# Прокси-серверы

*БГА, РТФ*
*Кафедра ИБ*

**Зензин Александр Степанович, к.т.н.**

# *Обзор*

1. Функции прокси-сервера
2. Прокси-серверы прикладного уровня и уровня соединений
3. «Проксификация» приложений
4. Системы обнаружения вторжений

# *Функции прокси-сервера*

**Прокси-сервер** — это особый тип *приложения*, которое выполняет функции посредника между клиентскими и серверными частями распределенных сетевых приложений, причем предполагается, что клиенты принадлежат внутренней (защищаемой) сети, а серверы — внешней (потенциально опасной) сети.

Роль транзитного узла позволяет прокси-серверу логически разорвать прямое соединение между клиентом и сервером с целью контроля процесса обмена сообщениями между ними.

Подобно сетевому экрану, прокси-сервер может эффективно выполнять свои функции только при условии, что контролируемый им трафик не пойдет обходным путем.

Прокси-сервер может быть установлен не только на платформе, где работают все остальные модули сетевого экрана (рис. 1, а), но и на любом другом узле внутренней сети или сети периметра (рис. 1, б). В последнем случае программное обеспечение клиента должно быть сконфигурировано таким образом, чтобы у него не было возможности установить прямое соединение с ресурсным сервером, минуя прокси-сервер.

