/system

```
set noexec_user_stack=1
set noexec_user_stack_log=1
```

Первая строка предотвращает выполнение команд вне стека, а вторая – заносит в журнал данные о произведенных попытках.
Существует несколько других проектов, предназначенных для повышения уровня защиты стека Linux. Один из них расположен по адресу http://www.openwall.com/linux/.

Отключение неиспользуемых учетных записей. В Unix создается набор учетных записей, необходимых для различных целей (например, владение некоторыми определенными файлами), которые никогда не используются для входа в систему. Такими учетными записями являются sys, uucp, nuucp и

listen. Для каждой учетной записи следует изменить их записи в файле /etc/shadow, чтобы предотвратить успешный вход в систему с их помощью

```
root:XUbBEEYtgskmk:10960:0:99999:7:::
bin:*LK*:10960:0:99999:7:::
daemon:*LK*:10960:0:99999:7:::
adm:*LK*:10960:0:99999:7:::
lp:*LK*:10960:0:99999:7:::
sync:*LK*:10960:0:99999:7:::
shutdown:*LK*:10960:0:99999:7:::
halt:*LK*:10960:0:99999:7:::
mail:*LK*:10960:0:99999:7:::
news:*LK*:10960:0:99999:7:::
uucp:*LK*:10960:0:99999:7:::
operator:*LK*:10960:0:99999:7:::
games:*LK*:10960:0:99999:7:::
gopher:*LK*:10960:0:99999:7:::
ftp:*LK*:10960:0:99999:7:::
nobody:*LK*:10960:0:99999:7:::
```

Второе поле в каждой строке представляет собой поле пароля. В случае с обычными пользовательскими учетными записями здесь располагается зашифрованный пароль. Для учетных записей, вход посредством которых запрещен, второе поле должно содержать какие-либо данные с символом "*". Символ "*" не соответствует ни одному реальному паролю и, таким образом, не может быть угадан или взломан. Посредством размещения в поле пароля соответствующих символов, таких как "LK", можно явным образом сообщать о том, что данная учетная запись заблокирована.

Обновления. Для исправления ошибок и устранения уязвимостей для Unix выпускаются обновления и «заплатки» аналогичные операционным системам семейства Windows. Обновления должны устанавливаться регулярно, чтобы минимизировать число уязвимостей. Различные поставщики систем Unix выпускают средства, помогающие в управлении обновлениями. Компания Sun предлагает программу Solaris Sunsolve Patch Manager[4], а Red Hat имеет онлайн-систему обновления в интернете:

http://www.redhat.com/apps/support/errata/.

Управление пользователями. Как и с любой операционной системой, управление сообществом пользователей является очень важным процессом для поддержки общей безопасности системы. В организации должна при-

---

[4]При загрузке обновлений для систем Solaris учитывайте, что Sun размещает многие обновления в кластере обновлений. Однако кластер обновлений может не содержать некоторых обновлений безопасности. Может понадобиться загрузить их в отдельном порядке и установить вручную

сутствовать специальная процедура управления пользователями, предусматривающая в деталях все действия, которые необходимо выполнить, чтобы предоставить сотруднику доступ к системе. В процедуре должны быть определены шаги, которые следует предпринять при увольнении сотрудника из компании.

Следующие разделы данной лекции содержат некоторые подробные рекомендации по управлению пользователями в системах Unix. Учитывайте, что существует множество вариаций систем Unix. Средства, используемые для управления пользователями, различны для каждого поставщика и версии операционной системы.

Добавление пользователей в систему. В большей части версий Unix имеются утилиты для добавления пользователей в систему. Здесь ключевыми задачами являются следующие[5]:

- добавление имени пользователя в файл паролей;
- присвоение соответствующего идентификатора пользователя;
- присвоение соответствующего группового идентификатора;
- определение соответствующей оболочки для входа в систему (некоторые пользователи могут вообще не иметь какой-либо оболочки);
- добавление имени пользователя в теневой файл;
- указание соответствующего начального пароля;
- определение соответствующего псевдонима электронной почты;
- создание домашнего каталога пользователя.

Добавление имени пользователя в файл паролей. Файл /etc/passwd содержит перечень всех имен пользователей, принадлежащих пользователям системы. Каждый пользователь должен иметь уникальное имя, состоящее из восьми или менее символов. Для каждой записи в файле паролей должно быть определено реальное лицо, ответственное за учетную запись. Данную информацию можно добавить в поле GECOS (пятое поле в каждой строке).

Каждому имени пользователя необходимо присвоить соответствующий идентификатор пользователя (UID). UID должен быть уникальным в рамках всей системы[6]. Как правило, идентификатор пользователя должны быть больше 100. Он ни в коем случае не должен быть равен 0, так как это идентификатор корневой учетной записи.

Каждый пользователь должен иметь главную группу. Присвойте этот номер имени пользователя в файле /etc/passwd. Обычные пользователи не должны быть членами группы «wheel», так как она используется в административных целях.

---

[5]Большая часть систем содержит утилиты по добавлению пользователей для обеспечения автоматического выполнения этой задачи. В Linux используется программа adduser. В системе Solaris эта утилита называется useradd

[6]Система использует UID для идентификации владельцев файлов в системе и, таким образом, не рекомендуется даже повторное использование UID

Определение соответствующей оболочки для входа в систему. Интерактивным пользователям необходимо предоставить оболочку для входа в систему. Как правило, это оболочки ksh, csh или bash. Пользователям, которые не будут осуществлять вход в систему, нужно предоставить программу, не являющуюся оболочкой. Например, если имеются пользователи, которые только проверяют электронную почту через POP или IMAP, им можно разрешить изменять свои пароли в интерактивном режиме. В данном случае существует возможность определить оболочку, указав в качестве нее /bin/passwd. При каждом подключении пользователей к системе через telnet им будет предоставляться возможность изменить пароль. По завершении этой операции пользователь будет выходить из системы.

Пароли не должны храниться в файле /etc/passwd, так как этот файл доступен для чтения всем пользователям, и с его помощью злоумышленник может осуществить взлом пароля. Пароли должны храниться в файле /etc/shadow. Следовательно, имя пользователя должно быть добавлено и в файл /etc/shadow.

После создания учетной записи следует установить начальный пароль. Большая часть утилит, используемая для добавления пользователей в системы, предлагает сделать это автоматически. В противном случае нужно войти в систему с правами пользователя и выполнить команду passwd. После этого появится предложение указать пароль для учетной записи. Начальные пароли должны быть сложными для угадывания и рекомендуется не использовать один и тот же для всех учетных записей. Если используется один и тот же начальный пароль, атакующий может использовать новые учетные записи, прежде чем у легального пользователя появится возможность войти в систему и изменить пароль.

При создании пользователя он автоматически получает адрес электронной почты «имя_пользователя@host». Если пользователь хочет иметь другой адрес электронной почты, такой как «имя.фамилия@host», то этот адрес можно присвоить посредством псевдонима электронной почты. Чтобы добавить псевдоним, измените файл /etc/aliases. Формат этого файла:

Alias: username

После создания псевдонима необходимо запустить программу newaliases, чтобы создать файл alias.db.

