

CIA Lab 6 - Lighttpd (Web)

Lab Author: *Ahmed SalahEldin Elkashef*

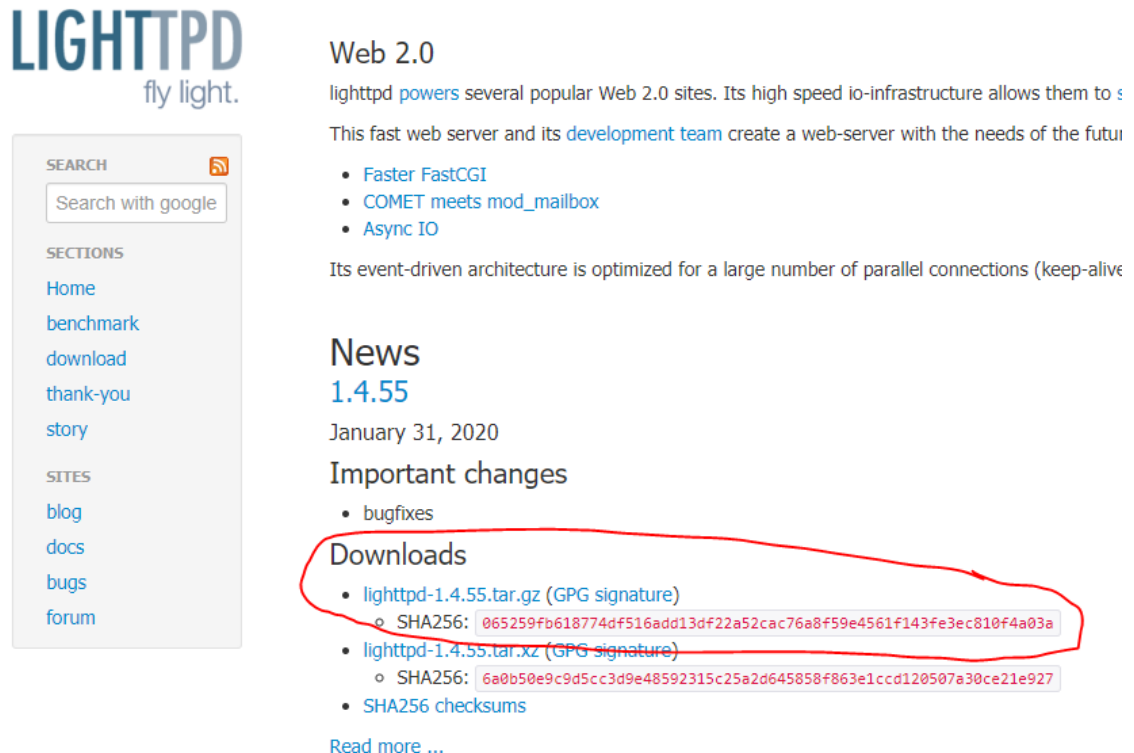
- **Task 1 - Install & Configure Virtual Hosts**

Fetch, verify, build and install the daemon.

Fetching & Verifying Source Code:

My Web server of choice is lighttpd, lighttpd or as pronounced “lighty” is a web server that is optimized for speed-critical environments, hence the name. Also it has two components in it’s name, “light” and “httpd” -> lighttpd

In order to fetch the web server, we go directly to the [official website](#) and we are presented with a section for “downloads” that contain two compressions of the web server code tar file. We download any of them and also the GPG signature file, we can also find the public key that we will need to import in order to start the verification process:



The screenshot shows the official Lighttpd website. On the left is a sidebar with a search bar and navigation links under 'SECTIONS' (Home, benchmark, download, thank-you, story) and 'SITES' (blog, docs, bugs, forum). The main content area has a header 'LIGHTTPD fly light.' followed by a 'Web 2.0' section describing its speed and event-driven architecture. Below that is a 'News' section dated January 31, 2020, titled '1.4.55' with 'Important changes' listed. The 'Downloads' section is circled in red and lists two tarballs: 'lighttpd-1.4.55.tar.gz (GPG signature)' and 'lighttpd-1.4.55.tar.xz (GPG signature)', each with its SHA256 checksum. A 'Read more ...' link is at the bottom.

LIGHTTPD
fly light.

SEARCH 
Search with google

SECTIONS
[Home](#)
[benchmark](#)
[download](#)
[thank-you](#)
[story](#)

SITES
[blog](#)
[docs](#)
[bugs](#)
[forum](#)

Web 2.0
lighttpd [powers](#) several popular Web 2.0 sites. Its high speed io-infrastructure allows them to s
This fast web server and its [development team](#) create a web-server with the needs of the futu

- [Faster FastCGI](#)
- [COMET meets mod_mailbox](#)
- [Async IO](#)

Its event-driven architecture is optimized for a large number of parallel connections (keep-alive

News
1.4.55
January 31, 2020
Important changes

- [bugfixes](#)

Downloads

- [lighttpd-1.4.55.tar.gz \(GPG signature\)](#)
 - SHA256: [065259fb618774df516add13df22a52cac76a8f59e4561f143fe3ec810f4a03a](#)
- [lighttpd-1.4.55.tar.xz \(GPG signature\)](#)
 - SHA256: [6a0b50e9c9d5cc3d9e48592315c25a2d645858f863e1ccd120507a30ce21e927](#)
- [SHA256 checksums](#)

[Read more ...](#)

```
ahmed@mail: ~/Downloads
ahmed@mail:~/Downloads$ sudo wget https://download.lighttpd.net/lighttpd/releases-1.4.x/lighttpd-1.4.55.tar.gz
[sudo] password for ahmed:
--2020-10-10 18:39:32-- https://download.lighttpd.net/lighttpd/releases-1.4.x/lighttpd-1.4.55.tar.gz
Resolving download.lighttpd.net (download.lighttpd.net)... 5.9.70.195, 2a01:4f8:161:80c9::2
Connecting to download.lighttpd.net (download.lighttpd.net)|5.9.70.195|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1122246 (1.1M) [application/x-gtar-compressed]
Saving to: 'lighttpd-1.4.55.tar.gz'

lighttpd-1.4.55.tar.gz      100%[=====] 1,07M  2,98MB/s  in 0,4s

2020-10-10 18:39:33 (2,98 MB/s) - 'lighttpd-1.4.55.tar.gz' saved [1122246/1122246]

ahmed@mail:~/Downloads$ sudo wget https://download.lighttpd.net/lighttpd/releases-1.4.x/lighttpd-1.4.55.tar.gz.asc
--2020-10-10 18:40:18-- https://download.lighttpd.net/lighttpd/releases-1.4.x/lighttpd-1.4.55.tar.gz.asc
Resolving download.lighttpd.net (download.lighttpd.net)... 5.9.70.195, 2a01:4f8:161:80c9::2
Connecting to download.lighttpd.net (download.lighttpd.net)|5.9.70.195|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 833 [text/plain]
Saving to: 'lighttpd-1.4.55.tar.gz.asc'

lighttpd-1.4.55.tar.gz.asc  100%[=====] 833  --.-KB/s  in 0s

2020-10-10 18:40:18 (171 MB/s) - 'lighttpd-1.4.55.tar.gz.asc' saved [833/833]

ahmed@mail:~/Downloads$ ls
lighttpd-1.4.55.tar.gz  lighttpd-1.4.55.tar.gz.asc  postfix-3.5.7
ahmed@mail:~/Downloads$
```

Still, we lack the public key for the verification of lighttpd, for that, we go to any trusted keyserver for GPG public keys, such as <http://keys.gnupg.net/> and we search for **lighttpd**.

