Прокси

В интернете часто можно встретить заявления, что прокси-сервер служит для ускорения загрузки за счет кэширования содержимого страниц, DNS и т.п. На самом деле это только одно из полезных свойств прокси-серверов, но не главное и присущее далеко не всем серверам.

Из многочисленных значений английского слова "proxy" в данном контексте применимы следующие: «доверенное лицо», «полномочный представитель». То есть некто, действующий от вашего имени и по вашему поручению. В компьютерной терминологии «прокси» — это программа, которая передает запросы вашего ПО в Интернет, получает ответы и передает их обратно. Необходимость в такой программе обычно возникает, если с пользовательского места нельзя работать в Интернете напрямую, но выход есть на другом компьютере в его локальной сети. Тогда на этом другом компьютере ставят прокси, а софт на всех остальных компьютерах сети настраивают таким образом, чтобы работа велась через него. Сейчас через прокси умеют работать практически все более-менее популярные программы, так или иначе предназначенные для работы с сетью. Таким образом, если хоть один компьютер в локальной сети имеет доступ «вовне», то и все остальные могут пользоваться этим соединением.

Возможна ли такая работа без прокси? Да, возможна — когда-то все так делали. Но такая работа требует выполнения определенных дополнительных условий и имеет свои минусы в сравнении с использованием прокси.

Условия следующие: каждому компьютеру должен быть выдан персональный «внешний» IP-адрес и построена схема маршрутизации. Это (получение IP) невозможно сделать без участия провайдера. Провайдеры идут на это, но, как правило, за отдельную плату — например, $5 за каждый IP в месяц. А если у вас сеть хотя бы в 50 машин? Это хлопотно, дорого и снижает уровень безопасности вцелом: каждый компьютер вашей сети станет потенциальной мишенью хакеров, вирусных атак и прочих «прелестей» Интернета. При правильной настройке соответствующего ПО это не очень страшно, но рядовые пользователи не склонны следить за безопасностью своих компьютеров, значит, будут дополнительные хлопоты у администраторов сети. И возможностей контролировать работу пользователей у администратора будет меньше, так как система децентрализована. Кстати, при таком способе подключения тоже нужна программа-посредник на том компьютере, который непосредственно подключен к интернету. Но при наличии реальных IP-адресов эта программа — обычный роутер (маршрутизатор) IP-пакетов. Она является либо частью операционной системы, либо, вообще, «встроена» в соответствующее оборудование. И к этой программе название «прокси» не применяют. Важное отличие маршрутизатора от прокси — при использовании маршрутизатора IP-пакеты остаются без изменений — в них сохраняются исходные адреса компьютеров локальной сети. А прокси всегда работает от своего «имени» и адреса клиентов не передаются, т.е. недоступны из Интернета. Маршрутизатор, меняющий адреса, уже является прокси (его называют NAT-proxy, NAT = network address translation).

1.1 Виды прокси

Упомянутый выше NAT-proxy — самый простой вид прокси. Теперь он даже входит в состав Windows 2000 и XP (а также в 98SE и Me — прим.ред.). Там он называется «Общий доступ к подключению интернета» и включается галочкой в свойствах модемного соединения. Этот прокси работает прозрачно для пользователя, никаких специальных настроек в программах не требуется. Но на этом удобства этого прокси заканчиваются. Влиять на работу «общего доступа» Windows (например, ограничивать список доступных

сайтов для отдельных пользователей) вы не сможете. Другие NAT-proxy могут быть более гибкими, но их общая проблема — универсальность. Они не «вникают» в тонкости тех прикладных протоколов, которые через себя пропускают, поэтому и не имеют средств управления ими. Специализированные прокси (свой вид для каждого протокола) имеют ряд преимуществ и с точки зрения администраторов, и с точки зрения пользователей. Ниже перечислены виды специализированных прокси.