Каждый пользователь должен иметь свой собственный домашний каталог. Этот каталог определяется в файле /etc/passwd. После создания каталога в соответствующем месте в системе (как правило, это каталог /home или /export), владельцем каталога назначается пользователь командой chown следующим образом: chown <username> <directory name>

Когда сотрудник увольняется из компании или переводится на другую работу, и его учетная запись становится ненужной, необходимо выполнить соответствующую процедуру по управлению пользователями. В системе Unix все файлы пользователей принадлежат UID пользователя. Следователь-

но, если пользовательский UID повторно используется для новой учетной записи, последняя будет предусматривать владение всеми файлами старого пользователя.

Изначально, если пользователю больше не требуется учетная запись, ее следует заблокировать. Это можно сделать посредством замены пароля пользователя в файле /etc/shadow символами <*LK*>. По прошествии определенного числа дней (как правило 30 дней), файлы пользователя могут быть удалены.

Управление системой Unix (относительно вопросов безопасности) заключается в ведении журнала и отслеживании системы на наличие признаков подозрительной активности. Системы Unix предоставляют достаточное количество информации о том, что происходит в системе, а также набор средств, которые могут использоваться для выявления подозрительной активности.

В большинстве случаев ведение системных журналов является стандартной процедурой, выполняемой в большинстве версий Unix, и в них заносится достаточный объем данных, связанных с безопасностью системы. В некоторых ситуациях требуется проведение дополнительного аудита. В Solaris для этого предусмотрен модуль Basic Security Module (BSM), который не включен в Solaris по умолчанию. Необходимость в дополнительных возможностях здесь определяется пользователем.

Чтобы включить BSM, выполните сценарий /etc/security/bsmconv. При этом запустится фоновая программа аудита, но перезагрузка системы не потребуется. Файл /etc/security/audit_control используется для определения конфигурации аудита. Полная информация по этому файлу находится в инструкции к ОС (man audit_control), однако для начала рекомендуется использовать следующую конфигурацию:

```
#identify the location of the audit file directory
dir: <directory>
#identify the file system free space percentage when a warning should occur
minfree: 20
#flags for what to audit. This example audits login, administrative
#functions and failed file reads, writes, and attribute changes
flags: lo,ad,-fm
#This set of flags tells the system to also audit login and administrative
#events that cannot be attributed to a user
naflags: lo,ad
```

Как только файл будет настроен, начнут создаваться записи аудита. Для закрытия текущего файла записи аудита и открытия нового файла используется команда audit -n. Команда praudit <имя файла аудита> предназначена для просмотра содержимого файла аудита.

5. ФАЙЛЫ ЖУРНАЛОВ

Большая часть систем Unix обеспечивает широкие возможности по ведению журналов в программе syslog. Syslog – это фоновая программа, выполняющая и фиксирующая данные журнала согласно настройке. Syslog настраивается через файл /etc/syslog.conf. Следует заметить, файлы журналов должны просматриваться только корневым пользователем, и никто не должен иметь возможность их изменять.

Большая часть файлов syslog.conf направляет сообщения журналов в /var/log/messages или /var/adm/log/messages. Правильно написанный syslog.conf должен содержать следующую команду конфигурации:

```
auth.info /var/log/auth.log
```

С помощью этой команды Unix собирает информацию о попытках входа, выполнения команды su, перезагрузке системы и других событиях, так или иначе связанных с безопасностью системы. Данная команда также позволяет программам TCP Wrappers заносить информацию в файл auth.log. Обязательно создайте файл /var/log/auth.log для фиксирования этой информации

```
#touch  /var/log/auth.log
#chown  root /var/log/auth.log
#chmod  600 /var/log/auth.log
```

В Solaris, при создании файла /var/adm/loginlog, можно фиксировать неудачные попытки входа в систему. Создайте файл следующим образом:

```
#touch   /var/adm/loginlog
#chmod  600 /var/adm/loginlog
#chown  root /var/adm/loginlog
#chgrp  sys /var/adm/loginlog
```

Убедитесь, что /var предоставлено достаточное количество свободного пространства для ведения файлов журнала. Если /var расположен в том же разделе, что и /, корневая файловая система переполнится при сильном увеличении файлов журнала. В этом случае рекомендуется размещать каталог /var в другой файловой системе.

5.1. Скрытые файлы

Скрытые файлы представляют собой потенциальную проблему для систем Unix. Любой файл, начинающийся с точки (<.>), не отображается при выполнении стандартной команды ls. Однако при использовании команды ls -a отобразятся все скрытые файлы. Хакеры научились использовать скрытые файлы для маскировки своих действий. Они могут расположить свои файлы в скрытом каталоге или в каталогах, которые трудно обнаружить администратору. Например, если назвать каталог <...>, то он может остаться незамеченным. Добавление пробела после третьей точки (<... >) делает каталог

труднодоступным, если не знать о наличии пробела. Чтобы отобразить все скрытые файлы и каталоги, имеющиеся в системе, выполните следующую команду: #find / -name '.*' -ls

Использование -ls вместо -print позволяет вывести более подробный список расположения файла. Следует периодически выполнять эту команду и проверять любые новые скрытые файлы.

Файлы, для которых разрешены полномочия Set UID (SUID) или Set Group ID (SGID), могут изменять идентификатор своего активного пользователя или группы в процессе выполнения. Некоторым файлам требуется такая возможность для выполнения своей работы, однако это должен быть ограниченный набор файлов, и ни один из них не должен находиться в домашних каталогах пользователей. Чтобы найти все файлы SUID и SGID, выполните следующие команды:

```
#find   /       -type f -perm -04000 -ls
#find   /       -type f -perm -02000 -ls
```

При построении системы необходимо выполнить данные команды и сохранить результаты их выполнения. Периодически следует выполнять эти команды и сопоставлять результаты с исходным списком. Любые обнаруженные изменения необходимо исследовать.

Файлы, общедоступные для записи, являются еще одной потенциальной ошибкой в конфигурации системы Unix. Такие файлы позволяют злоумышленнику создать сценарий, который при выполнении будет использовать их уязвимость. Если файлы SUID и SGID доступны для записи всем пользователям, у атакующего появляется возможность создать для самого себя самые обширные привилегии. Чтобы выявить все файлы общедоступные для записи выполните следующую команду:

```
#find / -perm -2 -type f -ls
```

Следует периодически выполнять эту команду, чтобы находить все общедоступные для записи файлы, имеющиеся в системе.

Уже описаны некоторые признаки, которые необходимо отслеживать в системе и которые могут означать проявление угрозы или проникновение в систему (скрытые файлы, файлы SUID и SGID, а также общедоступные для записи файлы). Существует несколько других способов проверки системы Unix на наличие подозрительной активности.

5.2. Смешанный режим

Интерфейс находится в смешанном режиме[7], когда в системе работает сниффер (сетевой анализатор пакетов). Сниффер переводит интерфейс в

---

[7]Solaris не выдает соответствующего отчета о том, что интерфейс находится в смешанном режиме. Причиной этого является ошибка в программном обеспечении ядра. Чтобы корректным образом проверить, находится ли интерфейс Solaris в смешанном режиме,

смешанный режим; при этом происходит фиксирование всей информации, проходящей через канал связи. Если во время работы интерфейса в данном режиме выполнить команду ifconfig -a, то появится сообщение о том, что интерфейс находится в состоянии PROMISC (признак того, что работает анализатор пакетов). Если сниффер запущен не администратором системы, необходимо провести исследование причин этих обстоятельств.

