**КОМПЬЮТЕРНОЕ УПРАВЛЕНИЕ ПРОИЗВОДСТВОМ**

**Проект учета пользовательских счетов для интернет-провайдеров на базе OS FreeBSD с применением программы «Billing ISP»**

**Курсовой проект выполнил студент Иванов М.Н.**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ МОРСКОЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**САНКТ-ПЕТЕРБУРГ**

**1999 г.**

**Предпроектное обследование объекта автоматизации.**

**Описание предметной области решаемой задачи.**

В настоящие время многие (ISP) интернет сервис провайдеров решают проблему учета пользовательских счетов, и проблему контроля трафика путем написания новых приложений, что зачастую приводит к частым сбоям данного ПО, и соответственно не оправдывает вложенные в него средства. Кроме того, такие продукты не способны обслуживать большое число пользовательских счетов и представлять всю обработанную информацию в компактной, удобной для работы и анализа форме. Большинство предлагаемых в настоящее время систем биллинга, т.е. систем учета отработанного "он-лайнового" времени пользователями Интернет-провайдера (ISP) основано, как правило, на анализе стандартных лог-файлов таких опирационных систем, как SCO Unix, SunOS, HpOS, AIX, IRIX раз в сутки, в неделю, в месяц и т.д. В то время как предлагаемая система биллинга основаная принципиально другой идее, заключающейся в контроле за каждой сессией пользователя в отдельности в реальном масштабе времени. Что позволяет значительно снизить время на обработку биллинг-инженером статистики работы каждого пользователя или группы, снизить трудоемкость занесения платежей пользователей на лицевые счета (базу данных этой программы) и соответственно позволяет провайдерам уменьшить количество обслуживающего персоонала, что непосредственно отражается на себестоимости предоставляемых услуг.

**Функции предметной области, реализуемой задачи.**

Основные функциональные преимущества такого подхода заключается в том, что отслеживаются и корректно отрабатываются, фиксируются и генерируются в отчеты такие данные как:

Регистрирование соединения любой продолжительности с точностью, равной одному кванту времени (например, 5 секунд). Квант времени задается системным администратором;

Исчерпывание средств на лицевом счете пользователя, если он находится в данный момент в режиме "он-лайн", и принудительное его отключение (эта ситуация очень актуальна, когда ISP предоставляет новому клиенту "тестовый час");

Возможность задания для каждого пользователя или для групп пользователей гибких прайс-листов с указанием цены в у.е. за 1 час "он-лайнового" времени в зависимости от времени суток и дня недели. Например, имеется ISP, у которого стоимость "дневного" (с 9 утра до 6 вечера) Интернета - $1, а "вечернего" (с 6 вечера до 9 утра) - $0,6. Пользователь звонит без четверти 6-ть вечера и работает 15 минут по тарифу $1 за час и 30 минут по тарифу $0,6 за час (всего 45 минут), а с его лицевого счета, соответственно, снимается сумма $0,25+$0,3, т.е. $0,55;

Переход пользователя с одного прайс-листа в другой, при исчерпывании средств на лицевом счете в первой схеме и при авансовом платеже по другому прайс-листу без отключения пользователя. Реально это означает ситуацию, когда работал пользователь по одному прайс-листу, и когда у него стали заканчиваться средства на лицевом счете, он сделал новый взнос, но уже по другому (например, "льготному") прайс-листу. Затем он доработал свои часы по "старому" прайс-листу и спокойно начал работу по "новому" прайсу-листу;

Удаленным пользователям предоставляется удобный www-интерфейс при помощи которого они могут полностью контролировать свою работу в Интернете вплоть до каждого модемного звонка на узел ISP, в том числе, разумеется, они могут в любое время посмотреть размер своего лицевого счета на текущий момент (момент генерирования web отчета из базы данных программы «Billing ISP».

Cистемному администратору (биллинг-инженеру) предоставляется достаточно простой в освоении стандартный для Unix систем режим командной строки, открытость, простота и возможность "затачивания" системы под свои конкретные особенности.