# Функции прокси-сервера

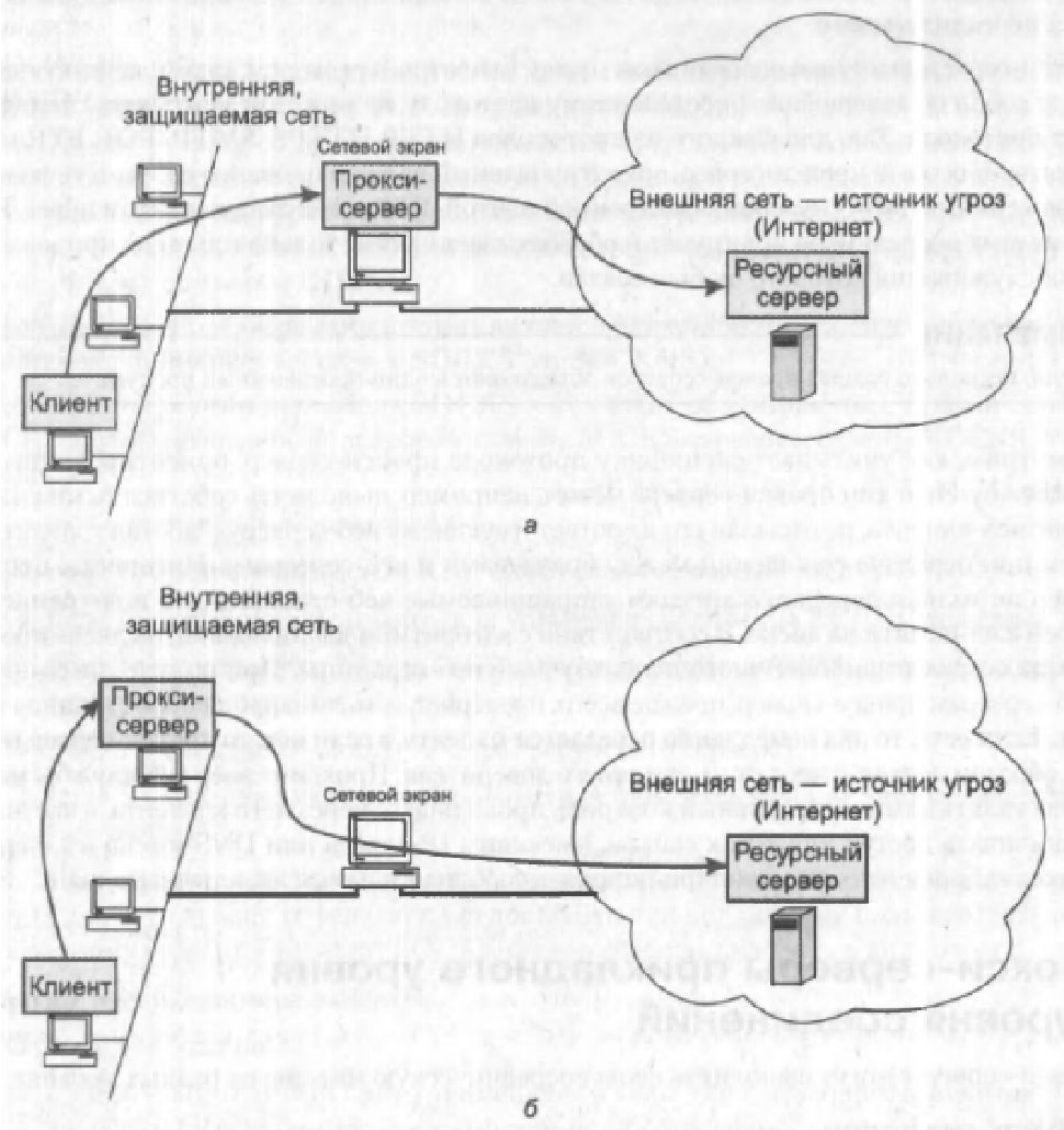


Рис. 1. Варианты расположения прокси-серверов: а — на сетевом экране, б — на узле внутренней сети

# *Функции прокси-сервера*

Когда клиенту необходимо получить ресурс от какого-либо сервера (файл, веб-страницу, почтовое сообщение), он посылает свой запрос прокси-серверу. Прокси-сервер анализирует этот запрос на основании заданных ему администратором правил и решает, каким образом он должен быть обработан (отброшен, передан без изменения ресурсному серверу, модифицирован тем или иным способом перед передачей, немедленно обработан силами самого прокси-сервера).

В качестве правил, которыми руководствуется прокси-сервер, могут выступать *условия пакетной фильтрации*. Правила могут быть достаточно сложными, например в рабочие часы блокируется доступ к тем или иным узлам и/или приложениям, а доступ к другим узлам разрешается только определенным пользователям, причем для FTP-серверов пользователям разрешается делать лишь загрузку, а выгрузка запрещается. Прокси-серверы могут также фильтровать почтовые сообщения по типу пересылаемого файла (например, запретить получение сообщений формата MP3) и по их контенту. К разным пользователям могут применяться разные правила фильтрации, поэтому часто на прокси-серверы возлагается задача аутентификации пользователей.

Если после всесторонней оценки запроса от приложения прокси-сервер констатирует, что запрос удовлетворяет условиям прохождения дальше во внешнюю сеть, то он выполняет **по поручению приложения**, **но от своего имени** процедуру соединения с сервером, затребованным данным приложением.

# *Функции прокси-сервера*

В некоторых случаях прокси-сервер может изменять запрос клиента. Например, если в него встроена функция трансляции сетевых, он может подменять в пакете запроса IP-адреса и/или номера TCP- и UDP-портов отправителя. Таким способом прокси-сервер лишает злоумышленника возможности сканировать внутреннюю сеть для получения информации об адресах узлов и структуре сети. Единственный адрес в таком случае, который может узнать злоумышленник, — это адрес компьютера, на котором выполняется программа прокси-сервера. Поэтому многие атаки, построенные на знании злоумышленником адресов узлов внутренней сети, становятся нереализуемыми.

Прокси-сервер, выступая посредником между клиентом и сервером, взаимодействующими между собой по совершенно определенному протоколу, не может не учитывать специфику этого протокола. Так, для каждого из протоколов HTTP, HTTPS, SMTP/POP, FTP, telnet существует особый прокси-сервер, ориентированный на использование соответствующими приложениями: веб-браузером, электронной почтой, FTP-клиентом, клиентом telnet. Каждый из этих посредников принимает и обрабатывает пакеты только того типа приложений, для обслуживания которого он был создан.

**ПРИМЕЧАНИЕ**

*Обычно несколько разных прокси-серверов объединяют в один программный продукт*.

# *Функции прокси-сервера*

Посмотрим, как учитывает специфику протокола прокси-сервер, ориентированный на веб-службу. Этот тип прокси-сервера может, например, выполнить собственными силами запрос веб-клиента, не отсылая его к соответствующему веб-серверу. Работая транзитным узлом при передаче сообщений между браузерами и веб-серверами Интернета, прокси-сервер не только передает клиентам запрашиваемые веб-страницы, но и сохраняет их в своей кэш-памяти на диске. В соответствии с алгоритмом кэширования, на диске прокси-сервера оседают наиболее часто используемые веб-страницы. При получении запросов к веб-серверам прокси-сервер, прежде всего, проверяет, есть ли запрошенная страница в его кэше. Если есть, то она немедленно передается клиенту, а если нет, то прокси-сервер обычным образом делает запрос от имени своего доверителя.

