КИРК УЭЙНГРОУ

UNIX

Полезные советы для системных администраторов

- ИЗМЕНЕНИЕ И ДОБАВЛЕНИЕ УЧЕТНЫХ ЗАПИСЕЙ ПОЛЬЗОВАТЕЛЕЙ
- МОНИТОРИНГ СЕРВЕРОВ И РАБОЧИХ СТАНЦИЙ
- ОБНАРУЖЕНИЕ УЯЗВИМЫХ МЕСТ В СЕТИ
- УСТРАНЕНИЕ НЕИСПРАВНОСТЕЙ ТЕКСТОВЫХ ТЕРМИНАЛОВ

АДМИНИСТРИРОВАНИЕ И ЗАЩИТА

que®

ДМК издательство

УДК 004.451.9UNIX
ББК 32.973.26-018.2
У97

Уэйнгроу К.
У97 UNIX: полезные советы для системных администраторов: Пер. с англ. – М.: ДМК Пресс. – 416 с.: ил. (Серия «Защита и администрирование»).

ISBN 5-94074-071-5

Данная книга предназначена для системных администраторов, обслуживающих компьютеры, на которых установлена операционная система UNIX. Предполагается, что читатель уже знаком с основными функциями и особенностями этой ОС. Автор книги рассказывает, как можно автоматизировать рутинную работу и подробно описывает процесс создания командных файлов, благодаря использованию которых значительно повышается производительность труда.

Приемы, рассмотренные в книге, были опробованы в разных версиях системы и в большинстве случаев подходят для каждой из них. Подробно освещаются такие темы, как администрирование сети, безопасность ОС, настройка компьютера, работа с учетными записями и файлами, эмуляция терминалов. Особое внимание уделено взаимоотношениям системного администратора с пользователями.

ISBN 0-7897-1927-4 (англ.)
ISBN 5-94074-071-5 (рус.)

СОДЕРЖАНИЕ

Об авторе 15
Предисловие 17

Глава 1
Вопросы администрирования 24

1.1. Сбор информации о системе 26
1.2. Копируйте ключевые файлы! 27
1.3. Запуск программы в последний день месяца 29
1.4. Отключение ненужных демонов 30
1.5. Перезапуск демонов 33
1.6. Применение fuser вместо ps 35
1.7. Изменение размера раздела подкачки «на лету» 37
1.8. Фоновые процессы и nohup 38
1.9. Перенаправление вывода в Null 40
1.10. Блокирование удаленного доступа 43
1.11. Быстрая перемотка лент 48
1.12. Генерация диапазона чисел 50
1.13. Удаление файлов, имя которых начинается с дефиса 51
1.14. Применение echo вместо ls 54
1.15. Создание больших тестовых файлов 55

1.16. Тестирование дисков 58
1.17. Завершение работы системы 62

Глава 2
Администрирование сети 66
2.1. Сетевая модель OSI 66
2.2. Поиск неисправности 68
2.3. Скрытие файлов в NFS 71
2.4. Удаленная настройка сетевого соединения 73
2.5. Завершение работы, остановка или перезагрузка системы по сети 76
2.6. Организация взаимодействия NFS3 и NFS2 78
2.7. Размонтирование занятых устройств 80
2.8. Статическая или динамическая маршрутизация? 84
2.9. Получение адреса Ethernet с помощью arp 87

Глава 3
Безопасность 89
3.1. Делегирование прав root нескольким администраторам 90
3.2. Полный путь к команде su 92
3.3. Мониторинг записей с правами root в файле паролей 94
3.4. Уязвимые места UNIX 96
3.5. Уровни прав доступа 99
3.6. Защита корневого каталога 102
3.7. Поиск файлов 103
3.8. Шифрование файлов 105
3.9. Блокирование экрана и его очистка 109
3.10. Мощные инструменты 111

Глава 4
Мониторинг системы 114
4.1. Мониторинг во время загрузки 115
4.2. Получение базовых значений данных 118

4.3. Мониторинг с помощью tail 121
4.4. Усечение log-файла 123
4.5. Мониторинг процесса по почте 126
4.6. Контроль объема дискового пространства 128
4.7. Поиск файлов, «пожирающих» дисковое пространство 132
4.8. Контроль изменений с помощью grep 133
4.9. Мониторинг с помощью ping 135
4.10. Контроль дампов памяти 138
4.11. Контроль аварийных файлов 140
4.12. Переход на летнее или зимнее время 142
4.13. Проверка времени 144

Глава 5
Учетные записи пользователей 147
5.1. Имена учетных записей пользователей 147
5.2. Пароли 150
5.3. Идентификаторы пользователей 152
5.4. Идентификаторы групп 155
5.5. Поле GECOS 157
5.6. Домашние каталоги 158
5.7. Оболочки и файл паролей 161
5.8. Работа с учетными записями 163
5.9. Конфигурационные файлы пользователей 167
5.10. Применение сокращений 168
5.11. Пользователи MS DOS 172
5.12. Смена оболочки 173
5.13. Поиск дисплея 174
5.14. Копирование файлов в несколько домашних каталогов 176
5.15. Уничтожение сеанса работы 177
5.16. Сброс пароля root без vi 179

