Описание функциональных характеристик

Версия 1.0

Аннотация

Настоящий документ содержит описание функциональных характеристик Angie PRO.

Angie PRO — эффективный, мощный и масштабируемый веб-сервер.

Оглавление

1 Общие сведения

Angie PRO – единственный коммерческий веб-сервер разработка которого локализована в России.

Веб-сервер — это класс программного обеспечения, предоставляющего доступ к сетевым ресурсам по протоколу HTTP конечным пользователям.

Angie PRO, например, может использоваться для работы интернет-сайтов, мобильных приложений, киосков самообслуживания в метрополитене, мультимедиа систем в поездах дальнего следования.

Каждый раз, когда пользователь открывает сайт, мобильное приложение, пользуется киоском самообслуживания в метрополитене, или даже пользуется мультимедиа системой в поезде Сапсан, запрос пользователя может быть обработан Angie PRO.

Angie PRO, это

- Веб-сервер общего назначения.

  Написан на языке C

- L4-L7 балансировщик

  Позволяет балансировать нагрузку между серверами как по протоколам TCP/UDP, так и по HTTP

- Proxy-Cache сервер

  Позволяет ускорят работу веб-сервисов с помощью гибкого механизма кеширования

- Доступен под все платформы

  Собирается и тестируется под Alpine, Debian, Oracle, RED OS, Rocky, Ubuntu

- Производительность

  Один из самых производительных веб-серверов в мире

Почему Angie PRO

- Совместимость с NGINX OSS

  Angie PRO полностью совместим с Nginx, таким образом любой существующий пользователь Nginx может без серьезных затрат и простоя сервисов перейти на Angie PRO.

- Расширенная статистика и real-time мониторинг

  Angie PRO имеет возможность полного мониторинга нагрузки сервера в режиме реального времени что позволяет динамически управлять

конфигурациями по профилю нагрузки и соблюдать полную доступность сервиса.

- Динамическое конфигурация Upstream

  Возможность управлять настройками upstream групп с помощью удобного REST интерфейса без остановки сервиса.

- Удаление элементов кэша

  Возможность удаления элементов cache через удобное API без остановки сервиса.

- Активная проверка состояния проксируемых серверов

  Проверка на "живучесть" и проксирование только на те upstream которые отвечают по заданному алгоритму.

- Динамическое хранилище ключ-значение

  Динамическое управление переменными конфигурации Angie PRO через HTTP API.

- DNS динамическое обновление
- Проксирование с привязкой сессий
- Репозиторий с динамическими сторонними модулями

  Angie PRO поддерживает большинство сторонних модулей NGINX и дает возможность без проблем устанавливать их, и предоставляет гарантию их работоспособности и поддержку.

- Синхронизация зон разделяемой памяти

  Возможно использовать зоны cache, limit_req и.т.д. в кластере Angie PRO.

- Сокрытие или персональный брендинг имени сервера в заголовках ответа

  Возможность изменить или скрыть название и версию веб-сервера от пользователей.

Перечень иностранного программного обеспечения, имеющего сходство функциональных характеристик с программным обеспечением Angie PRO: Nginx, Nginx Plus, Apache, Envoy, продукты, использующие решения NGINX (OpenResty, Tengine, Cloudfare), облачные решения Яндекса.

2 Функциональные характеристики

2.1 Операционные системы

Angie PRO может работать под управлением операционных систем Alpine, Debian, Oracle, RED OS, Rocky, Ubuntu. Версии и платформы операционных систем, для которых поддерживаются дистрибутивы приведены в руководстве по установке Angie PRO.

2.2 Функциональность управления Angie PRO

Процесс Angie PRO запускается как системный сервис. У Angie есть один главный и несколько рабочих процессов. Основная задача главного процесса — чтение и проверка конфигурации и управление рабочими процессами. Рабочие процессы выполняют фактическую обработку запросов. Angie использует модель, основанную на событиях, и зависящие от операционной системы механизмы для эффективного распределения запросов между рабочими процессами. Количество рабочих процессов задаётся в конфигурационном файле и может быть фиксированным для данной конфигурации или автоматически устанавливаться равным числу доступных процессорных ядер.

Управлять Angie можно также с помощью сигналов. Номер главного процесса по умолчанию записывается в файл /var/run/angie.pid. Изменить имя этого файла можно при конфигурации сборки или же в angie.conf директивой pid. Главный процесс поддерживает следующие сигналы: Управлять рабочими процессами по отдельности не нужно. Тем не менее, они тоже поддерживают некоторые сигналы, например: быстрое завершение, плавное завершение, переоткрытие лог-файлов, аварийное завершение для отладки.