Программа netstat используется для выяснения, какие сетевые соединения находятся в активном состоянии в системе Unix. Команду следует использовать следующим образом: netstat -an. Аргумент "n" сообщает netstat, что обработка IP-адресов не требуется.

```
#netstat -an
Active Internet connections (servers and established)
Prcto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 0.0.0.0:10000         0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:25            0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:515           0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:98            0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:113           0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:79            0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:513           0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:514           0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:23            0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:21            0.0.0.0:*              LISTEN
tcp        0      0 0.0.0.0:111           0.0.0.0:*              LISTEN
udp        0      0 0.0.0.0:10000         0.0.0.0:*
udp        0      0 0.0.0.0:518           0.0.0.0:*
udp        0      0 0.0.0.0:517           0.0.0.0:*
udp        0      0 0.0.0.0:111           0.0.0.0:*
raw        0      0 0.0.0.0:1             0.0.0.0:*
raw        0      0 0.0.0.0:6             0.0.0.0:*
```

Листинг 1.

Как видно из результирующих данных листинга 1, любая строка, содержащая слово «LISTEN», означает, что имеется программа, прослушивающая этот порт. Прослушиваться должны только сконфигурированные администратором порты. Если в системе присутствует прослушиваемый порт, который не конфигурировался администратором, систему необходимо проверить и выяснить, почему порт открыт.

необходимо использовать команду ifstatus, доступную по адресу ftp://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/ifstatus/

Адреса, отображаемые в столбце локальных адресов, заканчиваются номером локального порта (число после столбца справа). Этот номер порта используется для определения, является ли соединение входящим или исходящим. Например, если номер локального порта 23, то это входящее подключение к демону telnet. Если номер локального порта равен 1035, а номер внешнего порта – 23, то это исходящее соединение telnet.

Одна из проблем, связанных с программой netstat, заключается в том, что данная команда не сообщает, какой процесс поддерживает открытое состояние порта. Поиск процесса, связанного с определенным портом, может стать очень трудной задачей. Однако существует программа под названием lsof (http://ftp.cerias.purdue.edu/pub/tools/unix/sysutil/lsof/), которая предоставляет такую информацию. Сразу после установки программы выполните команду lsof -i, как показано ниже

```
#lsof -i
COMMAND     PID     USER    FD    TYPE   DEVICE SIZE  NODE NAME
portmap     311     root    4u    IPv4   301      UDP   *:sunrpc
portmap     311     root    5u    IPv4   302      TCP   *:sunrpc (LISTEN)
inetd       439     root    5u    IPv4   427      TCP   *:ftp (LISTEN)
inetd       439     root    6u    IPv4   428      TCP   *:telnet (LISTEN)
inetd       439     root    7u    IPv4   429      TCP   *:shell (LISTEN)
inetd       439     root    9u    IPv4   430      TCP   *:login (LISTEN)
inetd       439     root    10u   IPv4   431      UDP   *:talk
inetd       439     root    11u   IPv4   432      UDP   *:ntalk
inetd       439     root    12u   IPv4   433      TCP   *:finger (LISTEN)
inetd       439     root    13u   IPv4   434      TCP   *:auth (LISTEN)
inetd       439     root    14u   IPv4   435      TCP   *:linuxconf (LISTEN)
lpd         455     root    6u    IPv4   457      TCP   *:printer (LISTEN)
sendmail    494     root    4u    IPv4   495      TCP   *:smtp (LISTEN)
miniserv.   578     root    4u    IPv4   567      TCP   *:10000 (LISTEN)
miniserv.   578     root    5u    IPv4   568      UDP   *:10000
```

Листинг 2.

Как видно из результатов выполнения (листинг 2) программы, lsof выводит перечень всех открытых портов с указанием того, какие процессы поддерживают открытое состояние портов[8]. Убедитесь, что вам известно, какие функции выполняет каждый процесс и почему открыт соответствующий ему порт.

Администратор также должен изучать результаты выполнения команды ps. Эта программа выводит все активные процессы, имеющиеся в системе, что необходимо при поиске снифферов, так как сниффер может не отображаться в lsof или в netstat. В большинстве систем выполнение команды ps -ef

---

[8]lsof заменяет номер порта в столбце справа именем порта, если оно присутствует в файле /etc/services

выводит перечень процессов в системе. В тех версиях Unix, где эта команда не работает, следует выполнить команду ps -aux. Результаты выполнения данной команды показаны ниже

```
#ps -ef
UID        PID  PPID  C    STIME   TTY      TIME      CMD
root       1    0     0    13:09   ?        00:00:04  init
root       2    1     0    13:09   ?        00:00:00  [kflushd]
root       3    1     0    13:09   ?        00:00:00  [kupdate]
root       4    1     0    13:09   ?        00:00:00  [kpiod]
root       5    1     0    13:09   ?        00:00:00  [kswapd]
root       6    1     0    13:09   ?        00:00:00  [mdrecoveryd]
bin        3 11 1     0    13:09   ?        00:00:00  portmap
root       327  1     0    13:10   ?        00:00:00  /usr/sbin/apmd -p 10 -w 5 -W
root       380  1     0    13:10   ?        00:00:00  syslogd -m 0
root       391  1     0    13:10   ?        00:00:00  klogd
daemon     407  1     0    13:10   ?        00:00:00  /usr/sbin/atd
root       423  1     0    13:10   ?        00:00:00  crond
root       439  1     0    13:10   ?        00:00:00  inetd
root       455  1     0    13:10   ?        00:00:00  lpd
root       494  1     0    13:10   ?        00:00:00  sendmail: accepting connections
root       511  1     0    13:10   ?        00:00:00  gpm -t ps/2
xfs        528  1     0    13:10   ?        00:00:00  xfs -droppriv -daemon -port -1
root       570  1     0    13:10   tty1     00:00:00  login - root
root       571  1     0    13:10   tty2     00:00:00  /sbin/mingetty tty2
root       572  1     0    13:10   tty3     00:00:00  /sbin/mingetty tty3
root       573  1     0    13:10   tty4     00:00:00  /sbin/mingetty tty4
root       574  1     0    13:10   tty5     00:00:00  /sbin/mingetty tty5
root       575  1     0    13:10   tty6     00:00:00  /sbin/mingetty tty6
root       578  1     0    13:10   ?        00:00:00  perl /usr/libexec/webmin/miniser
root       579  570   0    13:10   tty1     00:00:00  -bash
root       621  579   0    13:17   tty1     00:00:00  ps -ef
```

Следует периодически проверять список процессов, работающих в системе. Если обнаруживается что-либо незнакомое, то необходимо выяснить, что это такое.

Когда злоумышленник успешно проникает в систему, он может попытаться изменить системные файлы для обеспечения продолжительного доступа к системе. Файлы, передаваемые в систему, обычно называются "rootkit", так как позволяют злоумышленнику осуществить доступ через корневую (root) учетную запись. В дополнение к таким программам как сниффферы, rootkit может содержать двоичные замещения для следующих файлов[9]:

---

[9]По адресу http://www.chkrootkit.org/ можно найти утилиту, которая помогает в проверке наличия в системе rootkit-ов

```
ftpd    passwd
inetd   ps
login   ssh
netstat telnetd
```

Как правило, любой исполняемый файл, который может тем или иным образом помочь злоумышленнику поддерживать доступ, является кандидатом на замещение. Лучший способ для определения был ли файл заменен – использовать криптографическую контрольную сумму. Лучше всего создавать контрольные суммы всех системных файлов при построении системы, после чего обновлять их во время установки системных обновлений. Контрольные суммы необходимо хранить на безопасной системе, чтобы злоумышленник не мог их изменить при изменении файлов.

