Введение

HTTP прокси-сервер представляет собой программу, которая принимает запросы от клиентов в виде URL-адресов и возвращает результат клиенту. Прокси-серверы используются в сетях, где клиенты не имеют прямого доступа к Интернету, но должны иметь возможность просмотра веб-страниц. Кроме того, прокси-сервер предпологает кэширование запросов выполненных однажды некоторым клиентом.

Многие компании и организации ставят в сетях firewall(файрволлы, брандмауэры), чтобы заблокировать весь входящий и исходящий трафик во внутренних локальных сетях. Это может быть сделано по соображениям безопасности, либо для ограничения списка лиц имеющих доступ в Интернет. Поскольку возможность просмотра веб-страниц является чрезвычайно полезной, прокси-сервер настраивается так, что веб-сайты могут быть доступны через него.

Крупные организации и провайдеры с большим количеством клиентов, также могут использовать прокси-сервер для снижения нагрузки на сеть, потому, что одной из главных задач прокси-сервера является кэширование страниц запросов клиентов, то любая страница запрошенная несколько раз будет возвращена из кэша, вместо того чтобы закачать ее заново. По этой причине, клиентам часто рекомендуют использовать прокси-сервер для доступа к сети Интернет.

Прокси-сервер полезен только тогда, когда браузер клиента настроен так, чтобы проводить запросы через сервер, вместо обращения к сайтам напрямую. На сегодня каждый браузер умеет работать через прокси-сервер.

Проект прокси-сервера Squid в свое время отделился от ныне платного проекта Harvest и разрабатывается несколькими энтузиастами во главе с Duane Wessels из Национальной лаборатории по исследованию сетей (National Laboratory for Applied Network Research). Сервер Squid — это высокопроизводительный кэширующий прокси-сервер, ориентированный прежде всего на работу с пользователями, которые занимаются активным серфингом в Интернете. Squid поддерживает работу пользователей с такими протоколами, как FTP, HTTP, HTTPS и GOPHER. В отличие от других подобных проектов, прокси-сервер Squid обладает интересной особенностью — выполнение запросов пользователей реализовано в нем как один большой неблокируемый процесс ввода-вывода, что обеспечивает более высокую производительность сервера в целом. Поскольку сервер Squid является кэширующим прокси-сервером, он поддерживает широкие возможности по построению иерархической структуры связи кэш-серверов на основе протоколов ICP/UDP (Internet Cache Protocol), HTCP/TCP и multicast. Такая система позволяет получить высокую производительность и оптимизировать пропускную способность канала в Интернет. Кэш сервера разделяется на виртуальный, который находится в оперативной памяти компьютера, и обычный, который хранится на жестком диске. Наиболее часто используемые объекты хранятся в оперативной памяти, что ускоряет процесс их отсылки клиентам. Также в виртуальной памяти хранится большая часть запросов DNS. Squid в полной мере поддерживает SSL (HTTPS), что обеспечивает конфиденциальность передаваемой пользователями информации и приватность их работы в Интернете. Также нельзя обойти вниманием широкие возможности по аутентификации пользователей на основе различных методик: NCSA, LDAP, MSNT, NTLM, PAM, SMB, SASL и др. Все дополнительные программы для аутентификации пользователей идут в комплекте с основным ядром программы. Как видно из перечисленных методик, Squid поддерживает авторизацию пользователей средствами сервисов не только на Linux, но и на Windows-платформе (MSNT и NTLMv1). В будущем ожидается поддержка сервиса NTLMv2, который используется в операционных системах Windows 2003 Server и Vista.

1. Описание предметной области

1.1 Постановка задач

Для компьютерной сети, выходящей в интернет через прокси-сервер Squid, настроить список контроля доступа. Сервер расположен на компьютере под управлением операционной системы семейства Linux.

1.2 Linux Mint

Linux Mint — дистрибутив, основанный на Ubuntu GNU/Linux. Linux Mint отличается от большинства дистрибутивов на основе Ubuntu GNU/Linux тем, что в результате изменений операционная система не теряет совместимость и те положительные качества, которыми наделена оригинальная Ubuntu. Разработчики Linux Mint вносят в дистрибутив Ubuntu ряд полезных изменений, исправляя некоторые недочеты и недостатки, тем самым делая ее доступной более широкой аудитории пользователей, а в довершение ко всему комплектуют операционную систему оригинальным интерфейсом и рядом собственных приложений (mintInstall, mintUpdate, mintBackup, mintConfig, mintAssistant, mintMenu, mintDesktop, mintNanny и других). Растущая популярность Linux Mint объясняется просто. Новые пользователи легко могут установить систему и сразу же включиться в работу. Профессионалы, в свою очередь, получают стабильную и легко конфигурируемую современную систему, на настройку которой они будут тратить значительно меньше времени. Девиз этого дистрибутива linux «from freedom came elegance» можно перевести так: свобода, привносящая элегантность.