Прокси-сервер веб-службы может осуществлять административный контроль проходящего через него контента, в частности ограничивать доступ клиента к сайтам, имеющим IP-адреса или DNS-имена из «черных списков». Более того, он может фильтровать сообщения на основе ключевых слов.

# *Прокси-серверы прикладного уровня и уровня соединений*

Прокси-серверы могут выполнять свою посредническую миссию на разных уровнях.

**ПРИМЕР-АНАЛОГИЯ**

*Рассмотрим пример, иллюстрирующий идею посредничества разного уровня. Для покупки акций инвестор (в нашем случае аналог клиентской части приложения) может прибегнуть к посредническим услугам брокера или трейдера. Брокер, точно следуя указаниям инвестора, покупает для него определенное количество акций определенного типа по определенной цене. Трейдер — это посредник более высокого уровня, которому инвестор поручает самостоятельно принимать решения о необходимых покупках, учитывая различные факторы, например состояние рынка.*

Различают прокси-серверы прикладного уровня и уровня соединений.

**Прокси-сервер прикладного уровня**, как это следует из его названия, умеет «вклиниваться» в процедуру взаимодействия клиента и сервера по одному из прикладных протоколов, например тому же HTTP, HTTPS, SMTP/POP, FTP или telnet. Чтобы выступать в роли посредника на прикладном уровне, прокси-сервер должен «понимать» смысл команд, «знать» форматы и последовательность сообщений, которыми обмениваются клиент и сервер соответствующей службы. Это дает возможность прокси-серверу проводить анализ содержимого сообщений, делать заключения о подозрительном характере того или иного сеанса.

# *Прокси-серверы прикладного уровня и уровня соединений*

**Прокси-сервер уровня соединений** выполняет свою посредническую миссию на транспортном уровне, контролируя TCP-соединение. Очевидно, что работая на более низком уровне, прокси-сервер обладает гораздо меньшим «интеллектом» и имеет меньше возможностей для выявления и предупреждения атак. Однако он обладает одним очень важным преимуществом перед прокси-сервером прикладного уровня — универсальностью, то есть он может быть использован любыми приложениями, работающими по протоколу TCP (а в некоторых случаях и UDP).

Примером прокси-сервера данного типа является разработанный достаточно давно, но все еще широко применяемый сервер **SOCKS** (от SOCKetS). В простейшей версии протокола SOCKS V4 клиент обменивается с прокси-сервером SOCKS двумя сообщениями: *запросом клиента* SOCKS-серверу и *ответом* SOCKS-сервера клиенту.

- *Запрос клиента* SOCKS-серверу:
  - поле 1 — номер версии SOCKS, 1 байт (для этой версии — 4);
  - поле 2 — код команды, 1 байт (для установки соединения TCP/IP код равен 1);
  - поле 3 — номер порта, 2 байта (TCP-порт запрашиваемого пользователем ресурсного сервера, например, для 21 для FTP);
  - поле 4 — IP-адрес, 4 байта (IP-адрес ресурсного сервера);
  - поле 5 — идентификатор пользователя (строка переменной длины, завершаемая байтом null).

SOCKS-сервер анализирует все полученные данные и на основании сконфигурированных для него правил определяет, предоставить или нет данному пользователю доступ к данному серверу.

# *Прокси-серверы прикладного уровня и уровня соединений*

Результат SOCKS-сервер сообщает клиенту в виде *ответа*.

- *Ответ* SOCKS-сервера клиенту:
  - О поле 1 — байт null;
  - поле 2 — код ответа, 1 байт (применяются коды для следующих вариантов ответа: запрос разрешен, запрос отклонен или ошибочен, запрос не удался из-за проблем с идентификацией пользователя);
  - несколько байтов, игнорируемых клиентом.

Если прокси-сервер сообщил в ответе, что запрос разрешен, то SOCKS-сервер начинает работать промежуточном звеном между клиентом и сервером (например, FTP), контролируя поток квитанции, которыми они обмениваются.

# *«Проксификация» приложений*

Заметим, что не каждое приложение, построенное в архитектуре клиент-сервер, непременно должно работать через прокси-сервер, а также не каждое из них имеет возможность работать через прокси-сервер.