Глава 6
Работа с файлами 181

6.1. Копирование файлов с атрибутами 181
6.2. Копирование файлов на удаленный компьютер 186
6.3. Где хранить временные файлы? 192
6.4. Работа с символьными ссылками 196
6.5. Поиск файлов с помощью grep 201
6.6. Поиск по нескольким шаблонам 203
6.7. Рекурсивное выполнение команд с помощью find 206
6.8. Перемещение или переименование групп файлов 209
6.9. Извлечение информации из страниц руководства 213
6.10. Преобразование файлов DOS 216
6.11. Разбиение файлов 217
6.12. Ограничение размера дампов памяти 220
6.13. Команды uuencode и uudecode 222

Глава 7
Дисплеи и эмуляция 227

7.1. Типы терминалов 228
7.2. Определение типа терминала 231
7.3. Применение stty 234
7.4. «Горячие» клавиши 239
7.5. Тестирование текстовых терминалов 241
7.6. Устранение неисправностей текстовых терминалов 247
7.7. Общие STDIN/STDOUT на двух терминалах 249
7.8. Обновление X терминала 252
7.9. Уничтожение ресурсов с помощью xkill 253
7.10. Изменение заголовка xterm 254
7.11. Управление мышью с клавиатуры 255
7.12. Подключение к удаленному X-серверу 256
7.13. Таблица ASCII в UNIX 259

Глава 8
Редакторы ... 262

8.1. Анатомия ed и vi ... 263
8.2. Шесть ступеней ed ... 264
8.3. Шесть ступеней vi ... 266
8.4. Настройка параметров vi ... 270
8.5. Сокращение команд vi ... 274
8.6. Создание макросов ... 286
8.7. Поиск и замена ... 291
8.8. Другие применения vi ... 295
8.9. Одновременное редактирование нескольких файлов ... 299
8.10. Редактирование, выполнение и снова редактирование ... 300
8.11. Считывание STDOUT в vi ... 302
8.12. Работа с vi при переполнении tmp ... 303

Глава 9
Пользователи ... 305

9.1. Шесть типов пользователей ... 306
9.1.1. Извиняющийся пользователь ... 307
9.1.2. Пользователь-догматик ... 308
9.1.3. Мнительный пользователь ... 309
9.1.4. Бета-пользователь ... 310
9.1.5. Пользователь, считающий себя администратором ... 312
9.1.5. Идеальный пользователь ... 313
9.2. Новые пользователи ... 314
9.2.1. Создание регистрационной записи ... 315
9.2.2. Встреча с новым пользователем ... 315
9.2.3. Система и регистрационная запись пользователя UNIX ... 316
9.2.4. Корпоративная политика ... 316
9.2.5. Рабочее окружение ... 317
9.3. Самореклама ... 318
9.3.1. Будьте на виду ... 318
9.3.2. Проверочные звонки ... 319
9.3.3. Поддерживайте контакт ... 319

9.4. Умейте произвести впечатление 321
9.4.1. Умейте слушать 321
9.4.2. Помогайте пользователям в мелочах 321
9.4.3. Избавляйте пользователей от страхов 321
9.4.4. Звоните производителю ПО 322
9.5. Обращение с рассерженным пользователем 322
9.5.1. Как успокоить пользователя 322
9.5.2. Рассмотрение жалобы 323
9.5.3. Обратная связь 324
9.6. Средства удаленной поддержки пользователей 325
9.7. Передача оборудования во временное пользование 325
9.7.1. Порядок передачи оборудования пользователям 326
9.7.2. Возврат оборудования 327
9.7.3. Что делать в случае, если оборудование не возвращают 328
9.8. Сообщение об отсутствии доступа к компьютеру 329
9.8.1. Сколько времени понадобится? 330
9.8.2. Сколько сообщений рассылать? 331
9.8.3. Как лучше сформулировать сообщение об отсутствии доступа к компьютеру 332
9.8.4. Как предупреждать пользователей 333
9.9. Забота пользователей об администраторах 333
9.10. Когда пользователи увольняются 334
9.10.1. Общайтесь с ними по сети! 334
9.10.2. Индивидуальный подход 335
9.10.3. Блокирование учетной записи 335

Глава 10
Профессия системного администратора 336
10.1. Три уровня администраторов 337
10.1.1. Младший уровень 338
10.1.2. Средний уровень 339
10.1.3. Старший уровень 339
10.1.4. Достижение статуса гуру 340
10.2. Функции администратора 341

10.3. Поиск работы, связанной с UNIX 343
10.3.1. Ваше первое место работы 344
10.3.2. Доступные ресурсы 344
10.4. Подготовка резюме администратора 346
10.4.1. Малые предприятия 347
10.4.2. Крупные корпорации 348
10.4.3. Грамотное оформление резюме специалиста по UNIX 348
10.4.4. Создание нескольких резюме 358
10.5. Подготовка к собеседованию 358
10.6. Типы собеседований 360
10.7. Поведение на собеседовании 362
10.8. Поиск сотрудников 365
10.9. Проведение собеседования с кандидатами 367
10.9.1. Собеседование по телефону 367
10.9.2. Проведение собеседования 368
10.10. Работа с торговыми представителями и инженерами службы поддержки 370
10.10.1. Их тактика 371
10.10.2. Работа с новыми торговыми представителями 371
10.10.3. Проверка цены 372
10.10.4. Получение гарантии 373
10.10.5. Кое-что о подарках 373
10.11. Взаимодействие со службой поддержки 373
10.12. Работа с инженерами службы поддержки 375