1.3 Прокси-сервер

Прокси-сервер (от англ. proxy — «представитель, уполномоченный») — служба в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам. Сначала клиент подключается к прокси-серверу и запрашивает какой-либо ресурс (например, e-mail), расположенный на другом сервере. Затем прокси-сервер либо подключается к указанному серверу и получает ресурс у него, либо возвращает ресурс из собственного кэша (в случаях, если прокси-сервер имеет свой кэш). В некоторых случаях запрос клиента или ответ сервера может быть изменён прокси-сервером в определённых целях. Также прокси-сервер позволяет защищать клиентский компьютер от некоторых сетевых атак.

1.4 Использование прокси-серверов

Чаще всего прокси-серверы применяются для следующих целей:

Обеспечение доступа с компьютеров локальной сети в Интернет.

Кэширование данных: если часто происходят обращения к одним и тем же внешним ресурсам, то можно держать их копию на прокси-сервере и выдавать по запросу, снижая тем самым нагрузку на канал во внешнюю сеть и ускоряя получение клиентом запрошенной информации.

Сжатие данных: прокси-сервер загружает информацию из Интернета и передаёт информацию конечному пользователю в сжатом виде. Такие прокси-серверы используются в основном с целью экономии внешнего трафика.

Защита локальной сети от внешнего доступа: например, можно настроить прокси-сервер так, что локальные компьютеры будут обращаться к внешним ресурсам только через него, а внешние компьютеры не смогут обращаться к локальным вообще (они «видят» только прокси-сервер).

Ограничение доступа из локальной сети к внешней: например, можно запретить доступ к определённым веб-сайтам, ограничить использование интернета каким-то локальным пользователям, устанавливать квоты на трафик или полосу пропускания, фильтровать рекламу и вирусы.

Анонимизация доступа к различным ресурсам. Прокси-сервер может скрывать сведения об источнике запроса или пользователе. В таком случае целевой сервер видит лишь информацию о прокси-сервере, например, IP-адрес, но не имеет возможности определить истинный источник запроса. Существуют также искажающие прокси-серверы, которые передают целевому серверу ложную информацию об истинном пользователе.

Многие прокси-серверы используются для нескольких целей одновременно. Некоторые прокси-серверы ограничивают работу несколькими портами: 80 (HTTP), 443 (Шифрованное соединение HTTPS), 20,21 (FTP).

1.5 Типы прокси-серверов

Зачастую предоставленное для прокси-сервера оборудование будет на самом деле выполнять или поддерживать множество типов прокси-сервисов. К примеру, прокси-сервер может предоставлять возможности кэширования и аутентификации в дополнение к основной функции обеспечения сетевого посредничества для приложений.

Ключевыми типами прокси-серверов, являются:

пересылающие прокси-серверы (forward proxies);

прозрачные прокси-серверы (transparent proxies);

кэширующие прокси-серверы (caching proxies);

прокси-сервер обеспечения безопасности (security proxies);

обратные прокси-серверы (reverse proxies).

Пересылающие прокси-серверы

Пересылающий прокси-сервер является прокси-сервером, который помогает пользователям из одной зоны безопасности выполнять запросы контента из "следующей" зоны, следуя направлению, которое обычно (но не обязательно) является исходящим (это значит, что клиент находится внутри, а сервер где-то в открытом Интернете).

С точки зрения безопасности простой прокси-сервер имеет целью обеспечение безопасности, состоящее в скрытии наименования (в терминах топологии внутренней сети) рабочей станции или процесса запрашивающего пользователя. Он может также применяться для скрытия некоторых других атрибутов сеанса пользователя.

Типичным примером этого типа являются корпоративные прокси-серверы, которые обслуживают внутренних пользователей посредством разрешения им доступа на внешние сайты для Web-браузинга или любого другого вида взаимодействия с Интернетом.

С точки зрения топологии (как в общем смысле, так и относительно ширины полосы пропускания) пересылающие прокси-серверы всегда относительно ограничены в терминах сетевой скорости по отношению к своим пользователям из-за более медленного WAN-соединения (соединения с глобальной сетью), которое обычно отделяет пересылающий прокси-сервер от реального контента в Интернете.