Если имеются подозрения нелегального проникновения в систему, пересчитайте контрольные суммы и сопоставьте их с исходными. Если они совпадают, то файлы изменены не были. Если же контрольные суммы различны, рассматриваемому файлу доверять не следует; его необходимо заменить оригиналом с установочного носителя.

Следующий раздел покажет пути проверки системы Unix на ошибки в конфигурации или на наличие неизвестных процессов и учетных записей.

6. ШАГ ЗА ШАГОМ

1. Начните с системы Unix, к которой имеется административный доступ (то есть имеется пароль к корневой учетной записи этой системы) и куда можно вносить изменения, не затрагивая рабочие приложения.

2. Найдите файлы загрузки и определите, какие приложения запускаются при загрузке системы. Выявите приложения, которые являются необходимыми для системы, и отключите все остальные.

3. Просмотрите файл inetd.conf и определите, какие службы включены. Определите службы, необходимые для системы и отключите все остальные. Не забудьте выполнить команду kill -HUP для процесса inetd, чтобы перезапустить его с использованием новой конфигурации.

4. Определите, используется ли в системе NFS. Внесите соответствующие изменения в файл dfstab.

5. Если система использует telnet или FTP, загрузите TCP Wrappers и установите программу в системе. Настройте TCP Wrappers на разрешение доступа только к telnet и FTP, согласно требованиям системы.

6. Найдите файл приветственного сообщения. Определите, используется ли корректное приветственное сообщение. Если это не так, разместите в системе корректное приветственное сообщение.

7. Выясните, настроены ли в системе требуемые ограничения на пароли согласно политике безопасности организации. Если это не так, внесите соответствующие настройки.

8. Определите, настроен ли в системе должным образом параметр umask по умолчанию. Если это не так, настройте umask соответствующим образом.

9. Определите требования для входа через корневую учетную запись. Если администраторам требуется осуществлять начальный вход с использованием их собственного идентификатора (ID), настройте соответствующим образом конфигурацию системы.

10. Проверьте систему на наличие неиспользуемых учетных записей. Все подобные учетные записи должны быть заблокированы.

11. Установите в системе соответствующие обновления.

12. Проверьте систему на некорректные пользовательские идентификаторы. В особенности следует искать учетные записи с UID, значение которого равно 0.

13. Убедитесь, что в системе ведется журнал подозрительной активности, и что файл syslog.conf настроен соответствующим образом.

14. Произведите в системе поиск скрытых файлов. Если будут найдены необычные скрытые файлы, исследуйте их, чтобы убедиться – в систему никто не проник.

15. Произведите поиск файлов SUID и SGID. Если будут обнаружены такие файлы, расположенные в каталогах пользователей, исследуйте их, чтобы убедиться – в систему никто не проник.

16. Произведите поиск файлов, общедоступных для записи. Если будут найдены такие файлы, либо устраните проблему посредством изменения разрешений (вначале выясните, для чего эти файлы используются), либо обратите на них внимание владельца.

17. Проверьте сетевые интерфейсы на наличие любых неправильных настроек.

18. Проверьте систему относительно прослушиваемых (активных) портов. Если обнаружится какое-либо несоответствие, найдите процесс, использующий порт и определите должен ли данный процесс работать в системе.

19. Проверьте таблицу процессов в системе и определите, выполняются ли какие-либо несоответствующие процессы.

Выводы

В зависимости от параметров проверяемой системы, на аудит может быть затрачено некоторое время. Кроме того, может потребоваться помощь различных пользователей системы. Как видим, гораздо легче вначале настроить систему корректным образом и затем поддерживать ее, чем осуществлять аудит системы и устранять проблемы.

7. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. При отключении службы, запускаемой автоматически, что необходимо сделать в файле загрузки?
2. Где находится файл конфигурации для inetd?
3. Как отключить службу для inetd?
4. Какие функции выполняет TCP Wrappers после установки?
5. Почему не следует размещать сообщение входа в /etc/motd?
6. Почему в системе Linux необходимо размещать сообщение в /etc/issue и /etc/issue.net?
7. Каким образом настройки возраста паролей в системе Solaris отличаются от аналогичных настроек в Linux?
8. Что устанавливает параметр umask?
9. Почему зашифрованные пароли пользователей должны храниться в файле shadow, а не в файле passwd?
10. Что должен проверить администратор, перед тем как включать BSM в системе Solaris?
11. Почему в файл syslog.conf должна быть включена строка <auth.info /var/log/auth.log>?
12. Почему общедоступный для записи файл SUID является потенциальной уязвимостью?
13. Какую информацию выводит команда netstat -m?
14. Каким образом использование lsof помогает защитить систему?
15. Какие данные выводит команда ps?

ПРИЛОЖЕНИЕ

xInetd

В большинстве клонов Linux в настоящее время система inetd заменена на использование системы xInetd.

Данное решение позволило обеспечить большую гибкость в реализации серверных приложений, большую безопасность и прозрачность в конфигурации. Демон xInetd продолжает выступать в роли супердемона при прослушивании портов, отвечающих в ОС за предоставление услуг для внешних и ряда внутренних вызовов со стороны клиентских приложений.

После установки ОС и xInetd администратор получает в свое распоряжение следующие файлы, задающие конфигурацию xInetd:

- основной файл конфигурации /etc/xinetd.conf, который содержит информацию объединенную в разделе defaults { ….. }, для указания параметров общих для всех служб, обрабатываемых с помощью xInetd и содержит каталог, обычно это /etc/xinetd.d, содержащий множество файлов, каждый из которых описывает отдельную службу с индивидуальными для нее параметрами;
- файлы /etc/hosts.deny и /etc/hosts.allow, содержимое которых xInetd учитывает при получении запросов на активацию сервисов.

Ниже более подробно рассматриваются параметры, описывающие возможное поведение xInetd.

Пример основного конфигурационного файла /etc/xinetd.conf:

```
defaults
{
        instances               = 25
        log_type                = FILE /var/log/servicelog
        log_on_success          = HOST PID
        log_on_failure          = HOST RECORD
        only_from               = 128.138.193.0 128.138.204.0 128.138.209.0
}
includedir   /etc/xinetd.d
```

В данном рзделе, defaults, допустимы следующие атрибуты:

- log_on_success – определяет логируемую информацию при удачном обращении
- log_on_failure – аналогично предыдущему, но при неудачном обращении к сервису
- only_from – разрешение обращения к сервисам с ограниченного диапазона IP-адресов
- instances – максимальное количество одновременно активируемых сервисов одного типа

Все файлы, находящиеся в каталоге /etc/xInetd.d и включаемые в конфигурацию с помощью оператора includedir в основном файле, имеют примерно одинаковую структуру и описывают параметры индивидуальные для каждой службы, которые могут перекрывать соответствующие параметры в секции defaults. Значение параметра, указываемого для атрибута service, должно совпадать с соответствующим именем в файле /etc/services. Ниже приведен пример описания службы telnet, для удаленного входа пользователей в систему.

```
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/etc/in.telnetd
        log_on_failure  += USERID
}
```

Данная конфигурация telnet свидетельствует о том, что запросы на нее должны приниматься портом 23 (будет взято из файла /etc/services), служба будет повторно использоваться, потоковый тип сокета, служба не будет ждать окончания работы вызванной ранее сессии для активации следующей сессии при поступлении вызова, резидент выполняется от имени пользователя root, сам файл, реализующий серверную часть, расположен в файле /usr/etc/in.telnetd, при успешном обращении в лог-файл будет дополнительно выводиться идентификатор пользователя.