Angie PRO позволяет осуществить управление в части:

- изменения конфигурации;
- ротации лог-файлов;
- обновления исполняемого файла на лету;
- управления из командной строки.

Изменение конфигурации

Для того чтобы Angie перечитал файл конфигурации, нужно послать главному процессу сигнал HUP. Главный процесс сначала проверяет синтаксическую правильность конфигурации, а затем пытается применить новую конфигурацию, то есть, открыть лог-файлы и новые слушающие сокеты. Если ему это не удаётся, то он откатывает изменения и продолжает работать со старой конфигурацией. Если же удаётся, то он запускает новые рабочие процессы, а старым шлёт сообщение о плавном выходе. Старые рабочие процессы закрывают слушающие сокеты и продолжают обслуживать старых клиентов. После обслуживания всех клиентов старые рабочие процессы завершаются.

Ротация лог-файлов

Лог-файлы нужно переименовать, а затем послать сигнал USR1 главному процессу. Он откроет заново все текущие открытые файлы и назначит им в качестве владельца непривилегированного пользователя, под которым работают рабочие процессы. После успешного открытия главный процесс закрывает все открытые файлы и посылает сообщение о переоткрытии файлов рабочим процессам. Они также открывают новые файлы и сразу же закрывают старые. В результате старые файлы практически сразу же готовы для дальнейшей обработки, например, их можно сжимать.

Обновление исполняемого файла на лету

Для обновления исполняемого файла сервера вначале нужно записать на место старого файла новый. Затем нужно послать сигнал USR2 главному процессу — он переименует свой файл с номером процесса в файл с суффиксом .oldbin, например, /usr/local/angie/logs/angie.pid.oldbin, после чего запустит новый исполняемый файл, а тот в свою очередь — свои рабочие процессы.

Старый процесс не закрывает свои слушающие сокеты и при необходимости ему можно сказать, чтобы он снова запустил свои рабочие процессы. Если работа нового исполняемого файла по каким-то причинам не устраивает, можно проделать одно из следующих действий:

- Послать старому главному процессу сигнал HUP. Старый главный процесс, не перечитывая конфигурации, запустит новые рабочие процессы. После этого можно плавно завершить все новые процессы, послав новому главному процессу сигнал QUIT.
- Послать новому главному процессу сигнал TERM. В ответ на это он пошлёт сообщение о немедленном выходе своим рабочим процессам, и все они практически сразу же завершатся. (Если новые процессы по каким-то причинам не завершаются, нужно послать им сигнал KILL, который заставит их завершиться.) По завершению нового главного процесса старый главный процесс автоматически запустит новые рабочие процессы.

Если новый главный процесс выходит, то старый главный процесс убирает суффикс .oldbin из имени файла с номером процесса.

Если же обновление прошло удачно, то старому процессу нужно послать сигнал QUIT, и останутся только новые процессы.

Управление из командной строки

Поддерживаются команды управления, обеспечивающие:

- вывод справки по параметрам командной строки
- использование альтернативного конфигурационного файла файл вместо файла по умолчанию.
- использование альтернативного лог-файла ошибок файл вместо файла по умолчанию
- задание глобальных директив конфигурации
- задание префикса пути angie, т.е. каталога, в котором будут находиться файлы сервера (по умолчанию — каталог /usr/local/angie)
- вывод только сообщений об ошибках при тестировании конфигурации
- отправка сигнала главному процессу: stop, quit, reopen, reload

- тестирование конфигурационного файла: Angie проверяет синтаксическую правильность конфигурации, а затем пытается открыть файлы, описанные в конфигурации
- тестирование конфигурационного файла: Angie проверяет синтаксическую правильность конфигурации, а затем пытается открыть файлы, описанные в конфигурации, а также вывод конфигурационных файлов в стандартный поток вывода
- вывод версии Angie PRO
- вывод версии Angie PRO, версии компилятора и параметров конфигурации сборки.

2.3 Функциональность конфигурирования Angie PRO

Angie PRO работает с текстовым конфигурационным файлом. Файл конфигурации *angie.conf* находится по пути conf-path, указанном при компиляции, по умолчанию в */etc/angie*.