Прозрачные прокси-сервера

Прозрачные прокси-сервера являются прокси-серверами, которые "находятся здесь", но не осведомляют пользователей в прямой форме о том, что они здесь находятся. В пересылающих прокси-серверах обычно существуют Linux/UNIX блоки, которые слушают весь трафик по определенному протоколу для определенного сегмента сети и перехватывают трафик, хотя пользовательский процесс в действительности не знает об их существовании. Фактически пользовательский процесс не общается с прокси-сервером, но общается с другим (конечным) сайтом, а прокси-сервер, в сущности, становится тем "человеком посередине", который "взламывает" соединение.

Прокси-сервер является непрозрачным, или объявленным, когда пользователи знают о том, что они общаются через прокси-сервер, потому что они обращаются (на языке прокси-сервера: HTTP) к прокси-серверу.

Прозрачные прокси-сервера сами по себе не являются на самом деле типом прокси-сервера, скорее любой прокси-сервер является либо прозрачным, либо объявленным по проекту.

Кэширующие прокси-серверы

Кэширующие прокси-серверы, как указано в их названии, являются прокси-серверами, которые сконфигурированы на повторное использование кэшированных образов контента, когда это доступно и возможно. Когда кэшированная ранее часть контента недоступна, то производится ее выборка и использование в контенте, но также с попыткой ее кэширования.

Наиболее важным аспектом для кэширующих прокси-серверов является необходимость обеспечения того, что кэширующие прокси-серверы кэшируют только то, что на самом деле можно кэшировать. Динамический, регулярно изменяющийся контент не лучший выбор для кэширования, так как это может оказать воздействие на стабильность приложения, основанного на этом контенте. В случае HTTP-контента заголовки HTTP отображают возможность кэширования контента посредством указателей "cache".

В большинстве случаев пересылающие прокси-серверы конфигурируются также для работы в качестве кэширующих прокси-серверов. Это явление используется настолько часто, что компания IBM включила это в название компонента своего Edge Server: IBM Caching Proxy.

Прокси-сервер обеспечения безопасности

В качестве необходимой для простых прокси-серверов функциональности прокси-серверы могут быть сконфигурированы для приведения в исполнение политик безопасности. Такие прокси-серверы обеспечения безопасности могут обрабатывать (либо выступать в качестве посредников при обработке) запросы аутентификации и авторизации. В этих случаях аутентификация пользователя клиента и авторизация клиента для доступа к определенному контенту контролируется самим прокси-сервером. Далее мандат безопасности посылается от прокси-сервера к конечным серверам с запросом, а конечный сервер должен быть сконфигурирован на оказание доверия предоставляемому прокси-серверу мандату.

Существует много различных продуктов и предложений, а также множество топологий на выбор, но с точки зрения выполнения прокси-функций безопасность является дополнительной функцией, которую может выполнять прокси-сервер.

В большинстве случаев функциональные возможности по обеспечению безопасности могут быть добавлены стандартному прокси-серверу в виде дополнительного программного модуля (плагина, от англ.- plug-in) (к примеру, IBM Tivoli WebSeal Plug-In для IBM WebSphere Edge Server). Существуют также и отдельные продукты, такие, как IBM Tivoli Access Manager for e-Business, которые служат только в качестве прокси-сервера обеспечения безопасности.

Обратные прокси-серверы

Обратные прокси-серверы имеют много общего кода с пересылающими прокси-серверами: фактически одни и те же продукты могут быть сконфигурированы одним или другим образом либо двумя сразу. Однако с функциональной и практической точки зрения в нашем случае мы рассматриваем обратные прокси-серверы как полностью другой инструмент.

Обратные прокси-серверы прозрачны, отчасти по определению. За обратным прокси-сервером пользователь вообще не знает о своем общении с прокси-сервером. Пользователь полагает, что общается с реальным предметом – сервером, на котором находится контент.

Обратные прокси-серверы обычно избраны и реализованы в целях обеспечения изоляции контента и зон. Однако, вы можете также добавить в обратные прокси-серверы функциональные возможности по кэшированию для обеспечения производительности заодно с преимуществами обеспечения безопасности.