**Основные качества и особенности предлагаемой системы биллинга**

Высокая точность подсчета "он-лайнового времени" отработанного пользователями;

Простая интеграция предлагаемой системы в существующую систему аутентификации DialUp-пользователей провайдера (забегая вперед, хочется отметить, что в настоящий момент наша система поддерживает только схемы TACACS+ и pppd);

Возможность развертывания предлагаемой системы параллельно с уже существующей системой биллинга провайдера для тестирования и отладки с целью окончательного запуска в эксплуатацию;

Автоматическое получение ежедневных (еженедельных, ежемесячных) отчетов отработанных часов и их стоимости по различным группам клиентов (например, "основной тариф", "льготный тариф", "бартер", "халява");

Возможность подключения SQL-сервера для генерации более гибкой системы статистик при помощи SQL-запросов;

Минимальные требования к аппаратным ресурсам сервера биллинга (предлагаемая система может функционировать даже без SQL-сервера). Однако, www-сервер (Apache) все-таки следует установить для того, чтобы удаленные пользователи имели доступ к своей статистики через привычный www-интерфейс;

Своевременное оповещение пользователя и системного администратора через e-mail о том, что размер лицевого счета пользователя приближается к концу;

Гибкое ведение прайс-листов по группам пользователей, их быстрая и несложная модификация (например, установка "праздничного тарифа" когда "народное гулянье" выпадает на середину недели);

**Возможность удаленного администрирования пользователей**

P.S. В выше изложенном тексте применяются некоторые профессиональные термины относящиеся к различным клонам Unix систем, а также к общесистемному профессиональному ПО такие как:( демон, Apache, pppd, домашний каталог пользователя, ядро, сервер биллинга, лог-файл, пользователь, www-интерфейс, SQL и т.д.), которые нуждаются в дополнительных комментариях. Описания данных терминов можно сравнить с полноценным книжным изданием, вследствие чего оно здесь не присутствует. Короткие пояснения можно получить у составителя данной курсовой работы.

**Организационно-экономическая сущность задачи.**

Использование данного комплекса программного обеспечения позволит небольшому интерне-провайдеру значительно снизить себестоимость услуг по коммутационному подключению своих абонентов, а также предоставить им удобную систему тарифных планов, тестовых подключений, систему анализа и мониторинга собственных счетов и статистики подключений, что естественно способствует росту клиентской базы и увеличению прибыли нашего ISP.

Также стоит отметить, что основная экономия средств происходит за счет использования абсолютно бесплатного программного обеспечения, а именно операционной системы FreeBSD и системы учета «Billing ISP», которые распространяются с открытым исходном кодом по лицензии GNU и не имеют ограничений на число копий и т.д. Наличие исходного кода данных продуктов дает возможность адаптации их под уже существующие бухгалтерские программы и системы учета. Также значительная экономия происходит за счет небольшого числа технического персонала обслуживающего данную систему. По персоналу можно отметить, что управлять данной системой могут специалисты низкой квалификации, т.е. именно система «BillinISP» не требует углубленного знания Unix подобных опирационных систем, сетевых технологий и сложного сетевого оборудования такого как CISCO, из чего следует, что з/п такого работника будет относительно не велика. В тоже время средняя з/п сертифицированного специалиста колеблется от 500$-1500$. Естественно, что для поддержки системы в актуальном состоянии такие работники необходимы, но за счет применения данной схемы их число можно значительно уменьшить без потери качества обслуживания клиентов.

Для справки некоторые коммерческие операционные системы могут достигать стоимости 5000$, при это с ограниченным числом установок и с ограниченным числом пользователей плюс к этому около 2000-3000$ может стоить какая ни будь посредственная биллинговая система.

Из всего выше сказанного видно, что при применении данного программного обеспечения интернет-провайдеры получают существенную выгоду.

**Разработка информационного обеспечения задачи.**

**Описание входной информации.**

Основным входным документом, с которым взаимодействует рассматриваемая биллинговая система это копия платежного поручения пользователя ISP. При регистрации клиента провайдер предлагает ему выбрать установленный тариф (прайс-лист) на услуги коммутируемого подключения на определенное (тарифом) время, по которому клиент будет перечислять установленную денежную сумму. После регистрации и предоплаты услуг нашего ISP на счет (в домашний каталог пользователя копируются определенный набор файлов см. ниже) пользователя поступает определенная прайс-листом сумма, которая в процессе работы пользователя в интернете уменьшается через определенный тарифом (прайс-листом) квант на определенную сумму. Соответственно все, что делает одна часть программы биллинга другая ее часть фиксирует все в базе данных и в том же личном каталоге пользователя в отдельный лог-файл на основании, которого и генерируются отчеты пользователю и администратору. Также к входной информации можно отнести: отчисления с пользовательского счета, проведенное время в интернете, время подключения (регистрации в системе). На основании изложенной выше информации вспомогательный модуль программы билинга может предоставлять подробную статистикуего счета пользователя, как администратору так и самому клиенту ISP.