Файл конфигурации состоит из контекстов:

- *events* – обработка соединений
- *http* – трафик HTTP
- *mail* – Mail трафик
- *stream* – TCP и UDP трафик

Директивы, расположенные вне этих контекстов, считаются директивами контекста *main*.

Функциональность конфигурирования Angie PRO включает в себя настройку:

- наследования;
- перезагрузки конфигурации;
- единиц измерения;
- настройку хэшей;
- настройку HTTPS-серверов.

Наследование

Вложенный контекст — включённый внутри родительского, — наследует директивы из родительского контекста тогда и только тогда, когда такие директивы не описаны в нём самом. При наличии, директива вложенного контекста переопределяет директиву родительского.

Перезагрузка конфигурации

При любом изменении конфигурации, для применения изменений процессов Angie PRO необходимо:

- либо перезапустить полностью, предварительно проверив конфигурацию синтаксически;

- либо перезагрузить, что позволит не прерывать обработку текущих соединений.

Единицы измерения

Размеры в конфигурационном файле можно указывать в байтах, килобайтах (суффиксы *k* и *K*) или мегабайтах (суффиксы *m* и *M*), например, *“1024”*, *“8k”*, *“1m”*.

Интервалы времени можно задавать в миллисекундах, секундах, минутах, часах, днях и т.д., используя соответствующие суффиксы.

В одном значении можно комбинировать различные единицы, указывая их в порядке от более к менее значащим, и по желанию отделяя их пробелами. Например, *“1h 30m”* задаёт то же время, что и *“90m”* или *“5400s”*. Значение без суффикса задаёт секунды. Рекомендуется всегда указывать суффикс.

Некоторые интервалы времени можно задать лишь с точностью до секунд.

Настройка хэшей

Для быстрой обработки статических наборов данных, таких как имена серверов, значения директивы map, MIME-типы, имена полей заголовков запросов, Angie использует хэш-таблицы. Во время старта и при каждой переконфигурации Angie подбирает минимально возможный размер хэш-таблиц с учётом того, чтобы размер корзины, куда попадают ключи с совпадающими хэш-значениями, не превышал заданного параметра (*hash bucket size*). Размер таблицы считается в корзинах. Подбор ведётся до тех пор, пока размер таблицы не превысит параметр *hash max size*. Для большинства хэшей есть директивы, которые позволяют менять эти параметры, например, для хэшей имён серверов директивы называются server_names_hash_max_size и server_names_hash_bucket_size.

Параметр *hash bucket size* всегда выравнивается до размера, кратного размеру строки кэша процессора. Это позволяет ускорить поиск ключа в хэше на современных процессорах, уменьшив число обращений к памяти. Если *hash bucket size* равен размеру одной строки кэша процессора, то во время поиска ключа число обращений к памяти в худшем случае будет равно двум — первый раз для определения адреса корзины, а второй — при поиске ключа внутри корзины. Соответственно, если Angie выдал сообщение о необходимости увеличить *hash max size* или *hash bucket size*, то сначала нужно увеличивать первый параметр.

Настройка HTTPS-серверов

Чтобы настроить HTTPS-сервер, необходимо включить параметр ssl на слушающих сокетах в блоке server, а также указать местоположение файлов с сертификатом сервера и секретным ключом.

Сертификат сервера является публичным. Он посылается каждому клиенту, соединяющемуся с сервером. Секретный ключ следует хранить в файле с ограниченным доступом (права доступа должны позволять главному процессу Angie читать этот файл). Секретный ключ можно также хранить в одном файле с сертификатом, при этом права

доступа к файлу следует также ограничить. Несмотря на то, что и сертификат, и ключ хранятся в одном файле, клиенту посылается только сертификат.

С помощью директив ssl_protocols и ssl_ciphers можно ограничить соединения использованием только “сильных” версий и шифров SSL/TLS.

Оптимизация HTTPS-сервера

SSL-операции потребляют дополнительные ресурсы процессора. На мультипроцессорных системах следует запускать несколько рабочих процессов, не меньше числа доступных процессорных ядер. Наиболее ресурсоёмкой для процессора является операция SSL handshake, в рамках которой формируются криптографические параметры сессии. Существует два способа уменьшения числа этих операций, производимых для каждого клиента: использование постоянных (keepalive) соединений, позволяющих в рамках одного соединения обрабатывать сразу несколько запросов, и повторное использование параметров SSL-сессии для предотвращения необходимости выполнения SSL handshake для параллельных и последующих соединений. Сессии хранятся в кэше SSL-сессий, разделяемом между рабочими процессами и настраиваемом директивой ssl_session_cache. В 1 мегабайт кэша помещается около 4000 сессий. Таймаут кэша по умолчанию равен 5 минутам. Он может быть увеличен с помощью директивы ssl_session_timeout. Вот пример конфигурации, оптимизированной под многоядерную систему с 10-мегабайтным разделяемым кэшем сессий:

Цепочки SSL-сертификатов

Некоторые браузеры могут выдавать предупреждение о сертификате, подписанном общеизвестным центром сертификации, в то время как другие браузеры без проблем принимают этот же сертификат. Так происходит потому, что центр, выдавший сертификат, подписал его промежуточным сертификатом, которого нет в базе данных сертификатов общеизвестных доверенных центров сертификации, распространяемой вместе с браузером. В подобном случае центр сертификации предоставляет “связку” сертификатов, которую следует присоединить к сертификату сервера. Сертификат сервера следует разместить перед связкой сертификатов в скомбинированном файле.

Полученный файл следует указать в директиве ssl_certificate:

Если сертификат сервера и связка сертификатов были соединены в неправильном порядке, Angie не запустится и выдаст сообщение об ошибке, поскольку Angie попытается использовать секретный ключ с первым сертификатом из связки вместо сертификата сервера.

Браузеры обычно сохраняют полученные промежуточные сертификаты, подписанные доверенными центрами сертификации, поэтому активно используемые браузеры уже могут иметь требуемые промежуточные сертификаты и не выдать предупреждение о сертификате, присланном без связанной с ним цепочки сертификатов.

Примечание

При тестировании конфигураций с SNI необходимо указывать опцию *-servername*, так как openssl по умолчанию не использует SNI.

Единый HTTP/HTTPS сервер

Можно настроить единый сервер, который обслуживает как HTTP-, так и HTTPS-запросы.

Выбор HTTPS-сервера по имени

Типичная проблема возникает при настройке двух и более серверов HTTPS, слушающих на одном и том же IP-адресе, когда браузер получает сертификат сервера по умолчанию независимо от запрашиваемого имени сервера. Это связано с поведением протокола SSL. SSL-соединение устанавливается до того, как браузер посылает HTTP-запрос, и Angie не знает имени запрашиваемого сервера. Следовательно, он лишь может предложить сертификат сервера по умолчанию.

Наиболее старым и надёжным способом решения этой проблемы является назначение каждому HTTPS-серверу своего IP-адреса.

SSL-сертификат с несколькими именами

Существуют и другие способы, которые позволяют использовать один и тот же IP-адрес сразу для нескольких HTTPS-серверов. Все они, однако, имеют свои недостатки. Одним из таких способов является использование сертификата с несколькими именами в поле SubjectAltName сертификата, например www.example.com и www.example.org. Однако, длина поля SubjectAltName ограничена.

Другим способом является использование wildcard-сертификата, например *.example.org. Такой сертификат защищает все поддомены указанного домена, но только на заданном уровне. Под такой сертификат подходит www.example.org, но не подходят example.org и www.sub.example.org. Два вышеуказанных способа можно комбинировать. Сертификат может одновременно содержать и точное, и wildcard имена в поле SubjectAltName, например example.org и *.example.org.

Лучше поместить сведения о файле сертификата с несколькими именами и файле с его секретным ключом на уровне конфигурации http, чтобы все серверы унаследовали их единственную копию в памяти:

Указание имени сервера

Более общее решение для работы нескольких HTTPS-серверов на одном IP-адресе — расширение Server Name Indication протокола TLS (SNI, RFC 6066), которое позволяет браузеру передать запрашиваемое имя сервера во время SSL handshake, а значит сервер будет знать, какой сертификат ему следует использовать для соединения. Сейчас SNI поддерживается большинством современных браузеров, однако может не использоваться некоторыми старыми или специализированными клиентами.

Примечание

В SNI можно передавать только доменные имена, однако некоторые браузеры могут ошибочно передавать IP-адрес сервера в качестве его имени, если в запросе указан IP-адрес. Полагаться на это не следует.

Чтобы использовать SNI в Angie PRO, соответствующая поддержка должна присутствовать как в библиотеке OpenSSL, использованной при сборке бинарного файла Angie PRO, так и в библиотеке, подгружаемой в момент работы. OpenSSL поддерживает SNI начиная с версии 0.9.8f, если она была собрана с опцией конфигурации *--enable-tlsext*. Начиная с OpenSSL 0.9.8j эта опция включена по умолчанию. Если Angie был собран с поддержкой SNI, то при запуске Angie с ключом “-V” об этом сообщается.