Нужно обратить внимание, что при этом виде сценария выигрыш в производительности за счет кэширования относится к производительности сети, так как обычно обратный прокси-сервер расположен близко к конечному серверу. Важным мотивом кэширования с помощью обратного прокси-сервера является разгрузка от подачи статического, кэшируемого контента от конечных серверов приложений. Это позволит зачастую более дорогим конечным серверам приложений немного освободиться и сфокусировать свое внимание на пропускных способностях своих процессоров при выполнении более сложных динамических задач и задач, связанных с транзакциями.

Однако когда на обратном прокси-сервере разрешено кэширование, важным становится обеспечение его должной защиты. Весь контент, даже если он полностью статичен, продолжает нуждаться в защите.

1.6 Прокси-сервер Squid

Squid — программный пакет, реализующий функцию кэширующего прокси-сервера для протоколов HTTP, FTP, Gopher и (в случае соответствующих настроек) HTTPS. Разработан сообществом как программа с открытым исходным кодом (распространяется в соответствии с GNU GPL). Все запросы выполняет как один неблокируемый процесс ввода/вывода. Используется в UNIX-like системах и в ОС семейства Windows NT. Имеет возможность взаимодействия с Active Directory Windows Server путём аутентификации через LDAP, что позволяет использовать разграничения доступа к интернет ресурсам пользователей, которые имеют учётные записи на Windows Server, также позволяет организовать «нарезку» интернет трафика для различных пользователей.

В сочетании с некоторыми межсетевыми экранами и маршрутизаторами Squid может работать в режиме прозрачного прокси-сервера. В этом режиме маршрутизатор вместо того, чтобы сразу пересылать http-запросы пользователя http-серверу в интернете, перенаправляет их прокси-серверу, который может работать как на отдельном хосте, так и на самом маршрутизаторе. Прокси-сервер обрабатывает запрос (с возможной отдачей содержимого из кэша), это содержимое направляется к запросившему пользователю, для которого оно выглядит как «ответ» сервера, к которому адресовался запрос. Таким образом, пользователь может даже не знать, что все запросы и ответы прошли через прокси-сервер.

Сервер Squid развивается в течение уже многих лет. Обеспечивает совместимость с большинством важнейших протоколов Интернета, а также с операционными системами:

GNU/Linux

FreeBSD

OpenBSD

NetBSD

BSDI

Mac OS X

OSF и Digital Unix

IRIX

SunOS/Solaris

NeXTStep

SCO Unix

AIX

HP-UX

Microsoft Windows

Описание архитектуры

Списки контроля доступа

Для контроля доступа к ресурсам и определения ряда действий используются списки контроля доступа (англ. access control list, acl). Каждый ACL может состоять из нескольких критериев (но только одного типа):

адрес (сеть) источника запроса, цели запроса

имя (доменное имя) источника запроса, имя цели запроса

части URL запроса

протокол

порт (получателя, отправителя, самого Squid’а)

метод (PUT или GET) при передаче данных по HTTP

браузер (User-agent)

ident (запрос к рабочей станции)

номер автономной системы отправителя/получателя (не для всех случаев)

авторизация на прокси-сервере

номер соединения (чаще всего используется для ограничения количества соединений)

SNMP

сертификаты пользователя

параметры запроса

внешние обработчики

Идентификация

Squid поддерживает несколько видов идентификации пользователей:

по IP-адресу (или доменному имени узла)

по переданным реквизитам (логин/пароль)

по идентификатору пользовательского агента (браузера)

Для идентификации по логину/паролю возможно использовать:

обычные логин/пароль

NTLM-авторизацию

внешние программы авторизации (определяющие формат авторизации)

Редиректоры

Squid имеет возможность переписывать запрашиваемые URL. Squid может быть сконфигурирован так, чтобы пропускать входящие URL через процесс редиректора выполняемого как внешний процесс (подобно dnsserver), который возвращает новый URL или пустую строку, обозначающую отсутствие изменений.

Редиректор – не является стандартной частью пакета Squid. Редиректор предоставляет администратору контроль за передвижениями пользователей. Использование редиректора в сочетании с прозрачным проксированием дает простой, но эффективный контроль, над доступом к порно. Программа-редиректор должна читать URL (один на строку) со стандартного входа и записывать измененные URL или пустые строки на стандартный выход. Нужно заметить, что программа-редиректор не может использовать буферизированный I/O. Squid дописывает дополнительную информацию после URL, которую редиректор может использовать для принятия решения.

SAMS (SQUID Account Management System) - программное средство для администрирования доступа пользователей к прокси-серверу Squid.

На данный момент SAMS настраивает работу редиректоров:

Редиректор SAMS - редиректор, работающий напрямую с базами SAMS

SquidGuard - очень мощный редиректор.