**Файл задания прайс-листа (тарифной схемы) - account.conf**

#

#

# Пример файла account.conf (.account.conf). # Лидирующие пробелы, пустые строки,

# строки, начинающиеся с символа "#" игнорируются.

# Обрабатываются лишь строки, начинающиеся с ключевого

# слова "price:". Количество строк "price" неограченно.

# Формат прайс-листа (тарифной схемы) -

# price: День\_недели, час\_начала-час\_окончания $стоимость\_в\_у.е.

# что соответствует промежутку времени

# час\_начала:00-час\_окончания:59

# Если при указании временные диапазоны пересекаются, то стоимость

# часа принимается последняя.

# Основная тарифная схема. # Цена: в будние дни с 10:00-18:00 - $1

# в остальное время - $0,6

#

#

comment: Поле\_comment\_будет\_автоматически\_выводиться\_при\_запуске

comment: демома\_в\_режиме\_получения\_сведений\_о\_размере\_лицевого

comment: счета\_пользователя.\_Удобно\_использовать\_для\_задания\_комментарий

comment: к\_прайс\_листу.

commenth:

commenth: Поле\_commenth\_выводится,\_если\_размер\_лицевого\_счета

commenth: выдается\_в\_html\_формате.\_Пробелы\_должны

commenth: заменяться\_на\_подчеркивания.\_Количество\_строк\_comment\_и

commenth: commenth\_не\_ограничено,\_однако\_суммарная\_длина\_каждой\_не

commenth: не\_должна\_превышать\_1000\_символов.

price: Monday, 0-9 $0.6

price: Monday, 10-17 $1

price: Monday, 18-23 $0,6

price: Tuesday, 0-9 $0.6

price: Tuesday, 10-17 $1

price: Tuesday, 18-23 $0,6

price: Wednesday, 0-9 $0.6

price: Wednesday, 10-17 $1

price: Wednesday, 18-23 $0,6

price: Thursday, 0-9 $0.6

price: Thursday, 10-17 $1

price: Thursday, 18-23 $0,6

price: Friday, 0-9 $0.6

price: Friday, 10-17 $1

price: Friday, 18-23 $0,6

price: Saturday, 0-23 $0.6

price: Sunday, 0-23 $0.6

Описание файлов в домашнем каталоге пользователя с "биллинговой информацией"

.pay - информация о начислениях (история начислений) на лицевой счет пользователя условных единиц или $. Файл имеет формат вида:

#

#

# Платежи клиента ivan

#

#

1999/02/27 13:00:01 Add pay | 10.5

1999/03/15 15:12:00 Add pay | 23

1999/05/05 12:30:40 Add pay | 6.5

Как видно, данный файл имеет два поля произвольной длины разделенные символом "|". Первое (левое) поле содержит комментарий или, другими словами, обоснование для второго (правого) поля, в котором содержится число с плавающей точкой, определяющее стоимость транзакции, т.е. стоимость биллинговой информации. Основная и единственная единица измерения биллинговой информации - условная единица или $. Если приведенный выше файл содержится в домашнем каталоге пользователя ivan, то, просуммировав второе (правое) поле, можно выяснить, что общий размер начислений на лицевой счет (или платежей) клиента ivan равняется 40 условным единицам. Открыв этот файл, системный администратор или пользователь ivan может не только узнать сколько вообще было начислено на данный лицевой счет, но и то, когда (кем) это было сделано (забегая вперед, хочется отметить, что подобный способ хранения биллинговой информации в обычных текстовых файлах, т.е. "дата, обоснование операции | размер", является основным для предлагаемой системы). Добавлять или изменять информацию в файлах .pay должен только системный администратор. Делать это можно как из командной строки, так и через веб-интерфейс;

.weekly - информация об отчислениях (история отчислений) с лицевого счета пользователя в условных единицах за фактическую работу за текущую неделю по каждому соединению (по каждой предоставленной услуге). Формат файла аналогичен формату файла .pay