Приложение 1
Основные концепции создания командных файлов 377
Создание командного файла 377
Рекурсивные командные файлы 379

Приложение 2
Карточка установки системы 381
Образец заполнения карточки установки системы 382

Приложение 3
Журнал регистрации происшествий 386
Образец заполнения журнала регистрации происшествий 387

Приложение 4
Утилиты администратора и рекомендованные организации 388
Утилиты для системного администрирования 388
Сетевые утилиты 390
Утилиты контроля безопасности 391
Рекомендованные организации 392

Приложение 5
Глоссарий 394

Предметный указатель 404

ГЛАВА 1

Вопросы администрирования

Изо дня в день администраторы UNIX сталкиваются со множеством мелких и крупных задач. В этой главе описаны те из них, решать которые вы вполне можете по отработанной схеме.

Прежде всего речь пойдет об автоматизации рутинных занятий, жизненно важной для системного администратора. Его работа протекает в довольно напряженной обстановке – если скучные задачи вновь и вновь приходится выполнять вручную, возникает понятное раздражение. В данной главе вы найдете примеры автоматизации повседневных действий, благодаря которой будете тратить на них вполовину меньше времени. Не забудьте также установить предупреждающие программы, чтобы первым узнавать о том, что выявлены какие-либо неполадки.

Много беспокойства причиняет администраторам и настройка системы. Следует быть осторожным: начав возиться с одним, вы рискуете упустить другое. Некоторые из приведенных примеров в той или иной мере относятся к настройке ОС. Перед тем как вносить изменения, попытайтесь оценить последствия и не забывайте: что подходит для одной системы, может вызвать проблемы на другой.

Итак, меняя системные настройки, вы должны всегда (а не только во время чтения данной главы!) задавать себе следующие вопросы:

- *нужно ли тестировать данную процедуру?* Если есть возможность проверить, как отразятся изменения на аналогичной системе, не упускайте шанс! Потратьте на это немного времени, иначе можете поплатиться за свою лень позже – например, в два часа ночи на следующий день. Если вы не располагаете тестовой системой для подобных экспериментов, воспользуйтесь рабочей. Руководство поддержит вас в этом начинании, если вы сможете четко обосновать свои действия;
- *если ли у вас альтернативный план?* Всегда прорабатывайте запасной вариант на случай сбоя в работе системы, независимо от того, сколь простым будет изменение. Надо знать наперед, что делать, если события станут развиваться по наихудшему сценарию;

- *как вносимые изменения повлияют на работу системы?* Следует всегда мыслить в перспективе. Иногда небольшое изменение в одном приложении может затормозить работу всей ОС или сети. В этом плане печальной известностью пользуется установка сторонних приложений;
- *как изменится работа сети?* Ваши действия могут отразиться на всей сети. Подобный эффект вызывают запуск NIS/YP, экспортирование файловых систем NFS и редактирование символьных ссылок. Сперва убедитесь, что вы не заденете другие системы без ведома их администраторов;
- *как скажется изменение на работе пользователей?* Это один из наиболее важных вопросов, которые вы должны задать себе. Если будут затронуты интересы пользователей, ваш телефон будет звонить не переставая;
- *следует ли кому-либо рассказывать о ваших планах?* Я подробно остановлюсь на данном вопросе в главе 9, пока же отмечу следующее: обмен информацией с другими сотрудниками – ключ к получению мощной поддержки. Чем больше людей знает о вашей работе, тем больше они ценят ваши способности;
- *каков наихудший сценарий развития событий?* Этот вопрос связан с идеей проработки альтернативного плана. При внесении изменений следует всегда предвидеть самый неудачный исход событий и прогнозировать свои действия на этот случай. Помните, что при работе с компьютерами всегда лучше перестраховаться;
- *правильное ли время выбрано для внесения изменений?* На этот вопрос можно дать несколько ответов. Большинство администраторов отмахиваются: «Какая разница, просто это нужно сделать!». Лучше всего приступить к делу вечером или в выходные: если что-то пойдет не так, вы сможете не спеша устранить ошибки. Но из этого правила есть одно исключение: круглосуточно работающие системы. Изменения, на которые не уйдет много времени и в тестировании которых должны участвовать пользователи, лучше всего вносить рано утром. Если несколько сотрудников, первыми приступивших к работе, обнаружат серьезные огрехи, вы можете быстро вернуть систему в первоначальное состояние. Пользователям нравится, когда администратор приходит на работу раньше них: они проникаются чувством, что вы сможете справиться с любыми трудностями;
- *существует ли лучший способ?* Большинство задач в UNIX можно выполнить несколькими способами. Попытайтесь определить, какой из них удобнее. Не бойтесь свежих идей. Даже если пользователь прибегает к вам и просит помочь ему незамедлительно, проанализируйте ситуацию, прежде чем бросаться к компьютеру;
- *если ли возможность приступить к действию сейчас?* Не следует проявлять поспешность. Пытайтесь распределить время так, чтобы вы могли полностью завершить работу. Быстро поставленная «заплатка» может спасти ситуацию, но не оставляйте проблему решенной наполовину. Если вынуждать пользователей слишком долго мириться с временными исправлениями, сослуживцы начнут терять веру в вас.