Стандартный SQUID - простейший редиректор, описанный в документации к SQUID.

Редиректор SAMS

Написан специально для SAMS, напрямую использует информацию, содержащуюся в базе данных. Позволяет включить различное перенапраление запросов для пользователей (регулируется шаблонами пользователей).

Редиректор SAMS обеспечивает:

ограничение доступа пользователей к SQUID

контроль времени доступа пользователей к SQUID

ограниечение доступа пользователей к запрещенным ресурсам (или доступ пользователей только к разрешенным ресурсам)

перенаправление запросов пользователей к баннерам, счетчикам и т.п.

Редиректор SquidGuard

Мощный редиректор с большими возможностями. В состав редиректора входят списки баннерных, порно и пр. доменов.

SAMS добавляет в файл конфигурации SquidGuard Squidguard.conf настройки на списки запрещенных доменов и перенаправления доступа SAMS. Настройки на списки, идущие с SquidGuard не изменяются и не удаляются.

При использовании редиректора SquidGuard в файл Squid.conf заносятся acl, разрешающие доступ всех пользователей к SQUID. Ограничение доступа пользователей организовано средствами редиректора.

Стандартный SQUID

Этот редиректор описан в документации на SQUID. Написан на perl. Редиректор создается после подачи команды на реконфигурирование SQUID, на основе списков перенаправления запросов. Быстрый и легкий редиректор, но не различает пользователей.

При использовании этого редиректора, ограничение доступа пользователей по спискам запрета доступа организовано с использованием ACL SQUID.

При использовании редиректора SQUID или если редиректор не используется вовсе, то в существует возможность - при отключении пользователей за превышение трафика у них остается доступ к URL и IP адресам, прописанным в списке "Локальные домены".

Ограничение максимальной скорости соединения

Ограничение максимальной скорости получения пользователем (пользователями) в Squid реализовано с помощью механизма англ. delay pools (дословно — «пулы задержки»). Механизм ограничения скорости работает по принципу бассейна (откуда и название pool (бассейн)), в который «втекает» и «вытекает» информация. Отдельные конфигурируемые подобным образом области памяти называются англ. bucket (ведро). У ведра есть параметры: «ёмкость», «скорость наполнения». Если пользователь (пользователи) получают информацию на скорости ниже, чем «скорость наполнения», то ведро всегда полно. Если пользователь кратковременно поднимает скорость получения информации выше скорости наполнения, то до момента, пока ведро не пусто, он не ограничивается по скорости, как только ведро становится пустым, клиент получает информацию со скоростью наполнения ведра. В случае наличия групповых и индивидуальных ведёр, они включаются последовательно.

Существует три типа (класса) delay pools:

Единое ведро (англ. aggregate bucket, class 1) ограничение на общую потребляемую полосу для всей группы. (параметры: ёмкость бассейна, скорость наполнения).

Единое ведро с автоматическим формированием индивидуальных вёдер (англ. single aggregate bucket as well as an "individual" bucket, class 2). Индивидуальные вёдра формируются из битов IP-адреса (c 25 по 32).

Единое ведро, сетевые вёдра и индивидуальные вёдра (англ. single aggregate bucket as well as a "network" bucket and a "individual" bucket, class 3). Сетевое ведро формируется по битам 17-24 IP-адреса.

Для каждого ведра указываются два параметра: ёмкость и скорость наполнения. −1 означает «без ограничения».

Попадание пользователей в то или иное ведро определяется списками доступа к вёдрам, они просматриваются в порядке упоминания в файле конфигурации до первого совпадения. Пользователи, не попадающие ни в одно из вёдер, в скорости не ограничиваются.

Обратное кэширование

Одной из особенностей Squid является возможность работать в режиме «обратного прокси-сервера» («reverse proxy»), так же известного как «ускоритель» («HTTP accelerator»). В этом случае вместо кэширования запросов нескольких пользователей к множеству сайтов, кэшируются запросы множества пользователей к нескольким сайтам. В этом режиме принятый запрос проверяется на «динамичность» (нужно ли каждый раз обрабатывать запрос с нуля) и «возраст» (актуальны ли ещё данные). Если данные ещё актуальны и не поменялись, то запрос не передаётся серверу, а отдаётся из кеша Squid. Таким образом, существенно снижается нагрузка на серверы (например, в Википедии запросы к страницам кэшируются, так как от просмотра их содержимое не меняется, при этом нагрузка на серверы существенно меньше — обработка запроса к кэшу много проще, чем обработка запроса к базе данных SQL, обработка вики-разметки и формирование веб-страницы).