Однако если Angie PRO, собранный с поддержкой SNI, в процессе работы подгружает библиотеку OpenSSL, в которой нет поддержки SNI, Angie выдаёт предупреждение:

```
Angie was built with SNI support, however, now it is linked
dynamically to an OpenSSL library which has no tlsext support,
therefore SNI is not available
```

3 Описание модулей Angie PRO

Модуль API

Модуль API реализует HTTP RESTful интерфейс для получения базовой информации о веб-сервере в формате JSON, а также статистики по клиентским соединениям, зонам разделяемой памяти, DNS-запросам, HTTP-запросам, кэшу HTTP-ответов, сессиям модуля stream и зонам модулей limit_conn http, limit_conn stream, limit_req и http upstream.

Модуль core

Описание модуля и его директив приведено в руководстве по эксплуатации Angie PRO.

Модуль http_access

Модуль позволяет ограничить доступ для определённых адресов клиентов.

Ограничить доступ можно также по паролю или по результату подзапроса. Одновременное ограничение доступа по адресу и паролю управляется директивой satisfy.

Модуль http_addition

Фильтр, добавляющий текст до и после ответа.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_addition_module.

Модуль http_auth_basic

Позволяет ограничить доступ к ресурсам с проверкой имени и пароля пользователя по протоколу “HTTP Basic Authentication”.

Ограничить доступ можно также по адресу или по результату подзапроса. Одновременное ограничение доступа по адресу и паролю управляется директивой satisfy.

Модуль http_auth_request

Предоставляет возможность авторизации клиента, основанной на результате подзапроса. Если подзапрос возвращает код ответа 2xx, доступ разрешается. Если 401 или 403 — доступ запрещается с соответствующим кодом ошибки. Любой другой код ответа, возвращаемый подзапросом, считается ошибкой.

При ошибке 401 клиенту также передаётся заголовок “WWW-Authenticate” из ответа подзапроса.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_auth_request_module.

Модуль может быть скомбинирован с другими модулями доступа, такими как http_access и auth_basic с помощью директивы satisfy.

Модуль http_autoindex

Обслуживает запросы, оканчивающиеся слэшом (‘/’), и выдаёт листинг каталога. Обычно запрос попадает к модулю http_autoindex, когда модуль http_index не нашёл индексный файл.

Модуль http_browser

Создаёт переменные, значения которых зависят от значения поля “User-Agent” в заголовке запроса.

Модуль http_charset

Добавляет указанную кодировку в поле “Content-Type” заголовка ответа. Кроме того, модуль может перекодировать данные из одной кодировки в другую с некоторыми ограничениями:

- перекодирование осуществляется только в одну сторону — от сервера к клиенту,
- перекодироваться могут только однобайтные кодировки
- или однобайтные кодировки в UTF-8 и обратно.

Модуль http_core

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.

Модуль http_dav

Предназначен для автоматизации задач управления файлами на сервере по протоколу WebDAV. Модуль обрабатывает HTTP- и WebDAV-методы PUT, DELETE, MKCOL, COPY и MOVE.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_dav_module*.

Модуль http_empty_gif

Выдаёт однопиксельный прозрачный GIF.

Модуль http_fastcgi

Позволяет передавать запросы FastCGI-серверу.

Модуль http_flv

Обеспечивает серверную поддержку псевдо-стриминга для файлов Flash Video (FLV).

Он специальным образом обрабатывает запросы с аргументом start в строке запроса, посылая в ответ содержимое файла с запрошенного смещения в байтах, добавив перед ним FLV-заголовок.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_flv_module.

Модуль http_geo

Создаёт переменные, значения которых зависят от IP-адреса клиента.

Модуль http_geoip

Создаёт переменные, значения которых зависят от IP-адреса клиента, используя готовые базы данных MaxMind.

При использовании баз данных с поддержкой IPv6 IPv4-адреса ищутся отображёнными на IPv6.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_geoip_module*.

Внимание

Для сборки и работы этого модуля нужна библиотека MaxMind GeoIP.

Модуль http_grpc

Позволяет передавать запросы gRPC-серверу.

Примечание

Для работы этого модуля необходим модуль http_v2.