Редактируя системные настройки, старайтесь держать в памяти перечисленные темы. Некоторые администраторы со временем становятся излишне самоуверенными, полагая, что они смогут преодолеть любые сложности, и терпят крах из-за того, что не задали себе один из простейших вопросов. Не пожалейте времени на их решение.

1.1. Сбор информации о системе

Необходимо собирать как можно больше информации о каждом новом компьютере.

Пример

Версии системы: AT&T, BSD.
Вы должны собрать следующую информацию:

- имя каждого компьютера;

```
% hostname
```

- псевдонимы;

```
% grep `hostname` /etc/hosts | awk '{ print $3}'
```

- сетевые адреса;

```
% grep `hostname` /etc/hosts | awk '{ print $1}'
```

- идентификатор;

```
% hostid
```

- серийный номер системы (обычно находится на задней стенке корпуса);
- производитель (обычно указан на передней панели системного блока);
- модель компьютера (обычно указана на передней панели);
- тип процессора:

```
% uname -a
```

- архитектура программ:

```
% uname -a
```

- архитектура ядра:

```
% uname -a
```

- объем основной памяти (выводится в момент загрузки):

```
% dmesg
```

- название операционной системы:

```
% uname -a
```

- версия операционной системы:

```
% uname -a
```

- версия ядра:

```
% uname -a
```

- конфигурация дисков:

```
% df
```

- дополнительная информация:
  - список смонтированных файловых систем NFS;
  - конфигурация NIS/YP;
  - список установленных пакетов системы;
  - перечень инсталлированных обновлений;
  - характеристики дисковых устройств;
  - регистрационные номера и коды установленного ПО;
  - символьные ссылки на каталоги;
  - настройки принтеров.

Зачем это нужно?

По мере расширения сети и увеличения числа поддерживаемых компьютеров полезно дополнять список настроек всех имеющихся систем.

Практический опыт

Даже самых опытных администраторов UNIX постигнет горькое разочарование, если, пытаясь заполнить квитанцию на замену жесткого диска или памяти, они не смогут ответить на простейшие вопросы типа «Каков объем диска?» или «Какие SIMM стояли в системе?». Когда приобретается новый компьютер, на сбор подобной информации уходит всего несколько секунд. Не теряйте ваши записи.

Если системный диск прикажет долго жить, но у вас останутся копия настроек и хорошая резервная копия, вы сможете быстро подобрать новый диск с теми же характеристиками, восстановить данные и «реанимировать» компьютер.

Другие источники информации

Страницы руководства:

```
df, hostname, hostid, uname
```

Страница SysInfo в Internet – http://www.MagniCorp.com/sysinfo/.

1.2. Копируйте ключевые файлы!

Одна из последних операций, которую необходимо выполнить перед переводом системы в рабочий режим, – резервное копирование ключевых файлов системы.

Пример

Версии системы: AT&T, BSD.

Ядро – файл, который необходимо скопировать в первую очередь. Часто называется `/kernel`, `/unix` или `/vmunix`.

Файл паролей следует копировать на случай его компрометации. Называется `/etc/passwd`.

Файл групп копируется по тем же соображениям, что и файл паролей. Называется `/etc/group`.

Таблица имен узлов копируется на случай повреждения или удаления из нее записей, которые могут понадобиться в будущем. Называется `/etc/hosts`.

Таблица файловых систем копируется для восстановления параметров файловых систем. Часто называется `/etc/fstab` или `/etc/vfstab`.

Файлы настройки sendmail `/usr/lib/sendmail.cf`, `/usr/lib/sendmail.fc` и `/usr/lib/sendmail.mc` копируются для быстрого восстановления работы sendmail после сбоя.

Файл настройки inetd (`/etc/inetd.conf`) часто содержит ошибки или подменяется хакерами.

Файлы настроек TTY. Если к системе подключены специализированные устройства, следует сохранять их параметры, которые обычно записаны в файлах `/etc/inittab`, `/etc/ttytab` и `/etc/ttys`.

Командные файлы начальной загрузки. Все командные файлы, выполняющиеся при запуске системы, необходимо сохранять в другом месте. Обычно они находятся в каталогах `/etc/init.d` или `/etc/rc#.d`.

Зачем это нужно?

Проще восстановить с резервного диска или ленты только нужные файлы, объем которых невелик. Их содержимое также следует периодически проверять из соображений безопасности.

Практический опыт

Отведите 5–10 Мб системного пространства для жизненно важных данных. Если в системе только один диск, копии и исходные файлы должны располагаться в разных разделах. Если все исходные материалы находятся в корневом разделе (`/`), сохраните копии в `/usr/`. Убедитесь, что для всех исходных файлов и соответствующих копий определены идентичные права доступа и владельцы.