Количество параметров, которые определяют поведение активируемого сервиса довольно велико, но не все они должны быть обязательно указаны. Ниже приведен перечень параметров, которыми можно тонко настроить поведение сервисов, активируемых с помощью xInetd:

- **service_name** – это имя сервиса

- **flags** – в качестве значения может быть использована любая комбинация из следующих значений:

REUSE – установить флаг SO_REUSEADDR на сокет сервера;
INTERCEPT – перехватывать пакеты или принимать соединения, по порядку проверяя, что они приходят из нужных мест;
NORETRY – избегать повторных попыток в случае неудачи;
IDONLY – соединение будет приниматься только от идентифицированных пользователей. На удаленной машине должен работать identification сервер;
NODELAY – для tcp сервиса будет установлен сокет;
TCP_NODELAY - для не tcp серисов никакого эффекта не будет;
DISABLE – сделать сервис недоступным;

KEEPALIVE – установка сокет флага SO_KEEPALIVE только для tcp сервисов

- **disabled** – при значении yes сервис не будет запускаться
- **socket_type** – возможные значения:
  stream – stream сокет
  dgram – dgram сокет
  raw – сервисы, требующие прямой доступ к IP
- **protocol** – определяет протокол, по которому будет работать сервер (tcp, udp, ...)
- **wait** – имеет два значения: «yes» и «no». Значение «yes» устанавливается только на stream сокетах. Если установлено «yes», то выполняется только один сервер для точно определенного порта. При значении «no» xinetd немедленно продолжает слушать порт
- **user** – определяет gid серверного процесса
- **server** – путь к исполняемой программе-сервера
- **server_args** – аргументы, с которыми будет запускаться серверная программа
- **only_from** – удаленные хосты, которым будет доступен сервис. В качестве значения принимает список хостов, IP адресов или сетей (из /etc/networks)
- **no_access** – хосты, которым данный сервис не будет доступен. Значения такие же, как и у only_from. Если не один из атрибутов (only_from и no_access) не указан, то сервис будет доступен всем
- **access_times** – интервалы времени в форме hour:min-hour:min, в которых сервис будет доступен. (Часы от 0 до 23 и минуты от 0 до 59.) Например: 9:00-12:00
- **log_type** – определяет куда сервис будет посылать логи. Значения:
  SYSLOG syslog_facility [syslog_level] – логи будут посылаться syslog демону;
  FILE file [soft_limit [hard_limit]] – логи будут писаться в указанный файл.
- **env** – значение атрибута – это список строк типа 'name=value'. Они будут добавлены в окружение перед тем как сервер будет запущен
- **passenv** – значение атрибута – это список переменных окружения из окружения xinetd, которые могут быть переданы серверу
- **port** – определяет порт сервиса. Если он указан в файле /etc/services, то он должен совпадать с ним
- **redirect** – позволяет tcp сервису делать редирект на другой хост. Значение – host:port
- **bind** – устанавливает интерфейс, на котором будет работать сервис. Синтаксис: bind=(ip address of interface)

- **interface** – синоним для bind
- **banner** – имя файла, который будет показываться при коннекте к сервису
- **banner_success** – имя файла, который будет показываться при удачном коннекте
- **banner_fail** – имя файла, который будет показываться при неудачном коннекте
- **cps** – атрибут имеет два аргумента. Первый устанавливает количество коннектов в секунду. Если это число будет превышено, сервис будет временно недоступен. Второй – число секунд, после которых сервис снова будет доступен
- **max_load** – загрузка. При достижении максимума, сервер перестает принимать запросы на соединение. Значение – число типа float
- **groups** – принимает значения "yes" или "no". Если значение атрибута "yes" – то сервер запустится с доступом к группам, которые имеют effective UID сервера. При значении "no" сервер запустится без привилегий групп. Атрибут должен иметь значение "yes" для большинства BSD систем. Этот атрибут может быть установлен в секции default
- **instances** – определяет число серверов, которые могут быть активны одновременно для сервиса (по умолчанию лимита нет). Значением этого атрибута может быть число, либо – UNLIMITED
- **nice** – устанавливает приоритет сервиса. Смотрите man nice 3 для получения более точной информации
- **enabled** – список имен задействованных сервисов

Для активации самого сервера xinetd, остановки, рестарта и т.д. обычно используется стандартный скрипт /etc/rc,d/init,d/xinetd с соответствующей командной строкой: start, stop, restart и т.д.

Для ограничения доступа к поднятым сервисам очевидно можно использовать привязку к конкретному IP-адресу (при наличии нескольких интерфейсов), доступ по времени, доступ по списку допустимых адресов клиентов и т.д.

Обеспечение безопасности системы Ubuntu

Рассмотрим также систему безопасности Ubuntu Linux как наиболее распространенной Unix-подобной системы на персональных компьютерах [6].

Сообщество Ubuntu уделяет много внимания проблеме безопасности системы. Комплекс мер по обеспечению безопасности затрагивает несколько аспектов:

1) обновление системы;
2) настройка межсетевого экрана (файерволла);
3) использование антивирусных программ;
4) шифрование данных;
5) обеспечение безопасности веб-браузеров.

Обновление Ubuntu

Обновление системы является важнейшим аспектом обеспечения безопасности компьютера. Для установки обновлений компьютер должен быть подключен к Интернету.

Перейдите к меню *Система -> Администрирование -> Менеджер обновлений*. Открывшееся окно предоставляет Вам доступ к обновлению системы и настройкам обновлений.

1. Нажмите кнопку **Проверить** для загрузки списка обновлений.
2. Введите Ваш пароль при появлении запроса.
3. После проверки онлайн-ресурсов в окне Менеджера обновлений появятся названия доступных компонентов. Нажмите кнопку *Установить обновления*.
4. После загрузки и установки всех обновлений последовательно нажмите кнопки **Закрыть** в окнах *Внесение изменений...* и *Менеджер обновлений*.
5. При необходимости перегрузите компьютер.

Для настройки времени и способа обновлений выберите пункт меню *Система -> Администрирование -> Источники приложений* и перейдите на вкладку *Обновления*

1. Секция *Обновления Ubuntu* дает возможность устанавливать как официальные, так и тестовые обновления. Однако установка обновлений, не поддерживаемых или не вошедших в официальный релиз, может привести к ошибкам в работе системы.
2. Установите частоту и автоматизацию загрузки и установки обновлений в секции *Автоматические обновления*. Рекомендуется установить опцию *Устанавливать обновления безопасности без подтверждения*.
3. Секция *Обновления релиза* – определите, какие релизы будут предложены для установки в случае их выхода – обычные или с долговременной поддержкой (LTS). Кроме того, можно запретить предлагать установку нового релиза.
4. После внесения всех необходимых изменений нажмите кнопку **Закрыть**.

5. Нажмите кнопку **Обновить** в открывшемся окне для повторной проверки доступных обновлений в соответствии с измененными параметрами.