Кроме того, «обратный прокси-сервер» способен распределять запросы между несколькими серверами, балансируя нагрузку и/или обеспечивая отказоустойчивость, то есть фактически предоставляет функциональность, аналогичную кластеру.

Режим прозрачного прокси-сервера

В сочетании с некоторыми межсетевыми экранами и маршрутизаторами Squid может работать в режиме прозрачного прокси-сервера (англ. transparent proxy). В этом режиме маршрутизатор вместо того, чтобы сразу пересылать HTTP-запросы пользователя HTTP-серверу в Интернете, перенаправляет их прокси-серверу, который может работать как на отдельном хосте, так и на самом маршрутизаторе. Прокси-сервер обрабатывает запрос (с возможной отдачей содержимого из кеша), это содержимое направляется к запросившему пользователю, для которого оно выглядит как «ответ» сервера, к которому адресовался запрос. Таким образом, пользователь может даже не знать, что все запросы и ответы прошли через прокси-сервер.

При таком подходе проксирования аутентификация не предусмотрена, так как прозрачность проксирования это и подразумевает.

2. Рабочий проект

2.1 Установка Squid

Squid – приложение позволяющее организовать прокси/кэширующий сервер для HTTP, FTP и некоторых других популярных протоколов. Поддерживается работа с защищенными TLS/SSL соединениями, кэширование DNS, возможно использование Squid в качестве прозрачного или реверсного прокси. Распространяется по лицензии GNU GPL. Работает во всех популярных вариантах Unix систем – GNU/Linux, \*BSD, Mac OS X, SunOS/Solaris, и некоторых других. Есть версия для Windows.

Для примера будет использоваться Linux Mint, но все сказанное касается и всех остальных дистрибутивов или ОС, за исключением особенностей установки в конкретном решении. Хотелось бы также отметить, что сейчас параллельно развивается две ветки: 2-x и 3-x. Третья ветка перешла в разряд STABLE в конце 2008 года и рекомендуема к использованию. По параметрам описываемых далее отличий у них практически нет, поэтому все описанное касается обеих версий.

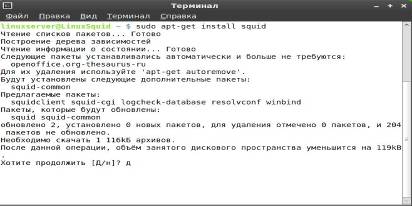


Рис. 3. Установка Squid в Mint.

После инсталляции Squid будет запущен с установками по умолчанию.

2.2 Настройка конфигурации

Все настройки Squid производятся в единственном файле /etc/Squid/Squid.conf, параметров внутри очень много. Просмотреть список параметров, убрав пустые и закомментированные строки, можно при помощи команды (рис. 4):

$ sudo grep -v «^#» /etc/Squid/Squid.conf | sed -e /^$/d’