#

#

# Работа клиента ivan за текущую неделю

#

#

1999/05/18 13:00:01 Time elapsed=40 sec., cost | 0.052

1999/05/19 15:12:00 Time elapsed=1200 sec., cost | 0.156

1999/05/19 16:30:40 Time elapsed=75 sec., cost | 0.101

Запись в файл .weekly делает демон после окончания соединения. В приведенном выше фрагменте файла .weekly в первом поле содержится следующая информация: дата окончания соединения (YYYY/MM/DD), вреня окончания соединения (HH:MM:SS), продолжительность соединения в секундах (Time elapsed). Во втором поле файла .weekly, отделенного символом "|" содержится стоимость транзакции (стоимость соединения, стоимость фактически предоставленной услуги в условных единицах);

.work - информация об отчислениях (история отчислений) с лицевого счета пользователя по за целые недели в сумме. В конце каждой недели второе поле файла .weekly суммируется и итоговая сумма заносится в файл .work.

1999/05/18 1999/05/25 cost | 5.011

1999/05/26 1999/06/01 cost | 2.133

Файлы .weekly в домашних каталогах пользователей "разбухают" вследствие того, что там протоколируется каждое соединение. С другой стороны, как показывает практика, большинству пользователей интересен лишь подробный отчет использования машинного времени за текущую и прошедшую неделю, поэтому информация, накопленная в файлах .weekly должна "компрессироваться". Если же пользователю нужно предоставить подробный отчет за работу двух- (трех-, четырех- и т.д.) недельной давности (что случается не так часто), то эту информацию системный администратор может "вытащить" из SQL-базы. Обновлением файлов .work, .weekly и .weekly.last должна заниматься специальная программа w\_update.pl, запускаемая по крону еженедельно;

.weekly.last - копия файла .weekly за прошлую неделю;

.current - текущий размер лицевого счета пользователя в условных единицах, файл обновляется после завершения каждой сессии данного пользователя, служит для "быстрого" вычисления размера лицевого счета пользователя, например, при старте нового соединения;

.time - наличие такого файла (под наличием файла подразумевается файл с данным именем любой длины, в том числе и нулевой, доступный для чтения в данный момент) сигнализирует о том, что баланс лицевого счета пользователя всегда положителен. В русском языке обычно под этим подразумевают сладкое слово "халява". Разумеется, этот файл прежде всего должны иметь, например, сотрудники ISP и их ближайшие родственники ;-)

.refused - наличие такого файла сигнализирует о том, что баланс лицевого счета пользователя всегда отрицателен, т.е. доступ ему в систему временно приостановлен;

.type - тип пользователя (например, "свой", "халява", "бартер", "деньги"). Служит для деления пользователей на группы, по которым в дальнейшем генерируется статистическая информация;

.account - тип (или индекс) прайс-листа для данного пользователя. Этот файл очень удобно использовать для задания прайс-листа отдельным группам пользователей. Вы создаете желаемый прайс-лист и помещаете его в каталог /var/statserv/etc, а пользователям в домашних каталогах указываете лишь ссылку на него. Если Вы хотите поменять прайс-лист для некоторой группы пользователей, то, Вы редактируете прайс-лист всего в одном месте (см. ниже "Алгоритм выбора тарифной схемы для пользователя при старте демона");

.account.conf - собственный прайс-лист для данного пользователя. См. структуру файла .account.conf. Этот файл следует применять в том случае, когда Вы хотите задать для пользователя индивидуальный прайс-лист;

.pay.next - авансовый платеж, или следующее начисление на лицевой счет пользователя после обнуления текущего лицевого счета. Может быть использовано в том случае, когда пользователь не исчерпал текущий лицевой счет, однако оплатил следующую услугу по прежнему или новому прайс-листу;

.account.next - то же, что и .account (см. выше), но только для авансового платежа.

**Описание выходных документов.**

В результате работы биллинговой программы вся информация о работе пользователь, как было сказано выше, фиксируется в лог-файлах и базе данных. Это является основным базисом для генерации отчетов и статистики. Извлекаемые данные могут быть представлены в качестве структурированных таблиц, либо в форме отчетов по запрашиваемым данным. Данная информация, также является подтверждением того, что пользователь работал в сети на случай претензий последнего.

Список информации - данных, которые предлагаются пользователю и системному администратору (биллинг-инженеру):

Время регистрации пользователя в конкретный день

Оставшаяся сумма на счету у пользователя

Время, проведенное в сети

Статистика работы в сети по дням, неделям и месяцам

Почтовое уведомление пользователя и администратора об истечении денежного взноса.