Search results for 'lighttpd'

Type	bits/keyID	Date	User ID
pub	4096R/ <u>F60A43D4</u>	2018-03-28	Matthias Kuehn <matthias@nerdbase.de> Matthias Kuehn (nitrox) <nitrox@osnanet.de> Matthias Kuehn (nitrox) <nitrox@lighttpd.net> Matthias Kuehn (nitrox) <mail@matthias-kuehn.de> Fingerprint=B67D 328E E653 7B2A 41A9 58F9 46F7 9307 F60A 43D4
pub	4096R/ <u>6FE198C8</u>	2016-07-16	Glenn Strauss (lighttpd) <gstrauss@gluelogic.com> Fingerprint=649D 0DD7 67FF 2062 02A7 6C51 58F1 4A78 6FE1 98C8
pub	2048R/ <u>C74635EB</u>	2014-08-25	Jenkins (autobuilder) <jenkins@lighttpd.net> Fingerprint=1EF7 AB71 F7C0 7CB1 8C85 07D0 4DCC DA8A C746 35EB
pub	1024D/ <u>AFFAC34C</u>	2012-04-04	Alexandre Z. Bacich (servidor lighttpd) <azbacich@gmail.com> Fingerprint=E131 3691 55FE 4374 5FDA 3BBF E8EF F3F5 AFFA C34C
pub	1024R/ <u>93AA5D2C</u>	2010-02-28	Launchpad Lighttpd Fingerprint=59AA 33B7 6324 3343 FE78 FA57 C30D B007 93AA 5D2C
pub	4096R/ <u>1E95BAD7</u>	2009-05-06	Stefan Bühler <stefan@stbuehler.de> Stefan Bühler <stbuehler@web.de> Stefan Bühler <stbuehler@freenet.de> Stefan Bühler <lighttpd@stbuehler.de> Stefan Bühler <stbuehler@lighttpd.net> Fingerprint=C7CA 1E9E 29DC 77F5 4808 94B2 E0E7 D017 1E95 BAD7
pub	1024D/ <u>9CCEB24D</u>	2005-01-23	*** KEY REVOKED *** [not verified] Matthias Kuehn (nitrox) <matthias@nerdbase.de> Matthias Kuehn (nitrox) <mail@homer.osnanet.de> Matthias Kuehn (nitrox) <nitrox@osnanet.de> Matthias Kuehn (nitrox) <mail@matthias-kuehn.de> Matthias Kuehn (nitrox) <nitrox@homer.nerdbase.de> Matthias Kuehn (nitrox) <nitrox@nitrox.osnanet.de> Matthias Kuehn (nitrox) <nitrox@nitrox.nerdbase.de> Matthias Kuehn (nitrox) (Lighty) <nitrox@lighttpd.net> Fingerprint=99DC 67A5 AD39 D1C1 EF94 1AE6 712E EF8A 9CCE B24D

As shown in the screenshot, there are a number of public keys that are available, through many years, the oldest of them is a revoked key from 2005. We go for the newest key and we get the key ID from it, which will be **F60A43D4**.

Now we retrieve the public key using the key ID and attempt to verify the file:

```
ahmed@mail: ~/Downloads
ahmed@mail:~/Downloads$ gpg --keyserver hkp://keyserver.ubuntu.com --recv-keys F60A43D4
gpg: key 46F79307F60A43D4: "Matthias Kuehn <mail@nerdbase.de>" not changed
gpg: Total number processed: 1
gpg: unchanged: 1
ahmed@mail:~/Downloads$ gpg --verify lighttpd-1.4.55.tar.gz lighttpd-1.4.55.tar.gz.asc
gpg: no valid OpenPGP data found.
gpg: the signature could not be verified.
Please remember that the signature file (.sig or .asc)
should be the first file given on the command line.
ahmed@mail:~/Downloads$ gpg --verify lighttpd-1.4.55.tar.gz.asc lighttpd-1.4.55.tar.gz
gpg: Signature made C6 01 feb 2020 05:50:33 MSK
gpg: using RSA key ACAA8EC8C9703EAE86EFB5FBFAF16D0F0
gpg: Good signature from "Glenn Strauss (lighttpd) <gstrauss@gluelogic.com>" [expired]
gpg: Note: This key has expired!
Primary key fingerprint: 649D 0DD7 67FF 2062 02A7 6C51 58F1 4A78 6FE1 98C8
Subkey fingerprint: ACAA 8EC8 C970 3EAE B349 258E 86EF B5FB AF16 D0F0
ahmed@mail:~/Downloads$
```

It looks like the key has expired. However, after checking the internet, it seems that this is the latest published public key for lighttpd, so we attempt a different form of verification using the SHA256 checksum published on the website, and we compute another one for our local file and then check them, if they are the same then we verify that our file is original as the one on the website:

```
ahmed@mail: ~/Downloads
ahmed@mail:~/Downloads$ ls
lighttpd-1.4.55 lighttpd-1.4.55.tar.gz lighttpd-1.4.55.tar.gz.asc postfix-3.5.7
ahmed@mail:~/Downloads$ sha256sum lighttpd-1.4.55.tar.gz
065259fb618774df516add13df22a52cac76a8f59e4561f143fe3ec810f4a03a lighttpd-1.4.55.tar.gz
ahmed@mail:~/Downloads$
```

Downloads

- [lighttpd-1.4.55.tar.gz](#) (GPG signature)
 - SHA256: `065259fb618774df516add13df22a52cac76a8f59e4561f143fe3ec810f4a03a`

Both of them seem to be matching:

065259fb618774df516add13df22a52cac76a8f59e4561f143fe3ec810f4a03a

Now we start the installation process based on the official documentation [here](#).

Building and Compiling:

As instructed, we decompress the zipped file and then cd into the resulting directory

```
ahmed@mail: ~/Downloads/lighttpd-1.4.55
ahmed@mail:~/Downloads/lighttpd-1.4.55$ ls
aclocal.m4      config.guess    depcomp         m4              NEWS            tests
ar-lib          config.h.in     distribute.sh.in Makefile.am     README          README.FreeBSD
AUTHORS         config.sub      doc             Makefile.in     README.FreeBSD  SConstruct
autogen.sh      configure       INSTALL         meson.build     SConstruct      scripts
CMakeLists.txt  configure.ac    install-sh      meson_options.txt
compile         COPYING        ltmain.sh       missing         src
ahmed@mail:~/Downloads/lighttpd-1.4.55$
```

Before we proceed, we read in the documentation that we will need the zlib and pcre libraries, we install and locate them, they are the **libpcre3-dev** (perl-compatible regular expressions) and **zlib1g-dev** (for data compression).

```
sudo apt install libpcre3-dev
sudo apt install zlib1g-dev
```

In addition to other libraries that were found missing during the installation:

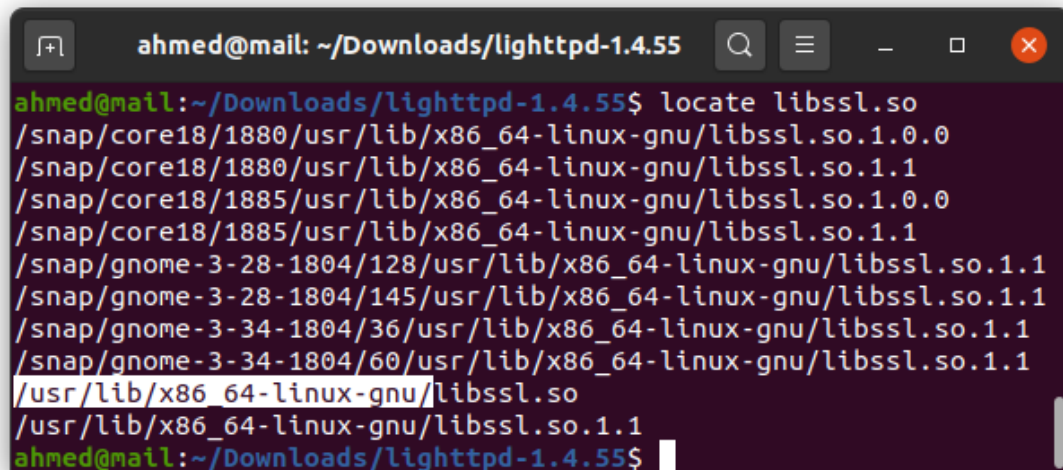
```
sudo apt install libbz2-dev
```

Libbz2 is a data compressor.

To enable the support for SSL/TLS before compilation, we use the `./configure` with a lot of options that can appear with `./configure --help`.

From the `--help`, we know that we need to use the `--with-openssl=<directory to SSL libraries>` flag.