Обновить систему можно также и из командной строки. Для этого могут быть использованы следующие два набора команд:

```
sudo apt-get update
sudo apt-get upgrade
```

или

```
sudo apt-get update
sudo apt-get dist-upgrade
```

Команда **sudo apt-get update** загружает список доступных обновлений, поэтому всегда должна идти первой.

Команда **sudo apt-get upgrade** обновляет все приложения, установленные на компьютере. В случае использования командной строки рекомендуем выполнять эту команду не реже чем раз в неделю.

Команда **sudo apt-get dist-upgrade** служит для обновления только самого дистрибутива Ubuntu. Используйте эту команду, когда знаете, что доступен новый дистрибутив.

Настройка межсетевого экрана

Частью ядра Ubuntu является мощный файерволл, называющийся *netfilter*. Однако он не активирован по умолчанию и должен быть настроен вручную, так как сама система является достаточно закрытой и не предусматривающей свободного обмена данными между выполняемыми приложениями и сетевыми ресурсами Интернета. Однако, несмотря на подобную принципиальную защиту, необходимость в дополнительных программных средствах не уменьшается.

Для конфигурирования встроенного файерволла Ubuntu предназначена программа Firestarter, использующая графический интерфейс. В терминале файерволл может быть настроен командой **uf w**.

Установка и настройка Firestarter

Установка приложения осуществляется через *Менеджер пакетов Synaptic (Система -> Администрирование -> Менеджер пакетов Synaptic)*.

1. После запуска *Synaptic* введите слово *firestarter* в строку быстрого поиска.

2. В списке доступных приложений выберите *Firestarter* и щелкните правой кнопкой мыши на его названии.

3. В выпадающем контекстном меню выберите пункт *Отметить для установки*.

4. Подтвердите установку дополнительных пакетов и нажмите **Применить** на верхней панели *Synaptic* и в появившемся окне описания установки.

5. После завершения процесса установки нажмите **Закрыть** в окне прогресса и выйдите из *Менеджера пакетов Synaptic*.

Для запуска приложения *Firestarter* перейдите в меню *Приложения -> Интернет -> Firestarter*.

1. Нажмите кнопку **Вперед** для начала настроек.

2. Выберите тип подключения к сети из выпадающего списка *Обнаруженные устройства*.

3. Поставьте галочку слева от опции *Включать межсетевой экран при дозвоне*, если Вы пользуетесь коммутируемым соединением с провайдером типа dial-up. В этом случае файерволл будет запускаться каждый раз, когда связываетесь со своим провайдером.

4. Установите галочку слева от опции *IP-адреса получаются по DHCP*, если провайдер предоставляет динамический IP-адрес при доступе в Интернет.

5. Нажмите кнопку **Вперед** для перехода к следующему окну.

6. Можно разрешить другим компьютерам в локальной сети использовать подключение к Интернету по одному IP-адресу, назначенному провайдером этому компьютеру. Для реализации этой возможности компьютер должен иметь, как минимум, две сетевые карты, одна из которых будет подключена к сети провайдера, а другая – к локальной сети дома или в офисе. В этом случае Ваш компьютер будет выступать в роли DHCP-сервера и предоставлять IP-адреса компьютерам локальной сети.

7. Нажмите кнопку **Вперед** для перехода к следующему окну.

8. Нажмите кнопку **Сохранить**, если Вы хотите сохранить настройки и запустить файерволл сразу, или уберите галочку слева от опции *Start firewall now* и нажмите **Сохранить** для сохранения настроек без запуска файерволла.

Настройка правил для входящих потоков данных

Для повторного запуска приложения Firestarter перейдите в меню *Приложения -> Интернет -> Firestarter*. По умолчанию, весь входящий и исходящий поток данных блокируется при включенном файерволле, если не добавлены правила пропуска данных от и к внешним источникам.

1. После запуска *Firestarter* перейдите на вкладку *Policy*.

2. Наведите указатель мыши на область *Allow Service...* и щелкните левой кнопкой. Область подсветится розовым цветом.

3. В главном меню программы перейдите к пункту *Policy -> Add Rule*.

4. В открывшемся диалоговом окне введите параметры службы, которой будет разрешено получать данные из Интернета.

5. Из выпадающего списка *Name* выберите приложение или службу доступа к сети. Например, программу BitTorrent.

6. Поле Порт заполнится значениями портов по умолчанию, установленных для каждого приложения или службы. Не изменяйте эти значения, если нет точных данных о номерах используемых портов!

7. Выберите опцию *Anyone* в секции *When the source is*, чтобы разрешить получение данных от любого сервера в Интернете.

8. Можно ограничить источники получения данных, введя сетевой адрес сервера в поле *IP, host or network*.

9. Добавьте свой комментарий в поле *Comment* (необязательно) и нажмите кнопку **Добавить** для сохранения настроек правила.

Настройка правил для исходящих потоков данных

Для разрешения исходящих потоков данных необходимо создать соответствующие правила файерволла[10].

1. На вкладке *Policy* в выпадающем списке *Editing* выберите пункт *Outbound traffic policy*.

2. Выберите разрешающую или запрещающую опцию, установив соответствующий переключатель.

3. Вне зависимости от выбранной политики безопасности наведите указатель мыши на нижнюю горизонтальную область окна и щелкните левой кнопкой. Область подсветится розовым цветом.

4. В главном меню программы выберите *Policy Add Rule*.

5. В открывшемся диалоговом окне введите параметры службы, которой будет разрешено или запрещено, в зависимости от выбранной политики, передавать данные в Интернет.

6. Из выпадающего списка *Name* выберите приложение или службу доступа к сети. Например, программу BitTorrent.

7. Поле Порт заполнится значениями портов по умолчанию, установленных для каждого приложения или службы. Не изменяйте эти значения, если нет точных данных о номерах используемых портов!

8. Выберите опцию *Anyone* в секции *When the source is*, чтобы разрешить или запретить передачу данных любому серверу.

9. Можно ограничить направление передачи данных, введя сетевой адрес сервера в поле *IP, host or network*.

10. Добавьте свой комментарий в поле *Comment* и нажмите кнопку **Добавить** для сохранения настроек правила.

Установка антивирусной защиты

Несмотря на наличие «иммунитета» у Ubuntu к заражению компьютерными вирусами, необходимость в установке антивирусных программ не отпадает. Это связано с тем, что компьютер может, не заражаясь сам, передавать вирусы, содержащиеся в файлах, скачанных из глобальной сети Интернет, или полученных в почтовых сообщениях, в свою локальную сеть, в ко-

---

[10] **Whitelist** (белый список) – опция, при выборе которой весь исходящий поток данных запрещен, кроме направлений, разрешенных специальными правилами. Также известна под названием Запрещающая политика. Обеспечивает максимальную защиту, однако требует добавления соответствующего правила для каждого направления и вида соединения.
Blacklist (черный список) – опция, при выборе которой весь исходящий поток данных разрешен, кроме направлений, запрещенных специальными правилами. Также известна под названием Разрешающая политика.

торую могут входить компьютеры, работающие под управлением операционных систем Windows или Mac.

Среди большого количества коммерческих антивирусных программ выделяется приложение с открытым исходным кодом *ClamAV*. Это мощный комплекс, который может обеспечить защиту на серверном уровне. Также можно установить его модификацию *ClamTK*, обладающую графическим интерфейсом понятным рядовому пользователю.