1.2 HTTP-proxy

HTTP-proxy — самый распространенный. Он предназначен для организации работы браузеров и других программ, использующих протокол HTTP. Браузер передает прокси-серверу URL ресурса, прокси-сервер получает его с запрашиваемого веб-сервера (или с другого прокси-сервера) и отдает браузеру. У HTTP-proxy широкие возможности при выполнении следующих запросов:

Можно сохранять полученные файлы на диске сервера. Впоследствии, если запрашиваемый файл уже скачивался, то можно выдать его с диска без обращения в интернет — увеличивается скорость и экономится внешний трафик (который может быть платным). Эта опция называется кэшированием и именно её очень любят администраторы и пользователи — настолько, что считают её главной функцией прокси. Однако приводимые оценки экономии (в описаниях встречалось от 30 до 60%) слишком оптимистичны. На деле получается не более 10-15%: современный Интернет очень динамичен, страницы часто формируются «на лету» и т.д. — такие данные кэшировать нельзя (веб-серверы обычно вставляют в HTTP-заголовки специальные указания об этом). Хотя многие прокси можно настроить так, чтобы эти указания частично игнорировались — например, перечитывать страницу не чаще одного раза в день…

Можно ограничивать доступ к ресурсам. Например, завести «черный список» сайтов, на которые прокси не будет пускать пользователей (или определенную часть пользователей, или в определенное время и т.д.). Ограничения можно реализовать по-разному. Можно просто не выдавать ресурс — например, выдавая вместо него страницу «запрещено администратором» или «не найдено». Можно спрашивать пароль и авторизованных пользователей допускать к просмотру. Можно, не спрашивая пароля, принимать решение на основании адреса или имени компьютера пользователя. Условия и действия могут быть очень разными.

Можно выдавать не тот ресурс, который запрашивается браузером. Например, вместо рекламных баннеров и счетчиков показывать пользователям прозрачные картинки, не нарушающие дизайн сайта, но существенно экономящие время и трафик за счет исключения загрузки картинок извне.

Можно ограничивать скорость работы для отдельных пользователей, групп или ресурсов. Например, установить правило, чтобы файлы *.mp3 закачивались на скорости не более 1кб/сек, дабы предотвратить забивание вашего канала трафиком меломанов, но не лишать их полностью этого удовольствия. Эта возможность, к сожалению, есть не во всех прокси. А присутствует, к примеру, в Eproxy. Она реализуется дополнением TrafC, который кроме ограничения пропускной способности (скорости) может ограничивать и суммарный трафик.

Ведутся журналы работы — можно подсчитывать трафик за заданный период, по заданному пользователю, выяснять популярность тех или иных ресурсов и т.д.

Можно маршрутизировать запросы — например, часть направлять напрямую, часть через другие прокси (прокси провайдера, спутниковые прокси и т.д.). Это тоже помогает эффективнее управлять стоимостью трафика и скоростью работы прокси вцелом.

Я перечислил основные, но не все возможности HTTP-proxy. Но уже одних перечисленных достаточно чтобы убедиться, что без HTTP-proxy в серьезной сети не обойтись.

1.3 FTP-proxy

FTP-proxy бывает двух основных видов в зависимости от протокола работы самого прокси. С ftp-серверами этот прокси, конечно, всегда работает по протоколу FTP. А вот с клиентскими программами — браузерами и ftp-клиентами (CuteFTP, FAR, и др.) прокси может работать как по FTP, так и по HTTP. Второй способ удобнее для браузеров, т.к. исторически является для них «родным». Браузер запрашивает ресурс у прокси, указывая протокол целевого сервера в URL — http или ftp. В зависимости от этого прокси выбирает протокол работы с целевым сервером, а протокол работы с браузером не меняется — HTTP. Поэтому, как правило, функцию работы с FTP-серверами также вставляют в HTTP-proxy , т.е. HTTP-proxy , описаный выше, обычно с одинаковым успехом работает как с HTTP, так и с FTP-серверами. Но при «конвертации» протоколов FTP<->HTTP теряется часть полезных функций протокола FTP. Поэтому специализированные ftp-клиенты предпочитают и специальный прокси, работающий с обеими сторонами по FTP. Такой прокси называется FTP-gate, чтобы подчеркнуть отличие от FTP-proxy в составе HTTP-proxy . Также этот прокси называется и в некоторых ftp-клиентах. Хотя встречаются и вносящие еще большую путаницу названия: например, в программе CuteFTP FTP-gate называют firewall, хотя firewall, в общем случае, — это вообще не прокси, а, фактически, программа обратного назначения — не для подключения к интернету, а для изоляции от него. FTP-gate поддерживают различные способы указания в FTP-протоколе целевого сервера, с которым FTP-клиент хочет работать, в настройке FTP-клиентов обычно предлагается выбор этого способа, например как показано на рисунке ниже:

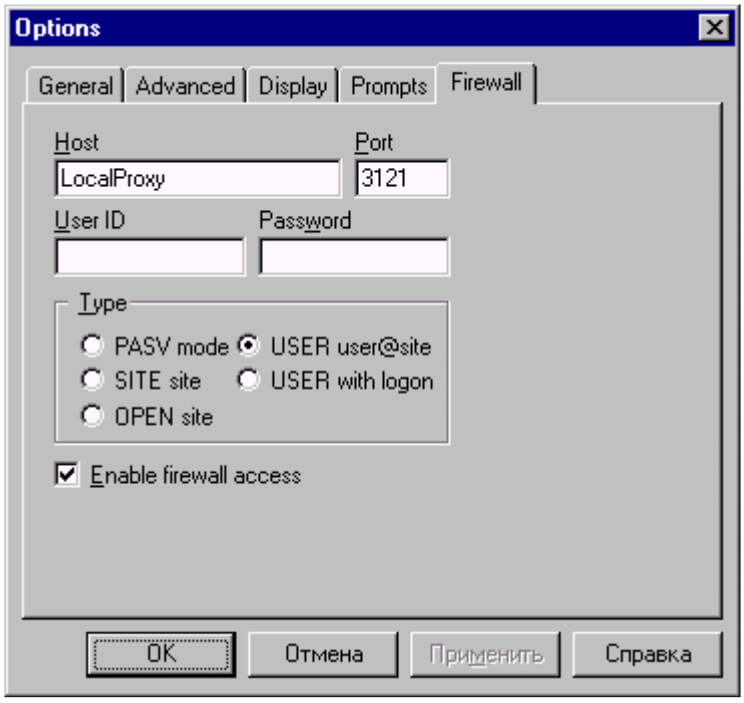

Рисунок 1.

Здесь USER user@site, OPEN site, и т.д. — способ указания сервера, с которым производится работа. Такое многообразие связано с тем, что нет общепринятого стандарта на этот вид прокси, и применяются такие хитрые добавки к стандартным командам FTP-протокола.

1.4 HTTPS-прокси

HTTPS-прокси — фактически часть HTTP-proxy. "S" в названии означает "secure", т.е. безопасный. Не смотря на то, что программно это часть HTTP-proxy , обычно HTTPS выделяют в отдельную категорию (и есть отдельное поле для него в настройке браузеров). Обычно этот протокол применяют, когда требуется передача секретной информации, например, номеров кредитных карт. При использовании обычного HTTP-proxy всю переда-

ваемую информацию можно перехватить средствами самого прокси (т.е. это под силу администратору локальной сети). Или на более низком уровне, например, tcpdump (т.е. и администратор провайдера и любого промежуточного узла и вообще — любой человек, имеющий физический доступ к маршрутам передачи ваших данных по сети, может при большом желании узнать ваши секреты). Поэтому в таких случаях применяют secure HTTP — всё передаваемое при этом шифруется. Прокси-серверу при этом дается только команда «соединится с таким-то сервером», и после соединения прокси передает в обе стороны шифрованный трафик. Но при этом отсутствует возможность узнать подробности (соответственно, отсутствуют и многие средства управления доступом — такие, как фильтрация картинок — не могут быть реализованы для HTTPS, т.к. прокси в этом случае неизвестно, что именно передается). Собственно, в процессе шифрации/дешифрации прокси тоже участия не принимает — это делают клиентская программа и целевой сервер. Наличие команды «соединиться с таким-то сервером» в HTTPS-прокси приводит к интересному и полезному побочному эффекту, которым все чаще пользуются разработчики клиентских программ. Так как после соединения с указанным сервером HTTPS-прокси лишь пассивно передает данные в обе стороны, не производя никакой обработки этого потока вплоть до отключения клиента или сервера, это позволяет использовать прокси для передачи почти любого TCP-протокола, а не только HTTP. То есть HTTPS-прокси одновременно является и простым POP3-прокси, SMTP-прокси, IMAP-прокси, NNTP-прокси и т.д. Никаких модификаций целевого сервера не требуется. Фактически HTTPS-прокси является программируемым mapping-proxy, как и Socks-proxy (о которых читайте ниже).