В некоторых версиях UNIX корневой раздел имеет небольшой объем. Иногда пользователи, видя, что он заполнен на 95%, решают, что все проблемы связаны с нехваткой дискового пространства в нем, и удаляют любые большие файлы, попадающиеся им на глаза, в том числе самый объемный файл в корневом каталоге – ядро системы. На подобные дерзости отваживаются немногие, но если уж вы столкнулись с такой ситуацией, то, располагая копией ядра, сможете быстро вернуть систему в рабочее состояние.

Кроме того, своевременно заархивировав перечисленные материалы, вы сможете сравнивать системные файлы с их копиями, проверяя тем самым, не были ли они скомпрометированы.

1.3. Запуск программы в последний день месяца

Данная команда определяет, наступил ли последний день текущего месяца, и, если это так, позволяет запустить нужный командный файл или программу.

Пример 1: использование командной оболочки

Версии системы: AT&T, BSD.

Командные оболочки: bsh, bash, ksh.

Синтаксис:

```
TZ={GMT|PST|EDT|…}-24 date +%d
```

`TZ` – это имя переменной часового пояса в системе UNIX. Прибавим к текущей дате один день (%d). Если сегодня последний день месяца (например, 31-е число), то завтра наступит первый день следующего месяца:

```
$ TZ=PST-24 date +%d
```

Теперь можно написать командный файл, который будет запускать программу, если результат сложения равен `1`:

```
#! /bin/sh

FILE=`runme`
if test `TZ=PST-24 date +%d` = 1; then
   $FILE
fi
```

Строка 1: выбор используемой командной оболочки (в данном случае – Bourne shell).

Строка 3: переменной присваивается имя запускаемой программы.

Строка 4: если переменная `TZ` получит значение `1`, сегодня последний день месяца.

Строка 5: если это утверждение верно, следует запустить программу, имя которой записано в переменной `FILE`.

Строка 6: если значение `TZ` не равно `1`, работа завершается.

Пример 2: использование Perl

Версии системы: AT&T, BSD.

Командные оболочки: Perl.

Ниже представлен стандартный подход, позволяющий добиться тех же результатов, что и в предыдущем примере, с помощью программы на языке Perl. Чтобы не запускать новый интерпретатор Perl, можно включить приведенный код в более крупную программу.

```
#!/usr/bin/perl

use POSIX;

@THE_DATE = localtime (time);
++$THE_DATE[3];
if ((localtime (POSIX::mktime (@THE_DATE)))[3] == 1) {
        exit 1;
}
exit 0;
```

Строка 1: показано, что файл представляет собой программу Perl.

Строка 3: используется модуль POSIX.

Строка 5: определение текущей даты и ее запись в массив `THE_DATE`.

Строка 6: к текущей дате прибавляется один день.

Строка 7: дата в массиве `THE_DATE` приводится к стандартному формату с помощью функции `mktime`. Проверяется, будет ли полученный день первым днем месяца.

Строка 8: если это первый день месяца, осуществляется выход с кодом завершения `1`.

Строка 10: в противном случае возвращается код `0`.

Зачем это нужно?

Выполнение некоторых программ (например, резервного копирования, фильтрации log-файлов и т.д.) бывает приурочено к последнему дню месяца. На первый взгляд кажется неуместным создавать для этого командные файлы. Почему бы не добавить еще одну строку в планировщик? Все дело в том, что cron может запускать программы только в заданный день (допустим, вы указываете 1-е или 31-е число), но не в состоянии определить наступление последнего дня месяца, поэтому придется прибегать к другим средствам.

Практический опыт

Этот нехитрый прием может быть полезен во многих случаях. Приведенный код легко подключить к существующим командным файлам, обрабатывающим статистику загрузки системы, использования дискового пространства, доступа пользователей и создающим ежемесячные отчеты разного рода. Приятно слышать, как босс или пользователи хвалят вас за то, что вы аккуратно предоставляете отчеты в конце каждого месяца – и не догадываются, насколько просто это делается!

Другие источники информации

Страницы руководства: `localtime`, `tzset`, `tzfile`, `crontab`.

1.4. Отключение ненужных демонов

Отключите все ненужные и неиспользуемые демоны. Для этого необходимо отредактировать файл `/etc/inetd.conf`, а также файлы или каталоги `rc`.

Пример 1: отключение демонов в inetd.conf

Версии системы: AT&T, BSD.

Отредактируйте файл `/etc/inetd.conf`, закомментировав в нем строки с ненужными демонами:

```
# vi /etc/inetd.conf

#
# Configuration file for inetd(1M). See inetd.conf(4).
#
# To re-configure the running inetd process, edit this file,
# then send the inetd process a SIGHUP.      kill -HUP [pID]
#
#talk     dgram    udp   wait     root     /usr/sbin/in.talkd    in.talkd
#ntalk    dgram    udp   wait     root     /usr/sbin/in.ntalkd   in.ntalkd
#uucp     stream   tcp   nowait   root     /usr/sbin/in.uucpd    in.uucpd
#
#finger   stream   tcp   nowait   nobody   /usr/sbin/in.fingerd  in.fingerd
#tftp     dgram    udp   wait     root     /usr/sbin/in.tftpd    in.tftpd
#bootps   dgram    udp   wait     root     /usr/sbin/in.bootpd   in.bootpd
#talk     dgram    udp   wait     root     /usr/sbin/tcpd        in.talkd
```

После изменения файла `/etc/inetd.conf` и отключения в нем ненужных демонов найдите идентификатор процесса (process ID, PID) демона `inetd` и перезапустите его командой `kill -HUP`.