Установка и настройка ClamTK

1. Запустите *Менеджер пакетов Synaptic*, перейдя к пункту меню *Система -> Администрирование -> Менеджер пакетов Synaptic*.

2. Введите свой пароль в соответствующий запрос.

3. В строку быстрого поиска введите *clamtk*.

4. В списке доступных приложений выберите *clamtk* и щелкните правой кнопкой мыши на его названии.

5. В выпадающем контекстном меню выберите пункт *Отметить для установки*.

6. Подтвердите установку дополнительных пакетов и нажмите **Применить** на верхней панели Synaptic и в появившемся окне описания установки.

7. После завершения процесса установки нажмите **Закрыть** в окне прогресса и выйдите из *Менеджера пакетов Synaptic*.

Для установки *ClamTK* с использованием командной строки запустите терминал *(Приложения -> Стандартные -> Терминал)* и введите команду **gksu clamtk**.

Для запуска приложения выберите пункт меню *Приложения -> Системные утилиты -> Virus Scanner*. Выберите режим обновления антивирусных баз. Затем последовательно нажмите кнопки **Save** и **Quit**.

Проверка на наличие вирусов

Для запуска приложения выберите пункт меню *Приложения -> Системные утилиты -> Virus Scanner*

1. В секции *Действия* выберите каталог или файл, который хотите проверить на наличие вирусов, и нажмите соответствующую кнопку:

- **Home** – ClamTK проверит содержимое домашнего каталога пользователя;
- **Файл** – приложение проверит выбранный файл;
- **Каталог** – ClamTK проверит файлы, содержащиеся в указанном Вами каталоге.

2. Нажатие на описанные кнопки начнет процесс проверки с установками по умолчанию. Для изменения некоторых параметров поставьте галочки слева от их названий в строке под кнопками:

- *Проверять скрытые* – в проверку включаются скрытые файлы и подкаталоги в выбранном для проверки каталоге;

• *Recursive (Рекурсивная проверка)* – включает в проверку вложенные подкаталоги;

• *Полностью* – включает дополнительные возможности проверки;

• *Игнорировать размер* – включает в проверку файлы, размер которых превышает 20 MB;

• *Сохранить журнал* – включает возможность ведения журнала проверок.

3. После запуска проверки в нижней части окна приложения появится секция *Сканирование*, предоставляющая информацию о прогрессе, количестве и наименовании сканируемых объектов, количестве найденных вирусов.

4. Секция Состояние отображает информацию о версии приложения, дате обновления антивирусной базы и дате последней проверки.

5. Для обновления антивирусной базы перейдите к пункту меню приложения *Помощь -> Обновить сигнатуры* или используйте сочетание клавиш **Ctrl+U**.

6. Нажмите кнопку **Выход** для завершения работы приложения.

Шифрование файлов и каталогов

Шифрованием файлов и каталогов называется процесс их кодирования во избежание несанкционированного доступа. При этом шифрованный объект становится доступным только после ввода соответствующего пароля.

В Ubuntu могут быть зашифрованы как отдельные файлы или каталоги, так и создан специальный каталог, содержимое которого – вновь созданные и перемещенные в него объекты – будет автоматически шифроваться.

Создание частного каталога шифрования /Private

Частный каталог шифрования /Private создается в домашнем каталоге пользователя. Файлы и каталоги, помещаемые в него, сохраняются в специальном хранилище, которое, на самом деле, является файловым архивом. В начале сеанса пользователя содержимое архива монтируется в каталог /Private и демонтируется при его завершении. Таким образом, зашифрованные данные являются недоступными для любого другого пользователя на данном компьютере.

Перед созданием частного каталога шифрования необходимо установить дополнительное приложение, предназначенное для создания и управления каталогом.

1. Запустите *Менеджер пакетов Synaptic*, перейдя к пункту меню *Система -> Администрирование -> Менеджер пакетов Synaptic*.

2. Введите свой пароль в соответствующий запрос.

3. В строку быстрого поиска введите **ecryptfs-ut i ls**.

4. В списке доступных приложений выберите *ecryptfs-utils* и щелкните правой кнопкой мыши на его названии.

5. В выпадающем контекстном меню выберите пункт *Отметить для установки*.

6. Подтвердите установку дополнительных пакетов и нажмите **Применить** на верхней панели Synaptic и в появившемся окне описания установки.

7. После завершения процесса установки нажмите **Закрыть** в окне прогресса и выйдите из *Менеджера пакетов Synaptic*.

Для установки приложения с использованием командной строки запустите терминал *(Приложения -> Стандартные -> Терминал)* и введите команду **sudo apt-get install ecryptfs-utils**

Введите свой пароль пользователя для заимствования прав суперпользователя.

Запуск приложения осуществляется только из терминала следующей командой **ecryptfs-setup-private**

Введите пароль учетной записи, затем введите пароль монтирования и демонтирования каталога. Запомните эти пароли, чтобы не потерять доступ к данным, помещенным в зашифрованный каталог.

После завершения установки каталога можно выйти из терминала при помощи команды **exit**.

Для входа в папку /Private откройте свой домашний каталог в Файловом менеджере *(Переход -> Домашняя папка)*, затем дважды щелкните левой кнопкой мыши на значке каталога /Private. Эмблема в виде замка в правом верхнем углу значка означает, что эта папка зашифрована.

Каталог содержит два файла – **README.txt** и **Access-Your-Private-Data.desktop**. В первом файле содержится информация о том, что монтирование каталога не было произведено для защиты данных, а для монтирования необходимо либо запустить исполняемый файл **Access-Your-Private-Data.desktop**, находясь в файловом менеджере, либо ввести команду **ecryptfs-mount-private** в терминале. Введите пароль монтирования в появившемся запросе.

Файлы и каталоги, копируемые или перемещаемые в каталог /Private, автоматически помещаются в файловое хранилище и шифруются без участия пользователя.

Индивидуальное шифрование файлов и каталогов

Процесс шифрования отдельных файлов и каталогов полностью отличается от вышеописанного использования шифрованного каталога /Private.

Шифрование заключается в копировании шифруемого файла или каталога в новый (зашифрованный) и удалении исходного. Затем для доступа к файлу или каталогу должен быть произведен обратный процесс дешифрации. При внесении последующих изменений файл или каталог должен быть зашифрован заново целиком. Таким образом, индивидуальное шифрование более применимо для редко используемых или архивных файлов и каталогов.

Для шифрования и дешифрования файлов и каталогов необходимо один раз создать ключ шифрования.

1. Перейдите к меню *Приложения -> Стандартные -> Пароли и ключи шифрования*.

2. В меню приложения выберите пункт *Файл -> Создать* или используйте сочетание клавиш **Ctrl+N**.

4. Выберите пункт *Ключи PGP* и нажмите кнопку **Продолжить**.

5. Введите личные данные и нажмите кнопку **Создать**.

6. Задайте пароль для ключа и нажмите кнопку **ОК**. Создание ключа требует достаточно много времени в зависимости от быстродействия компьютера.

7. После окончания процесса создания ключа он появится в списке на вкладке *Личные ключи* основного окна приложения.

8. Закройте приложение.

После создания личного ключа можно его использовать для шифрования файлов и каталогов. Этот ключ может быть использован многократно для шифрования любых файлов и каталогов в файловой системе Ubuntu.