Модуль http_gunzip

Фильтр, распаковывающий ответы с “Content-Encoding: gzip” для тех клиентов, которые не поддерживают метод сжатия “gzip”. Модуль будет полезен, когда данные желательно хранить сжатыми для экономии места и сокращения затрат на ввод-вывод.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_gunzip_module.

Модуль http_gzip

Фильтр, сжимающий ответ методом gzip, что позволяет уменьшить размер передаваемых данных в 2 и более раз.

Примечание

При использовании протокола SSL/TLS сжатые ответы могут быть подвержены атакам BREACH.

Модуль http_gzip_static

Позволяет отдавать вместо обычного файла предварительно сжатый файл с таким же именем и с расширением “.gz”.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_gzip_static_module*.

Модуль http_headers

Позволяет выдавать поля заголовка “Expires” и “Cache-Control”, а также добавлять произвольные поля в заголовок ответа.

Модуль http_image_filter

Фильтр для преобразования изображений в форматах JPEG, GIF, PNG и WebP.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_image_filter_module.

Примечание

Для сборки и работы этого модуля необходима библиотека libgd. Рекомендуется использовать самую последнюю версию библиотеки.

Для преобразования изображений в формате WebP библиотека libgd должна быть собрана с поддержкой WebP.

Модуль http_random_index

Обслуживает запросы, оканчивающиеся слэшом (‘/’). Такие запросы также могут обслуживаться модулями Модуль http_autoindex и Модуль http_random_index.

Модуль http_js

Позволяет задавать обработчики location и переменных на njs — подмножестве языка JavaScript.

Модуль http_limit_conn

Позволяет ограничить число соединений по заданному ключу, в частности, число соединений с одного IP-адреса.

Учитываются не все соединения, а лишь те, в которых имеются запросы, обрабатываемые сервером, и заголовок запроса уже прочитан.

Модуль http_limit_req

Позволяет ограничить скорость обработки запросов по заданному ключу или, как частный случай, скорость обработки запросов, поступающих с одного IP-адреса. Ограничение обеспечивается с помощью метода “leaky bucket”.

Модуль http_log

Модуль записывает логи запросов в указанном формате.

Логи записываются в контексте location’а, где заканчивается обработка. Это может быть location, отличный от первоначального, если в процессе обработки запроса происходит внутреннее перенаправление.

Модуль http_map

Создаёт переменные, значения которых зависят от значений других переменных.

Модуль http_memcached

Позволяет получать ответ из сервера memcached. Ключ задаётся в переменной $memcached_key. Ответ в memcached должен быть предварительно помещён внешним по отношению к Angie способом.

Модуль http_mirror

Позволяет зеркалировать исходный запрос при помощи создания фоновых зеркалирующих подзапросов. Ответы на зеркалирующие подзапросы игнорируются.

Модуль http_mp4

Обеспечивает серверную поддержку псевдо-стриминга для файлов в формате MP4. Такие файлы обычно имеют расширения .mp4, .m4v и .m4a.

Псевдо-стриминг работает в паре с совместимым медиаплеером. Плеер посылает серверу HTTP-запрос с указанием точки времени старта в аргументе start строки запроса (время задаётся в секундах), а сервер в ответ посылает поток, у которого начальная позиция соответствует запрошенному времени, например:

```
http://example.com/elephants_dream.mp4?start=238.88
```

Это позволяет в любой момент времени выполнить произвольное позиционирование, а также начать воспроизведение с середины временной шкалы.

В форматах, основанных на H.264, метаданные, необходимые для поддержки позиционирования, хранятся в так называемом “moov-атоме”. Это часть файла, которая содержит индексную информацию для всего файла.

До начала воспроизведения плееру необходимо прочитать метаданные. Для этого он отсылает специальный запрос с аргументом *start=0*. Многие кодирующие программы добавляют метаданные в конец файла. Это неоптимально для псевдо-стриминга, поскольку плееру потребуется загрузить файл целиком прежде чем начать воспроизведение. Если метаданные находятся в начале файла, Angie достаточно начать отправлять в ответ содержимое файла. Если же метаданные находятся в конце файла, потребуется прочитать весь файл и подготовить новый поток, в котором метаданные предшествуют медийным данным. Это требует дополнительного процессорного времени, памяти и дискового ввода/вывода, поэтому лучше заранее подготовить исходный файл для псевдо-стриминга, нежели делать это для каждого запроса.

