Nginx HTTP Server Learning Notes

Wednesday, April 23, 2014 14:23

1. **Installing Nginx**
    a. Compile Module Options
        i. Modules enabled by default

Modules enabled by default	Description
--without-http_charset_module	Disables the Charset module for re-encoding web pages.
--without-http_gzip_module	Disables the Gzip compression module.
--without-http_ssi_module	Disables the Server Side Include module.
--without-http_userid_module	Disables the User ID module providing user identification via cookies.
--without-http_access_module	Disables the Access module allowing access configuration for IP address ranges.
--without-http_auth_basic_module	Disables the Basic Authentication module.
--without-http_autoindex_module	Disables the Automatic Index module.
--without-http_geo_module	Disables the Geo module allowing you to define variables depending on IP address ranges.
--without-http_map_module	Disables the Map module that allows you to declare map blocks.
--without-http_referer_module	Disables the Referer control module.
--without-http_rewrite_module	Disables the Rewrite module.
--without-http_proxy_module	Disables the Proxy module for transferring requests to other servers.
--without-http_fastcgi_module	Disables the FastCGI module for interacting with a FastCGI process.
--without-http_memcached_module	Disables the Memcached module for interacting with the *memcache daemon*.
--without-http_limit_zone_module	Disables the Limit Zone module for restricting resource usage according to defined zones.
--without-http_limit_req_module	Disables the Limit Requests module allowing you to limit the amount of requests per user.
--without-http_empty_gif_module	Disables the Empty Gif module for serving a blank GIF image from memory.
--without-http_browser_module	Disables the Browser module for interpreting the User Agent string.
--without-http_upstream_ip_hash_module	Disables the Upstream module for configuring load-balanced architectures.

        ii. Modules disabled by default

Modules DISabled by default	Description
--with-http_ssl_module	Enables the SSL module for serving pages using HTTPS.
--with-http_realip_module	Enables the Real IP module for reading the real IP address from the request header data.
--with-http_addition_module	Enables the Addition module which lets you append or prepend data to the response body.
--with-http_xslt_module	Enables the XSLT module for applying XSL transformations to XML documents. Note: You will need to install the libxml2 and libxslt libraries on your system if you wish to compile these modules.
--with-http_image_filter_module	Enables the Image Filter module that lets you apply modification to images. Note: You will need to install the libgd library on your system if you wish to compile this module.
--with-http_geoip_module	Enables the GeoIP module for achieving geographic localization using MaxMind's GeoIP binary database. Note: You will need to install the libgeoip library on your system if you wish to compile this module.
--with-http_sub_module	Enables the Substitution module for replacing text in web pages.
--with-http_dav_module	Enables the WebDAV module (Distributed Authoring and Versioning via Web).
--with-http_flv_module	Enables the FLV module for special handling of .flv (flash video) files.
--with-http_gzip_static_module	Enables the Gzip Static module for sending pre-compressed files.
--with-http_random_index_module	Enables the Random Index module for picking a random file as the directory index.
--with-http_secure_link_module	Enables the Secure Link module to check the presence of a keyword in the URL.
--with-http_stub_status_module	Enables the Stub Status module, which generates a server statistics and information page.
--with-google_perftools_module	Enables the Google Performance Tools module

        iii. Miscellaneous Options

Mail server proxy options	
--with-mail	Enables mail server proxy module. Supports POP3, IMAP4, SMTP. It is disabled by default.
--with-mail_ssl_module	Enables SSL support for the mail server proxy. It is disabled by default.
--without-mail_pop3_module	Disables the POP3 module for the mail server proxy. It is enabled by default when the mail server proxy module is enabled.
--without-mail_imap_module	Disables the IMAP4 module for the mail server proxy. It is enabled by default when the mail server proxy module is enabled.
--without-mail_smtp_module	Disables the SMTP module for the mail server proxy. It is enabled by default when the mail server proxy module is enabled.

Event management:	Allows you to select the event notification system for the Nginx sequencer. For advanced users only.
--with-rtsig_module	Enables the rtsig module to use rtsig as event notification mechanism.
--with-select_module	Enables the select module to use select as event notification mechanism. By default, this module is enabled unless a better method is found on the system—kqueue, epoll, rtsig, or poll.
--without-select_module	Disables the select module.
--with-poll_module	Enables the poll module to use poll as event notification mechanism. By default, this module is enabled if available, unless a better method is found on the system—kqueue, epoll, or rtsig.
--without-poll_module	Disables the poll module.

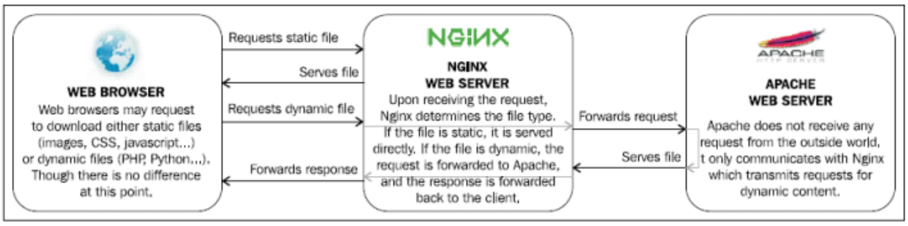
User and group options	
--user=⋯	Default user account for starting the Nginx worker processes. This setting is used only if you omit to specify the user direc tive in the configuration file.
--group=⋯	Default user group for starting the Nginx worker processes. This setting is used only if you omit to specify the group direct ive in the configuration file.