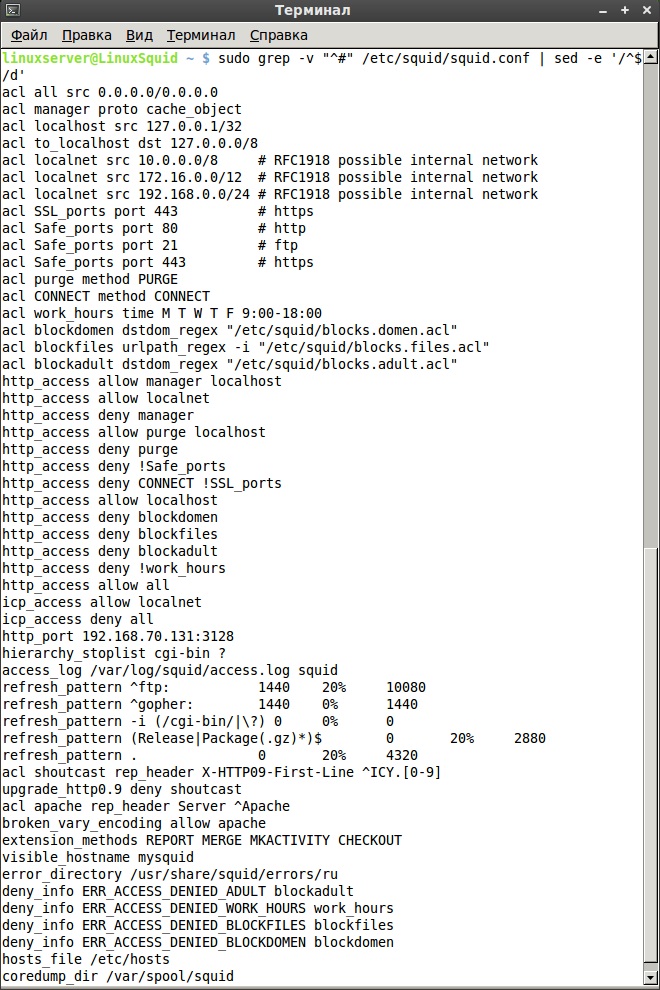


Рис. 4. Список параметров.

Создание acl (Access Control List) с именем all для абсолютно всех ip-адресов:

acl all src 0.0.0.0/0.0.0.0

Создание acl (Access Control List) с именем localhost для 127.0.0.1/32 ip-адресов:

acl localhost src 127.0.0.1/32

Создание acl (Access Control List) с именем to\_localhost для 127.0.0.0/8 ip-адресов:

acl to\_localhost dst 127.0.0.0/8

Указание сети, с которой можно присоединяться без авторизации:

acl localnet src 10.0.0.0/8

acl localnet src 172.16.0.0/12

acl localnet src 192.168.0.0/24

Описание портов:

acl SSL\_ports port 443 – https порт

acl Safe\_ports port 80 – http порт

acl Safe\_ports port 21 – ftp порт

acl Safe\_ports port 443 – https порт

Включение поддержки проброски соединения с помощью команды протокола CONNECT:

acl CONNECT method CONNECT

Описывает рабочее время с понедельника по пятницу:

acl work\_hours time M T W T F 9:00-18:00

Описывает путь к файлу со списком доменов:

acl blockdomen dstdom\_regex "/etc/squid/blocks.domen.acl" – в этом файде содержатся список доменов.

Описывает путь к файлу со списком файлов:

acl blockfiles urlpath\_regex -i "/etc/squid/blocks.files.acl" – в этом файде содержатся данные о расширениях.

Описывает путь к файлу со списком значений адресса:

acl blockadult dstdom\_regex "/etc/squid/blocks.adult.acl" – в этом файде содержатся регулярные выражения для интернет ресурсов.

Пропуск (allow) или запрет (deny) для указанных портов. Порядок http\_acces важен, идет сверху вниз:

http\_access allow manager localhost

http\_access allow localnet

http\_access deny manager

http\_access allow purge localhost

http\_access deny purge

http\_access deny!Safe\_ports

http\_access deny CONNECT!SSL\_ports

http\_access allow localhost

http\_access deny blockdomen

http\_access deny blockfiles

http\_access deny blockadult

http\_access deny!work\_hours

Разрешение acl all доступ:

http\_access allow all

Разрешение или запрет доступа к ICP порту, основанное на заявленных списках доступа:

icp\_access allow localnet

icp\_access deny all

Адреса сокетов, на которых Squid будет ожидать запросы HTTP клиентов:

http\_port 192.168.70.131:3128

В этих файлах размещаются журналы запросов клиентов. На каждый HTTP и ICP запрос отводится одна строка:

access\_log /var/log/squid/access.log squid

Этот тэг определяет имя хоста(hostname), которое будет отображатся в сообщениях об ошибках, и т.д. в данном случае используется имя mysquid:

visible\_hostname mysquid

Директория ошибок:

error\_directory /usr/share/squid/errors/ru

Выводит ошибки для определенных ACL:

deny\_info ERR\_ACCESS\_DENIED\_ADULT blockadult

deny\_info ERR\_ACCESS\_DENIED\_WORK\_HOURS work\_hours

deny\_info ERR\_ACCESS\_DENIED\_BLOCKFILES blockfiles

deny\_info ERR\_ACCESS\_DENIED\_BLOCKDOMEN blockdomen

Расположение локальной базы данных связей IP адрес-имя узла:

hosts\_file /etc/hosts

По умолчанию Squid оставляет файлы ядра в папке, из которой он был запущен:

coredump\_dir /var/spool/squid

Формат Squid.conf стандартен для Unix, каждая запись состоит из строк вида: параметр значение.

Возможно использование переменных. Cтроки начинающиеся со знака решетки (#) являются комментариями. Для удобства настройки, все параметры разбиты по секциям. Такое разбиение чисто условно и можно прописывать свои параметры в любое место файла, лишь бы было понятно. Возможно подключение внешнего файла с настройками при помощи include. Единственное о чем следует помнить – установки применяются в порядке очередности. После установки в /usr/share/doc/Squid можно найти документацию и примеры конфигурационных файлов.