1.5 Mapping-proxy

Mapping-proxy — способ заставить работать через прокси те программы, которые умеют работать с Интернетом только напрямую. При настройке такого сервиса администратор как бы создает «копию» целевого сервера, но доступную через один из портов прокси-сервера для всех клиентов локальной сети — устанавливает локальное «отображение» заданного сервера. Например, пользователи локальной сети хотят работать с почтовым сервером Mail.ru не через браузер, а с использованием почтовой программы Outlook Express или TheBat. Эти программы не умеют работать через прокси (кроме случая, когда Outlook получает почту по HTTP с hotmail.com — тогда он, как и браузер, пользуется HTTP-proxy). Простейший способ работать с Mail.ru по POP3, т.е. установить через прокси локальное отображение сервера pop.mail.ru. И в Outlook вместо pop.mail.ru написать имя прокси-сервера и порт отображения. Outlook будет соединяться с прокси-сервером («думая», что это почтовый сервер), а прокси при этом будет соединяться с pop.mail.ru и прозрачно передавать всю информацию между Outlook и pop.mail.ru, таким образом «превращаясь» на время соединения в POP3-сервер. Неудобство Mapping-proxy в том, что для каждого необходимого внешнего сервера нужно вручную устанавливать отдельный порт на прокси. Но зато не требуется модификации ни серверов, ни клиентов. Особенно это помогает в случае необходимости «проксирования» многочисленных «доморощенных» протоколов, реализованных в играх или финансовых программах. Почему-то они часто игнорируют существование прокси и стандартных протоколов. Такие программы можно «обмануть» и направить через прокси практически всегда, если они не делают другой глупости — передачи клиентского IP-адреса внутри протокола и пытаются с ним соединяться напрямую еще раз (что невозможно, т.к. локальные адреса недоступны извне).

1.6 Socks-proxy

Socks-proxy. Об этом виде прокси будет вестись отдельный разговор, так как на данный момент это самый «прогрессирующий» протокол. Итак, начнем…

Разработан он Дейвом Кобласом (Dave Koblas) из компании SGI. С начала существования протокол пережил несколько больших модификаций, а на сегодняшний день «в работе» находятся две версии протокола: Socks4 и Socks5. Не смотря на то, что Socks5 бо-

лее «продвинут», сейчас с одинаковой степени распространены сервера как с поддержкой старой, так и новой версии. Протокол представляет собой транслятор (что-то вроде прокси сервера), но в отличие от обычных прокси Socks-клиент «сидит» между прикладным и транспортным уровнем в сетевой модели, а Socks-сервер находиться на прикладном уровне. Это означает, что такой сервер не привязан больше к протоколам высокого уровня. Сам протокол разработан для того, чтобы приложения, работающие на основе tcp и udp, могли использовать ресурсы сети, доступ к которым ограничен архитектурой или настройками (например, доступ к ресурсам Интернета из локальной сети для приложений, у которых вообще не предусмотрена работа с использованием прокси).

Теперь рассмотрим отличия версий протокола. Socks4 решает вопрос незащищенного пересечения межсетевых экранов приложениями клиент/сервер, основанными на протоколе TCP. Socks5 (RFC 1928), является дальнейшим расширением четвертой версии и включает в себя UDP. Он расширяет общую рамочную структуру (придавая ей, возможность использования мощных обобщенных схем идентификации) и систему адресации, включая в нее имя домена и адреса IP v.6. Если выражаться короче, то Socks4 поддерживает tcp, а Socks5 поддерживает tcp, udp, авторизацию и удаленный dns-запрос.

Теперь можно рассказать, каким образом возможно применение этой технологии для анонимности в Интернет. Для начала скажу, что так как главной «рабочей единицей» в WWW является браузер, использующий протокол HTTP (который, в свою очередь, работает на tcp), то нам безразлично, какую версию Socks мы будем использовать.