We therefore have to locate the SSL libraries at first:



```
ahmed@mail: ~/Downloads/lighttpd-1.4.55
ahmed@mail:~/Downloads/lighttpd-1.4.55$ locate libssl.so
/snap/core18/1880/usr/lib/x86_64-linux-gnu/libssl.so.1.0.0
/snap/core18/1880/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/snap/core18/1885/usr/lib/x86_64-linux-gnu/libssl.so.1.0.0
/snap/core18/1885/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/snap/gnome-3-28-1804/128/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/snap/gnome-3-28-1804/145/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/snap/gnome-3-34-1804/36/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/snap/gnome-3-34-1804/60/usr/lib/x86_64-linux-gnu/libssl.so.1.1
/usr/lib/x86_64-linux-gnu/libssl.so
/usr/lib/x86_64-linux-gnu/libssl.so.1.1
ahmed@mail:~/Downloads/lighttpd-1.4.55$
```

The directory is `/usr/lib/x86_64-linux-gnu/`

We run `./configure --with-openssl=/usr/lib/x86_64-linux-gnu/`

And it appears that the OpenSSL libs successfully were discovered and lighttpd was configured with the SSL features:

```
ahmed@m...  
Plugins:  
enabled:  
  mod_access  
  mod_accesslog  
  mod_alias  
  mod_auth  
  mod_authn_file  
  mod_cgi  
  mod_compress  
  mod_deflate  
  mod_dirlisting  
  mod_evhost  
  mod_expire  
  mod_extforward  
  mod_fastcgi  
  mod_flv_streaming  
  mod_indexfile  
  mod_openssl  
  mod_proxy
```

```
ahmed@m...  
Features:  
enabled:  
  auth-crypt  
  compress-bzip2  
  compress-deflate  
  compress-gzip  
  large-files  
  network-ipv6  
  network-openssl  
  regex-conditionals
```

Then make:

```
ahmed@mail: ~/Downloads/lighttpd-1.4.55  
make[5]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot/www'  
make[4]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot/www'  
make[4]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot'  
make[4]: Nothing to be done for 'all-am'.  
make[4]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot'  
make[3]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot'  
make[3]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[3]: Nothing to be done for 'all-am'.  
make[3]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[2]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
make[2]: Nothing to be done for 'all-am'.  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
ahmed@mail:~/Downloads/lighttpd-1.4.55$
```

```
ahmed@mail: ~/Downloads/lighttpd-1.4.55  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests/docroot'  
make[2]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[3]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[3]: Nothing to be done for 'install-exec-am'.  
make[3]: Nothing to be done for 'install-data-am'.  
make[3]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[1]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55/tests'  
make[1]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
make[2]: Entering directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
make[2]: Nothing to be done for 'install-exec-am'.  
make[2]: Nothing to be done for 'install-data-am'.  
make[2]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
make[1]: Leaving directory '/home/ahmed/Downloads/lighttpd-1.4.55'  
ahmed@mail:~/Downloads/lighttpd-1.4.55$
```

Now, we redo the steps above to add additional modules that will be later used by this lab, we have a look at the possible options and we find the following:

Plugins:

enabled:

Mod_access
Mod_accesslog
Mod_alias
Mod_auth
Mod_authn_file
Mod_cgi
Mod_compress
Mod_deflate
Mod_dirlisting
Mod_evhost
Mod_expire
Mod_extforward
Mod_fastcgi
Mod_flv_streaming
mod_indexfile
mod_openssl
mod_proxy
mod_redirect
mod_rewrite
mod_rrdtool
mod_scgi
mod_secdownload
mod_setenv
mod_simple_vhost
mod_sockproxy
mod_ssi
mod_staticfile
mod_status
mod_userdir
mod_usertrack
mod_vhostdb
mod_webdav
mod_wstunnel

disabled:

mod_authn_gssapi
mod_authn_ldap
mod_authn_mysql
mod_authn_pam
mod_cml
mod_geoip
mod_magnet
mod_maxminddb
mod_mysql_vhost
mod_trigger_b4_dl
mod_vhostdb_dbi
mod_vhostdb_ldap
mod_vhostdb_mysql
mod_vhostdb_pgsq

Features:

enabled:

Auth-crypt
Compress-bzip2
Compress-deflate
Compress-gzip
Large-files
Network-ipv6
Network-openssl
Regex-conditionals

disabled:

dbi
geoip
kerberos
ldap
lua
maxminddb
mysql
pam
postgresql
stat-cache-fam
storage-gdbm
storage-memcached
webdav-locks
webdav-properties

Using the list above, we understand what possible features we can enable to achieve the other tasks in the lab:

For a wholistic approach, we will install with the support with all the features we might need, and configure them whenever needed:

```
./configure --with-geoip --with-lua --with-maxminddb --with-memcached  
--with-fam --with-openssl=/usr/lib/x86_64-linux-gnu/
```

We install the following packages:

```
sudo apt install libgamin-dev  
sudo apt install libgeoip-dev  
sudo apt install libmaxminddb-dev  
sudo apt install libmemcached-dev
```

For Log Analysis, we use a third-party Log analyzer ([AwStats Software](#))

After configuration step is complete, we shall have the following features:

Features:

enabled:

Auth-crypt
Compress-bzip2
Compress-deflate
Compress-gzip
Geoip
Large-files
Maxminddb
Network-ipv6
Network-openssl
Regex-conditionals
stat-cache-fam
storage-memcached

disabled:

dbi
kerberos
ldap
lua
mysql
pam
postgresql
storage-gdbm
webdav-locks
webdav-properties

Now we `make` and `sudo make install` and it completes successfully.

After installation is complete, we do the following steps:

- 1) We move the `/doc/config` files to the `/etc/lighttpd` directory so that we have a sample configuration file `lighttpd.conf` that we can configure easier.
- 2) We create the structure of the directories inside `/var/log` and the `/srv/www` according to the configuration file

After the config files are copied to the `/etc/lighttpd/` directory, we have the following files:

```
ahmed@mail: /etc/lighttpd
ahmed@mail:/etc/lighttpd$ ls
conf.d  lighttpd.conf  lighttpd.conf.complete  Makefile  Makefile.am  Makefile.in  modules.conf  vhosts.d
ahmed@mail:/etc/lighttpd$ ls conf.d/
access_log.conf  compress.conf  expire.conf  Makefile  mod.template  scgi.conf  status.conf
auth.conf        debug.conf    fastcgi.conf  Makefile.am  mysql_vhost.conf  secdownload.conf  trigger_b4_dl.conf
cgi.conf         dirlisting.conf  geoip.conf  Makefile.in  proxy.conf  simple_vhost.conf  userdir.conf
cml.conf         evhost.conf    magnet.conf  mime.conf    rrdtool.conf  ssi.conf  webdav.conf
ahmed@mail:/etc/lighttpd$ ls vhosts.d/
Makefile  Makefile.am  Makefile.in  vhosts.template
ahmed@mail:/etc/lighttpd$
```

I highlighted the most important files that will help me soon in the setup of Virtual hosts, SSL and logging.

Secondly, we understand from the template `lighttpd.conf` file that the directory structure is as follows:

Virtual hosts directory -> `"/srv/www/vhosts"`
Document root -> `"/srv/www/htdocs"`
Logging directory -> `"/var/log/lighttpd"`
Configuration directory -> `"/etc/lighttpd"`

Therefore, we create all the directories as specified and we proceed for a small testing for the web server's functionality.

We make a test page and we place it in the `htdocs` directory:

```
ahmed@mail: /srv/www/htdocs
ahmed@mail:/srv/www/htdocs$ ls
index.html
ahmed@mail:/srv/www/htdocs$ cat index.html
hello, testing.
ahmed@mail:/srv/www/htdocs$
```

Now we configure logging by adding the following two lines in the `access_log.conf` file in the `conf.d` directory:

```
accesslog.filename = "/var/log/lighttpd/lighttpd.log"
```

```
ahmed@mail: /etc/lighttpd/conf.d
GNU nano 4.8      access_log.conf
#####
## Corresponding documentation:
##
## https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModAccesslog
##
server.modules += ( "mod_accesslog" )

##
## Default access log.
##
accesslog.filename      = log_root + "/lighttpd.log"

##
## The default format produces CLF compatible output.
## For available parameters see access.txt
##
```

The previous lines of configuration add the access log module and specify the location for these logs.