Модуль также поддерживает аргумент *end* HTTP-запроса, задающий время окончания воспроизведения потока. Аргумент *end* задаётся совместно с аргументом *start* или самостоятельно:

```
http://example.com/elephants_dream.mp4?start=238.88&end=555.55
```

Для запроса с ненулевыми аргументами *start* или *end* Angie считывает из файла метаданные, готовит поток с запрошенным диапазоном и отправляет его клиенту. Это тоже требует дополнительных ресурсов, как указано выше.

Если аргумент *start* указывает на видеокадр, не являющийся ключевым, то начало такого видео может воспроизводиться с ошибками. В этом случае к запрашиваемому видео могут быть добавлены ближайший к точке *start* ключевой кадр и все промежуточные кадры между ними. При воспроизведении эти кадры будут скрыты при помощи edit-листа.

Если запрос, обрабатываемый этим модулем, не содержит аргументов *start* и *end*, дополнительные ресурсы не тратятся, а файл отсылается непосредственно как статический ресурс. Некоторые плееры также поддерживают запросы с указанием диапазона запрашиваемых байт (byte-range requests), для них этот модуль не требуется.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_mp4_module*.

Примечание

Если ранее использовался сторонний модуль mp4, следует его отключить.

Схожая поддержка псевдо-стриминга для FLV-файлов обеспечивается модулем http_flv.

Модуль http_perl

Модуль позволяет писать обработчики location и переменных на Perl, а также вставлять вызовы Perl в SSI.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_perl_module*.

Внимание

Для сборки этого модуля необходим Perl версии 5.6.1 и выше. Компилятор C должен быть совместим с тем, которым был собран Perl.

Модуль http_proxy

Позволяет передавать запросы другому (проксируемому) серверу.

Модуль http_random_index

Обслуживает запросы, оканчивающиеся слэшом (‘/’), и выдаёт случайный файл в качестве индексного файла каталога. Модуль выполняется до модуля http_index.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_random_index_module.

Модуль http_realip

Позволяет менять адрес и необязательный порт клиента на переданные в указанном поле заголовка.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра --with-http_realip_module.

Модуль http_referer

Позволяет блокировать доступ к сайту для запросов с неверными значениями поля “Referer” в заголовке. Следует иметь в виду, что подделать запрос с нужным значением поля “Referer” не составляет большого труда, поэтому цель использования данного модуля заключается не в стопроцентном блокировании подобных запросов, а в блокировании массового потока запросов, сделанных обычными браузерами. Нужно также учитывать, что обычные браузеры могут не передавать поле “Referer” даже для верных запросов.

Модуль http_rewrite

Позволяет изменять URI запроса с помощью регулярных выражений PCRE, делать перенаправления и выбирать конфигурацию по условию.

Директивы break, if, return, rewrite и set обрабатываются в следующем порядке:

- последовательно выполняются директивы этого модуля, описанные на уровне *server*;
- в цикле:
  - ищется *location* по URI запроса;
  - последовательно выполняются директивы этого модуля, описанные в найденном *location*;

- цикл повторяется, если URI запроса изменялся, но не более 10 раз.

Модуль http_scgi

Позволяет передавать запросы SCGI-серверу.

Модуль http_secure_link

Позволяет проверять аутентичность запрашиваемых ссылок, защищать ресурсы от несанкционированного доступа, а также ограничивать срок действия ссылок.

Правильность запрашиваемой ссылки проверяется сравнением переданного в запросе значения контрольной суммы со значением, вычисляемым для запроса. Если ссылка имеет ограниченный срок действия и он истёк, ссылка считается устаревшей. Результат этих проверок делается доступным в переменной $secure_link.

Модуль реализует два альтернативных режима работы. В первом режиме, который включается директивой secure_link_secret, можно проверить аутентичность запрашиваемых ссылок и защитить их от несанкционированного доступа. Второй режим включается директивами secure_link и secure_link_md5, и позволяет также ограничить срок действия ссылок.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_secure_link_module*.

Модуль http_slice

Фильтр, который разбивает запрос на подзапросы, каждый из которых возвращает определённый диапазон ответа. Фильтр обеспечивает более эффективное кэширование больших ответов.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_slice_module*.

Модуль http_split_clients

Создаёт переменные для A/B тестирования (также известного как “split-тестирование”).

Модуль http_ssi

Фильтр, обрабатывающий команды SSI (Server Side Includes) в проходящих через него ответах.

Модуль http_ssl

Обеспечивает работу по протоколу HTTPS.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_ssl_module*.