Вследствие того, что Socks не имеет никакого отношения к http, то данный вид прокси игнорирует все вопросы, связанные с модернизацией заголовков http-запросов. Socks-сервер будет передавать все данные в чистом виде от первого лица — то есть от себя. Другими словами можно сказать (используя терминологию из http), что все Socks-серверы «анонимны». Socks не передает информацию о вашем ip-адресе, потому что это не предусмотрено его технологией. Соответственно отпадает множество проблем — например, кроме того, что он не передает ip-адрес, он, естественно, не модернизирует http-заголовки — и web-сервер никаким образом не может определить, что вы используете прокси. Для него работа с вами будет абсолютно прозрачной, как если бы вы работали с ним непосредственно. С той лишь разницей, что он будет видеть совсем другой ip-адрес.

Помните проблему с использованием различных прокси серверов для разных протоколов (http, ftp, https и т.д.)? Так как все эти протоколы (в браузере) работают на основе tcp, то Socks-прокси без проблем берет их всех на себя, то есть больше не надо мучаться, прописывая для каждого протокола свой прокси-сервер, а тем более — искать их. Достаточно одного Socks.

Технология Socks легко поддерживает выстроение в цепь. Здесь следует отметить, что некоторые http прокси-серверы тоже могут выстраиваться в цепь, но в этом случае возникает много проблем. Во-первых, из 100% рабочих прокси-серверов, анонимными будут процентов 10, из них, возможно, 1% будет поддерживать возможность перенаправлять запросы, то есть выстраиваться в цепь. Во-вторых, использование такой возможности http прокси браузером прямо не предусмотрена, но если все же использовать некоторые методы для этого, то останется множество брешей, главной из которых будет потенциальная возможность передачи данных напрямую, минуя прокси. Таким методом я не пользовался и не собираюсь, для этого есть средство лучше — это Socks.

Что дает возможность выстраивания в цепь Socks-серверов? Думаю это очевидно, Socks-серверы могут находиться в разных частях планеты, передавая информацию друг

другу по вашему желанию. Все данные, которые «ходят» между браузером и web-сервером, будут передаваться через все серверы, которые вы выстроили в цепь, возможно, не раз обойдя земной шар, пока не достигнут цели. Ваша анонимность обеспечена.

Основная сложность работы с Socks состоит в том, что кто-то должен проводить работу по замене сетевых системных вызовов версиями SOCKS (этот процесс обычно называется «SOCKSификацией» (Socksify) приложения). Другими словами нужно, каким-нибудь методом заставить приложение поддерживать Socks протокол. Уже во многих операционных системах этот процесс встроен в саму ос, (например, на машинах Solaris) можно автоматически «SOCKSифицировать» приложение, поставив общую библиотеку SOCKS перед "shared libc" в вашей строке поиска библиотек (переменная среды LD_LIBRARY_PATH в системах Solaris). В Linux SOCKS-ифицировать приложение тоже довольно просто: runsocks-приложение. Как дело обстоит в Win32?. К счастью и в Windows можно сделать такого рода поддержку. Об этом позаботились создатели программы SocksCap фирмы NEC USA, Inc. Эта программа с графическим интерфейсом позволит легко «SOCKSифицировать» почти любое приложение, использующее сеть на основе tcp/ip.

В настройках браузера должно быть отключено использование прокси-сервера (установлено непосредственное подключение к Интернету).

1.7 Цепь из Socks

Цепь из прокси серверов представляет собой аналогию этой самой конструкции — то есть цепи. Здесь все очевидно: первое звено является www-клиентом (браузер), последнее — www-сервер. Между ними можно поставить неограниченное количество Socks-серверов. Браузер и SocksCap не поддерживают этот режим работы. Зато есть программа, которая называется SocksChain, с помощью которой можно выстраивать Socks-прокси в цепочки. Однако возможны и проблемы: даже если один из серверов в цепи будет неработоспособен — вся цепь не будет работать. Сначала идите на страницу проверки — в случае, когда в поле REMOTE_ADDR стоит ip-адрес последнего Socks-сервера в цепи, все работает нормально. Если сделать слишком длинную цепь — работа может существенно замедлиться.