Версия системы: AT&T.

```
# ps -ef | grep inetd
root    124     1       ?        S  30:57    /usr/sbin/inetd -s
ugu     10377   10378   pts/4    S  0:00     grep inetd

# kill -HUP 124
```

Версия системы: BSD.

```
# ps -ax | grep inetd
124     ?       S     30:57    /usr/sbin/inetd -s
10377   pts/4   S     0:00     grep inetd

# kill  -HUP  124
```

Если включена регистрация, вы можете убедиться в том, что демон `inetd` был перезапущен, проверив системные log-файлы (`/var/adm/messages` или `/var/adm/SYSLOG`). При последующем выводе таблицы процессов вы обнаружите, что идентификатор процесса, как и предполагалось, не изменился. Выполнение команды `kill -HUP` не приводит к завершению процесса – ему просто посылается сигнал разрыва соединения. Многие демоны, в том числе и `inetd`, перехватывают данный сигнал и перечитывают файл конфигурации, продолжая свою работу.

Если процесс не перезапустился и демоны по-прежнему доступны, можно вручную завершить работу `inetd` и запустить его снова (хотя это и не рекомендуется). По возможности следует использовать одну команду:

```
#kill 124; /usr/etc/inetd
```

Затем необходимо снова проверить таблицу процессов командой `ps -ef` или `ps -ax`, чтобы убедиться, что драйвер запущен. На этот раз идентификатор его процесса будет другим.

Пример 2: отключение демонов в каталогах rc

Версия системы: BSD.

Демоны и приложения могут также запускаться из файлов и каталогов `rc`. Это командные файлы, которые выполняют функции по обслуживанию файловой системы и запускают системные демоны UNIX.

Вносить изменения в эту область опасно, поэтому вы должны четко осознавать, что вы делаете. В результате изъятия демона или процесса из файлов система может «зависнуть» во время загрузки. Если это произойдет и вы не сможете запустить ОС даже в однопользовательском режиме, придется использовать дискеты или компакт-диск. Итак, перед изменением любых rc-файлов рекомендуется выполнить резервное копирование.

После внесения нужных изменений в rc-файлы необходимо перезагрузить компьютер и убедиться, что внесенные изменения вступили в силу. Некоторые администраторы завершают работу связанных с удаленными демонами процессов и перезапускают систему через несколько часов, когда пользователи уйдут на обед или домой. Это вполне допустимо, если вы не слишком рассеянны!

Мне часто приходилось видеть, как администратор отвлекался от дел и забывал о том, что в rc-файлы были внесены изменения. Что же бывает в таких случаях? Увидите сами. Через несколько дней или недель система по той или иной причине будет перезагружена или в ней возникнет сбой. Когда начнется процесс начальной загрузки, изменения, которые вы внесли и не протестировали, помешают запуску. И вот итог: в вашем списке нерешенных проблем появится еще одна. Хуже всего, если вы отсутствуете – тогда другому администратору придется действовать наугад, не зная, какие изменения вы внесли.

Зачем это нужно?

В большинстве случаев готовые компьютеры поставляются с необходимым программным обеспечением. При установке системы с нуля по умолчанию будет инсталлировано намного больше приложений, чем вам нужно. Поскольку в мире намного больше систем, чем администраторов, производители пытаются облегчить жизнь пользователя, устанавливая самые разные компоненты.

Две основные причины, по которым стоит удалять ненужное программное обеспечение, – повышение безопасности и производительности. Отключение ненужных демонов обеспечивает дополнительную защиту, уменьшает число системных функций, которые необходимо администрировать, и высвобождает память и процессорное время, которые «пожираются» ненужными операциями.

Если вы не планируете использовать определенные службы, ликвидируйте их. К каждой системе надо подходить индивидуально. Если нет особой необходимости, системы, требующие более надежной защиты, не должны отвечать на запросы `tftp`, `talkd` или `fingerd` и даже `ftp` или `telnet`. Отключите соответствующие демоны. Пользователям, которым не понадобится доступ к сети, ни к чему `bind`, `YP/NIS`, `bootpd`, `sendmail`, `routed` и другие сетевые службы – будет разумно от них отказаться.

Практический опыт

В идеале программисты не должны выполнять компиляцию программ на файловых серверах, занимая все процессорное время. Но в реальности ничто их от этого не удержит, кроме отключения `telnetd`, `rshd` и `rlogind`. К сожалению, впоследствии можно будет администрировать систему только локально. Что же, всем нам приходится идти на жертвы!.. Из соображений безопасности аналогичные меры следует предпринять и на брандмауэрах, но там отключается намного больше демонов.

Другие источники информации

Страницы руководства: `inetd`, `inetd.conf`, `rc`.