Список приложений (точнее их клиентских частей), которые должны передавать свои запросы во внешнюю сеть исключительно через прокси-сервер, определяется администратором. А чтобы эти приложения имели возможности для такого режима выполнения, их программы должны быть соответствующим образом написаны.

Точнее приложения должны быть оснащены средствами, которые распознавали бы запросы к внешним серверам и перед отправкой преобразовывали эти запросы так, чтобы все они попадали на соответствующий прокси-сервер, а не передавались в соответствии со стандартным протоколом прямо на сервер-адресат. Эти средства должны также поддерживать протокол обмена сообщениями приложения-клиента с прокси-сервером. В последние годы в большинстве приложений, ориентированных на работу через Интернет, предусмотрена встроенная поддержка прокси-сервера. Такой поддержкой, например, оснащены все веб-браузеры и все клиенты электронной почты, которыми мы сейчас пользуемся.

**«Проксификация»** приложения, изначально не рассчитанного на работу через прокси-сервер, требует изменения исходного кода с последующей перекомпиляцией — очевидно, что такая работа не представляет сложностей для разработчиков данного приложения, но не всегда под силу обслуживающему персоналу сети.

## «Проксификация» приложений

Задача последних заключается в приобретении готовых приложений, совместимых с используемым в сети прокси-сервером. Однако даже приобретение готового «проксифицированного» клиента не делает его готовым к работе — необходимо еще конфигурирование, в частности нужно сообщить клиенту адрес узла сети, на котором установлен соответствующий прокси-сервер.

Как можно было бы предположить, процедура «проксификации» значительно упрощается для прокси-сервера уровня соединений, в частности SOCKS-сервера. Для «проксификации» приложения в этом случае достаточно внести простейшие исправления в исходный текст, а затем выполнить его перекомпиляцию и связывание с библиотекой процедур SOCKS. Исправления сводятся к замене всех стандартных вызовов сетевых функций версиями этих функций из библиотеки SOCKS, в частности стандартный вызов listen () заменяется вызовом rlisten(), вызов bind() — вызовом rbind(), вызов accept () — вызовом raccept().

Имеется еще один подход к «проксификации» — встраивание поддержки прокси-сервера в операционную систему. В этом случае приложения могут оставаться в полном «неведении» о существовании в сети прокси-сервера, за них все необходимые действия выполнит ОС. Помимо основных функций, многие прокси-серверы *способны обнаруживать вирусы* еще до того, как они попали во внутреннюю сеть. К другим полезным (для администрации и службы безопасности) вспомогательным функциям прокси-сервера относится *сбор статистических данных* о доступе пользователей в Интернет: когда и какие сайты посещал тот или иной пользователь, сколько времени продолжалось каждое посещение.

# *Системы обнаружения вторжений*

***Система обнаружения вторжений*** *(Intrusion Detection System, IDS) — это программное или аппаратное средство, предназначенное для предупреждения, выявления и протоколирования некоторых типов сетевых атак*.

В отличие от сетевых экранов и прокси-серверов, которые строят защиту сети исключительно на основе анализа сетевого трафика, системы обнаружения вторжений учитывают в своей работе различные подозрительные события, происходящие в системе.

Существуют ситуации, когда сетевой экран оказывается проницаемым для злоумышленника, например, когда атака идет через туннель VPN из взломанной сети или инициатором атаки является пользователь внутренней сети и т. п. И дело здесь не в плохой конфигурации межсетевого экрана, а в самом принципе его работы. Экран, несмотря на то что обладает памятью и анализирует последовательность событий, конфигурируется на блокирование трафика с заранее предсказуемыми признаками, например по IP-адресам или протоколам. Так что факт взлома внешней сети, с которой у него был установлен защищенный канал и которая до сих пор вела себя вполне корректно, в правилах экрана отразить нельзя. Точно так же, как и неожиданную попытку легального внутреннего пользователя скопировать файл с паролями или повысить уровень своих привилегий. Подобные подозрительные действия может обнаружить только система со встроенными агентами во многих точках сети, причем она должна следить не только за трафиком, но и за обращениями к критически важным ресурсам операционных систем отдельных компьютеров, а также иметь информацию о перечне подозрительных действий (сигнатур атак) пользователей. Таковой и является система обнаружения вторжений. Она не дублирует действия межсетевого экрана, а **дополняет** их, производя, кроме того, автоматический анализ всех журналов событий, имеющихся у сетевых устройств и средств защиты, чтобы попытаться найти следы атаки, если ее не удалось зафиксировать в реальном времени.