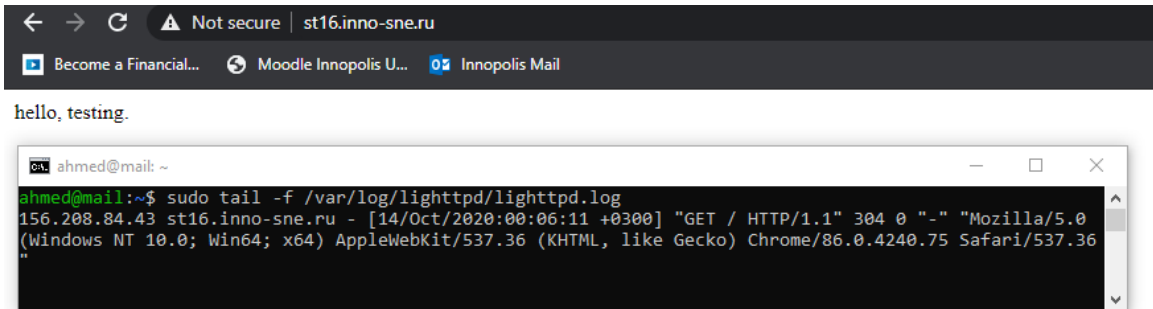
Then check the syntax of the configuration file with:

```
lighttpd -t -f /etc/lighttpd/lighttpd.conf
```

It gives a SYNTAX OK message, now we run the server with:

```
lighttpd -D -f /etc/lighttpd/lighttpd.conf
```

The server looks like up and running:

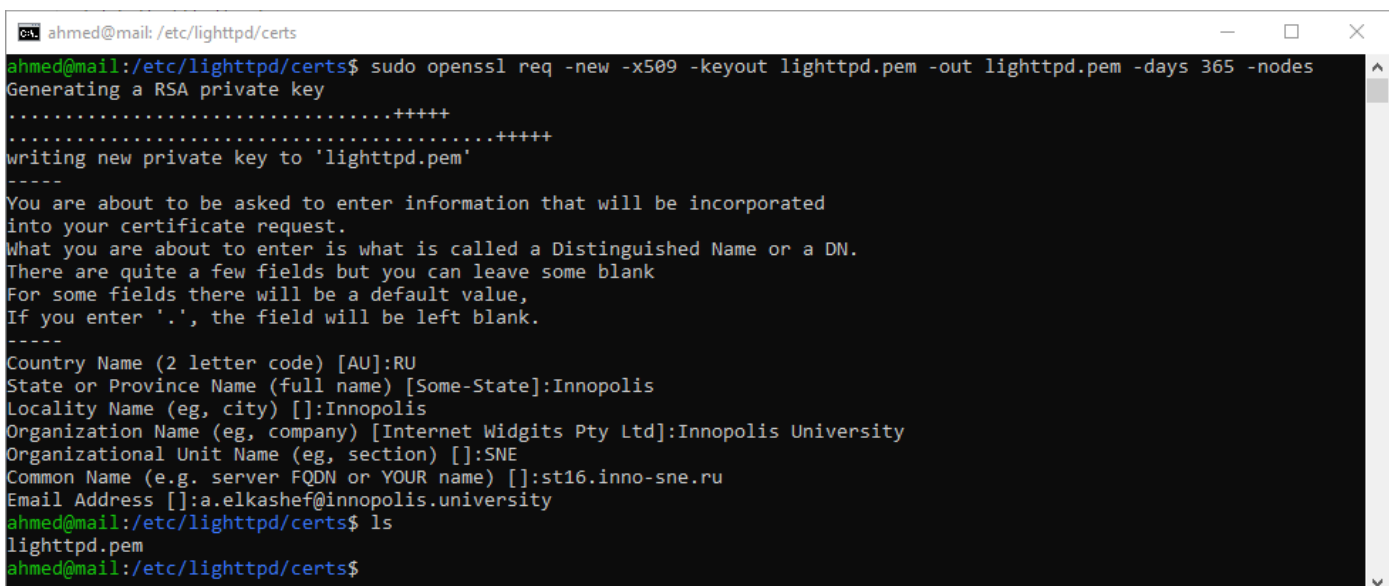


Enabling SSL

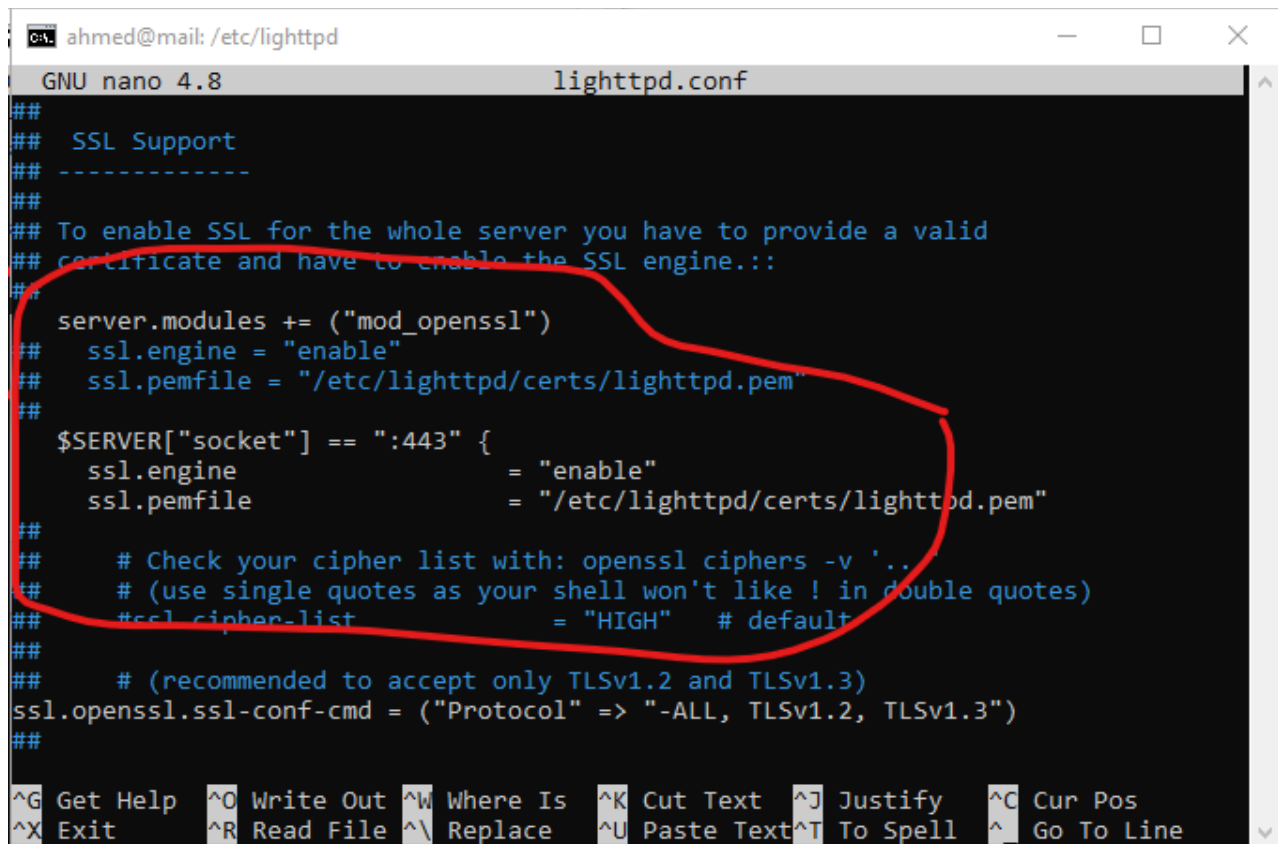
According to the documentation, we can have SSL using a self-signed certificate, we place that in the `/etc/lighttpd/certs` directory and then refer to it from within the configuration file.

We generate this certificate using the following openssl command:

```
openssl req -new -x509 -keyout lighttpd.pem -out lighttpd.pem -days 365 -nodes
```