1.5. Перезапуск демонов

Если демон часто завершает свою работу в аварийном порядке, приведенная ниже процедура позволяет следить за ним и при необходимости перезапускать.

Пример

Версия системы: AT&T.

Оболочки: csh, ksh.

Представленный командный файл оболочки csh будет следить за таблицей процессов, проверять наличие в ней определенных демонов и, если нужно, перезапускать их.

```
#!/bin/csh

foreach DAEMON ( MonitorSuLog.pl MonitorLogins.pl DiskHogs.pl )
  ps -e | fgrep "$DAEMON:t" | cut -c1-8 > /dev/null
  if ( $status > 0 ) then
```

```
      echo "Restarting $daemon"
      date
      $DAEMON &
    endif
  end
```

Строка 1: выбор используемой командной оболочки.

Строка 3: обработка каждого из перечисленных демонов.

Строка 4: поиск определенного демона в таблице процессов и анализ результата. Чтобы данные не выводились на экран, вывод команды `cut` перенаправляется в `/dev/null`.

Строка 5: если демон был найден в таблице процессов, значение переменной статуса будет больше `0`; выполнятся строки 6–8. Если демон не существует, произойдет переход к строке 10.

Строка 6: направление сообщения о перезагрузке демона на стандартный вывод.

Строка 7: вывод текущих даты и времени.

Строка 8: запуск демона.

Строка 9: завершение проверки.

Строка 10: если требуется проверка следующего демона, остается выбрать его и проверить, перейдя на строку 4; если же список проверяемых демонов закончился, – выйти из программы.

Чтобы постоянно следить за состоянием демонов с помощью данной программы, поместите соответствующую запись в crontab. Для редактирования crontab выполните следующие команды:

```
# crontab -l > /tmp/crontab.txt
# vi /tmp/crontab.txt
```

Чтобы программа проверки запускалась каждые десять минут, внесите в файл `crontab.txt` запись:

```
0,10,20,30,40,50 * * * * /usr/local/bin/monitor_daemons
```

Если включена регистрация, вывод программы cron регистрируется и, возможно, отправляется пользователю, под именем которого запущена задача cron. Если, по вашему мнению, аварийное завершение работы демонов не будет регулярным, лучше использовать приведенную выше запись, чтобы знать, насколько часто оно происходит. Может оказаться, что демоны «умирают» постоянно; в таком случае отредактируйте запись и перенаправьте вывод в `/dev/null`:

```
0,10,20,30,40,50 * * * * /usr/local/bin/monitor_daemons > /dev/null 2>&1
```

После изменения файла `crontab.txt` импортируйте записи из него в cron и удалите файл по соображениям безопасности.

```
# /bin/crontab /tmp/crontab.txt
# rm /tmp/crontab.txt
```

Зачем это нужно?

Представленная процедура – не панацея, а всего лишь временное решение проблемы. Правильно написанные демоны не должны «умирать», но иногда это случается. Если система не обеспечена технической поддержкой, ваш удел – самостоятельно выходить из положения. К тому же вы хорошо знаете, как долго придется добираться до нужного сотрудника службы поддержки, а время дорого...

Практический опыт

Практика показывает, что демоны иногда завершают работу в аварийном порядке по неизвестной причине. Никого не радует, если демон DNS «отдает концы» дважды в месяц. Если проблема возникает редко, найти ее источник совсем не просто.

Предложенный вашему вниманию прием позволяет быстро устранить неполадки и, как минимум, предотвратит выход из строя вашего пейджера. Также эту методику удобно использовать для мониторинга процесса – вместо перезагрузки демона можно начинать новый процесс, если старый «падает». Иногда проверка или обработка данных допускается только после завершения какой-либо другой программы, поэтому слежение за определенными процессами позволяет автоматизировать работу.

Другие источники информации

Страницы руководства: `cron`, `crontab`, `ps`, `test`.

1.6. Применение fuser вместо ps

Альтернативный способ получения идентификатора процесса (PID) – использование команды `fuser`. Она надежнее `ps`, а иногда и срабатывает быстрее.

Пример

Версии системы: AT&T, BSD.

Оболочки: все.

Синтаксис:

/usr/sbin/fuser *файлы*

Образец:

/usr/sbin/fuser /bin/csh

Команда `fuser` выводит идентификаторы всех процессов, запущенных из файла с заданным именем. Если ей передается имя каталога, то выводятся имена всех процессов, открывших один или несколько файлов в этом каталоге на чтение. Чтобы команда выполнялась корректно, необходимо полностью задать имя файла; в противном случае на стандартный вывод будет направлена подсказка, в которой приведен верный синтаксис.

Данная команда имеет одну особенность – для работы с ней необходимо иметь доступ на чтение /dev/kmem и /dev/mem. Это связано с тем, что fuser напрямую обращается к системной памяти через указанные устройства.

```
# fuser /bin/csh
/bin/csh:    1485t  1106t
```

Буква t в конце идентификаторов процесса означает, что у каждого из них имеется собственный текстовый сегмент исполняемого файла.