Other options	

--with-ipv6	Enables IPv6 support.
--without-http	Disables the HTTP server.
--without-http-cache	Disables HTTP caching features.
--add-module=PATH	Adds a third-party module to the compile process by specifying its path. This switch can be repeated indefinitely if you wish to compile multiple modules.
--with-debug	Enables additional debugging information to be logged.

2. Apache and Nginx Together

a. The reverse proxy mechanism


WEB BROWSER
Web browsers may request to download either static files (images, CSS, javascript...) or dynamic files (PHP, Python...). Though there is no difference at this point.
Requests static file
Serves file
Requests dynamic file
Forwards response
NGINX
NGINX WEB SERVER
Upon receiving the request, Nginx determines the file type. If the file is static, it is served directly. If the file is dynamic, the request is forwarded to Apache, and the response is forwarded back to the client.
Forwards request
Serves file
APACHE
APACHE WEB SERVER
Apache does not receive any request from the outside world, t only communicates with Nginx which transmits requests for dynamic content.

There are now two web servers running and processing requests:

- Nginx positioned as a frontend server (in other words, as reverse proxy) receives all the requests coming from the outside world. It filters them, either serving static files directly to the client or forwarding dynamic content requests to Apache.
- Apache runs as a backend server; it only communicates with Nginx. It may be hosted on the same computer as the frontend, in which case, the listening port must be edited to leave port 80 available to Nginx. Alternatively, you can employ multiple backend servers on different machines and share the load.

b. Advantages and Disadvantages

i. The main purpose of setting up Nginx as frontend and giving Apache a simple backend role is to improve the serving speed. As we established, a great amount of requests coming from clients are for static files, and static files are served much faster by Nginx.
ii. On the other hand, you are still deporting requests for dynamic content to Apache
iii. Besides, since Nginx is installed as the frontend, it implies that it receives raw requests from users. This implies that the URI comes in its original form, which can lead to confusion for Nginx; it will not be able to make the difference between static and dynamic content. **You have two choices to solve this issue—either port your rewrite rules to Nginx or redirect any request that results in a 404 error to the Apache backend.**

c. Reconfiguring overview

At this point, you probably have the following architecture set up on your server:

- A web server application running on port 80, such as Apache
- A dynamic server-side script processing application such as PHP, communicating with your web server via CGI, FastCGI, or as a server module

The new configuration that we are going towards will resemble the following:

- Nginx running on port 80
- Apache or another web server running on a different port, accepting requests coming from local sockets only
- The script processing application configuration will remain unchanged

3. From Apache to Nginx

a. Feature comparison between Apache and Nginx

Features	Nginx	Apache
Request management How does the web server process requests?	**Event-driven architecture** Requests are accepted using asynchronous sockets and aren't processed in separate threads, in order to reduce memory and CPU overhead.	**Synchronous sockets, threads, and processes** Each request is in a separate thread or process and uses synchronous sockets.
Programming language Which language was the web server written in?	**C** The C language is notably low-level and offers more accurate memory management.	**C and C++** Although Apache was written in C, many modules were designed with C++.
Portability Which operating systems are supported?	**Multiplatform** Nginx runs under Windows, GNU/Linux, Unix, BSD, Mac OS X, and Solaris.	**Multiplatform** Apache runs under Windows, GNU/Linux, Unix, BSD, Mac OS X, Solaris, Novell NetWare, OS/2, TPF, OpenVMS, eCS, AIX, z/OS, HP-UX, and so on.
Year of birth How long ago did the development start?	**2002** While Nginx is younger than Apache, it was intended for a more modern era.	**1994** Apache is one of the numerous open source projects initiated in the 90s that contributed to making the World Wide Web what it is today.
HTTPS support Can the web server deliver secure web pages?	**Supported as module** If you want HTTPS support, you need to make sure to compile Nginx with the proper module.	**Supported as module** Apache comes with HTTPS support via a module included by default.
Virtual Hosting Can the web server host multiple websites on the same computer?	**Supported natively** Nginx natively supports virtual hosting, but is not configured by default to accept per-virtual-host configuration files (more details further in this chapter).	**Supported natively** Apache natively supports virtual hosting and offers the possibility to include one configuration file per folder (.htaccess).
CGI Support Does the web browser support CGI and FastCGI?	**FastCGI only** Nginx supports FastCGI via a module that is included by default at compile time.	**CGI and FastCGI** Both protocols are exploitable via modules that can be loaded into Apache.
Module system How does the web server handle modules?	**Static module system** Modules must be included at compile time.	**Dynamic module system** Modules can be loaded and unloaded dynamically from configuration files.

b. Performance

The same tests can be applied to Apache in order to establish direct performance comparisons. In fact, many admin bloggers an d technicians have already done so, and the general trend is unquestionably in favor of Nginx on all aspects:

- The RPS rate is generally much higher with Nginx, sometimes twice higher than Apache's. In other words, Nginx is able to serve twice as many pages as Apache in the same lapse of time.
- Response times are lower on Nginx—as the request count grows, Apache becomes slower and slower to serve pages.
- Apache tends to use slightly more bandwidth than Nginx for serving the same requests. This can be interpreted in two ways—either Apache generates more traffic overhead, or it is able to transfer data at a faster rate by better occupying the available bandwidth (it's still unsure as to which of these assumptions is the most valid).

In conclusion on the field of performance, Nginx wins hands down. It's clearly the main reason why so many have switched to t he lightweight Russian web server.

c. Conclusion
If you are in the market for high scale projects with limited resources at your disposal, Nginx comes in as a great solution. Apache is a good option to get your projects started when your knowledge of web servers and hosting is limited, but as soon as you meet success, you, your server, and you r visitors may eventually find it inconsistent.