Общая структурированная таблица статистики за определенный период времени

При генерации выше указанной информации используются дополнительные модули программы «Billing ISP» и системные программы Unix такие как, CGI- модули (для обращения к базам данных и генерации HTML кода и форм или писем), Apache web server (для вывода на экран HTML кода сгенерированного CGI программой), MTA Sendmail (для отправки электронного письма пользователю об окончании счета).

**Описание технологии и алгоритмов решения задачи и их машинная реализация.**

**Описание ввода в базу данных входной информации.**

Отличительная черта рассматриваемой программы от других вариантов данной курсовой работы, является полностью ароматическая генерация и занесение в базу данных информации за исключением создания самих счетов пользователей. Это определяется в основном спецификой данного программного продукта и операционной среды, в которой он работает.

Алгоритм начисления условных единиц на лицевой счет пользователя

Если файл .pay в домашнем каталоге пользователя отсутствует, то занести размер платежа в файл .pay, а индекс прайс-листа в файл .account. Переход к пункту 3;

Вычислить текущий размер лицевого счета пользователя. Если он отрицателен или равен нулю, то очередной платеж заносится в файл .pay, а выбранный индекс прайс-лист в файл .account. Если текущий размер лицевого счета пользователя положителен, то очередной платеж заносится в файл .pay.next, а выбранный индекс прайс-листа в файл .account.next;

Обновить файл .current с текущим размером лицевого счета;

Занести платеж в таблицу "ПЛАТЕЖИ" базы данных (опционально).

Алгоритм фиксации в базе статистической информации

Введение.

Рассматриваемый программный продукт напрямую зависит от системных вызовов операционной среды, в которой он работает, а также и от некоторых приложений, например PPPD (название этого демона происходит от названия протокола соединения пользователя и провайдера Point to Point Protocol), syslog (системная программа, которая фиксирует пребывание пользователя в системе, а также фиксирует в лог-файлы сообщения ядра ОС). В связи с тем, что описания данных продуктов это тема отдельной курсовой работы, данный алгоритм будет описан поверхностно.

При соединении пользователя к провайдеру запускается программа Mgetty, которая управляет и инициализирует работу модема. Запуск данной програмы фиксируется в системных лог-файлах системы. После ее запуска она активизирует, в нашем случаи демон PPPD, который в свою очередь принемает и регистрирует пользовательские запросы и после проверки всего необходимого впускает или нет в систему. Все действия данных сервисов после соединения отслеживает программа syslog, которая и генерирует основную базу действий пользователя в системе.

Billing ISP, как уже говорилось выше напрямую взаимодействует с PPPD проверяя актуальность данного подключения и в случаи удачного входа изменяет (уменьшает) за определенный квант времени счет пользователя.

Также демон биллинга с момента соединения пользователя начинает вести подсчет времени пребывания пользователя с соответствующим изменением в SQL базе полей.

Обобщенный алгоритм решения задачи.

Ядром предлагаемой системы является специально написанный демон "биллинга" (в дальнейшем просто демон), осуществляющий контроль за израсходованнным временем и/или условными единицами пользователя, находящегося в данный момент в режиме "он-лайн". Демон запускается в момент входа пользователя в систему, по истечении одного кванта времени (например, 5 секунд) снимает с лицевого счета пользователя стоимость одного кванта в соответствии с действующей в данный момент ценой, которая, кстати, может меняться в зависимости от времени суток, а после завершения сеанса связи демон прекращает свою работу, протоколируя информацию о продолжительности сеанса связи и отработанной стоимости в специальный лог-файл в домашнем каталоге пользователя и, при необходимости, в общую SQL-базу данных. Другими словами, отдельная копия демона постоянно присутствует в памяти сервера биллинга и "следит" за пользователем на протяжении всего сеанса связи. Информация о начислениях на лицевой счет пользователя и снятия с лицевого счета за фактически отработанное "он-лайновое" время (так называемая "биллинговая информация") хранится в домашних каталогах пользователей в обычных текстовых файлах. Т.е. для каждого пользователя создается свой набор файлов с биллинговой информацией. Соответственно, вычисление размера лицевого счета пользователя происходит на основании содержимого этих файлов. Такое распределенное хранение биллинговой информации по каждому пользователю обеспечивает простоту построения системы, а значит надежность, и предоставляет возможность организации несложной системы доступа к лицевым счетам через www-интерфейс. В тоже время, такая же информация, но немного в другом виде хранится в SQL-базе, однако она используется исключительно для генерации статистической информации предоставляемой системному администратору.