Для завершения работы процесса в fuser можно задать параметр (-k). Например, чтобы уничтожить все процессы csh, выполните следующую команду:

```
# fuser -k /bin/csh
/bin/csh:    1485t  1106t
```

Она заменяет целый набор команд, которые вы, возможно, не раз выполняете в течение дня:

```
# ps -ef | grep csh

root  1484  1485    1  17:54:02  pts/1    0:00 /bin/csh
root  1116  1117    1  17:54:16  pts/1    0:00 grep csh
root  1090  1091    0    Aug 09  pts/1    0:00 /bin/csh

# kill  1484  1090
```

Если с определенным процессом связано несколько других, можно легко написать программу для завершения работы приложения и всех связанных с ним демонов.

Допустим, файл приложения называется bsr и находится в каталоге /sbin. Оно зависит от нескольких демонов – bsrqqd, bsrexecd и bsrojbd, которые запускаются независимо. Для решения вышеназванной задачи можно быстро набросать командный файл:

```
#! /bin/sh

fuser -k /sbin/bsr
fuser -k /sbin/bsrqqd
fuser -k /sbin/bsrexecd
fuser -k /sbin/bsrojbd
```

Строка 1: выбор используемой оболочки.

Строки 3–6: поиск и уничтожение процесса.

Зачем это нужно?

Работа с командой fuser проста и эффективна. Поиск нужного процесса и всех прочих, связанных с ним, на большом сервере может отнять довольно много времени. Данная команда позволяет быстро найти информацию о процессе и при необходимости завершить его работу.

Практический опыт

Я привык использовать `fuser` для уничтожения определенных процессов и написал несколько командных файлов, подобных приведенному выше; они предназначены для завершения работы различных пользовательских приложений, сеансов X Window, командных оболочек и других вещей. Если вы можете выполнять команды в удаленной системе, достаточно запустить в ней оболочку и быстро удалить процессы. Пользователям кажется, что вы сделали это, даже не входя в ОС: чудеса, да и только!

Другие источники информации

Страницы руководства: `fuser`, `kill`, `ps`.

1.7. Изменение размера раздела подкачки «на лету»

Если после установки UNIX вы вдруг обнаружите, что указали недостаточный объем раздела подкачки, эту оплошность легко исправить. Создайте файл подкачки и сделайте его активным.

Пример

Версии системы: AT&T, BSD.

Синтаксис:

mkfile size[m] *имя_файла*

Основная операция выполняется при помощи команды `mkfile`. Ее просто использовать, поэтому увеличение размера файла подкачки не представляет проблем. Определив диск, который меньше используется (чтобы сбалансировать производительность ввода/вывода), можно приступить к работе.

```
# mkfile 200m /disk2/swap_200MB
```

Таким образом создается двухсотмегабайтный файл подкачки на разделе `/disk2`. Если в вашей системе отсутствует `mkfile`, можно воспользоваться командой `dd`:

```
# dd if=/dev/zero of=/disk2/swap_200MB bs=1024k count=200
```

Теперь полученный файл нужно сделать активным. В зависимости от установленной у вас версии UNIX выполните одну из следующих команд:

- версии Irix и Solaris

  ```
  # swap -a /disk2/swap_200MB
  ```

- версии HP-UX, SunOS

  ```
  # swapon -a /disk2/swap_200MB
  ```

- Linux

  ```
  # swapon /disk2/swap_200MB
  ```

Полученный файл подкачки будет использоваться, пока система не завершит работу. После перезагрузки ОС он останется на месте, но перестанет быть активным, и для его подключения необходимо будет снова выполнить команду `swap` или `swapon`.

Чтобы файл автоматически активировался при загрузке системы, необходимо внести соответствующую ссылку в таблицу файловых систем. Файл таблицы называется `/etc/fstab` или `/etc/vfstab`. Добавьте в него следующую строку:

```
/disk2/swap_200MB    swap     swap     rw 0    0
```

Формат записей в таблице файловых систем варьируется в разных версиях UNIX, поэтому уточните его в документации вашей ОС.

Зачем это нужно?

Раньше в качестве системного выбирался самый быстрый из доступных дисков. Но благодаря росту производительности шин SCSI сегодня необязательно создавать область подкачки только на системном диске. Его можно разместить в любом месте системы, но при этом желательно сбалансировать частоту обращения к разным дискам, чтобы добиться максимальной эффективности работы.

Практический опыт

Иногда пользователи жалуются, что запускаемые приложения занимают слишком много системных ресурсов. Если причиной тому – нехватка пространства подкачки, сообщите, что вы намерены немедленно устранить проблему. Увеличив размер файла подкачки «на лету» за то время, пока клиент беседует с вами по телефону, вы создадите у него впечатление, что дополнительные ресурсы появились «из воздуха». Этот простой трюк поднимет ваш авторитет в глазах пользователя.

Другие источники информации

Страницы руководства: `fstab`, `mkfile`, `swap`, `swapon`, `vfstab`.

1.8. Фоновые процессы и nohup

Если вам нужно, чтобы процесс продолжал работать после отключения системы, используйте команду `nohup`. Она нечувствительна к «зависаниям» терминалов, выходу пользователя из системы или из оболочки.

Пример 1: обычный вызов nohup

Версии системы: AT&T, BSD.

Оболочки: все.

Синтаксис:

nohup *команда аргументы*