Примечание

Для сборки и работы этого модуля нужна библиотека OpenSSL.

Модуль http_stub_status

Предоставляет доступ к базовой информации о состоянии сервера.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_stub_status_module*.

Модуль http_sub

Фильтр, изменяющий в ответе одну заданную строку на другую.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_sub_module*.

Модуль http_upstream

Предоставляет контекст для описания группы серверов, которые могут использоваться в директивах proxy_pass, fastcgi_pass, uwsgi_pass, scgi_pass, memcached_pass и grpc_pass.

Модуль http_userid

Выдаёт куки для идентификации клиентов. Для записи в лог полученных и выданных кук можно использовать встроенные переменные $uid_got и $uid_set. Модуль совместим с модулем mod_uid для Apache.

Модуль http_uwsgi

Позволяет передавать запросы uwsgi-серверу.

Модуль http_v2

Позволяет обеспечивает поддержку HTTP/2.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_v2_module*.

Модуль http_xslt

Фильтр, преобразующий XML-ответ с помощью одного или нескольких XSLT-шаблонов.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-http_xslt_module*.

Примечание

Для сборки и работы этого модуля нужны библиотеки libxml2 и libxslt.

Модуль stream_access

Модуль позволяет ограничить доступ для определённых адресов клиентов.

Модуль stream_core

Модуль по умолчанию не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *–with-stream*.

Модуль stream_geo

Создаёт переменные, значения которых зависят от IP-адреса клиента.

Модуль stream_geoip

Создаёт переменные, значения которых зависят от IP-адреса клиента, используя готовые базы данных MaxMind.

При использовании баз данных с поддержкой IPv6 IPv4-адреса ищутся отображёнными на IPv6.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-stream_geoip_module*.

Внимание

Для сборки и работы этого модуля нужна библиотека MaxMind GeoIP.

Модуль stream_js

Позволяет задавать обработчики на njs — подмножестве языка JavaScript.

Модуль stream_limit_conn

Позволяет ограничить число соединений по заданному ключу, в частности, число соединений с одного IP-адреса.

Модуль stream_log

Модуль записывает логи запросов в указанном формате.

Модуль stream_map

Создаёт переменные, значения которых зависят от значений других переменных.

Модуль stream_proxy

Позволяет проксировать потоки данных по TCP, UDP и UNIX-сокетам.

Модуль stream_realip

Позволяет менять адрес и порт клиента на переданные в заголовке протокола PROXY. Протокол PROXY должен быть предварительно включён при помощи установки параметра *proxy_protocol* в директиве listen.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-stream_realip_module*.

Модуль stream_return

Позволяет отправить заданное значение клиенту и после этого закрыть соединение.

Модуль stream_set

Позволяет устанавливать значение переменной.

Модуль stream_split_clients

Создаёт переменные для A/B тестирования (также известного как “split-тестирование”).

Модуль stream_ssl

Обеспечивает необходимую поддержку для работы прокси-сервера по протоколу SSL/TLS.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-stream_ssl_module*.

Примечание

Для сборки и работы этого модуля нужна библиотека OpenSSL.

Модуль stream_ssl_preread

Позволяет извлекать информацию из сообщения ClientHello без терминирования SSL/TLS, например имя сервера, запрошенное через SNI или протоколы, указанные в ALPN.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-stream_ssl_preread_module*.

Модуль stream_upstream

Предоставляет контекст для описания группы серверов, которые могут использоваться в директиве proxy_pass.

Модуль mail_core

Модуль по умолчанию не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *–with-mail*.

Модуль mail_auth_http

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.

Модуль mail_proxy

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.

Модуль mail_realip

Позволяет менять адрес и необязательный порт клиента на переданные в указанном поле заголовка адрес и порт клиента на переданные в заголовке протокола PROXY. Протокол

PROXY должен быть предварительно включён при помощи установки параметра *proxy_protocol* в директиве listen.

Модуль mail_ssl

Обеспечивает работу почтового прокси-сервера по протоколу SSL/TLS.

По умолчанию этот модуль не собирается, его сборку необходимо разрешить с помощью конфигурационного параметра *--with-mail_ssl_module*.

Примечание

Для сборки и работы этого модуля нужна библиотека OpenSSL.

Модуль mail_imap

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.

Модуль mail_pop3

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.

Модуль mail_smtp

Описание модуля и его директив приведено в руководстве по эксплуатации Angie Pro.