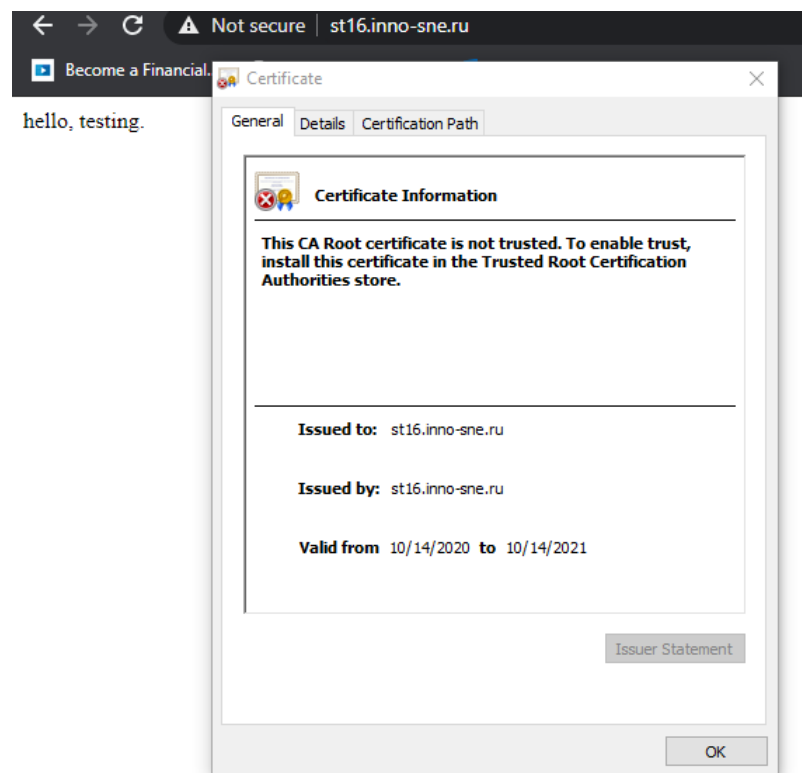


Now we have a self-signed certificate, we edit the lighttpd.conf file:



```
ahmed@mail: /etc/lighttpd
GNU nano 4.8 lighttpd.conf
##
## SSL Support
## -----
##
## To enable SSL for the whole server you have to provide a valid
## certificate and have to enable the SSL engine.::
##
server.modules += ("mod_openssl")
# ssl.engine = "enable"
# ssl.pemfile = "/etc/lighttpd/certs/lighttpd.pem"
#
$SERVER["socket"] == ":443" {
    ssl.engine          = "enable"
    ssl.pemfile         = "/etc/lighttpd/certs/lighttpd.pem"
#
# # Check your cipher list with: openssl ciphers -v '...'
# # (use single quotes as your shell won't like ! in double quotes)
# # ssl_cipher_list      = "HIGH" # default
#
# # (recommended to accept only TLSv1.2 and TLSv1.3)
ssl.openssl.ssl-conf-cmd = ("Protocol" => "-ALL, TLSv1.2, TLSv1.3")
##
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

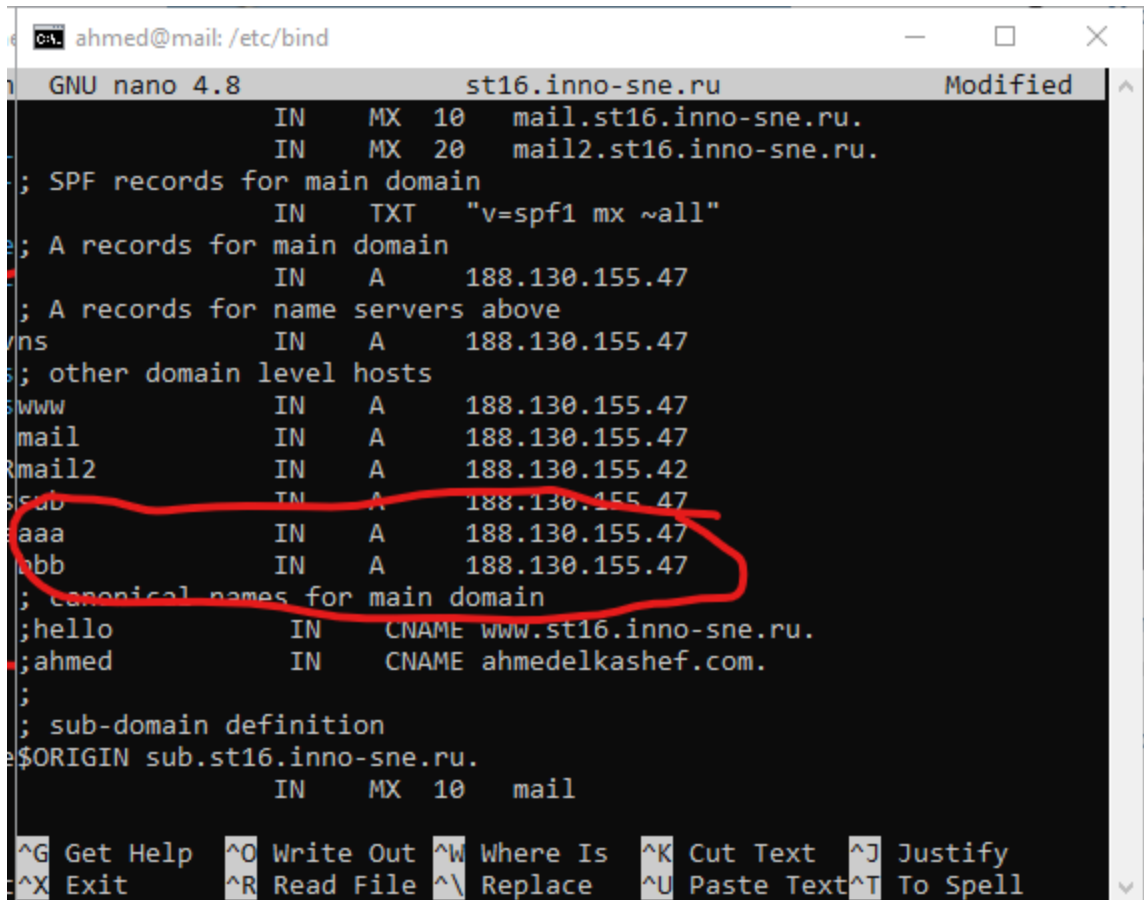
We restart the server and test:



Define the root directory and then two virtual hosts (and configure DNS records or wildcard accordingly):

- `aaa.stX.inno-sne.ru`
- `bbb.stX.inno-sne.ru`

For this new change, we add two new records to our BIND9 Zone file in `/etc/bind/named.conf`:



```
ahmed@mail: /etc/bind
GNU nano 4.8 st16.inno-sne.ru Modified
;
IN      MX  10  mail.st16.inno-sne.ru.
IN      MX  20  mail2.st16.inno-sne.ru.
; SPF records for main domain
IN      TXT    "v=spf1 mx ~all"
; A records for main domain
IN      A      188.130.155.47
; A records for name servers above
IN      A      188.130.155.47
; other domain level hosts
www     IN      A      188.130.155.47
mail    IN      A      188.130.155.47
mail2   IN      A      188.130.155.42
sub     IN      A      188.130.155.47
aaa     IN      A      188.130.155.47
bbb     IN      A      188.130.155.47
; canonical names for main domain
;hello  IN      CNAME  www.st16.inno-sne.ru.
;ahmed  IN      CNAME  ahmedelkashef.com.
;
; sub-domain definition
$ORIGIN sub.st16.inno-sne.ru.
IN      MX  10  mail
```

Create a simple, unique HTML page for each virtual host to make sure that the server can correctly serve it.

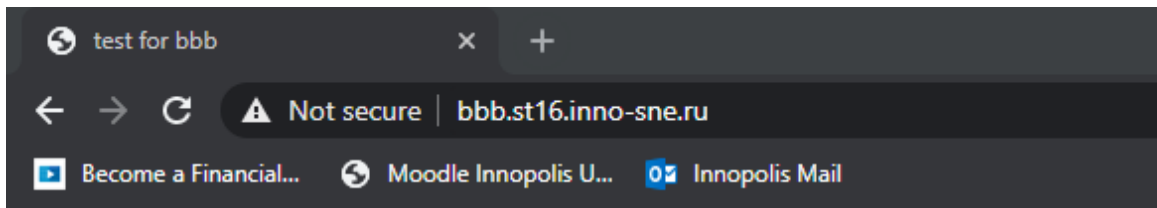
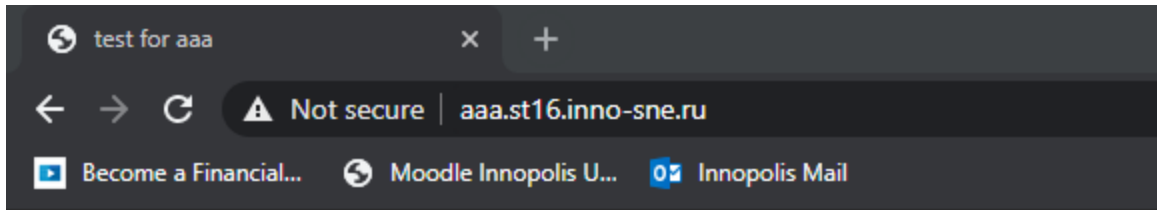
We follow the documentation, and create two basic files in the virtual hosts directory, which is - according to the `lighttpd.conf` file - is at `/srv/www/vhosts`

1st index.html file -> `/srv/www/vhosts/aaa.st16.inno-sne.ru`
2nd index.html file -> `/srv/www/vhosts/bbb.st16.inno-sne.ru`

Inside the `lighttpd.conf`, we then add the following configuration:

```
$HTTP["host"] == "aaa.st16.inno-sne.ru" {
    server.document-root = vhosts_dir + "/aaa.st16.inno-sne.ru"
    var.server_name = "aaa.st16.inno-sne.ru"
}
$HTTP["host"] == "bbb.st16.inno-sne.ru" {
    server.document-root = vhosts_dir + "/bbb.st16.inno-sne.ru"
    var.server_name = "bbb.st16.inno-sne.ru"
}
```

After this, we test both:



Check the configuration syntax, start the daemon and enable it at boot time.

To check the configuration syntax, we run the following command:

```
lighttpd -t -f /etc/lighttpd/lighttpd.conf
```

To start the daemon, we run the following command:

```
lighttpd -D -f /etc/lighttpd/lighttpd.conf
```

Use curl to display the contents of a full HTTP/1.1 session served by your server.
Explain the meaning of each request and reply header.

For aaa.st16.inno-sne.ru:

```
ahmed@mail: ~
ahmed@mail:~$ curl -v -k https://aaa.st16.inno-sne.ru
* Trying 188.130.155.47:443...
* TCP_NODELAY set
* Connected to aaa.st16.inno-sne.ru (188.130.155.47) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crt
*   CPath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server accepted to use http/1.1
* Server certificate:
*   subject: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com
*   start date: Oct 13 22:07:11 2020 GMT
*   expire date: Oct 13 22:07:11 2021 GMT
*   issuer: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com
*   SSL certificate verify result: self signed certificate (18), continuing anyway
.
> GET / HTTP/1.1
> Host: aaa.st16.inno-sne.ru
> User-Agent: curl/7.68.0
> Accept: */*
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Content-Type: text/html
< Accept-Ranges: bytes
< ETag: "2570617897"
< Last-Modified: Tue, 13 Oct 2020 23:18:45 GMT
< Content-Length: 131
< Date: Tue, 13 Oct 2020 23:21:18 GMT
< Server: lighttpd/1.4.55
<
<html>
  <head>
    <title>test for aaa</title>
  </head>
  <body>
    <h1>welcome to my page, this is only for aaa<h2>
  </body>
</html>
```

For bbb.st16.inno-sne.ru:

```
ahmed@mail: ~
ahmed@mail:~$ curl -v -k https://bbb.st16.inno-sne.ru
* Trying 188.130.155.47:443...
* TCP_NODELAY set
* Connected to bbb.st16.inno-sne.ru (188.130.155.47) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
* CAfile: /etc/ssl/certs/ca-certificates.crt
* Capath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server accepted to use http/1.1
* Server certificate:
* subject: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com
* start date: Oct 13 22:07:11 2020 GMT
* expire date: Oct 13 22:07:11 2021 GMT
* issuer: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com
* SSL certificate verify result: self signed certificate (18), continuing anyway
.
> GET / HTTP/1.1
> Host: bbb.st16.inno-sne.ru
> User-Agent: curl/7.68.0
> Accept: */*
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Content-Type: text/html
< Accept-Ranges: bytes
< ETag: "2149640773"
< Last-Modified: Tue, 13 Oct 2020 23:19:17 GMT
< Content-Length: 131
< Date: Tue, 13 Oct 2020 23:22:06 GMT
< Server: lighttpd/1.4.55
<
<html>
  <head>
    <title>test for bbb</title>
  </head>
  <body>
    <h1>welcome to my page, this is only for bbb<h2>
  </body>
</html>
```

To explain the headers one by one:

* TCP_NODELAY set	setting this means that TCP means that segments are always sent as soon as possible
* ALPN, offering h2	ALPN (Application Layer Protocol Negotiation) is a TLS extension that includes the protocol negotiation within the exchange of hello messages. Here ALPN offers HTTP/2 negotiation
* ALPN, offering http/1.1	Here ALPN offers HTTP/1.1 negotiation
* successfully set certificate verify locations: * CAfile: /etc/ssl/certs/ca-certificates.crt CApath: /etc/ssl/certs	Determining and setting the path for the SSL certificate directory and files
* TLSv1.3 (OUT), TLS handshake, Client hello (1): * TLSv1.3 (IN), TLS handshake, Server hello (2): * TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8): * TLSv1.3 (IN), TLS handshake, Certificate (11): * TLSv1.3 (IN), TLS handshake, CERT verify (15): * TLSv1.3 (IN), TLS handshake, Finished (20): * TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1): * TLSv1.3 (OUT), TLS handshake, Finished (20):	TLS Handshake between client and server: (OUT) means outgoing from the client to the server (IN) means ingoing from the server to the client * Client says Hello, server replies with: 1) Hello 2) SSL Certificate 3) Server's chosen cipher suite * Client exchanges the cipher and connects.
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384	Both client and server have established the TLS connection.
* ALPN, server accepted to use http/1.1	The server accepted the offer to go with HTTP/1.1
* Server certificate: * subject: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com * start date: Oct 13 22:07:11 2020 GMT * expire date: Oct 13 22:07:11 2021 GMT * issuer: C=RU; ST=Tatarstan; L=Innopolis; O=Innopolis-University; OU=SNE; CN=st16.inno-sne.ru; emailAddress=ahmedelkashef2012@gmail.com * SSL certificate verify result: self signed certificate (18), continuing anyway.	Displayed information about the SSL Self-Signed certificate. All those information were prefilled at the moment of using openssl command to create it.
> GET / HTTP/1.1	GET request from client to server
> Host: aaa.st16.inno-sne.ru	A third piece of information to identify a host, in addition to IP and port number. (useful in the case of many virtual hosts on a domain), if it is not specified, the root Web domain will act as the default Web server.
> User-Agent: curl/7.68.0	The user agent to initiate the request, it is initiated by curl, version 7.68.0
> Accept: /*/*	Accept any MIME type, MIME (Multipurpose Internet Mail Extensions) are types of data and their extensions. E.g. if the server is sending back images, it needs to specify if its a JPG or PNG, this information is specified in its MIME type header.
< HTTP/1.1 200 OK	Success message from the server (200 OK)

< Content-Type: text/html	The type of document to be received (type/subtype) Type: text Subtype: html
< Accept-Ranges: bytes	Here, the server indicates that bytes are the units to define a 'range', this is useful in the case of partial requests, if the server sets this to "none" then it doesn't support partial requests. You can know the full length of the data with the Content-length header. Example of partial request: <i>curl http://i.imgur.com/z4d4kWk.jpg -i -H "Range: bytes=0-1023"</i>
< ETag: "2570617897"	Entity Tag: one of the mechanisms that HTTP provide for web cache validation. ETag is an identifier assigned by a Web server to a specific version of a resource found at a URL.
< Last-Modified: Tue, 13 Oct 2020 23:18:45 GMT	Last time this data was modified
< Content-Length: 131	The full length of the data
< Date: Thu, 15 Oct 2020 00:04:46 GMT	The date of the request
< Server: lighttpd/1.4.55	The version and the type of the server itself

• Task 2 - SSL/TLS

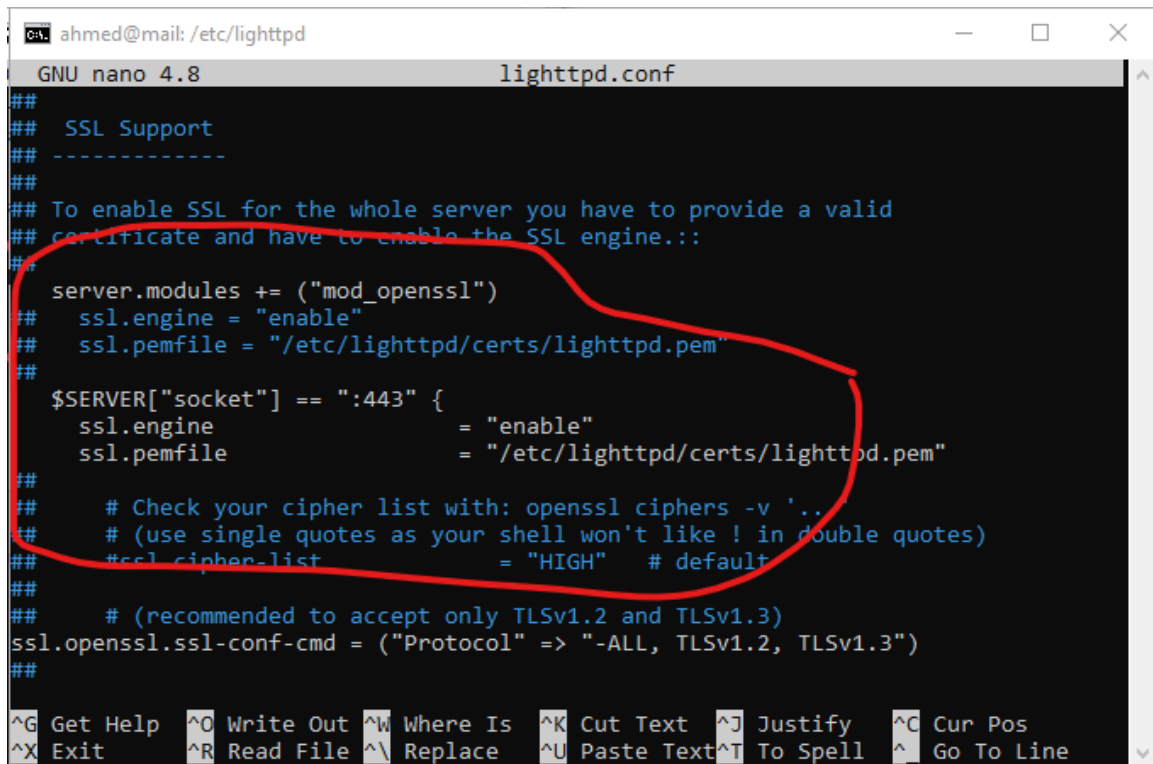
Enable SSL/TLS and tune the various settings to make it as secure as possible. Describe how you created your own certificate(s) e.g. with Let's encrypt or self-signed and re-validate every virtual-host versus root directory. Explain your security tuning process.

As described earlier, SSL/TLS was enabled from the very start during the building and compilation process, and then was configured on two steps:

- 1) Creating a self signed certificate using the following openssl command:

```
openssl req -new -x509 -keyout lighttpd.pem -out lighttpd.pem -days 365 -nodes
```

- 2) Configuring SSL/TLS to work on the whole server (all virtual hosts) from the lighttpd.conf file in the SSL section as below:



```
ahmed@mail: /etc/lighttpd
GNU nano 4.8 lighttpd.conf
##
##  SSL Support
##  -----
##
##  To enable SSL for the whole server you have to provide a valid
##  certificate and have to enable the SSL engine.:
##
server.modules += ("mod_openssl")
#  ssl.engine = "enable"
#  ssl.pemfile = "/etc/lighttpd/certs/lighttpd.pem"
##
$SERVER["socket"] == ":443" {
    ssl.engine          = "enable"
    ssl.pemfile         = "/etc/lighttpd/certs/lighttpd.pem"
##
#  # Check your cipher list with: openssl ciphers -v '...'
#  # (use single quotes as your shell won't like ! in double quotes)
#  # ssl-cipher-list = "HIGH" # default
##
#  # (recommended to accept only TLSv1.2 and TLSv1.3)
ssl.openssl.ssl-conf-cmd = ("Protocol" => "-ALL, TLSv1.2, TLSv1.3")
##
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File ^\ Replace  ^U Paste Text ^T To Spell  ^_ Go To Line
```

Accept only TLSv1.2 and TLSv1.3 by enabling this option:

```
Ssl.openssl.ssl-conf-cmd = ("Protocol" => "-All, TLSv1.2, TLSv1.3")
```

Also the certificate - since it is self-signed - is not trusted by the testing website.

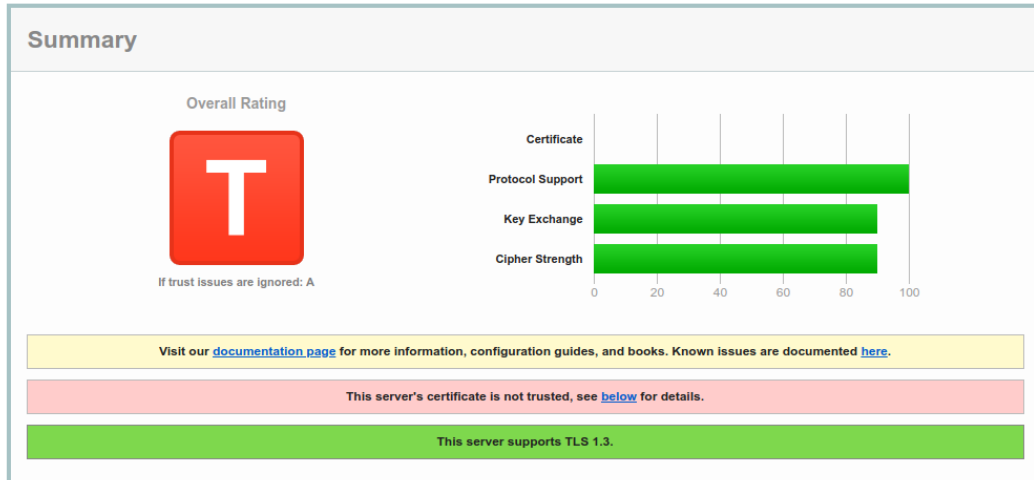
Finally, testing it:

You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [aaa.st16.inno-sne.ru](#)

SSL Report: [aaa.st16.inno-sne.ru](#) (188.130.155.47)

Assessed on: Wed, 14 Oct 2020 07:58:33 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)

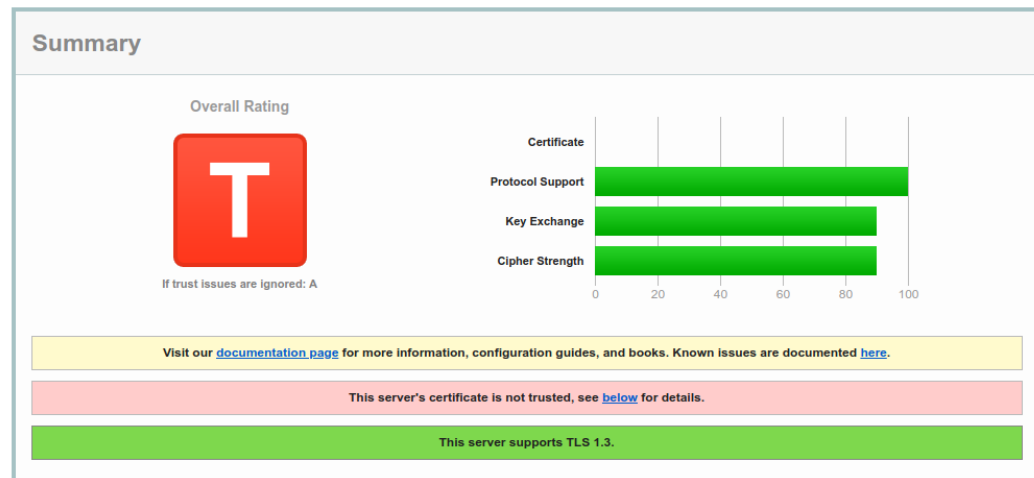


You are here: [Home](#) > [Projects](#) > [SSL Server Test](#) > [bbb.st16.inno-sne.ru](#)

SSL Report: [bbb.st16.inno-sne.ru](#) (188.130.155.47)

Assessed on: Wed, 14 Oct 2020 07:42:36 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)



• Task 3 - Web Server Performance

Investigate what configuration options there are that can potentially improve the performance of the web server.

The main idea about the performance in lighttpd, is that it is a single-threaded server, and its main resource limit are the file-descriptors. Therefore, our optimization will focus on that aspect.

According to the documentation, those options that can potentially improve the performance are the following:

- Out-of-fd condition
- Increase Number of Maximum Connections
- Disabling HTTP Keep-Alives
- stat() cache

Out-of-fd condition: As described, file descriptors are the main resource limit for the lighttpd server. Also, taking into consideration that a simple request for a PHP page can result in the usage of 3 file descriptors:

- 1) TCP/IP socket to the client
- 2) TCP/IP and Unix domain socket to the FastCGI process
- 3) filehandle to the file in the document root to check if it exists

Therefore, if the server runs out of File-descriptors, an error like this has a high probability of being found:

```
... accept() failed: Too many open files
```

Increase Number of Maximum Connections: The solution for this, is to increase the server.max-fds limit.

The current server.max-fds in the configuration file is: 1024, which we increase to 2048.