2.3 Запуск прокси-сервера Squid

Для запуска прокси-сервера Squid используется команда (Рис. 5):

$ sudo /etc/init.d/Squid start

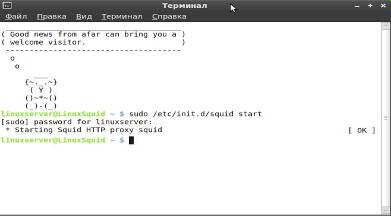


Рис. 5. Запуск Squid.

Так же нужно настроить клиентские машины для доступа в интернет через прокси-сервер Squid (Рис. 6).

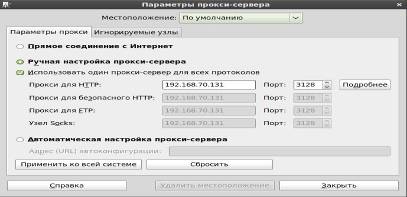


Рис. 6. Направление всего трафика через прокси-сервер.

Теперь при попытке доступа к заблокированным ресурсам вместо них будут открываться надписи со сведениями причины блокировки:

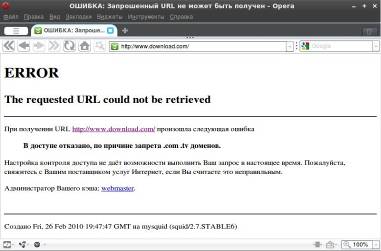


Рис. 7. Запрет доступа к развлекательным и зарубежным доменам.

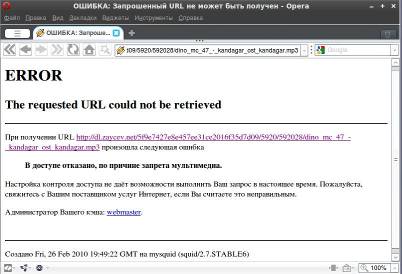


Рис. 8. Запрет доступа к файлам мультимедиа.



Рис. 9. Запрет доступа к порнографии.



Рис. 10. Запрет доступа в не рабочее время.

Заключение

В настоящее время на рынке программного обеспечения предствленно множество разнообразных программных прокси-серверов. Большинство из них имеет два основных недостатка: они коммерческие и не поддерживают ICP (ICP используется для обмена информации о наличии URL в соседнем кэше). Squid – это лучший выбор для кэширующего прокси-сервера, так как он надежный, бесплатный и поддерживает ICP. В настоящее время это наиболее производительный прокси-сервер, превосходящий по функциональности Microsoft ISA Server 2000.

Производный от “кэширующего” программного обеспечения ARPA-funded Harvest research project, разработанного в National Laboratory for Applied Network Research and funded by the National Science Foundation, Squid предлагает высокопроизводительное кэширование для веб клиентов, он также поддерживает FTP, HTTP и HTTPS объекты данных. Squid хранит часто используемые объекты в RAM, поддерживает надежную базу данных объектов на диске, имеет комплексных механизм контроля доступа и поддерживает SSL протокол для посредничества в безопасных соединениях. В дополнение к этому, он поддерживает иерархические связи с другими прокси-серверами, базирующимися на Squid. Squid ведет достаточно подробные логи об интернет-активности пользователей.

Squid используется в UNIX-like системах и в ОС семейства Windows NT. Имеет возможность взаимодействия с Active Directory Windows Server путём аутентификации через LDAP, что позволяет использовать разграничения доступа к интернет ресурсам пользователей, которые имеют учётные записи на Windows Server, также позволяет организовать «нарезку» интернет трафика для различных пользователей.

Список литературы

1. Бруй В. В., Карлов С. В. Б67 “LINUX-сервер: пошаговые инструкции инсталляции и настройки.” – М.: Изд-во СИП РИА, 2003. – 572 с. ISBN 5-89354-153-7
2. http://www.Squid-cache.org/- Домашняя страница проекта Squid
3. http://Squid.visolve.com/ - Руководство, советы по настройке
4. http://Squid.opennet.ru/ - FAQ, форум, ссылки на русскоязычные ресурсы, посвященные Squid
5. http://www.bog.pp.ru/ - Установка, настройка и использование
6. http://www.break-people.ru/ - Файл Squid.conf на русском, по секциям