1. Откройте домашний каталог в *Файловом менеджере (Переход -> Домашняя папка)*.

2. Наведите указатель мыши на значок каталога для шифрования (например, Документы) и нажмите правую кнопку.

3. В выпавшем контекстном меню выберите пункт *Зашифровать*.

4. Выберите созданный ключ шифрования в окне *Выбрать получателей* и нажмите кнопку **ОК**.

5. В случае выбора каталога или нескольких файлов появится окно *Зашифровать несколько файлов*, позволяющее выбрать одну из двух опций:

- *Зашифровать каждый файл отдельно* – в этом случае каждый из выбранных файлов будет зашифрован по отдельности и размещен в каталоге с исходным файлом, но с новым расширением;
- *Зашифровать в упакованном виде* – все выбранные файлы будут заархивированы в один общий архивный файл, а затем этот новый архив будет защищен паролем.

6. Какая бы из опций ни была выбрана, будет создан один или несколько файлов с расширением .pgp – это и есть зашифрованные файлы.

7. При необходимости исходные файлы или каталоги могут быть удалены.

Дешифрование файлов и каталогов

Для дешифрования необходимо сделать несколько простых действий:

1. Откройте в Файловом менеджере расположение шифрованных файлов с расширением .pgp.

2. Наведите указатель мыши на название файла, подлежащего дешифрованию, и дважды щелкните на нем левой кнопкой.

3. Выберите название и расположение для дешифрованного файла в появившемся диалоге и нажмите кнопку **Сохранить**.

4. Введите пароль, использовавшийся при создании Вашего ключа шифрования, и нажмите кнопку **ОК**.

5. После окончания процесса дешифрования в указанном каталоге появится файл с заданным именем. Скорость дешифрования зависит от размера файла и быстродействия компьютера.

6. Шифрованный файл с расширением .pgp теперь может быть удален, так как при внесении изменений в дешифрованный файл процедуру шифрования необходимо будет производить вновь, с самого начала.

Система безопасности ОС Android

В последнее время интенсивно развивается Android – операционная система для мобильных телефонов, планшетных компьютеров и смартбуков, основанная на ядре Linux. Изначально она разрабатывалась компанией Android Inc., которую затем приобрела Google. Впоследствии, Google инициировала создание Open Handset Alliance (OHA), которая в настоящее время и занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки и писать приложения на Си и других языках программирования с помощью Android Native Development Kit.

Рассмотрим данную OS с точки зрения безопасности системы. Android является разделенной по полномочиям операционной системой, в которой каждое приложение работает с отдельными системными идентификационными данными (идентификатор пользователя Linux и групповой ID). Части системы также разделяются на отдельные идентификационные данные. Linux, таким образом, изолирует приложения друг от друга и от системы.

Дополнительные, более детальные средства защиты обеспечиваются с помощью механизма «разрешений», который обеспечивает соблюдение ограничений для операций, которые выделенный процесс может выполнить, и полномочия URI для предоставления оперативного доступа к определенным данным.

Архитектура безопасности

Центральной точкой проекта архитектуры безопасности Android является то, что по умолчанию ни у какого приложения нет разрешения на выполнение операций, которые неблагоприятно воздействовали бы на другие приложения, операционную систему или процессы пользователя. Эти операции включают в себя чтение или запись частных данных пользователя (таких как контакты или электронные письма), чтение или запись файлов другого приложения, выполнение сетевого доступа, незасыпание устройства и т.д.

Поскольку приложения выполняются каждое в своей отдельной области памяти («песочнице») и изолированы друг от друга, они должны явно ука-

зывать, как ими будут использоваться совместные ресурсы и данные. Это достигается путем объявления полномочий, в которых они нуждаются для дополнительных возможностей, не обеспеченных основной «песочницей». Приложения статически объявляют полномочия, которых они требуют, и система Android требует подтверждения пользователя для этого (например, во время установки приложения). У Android нет никакого механизма для того, чтобы предоставить полномочия динамически (во время выполнения), так как это нанесет ущерб безопасности.

Ядро несет полную ответственность за отделение «песочниц» приложений друг от друга. Все типы приложений – Java, нативные и гибридные – изолированы подобным образом и безопасны друг для друга.

Подпись приложения

Все приложения Android (.apk файлы) должны быть подписаны сертификатом, закрытый ключ которого создается их разработчиком. Этот сертификат идентифицирует автора приложения. Сертификат необязательно должен быть подписан центром сертификации: для приложений Android допустимо использовать собственные (самоподписанные) сертификаты.

Идентификаторы пользователей и доступ к файлу

Во время установки Android дает каждому программному пакету отдельный идентификатор пользователя Linux (UID). Идентификационные данные остаются постоянными на все время жизни пакета на данном устройстве. На другом устройстве у того же самого пакета может быть другой UID. Важно то, что разные пакеты имеют различные UID на данном устройстве.

Поскольку осуществление безопасности происходит на уровне процесса, код любых двух пакетов не может обычно работать в том же самом процессе, так как пакеты должны работать как различные пользователи Linux. Можно использовать атрибут sharedUserId из AndroidManifest.xml в явном теге каждого пакета, чтобы присваивать их такому же идентификатору пользователя. В целях безопасности эти два пакета тогда обрабатываются как одно приложение, с одним идентификатором пользователя и полномочиями файла. Обратите внимание – для того, чтобы сохранить безопасность, только двум приложениям с одинаковой цифровой подписью (и с запросом такого же sharedUserId), дадут такой же идентификатор пользователя.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Робачевкий А. М. Операционная система UNIX. / А. М. Робачевкий, С. А. Немнюгин, О. Л. Стесик. – 2-е изд. – СПб.: БХВ-Петербург, 2010. – 656 с.

2. Шимонски Роберт. Освой самостоятельно Unix. 10 минут на урок = Sams Teach Yourself Unix in 10 Minutes. / Роберт Шимонски. – М.: «Вильямс», 2006. – 272 с.

3. Реймонд Эрик С. Искусство программирования для Unix = Art of Unix Programming. / Эрик С. Раймонд. – М.: «Вильямс», 2005. – 544 с.

4. Роббинс А. Unix. Справочник. / А. Роббинс: пер. с англ. 4-е изд. – М.: «КУДИЦ-ПРЕСС», 2007. – 864 с.

5. Tanenbaum Andrew Stuart. Operating Systems: Design and Implementation, ISBN 0-13- 638677-6

6. Голобродский К.В. Знакомьтесь: Ubuntu / К.В. Голобродский. – Ростов-на-Дону: Феникс, 2010. – 160 с.

7. Security and Permissions / http://developer.android.com

Учебное издание

В.В. Белоусов
В.И. Бондаренко
В.В. Данилов
А.А. Каргин
Ю.А. Кожемякин

Обеспечение безопасности в UNIX-подобных операционных системах Лекции и практические работы по специальности 6.170101 «Безопасность информационных и коммуникационных систем»

Редактор Е.И.Хвостова

Компьютерная верстка Н.Л. Попова

План изд. 2011г., поз. № 62

Підписано до друку 09.12.2011 р.
Формат 60 х 84/16. Папір офсетний.
Друк – цифровий. Умовн.-друк. арк. 2,79.
Тираж 100 прим. Зам. №056

Видавництво Донецького національного університету
83001, м. Донецьк, вул. Університетська, 24.
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру
серія ДК №1854 від 24.06.2004 р.