Алгоритм выполнения отдельных модулей.

Алгоритм вычисления лицевого счета при входе пользователя в систему

Данный алгоритм должна реализовывать программа, выполняющая аутентификацию пользователя (TACACS+ или pppd) на этапе подключения его к Интернету. В этот случае основную роль должен играть файл .current, который содержит уже вычисленное значение размера лицевого счета, т.к. в данный момент размер лицевого счета должен быть получен практически мгновенно.

Если файл .refused присутствует в домашнем каталоге пользователя, то текущий размер лицевого счета принимается отрицательным. Переход к пункту 5;

Если файл .time присутствует в домашнем каталоге пользователя, то текущий размер лицевого счета принимается положительным. Переход к пункту 5;

Если файл .current присутствует в домашнем каталоге пользователя, то из него берется текущий размер лицевого счета (файл **.current** обновляется каждый раз после завершения работы демона). Переход к пункту 5;

Текущий размер лицевого счета вычисляется по формуле : "общая сумма начислений из файла **.pay** минус общая сумма отчислений из файла **.work** минус общая сумма отчислений из файла **.weekly**";

Если текущий размер лицевого счета положителен, доступ в систему пользователю разрешатеся, в противном случае - запрещается.

Алгоритм выбора прайс-листа (тарифной схемы) для пользователя при старте демона

Если в домашнем каталоге пользователя находится файл .account.conf, то прайс-лист для данного пользователя читается из этого файла;

Если в домашнем каталоге пользователя присутсвует файл .account, то из него читается первая строчка, которая добавляется в слову account, к полученной строке добавляется расширение .conf, и, таким образом файл c прайс-листом с полученным именем читается из каталога /var/statserv/etc;

Если файлы .account.conf и .account отсутствуют в домашнем каталоге пользователя, то прайс-лист для данного пользователя читается из файла /var/statserv/etc/account.conf (прайс-лист по умолчанию)

Действия, выполняемые демоном при старте

Анализирует командную строку. В качестве первого параметра ему передается имя пользователя, в качестве второго - NAS-порт (или устройство, например, /dev/cuaa2), в качестве третьего - адрес NAS-сервера (третий параметр нужен только в том случае когда у провайдера более одного NAS'a);

Выбирает прайс-лист (тарифную схему) для пользователя (см. "Алгоритм выбора прайс-листа для пользователя при старте демона");

Проверяет присутствие в домашнем каталоге файла .time, если он там есть - взводит соответствующий флажок в переменную us.unoff=1;

Проверяет присутствие в домашнем каталоге файла .refused, если он там есть - взводит соответствующий флажок в переменную us.refused=1;

Вычисляет значение лицевого счета пользователя (см. пункт 4 "Алгоритма вычисления лицевого счета при входе пользователя в систему"). Даже если размер лицевого счета отрицателен, демон все равно продолжит свою работу, поскольку по истечении первого же кванта времени он, при необходимости, подаст сигнал на отключение этого пользователя;

Демонизируется ;-);

Записывает свой PID в файл с именем NAS-порта в каталог /var/run (если у провайдера больше одного NAS'a - то необходимо модифицировать демон, чтобы избежать конфликтов в каталоге /var/run между NAS'ами);

Программирует собственный таймер на заданный квант времени и входит в бесконечный цикл, вывести из которого может только SIGHUP.

Действия, выполняемые демоном при истечении одного кванта времени

Обновить счетчик (в секундах) продолжительности текущего соединения, в соответствии с действующим тарифом вычислить стоимость одного кванта времени, обновить переменную в которой накапливается стоимость текущей сессии;

Проверить, присутствует ли пользователь все еще в системе - просмотреть файл **/var/run/utmp**. Если пользователя в системе нет, то вызвать процедуру, выполняемую при завершении сессии;

Если пользователь не исчерпал свой лицевой счет, то ждать истечение следующего кванта времени. В противном случае описанные ниже действия возникают при исчерпывании средств на лицевом счете:

Проверить, является ли этот пользовать "привелегированным". Для этого посмотреть на флажок us.unoff. Если us.unoff==1, то ждать истечение следующего кванта времени;

Проверить, была ли вызвана программа **/var/statserv/bin/killuser**, если да - то ждать истечение следующего кванта времени. Дело в том, что из-за особенностей построения некоторых систем аутентификации, при исчерпывании средств на лицевом счете, фактически пользователь отключается не сразу после вызова программы **/var/statserv/bin/killuser**, а через некоторый интервал времени;

Если файл .pay.next отсутствует в домашнем каталоге пользователя, значит, пользователя необходимо принудительно отключить. Переход к пункту 9;

Прочитать сумму из файла .pay.next. Переписать ее в файл .pay. Удалить файл .pay.next. Вычислить текущий размер лицевого счета пользователя. Если он отрицателен, то переход к пункту 9;

Перечитать новый прайс-лист из файла .account.next. Удалить файл .account.conf, если он присутствует. Переименовать файл .account.next в файл .account и ждать истечение следующего кванта времени.

Вызвать программу /var/statserv/bin/killuser с параметрами имя\_пользователя и NAS-порт, которая пошлет сигнал на отключение пользователя;

Действия, выполняемые демоном при завершении сессии

При завершении сессии сервер аутентификации TACACS (или pppd) читает PID демона из каталога /var/run/ и посылает ему SIGHUP (возможен, также другой вариант, когда демон постоянно сканирует файл utmp и выполняет ниже описанные действия, если пользователь "изчез" из файла utmp). Демон удаляет файл со своим PID-ом из /var/run/, записывает сведения о только что завершенной сессии в файл .weekly, обновляет файл .current с текущим размером лицевого счета пользователя и вызывает скрипт /var/statserv/bin/close\_session. с параметрами имя\_пользователя, NAS-port, продолжительность\_сессии, стоимость\_сессии.

**Контрольный пример.**

**Описание использования демона биллинга**

Демон billing является ядром предлагаемой системы учета пользователей для ISP. Он может работать в двух режимах - в "основном" режиме (т.е. режиме демона) для контроля лицевого счета пользователя в реальном масштабе времени и в "вспомогательном" режиме выдачи сведений о размере лицевого счета пользователя или контроля правильности задания тарифной схемы. Ниже приведены все возможные ключи запуска и параметры командной строки:

/var/statserv/bin/billing

Usage error: billing [-vdeashtpPrRwW] [username] [port] [nas]

-v show version and exit

-d daemonize billing

-e increase debug level in daemonize mode

-a show current price

-c show current account for sysadmin

-s show current account

-h show current account in HTML format

-t total user account

-p show pay -P show pay history

-n show pay.next -N show pay.next history

-r show work -R show work history

-w show weekly -W show weekly history

Режим демона - запуск с ключом -d. Далее следуют обязательные параметры - имя пользователя, NAS-порт (порт модема) и имя NAS-сервера (если NAS-сервер у ISP один, то этот параметр выбирается произвольно, однако совсем опускать его нельзя). Пример:

/var/statserv/bin/billing -d jackson Async2 access.provider.net

Режим выдачи сведений о размере лицевого счета пользователя - запуск с ключом -s и единственным параметром - именем пользователя. Пример:

/var/statserv/bin/billing -s jackson

В данном варианте на стандартный вывод ничего не выводится, а "знак" лицевого счета отражается в коде возврата. Если размер лицевого счета больше либо равен нулю, т.е. доступ пользователю в систему разрешен, то billing возвращает число 0, если меньше нуля, т.е. доступ пользователю в систему ограничен, то billing возвращает число 1.

/var/statserv/bin/billing -st jackson

На стандартный вывод выводится общий размер лицевого счета пользователя в plain text. См. п.4 алгоритма вычисления размера лицевого счета пользователя

/var/statserv/bin/billing -spt jackson

На стандартный вывод выводится общий размер начислений на лицевой счет пользователя и его общий размер в plain text. Алгоритм вычисления лицевого счета тот же самый. Ключ -P задает вывод истории начислений.

/var/statserv/bin/billing -c jackson

Соответствует вызову

/var/statserv/bin/billing -stpnrw jackson

т.е. наиболее "употребительному" использованию billing с точки зрения системного администратора. Вызов billing с ключом -h, например

/var/statserv/bin/billing -shtpnrw jackson

выводит информацию о лицевом счете пользователя в HTML-формате для того, чтобы ее можно было предоставлять пользователю через www-интерфейс;

Режим проверки алгоритма выбора прайс-листа для пользователя. Вызов:

/var/statserv/bin/billing -a jackson

Предназначен исключительно для системного администратора чтобы контролировать правильность задания тарифной схемы для данного пользователя.