```
## By default lighttpd would not change the operation system default.  
## But setting it to 2048 is a better default for busy servers.  
##  
server.max-fds = 2048
```

Decreasing HTTP Keep-Alives: The default configuration contains the following:

```
server.max-keep-alive-requests = 16  
server.max-keep-alive-idle = 5  
server.max-read-idle = 60  
server.max-write-idle = 360
```

This configuration translates to:

“Handle at maximum 16 keep-alive requests in a row on a single connection, and wait 5 seconds before lighttpd drops the unused keep-alive connection.”

Under heavy usage conditions, and with the current defaults, we can easily lead our server to run out of file descriptors as explained before. In order to fix this, we configure the server to use a less amount of keep-alives in the same time, and hence keep our server away from running out of file descriptors: **4 keep-alives instead of 16**

```
server.max-keep-alive-requests = 4
server.max-keep-alive-idle = 4
```

Stat() cache: stat() system call is expensive, it produces so much information and caching it saves a lot of time and context switches.

stat() is usually used to check the existence of a file. Everytime the server is asked for a specific file, it runs this system call and then returns back to the client with an answer. Instead, we can opt to run it only once and monitor the directory of that file for any changes, and only then, we can run it again and update the cached information with the new results.

There are three options for this caching engine:

- 1) No cache engine -> disable
- 2) Fam -> caching engine to keep track of directories
- 3) Simple -> caches the stat() result for 1 second

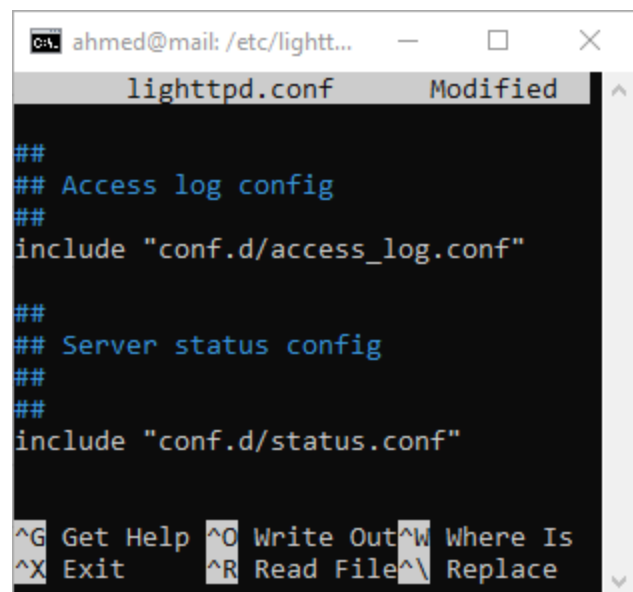
The current setting was to “**disable**”, we set it to “**fam**”.

Also look at how you can check the (current) load on the web server using e.g. the Apache mod status module.

This can be achieved in lighttpd using the (mod_status) module.

We start by loading it to the lighttpd.conf file.

Then by adding the configuration in the `conf.d/status file`, which will contain a conditional statement, that will enable only `127.0.0.1` (the server itself) from opening certain URLs to check for the server status. In addition, we also enable another directive for server configuration information. Screen shots are below:



```
ahmed@mail: /etc/lighttt...
lighttpd.conf Modified
##
## Access log config
##
include "conf.d/access_log.conf"
##
## Server status config
##
include "conf.d/status.conf"
^G Get Help ^O Write Out ^W Where Is
^X Exit ^R Read File ^\\ Replace
```

```

ahmed@mail: /etc/lighttpd/conf.d
GNU nano 4.8      status.conf      Modified
#####
##
##  Status Module
##  -----
##
## See https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModStatus
##
server.modules += ( "mod_status" )

$http["remoteip"] == "127.0.0.0/8" {
##
## configure urls for the various parts of the module.
##
    status.status-url      = "/server-status"
    status.config-url      = "/server-config"
##
## add JavaScript which allows client-side sorting for the connection
## overview
##
    status.enable-sort      = "enable"
}
##
#####
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text   ^J Justify
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell

```

Testing it from the same server computer:

Server Config:

Status - Mozilla Firefox

Telegram Web x Status x Status x

← → ↻ 🏠 🔒 https://127.0.0.1/server-config

lighttpd/1.4.55

Server-Features	
RegEx Conditionals	enabled
Network Engine	
fd-Event-Handler	linux-sysepoll
Config-File-Settings	
Loaded Modules	indexfile access accesslog status openssl dirlisting staticfile

Server status:

Status - Mozilla Firefox

Telegram Web x Status x Status x +

https://127.0.0.1/server-status

Server-Status (lighttpd/1.4.55)

Hostname	127.0.0.1 ()
Uptime	1 min 10 s
Started at	2020-10-15 00:39:44
absolute (since start)	
Requests	2 req
Traffic	705.00 byte
average (since start)	
Requests	0 req/s
Traffic	10.07 byte/s
average (5s sliding average)	
Requests	0 req/s
Traffic	0.00 byte/s

1 connections

h

- 0 k = keep-alive
- 0 . = connect
- 0 q = req-start
- 0 r = read
- 0 Q = req-end
- 0 R = readpost
- 1 h = handle-req
- 0 s = resp-start
- 0 W = write
- 0 S = resp-end
- 0 E = error
- 0 C = close

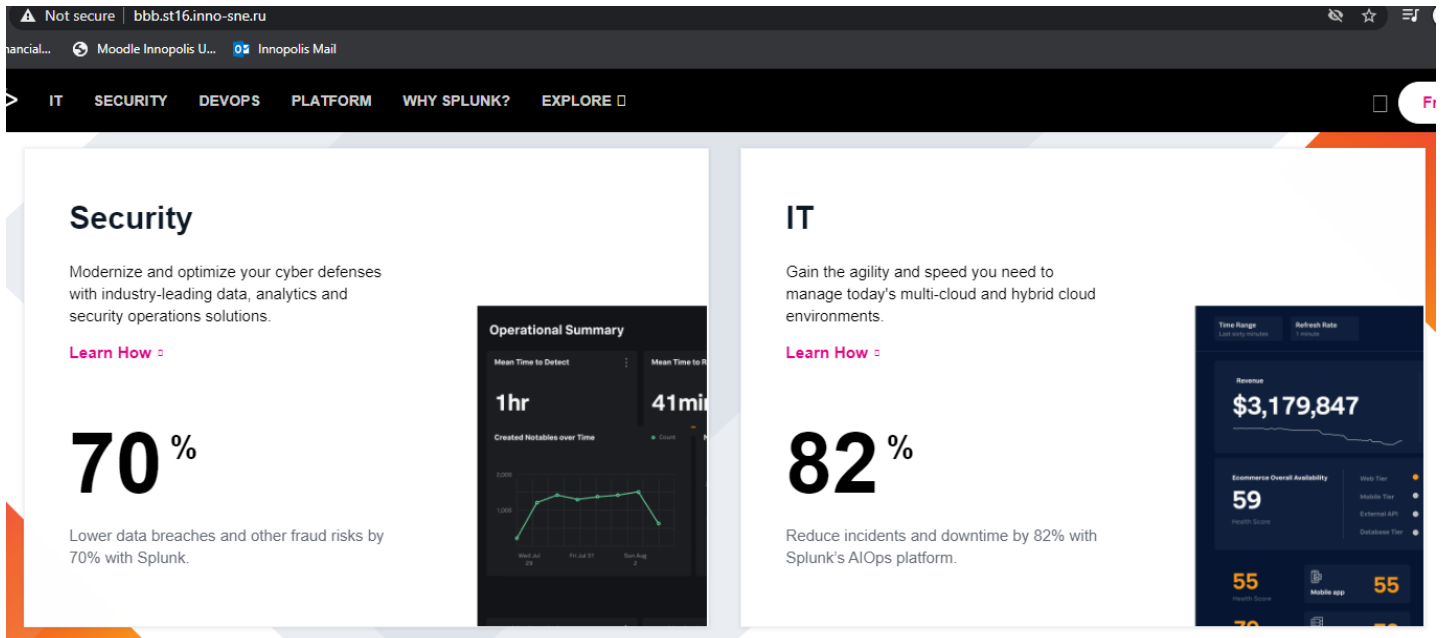
Connections

Client IP:	Read:	Written:	State:	Time:	Host:	URI:	File:
127.0.0.1	0/0	0/5222	handle-req	0	127.0.0.1	/server-status (/server-status)	

Using a standard benchmarking tool (e.g. ab, siege, etc.) evaluate the performance of your server before and after optimizations for both the static page and the dynamic page. Try to maximize the number of requests per second. Explain all the changes made.

First, we need to create two pages, 1 static page and place it in the aaa.st16.inno-sne.ru and the other one is a dynamic page and place it in the bbb.inno-sne.ru page.

Static page is a simple html page with only one tag and it is the current page from the other tasks.
Dynamic page, will be clone a website page and place it in the appropriate directory for bbb:



We use siege for this, and install it via apt:

```
sudo apt install siege
```

And start our testing on the aaa and the bbb respectively, using the same number of users (255 default limit) in addition to the -b flag which runs the test with NO DELAY for throughput benchmarking.

```
siege -c255 -t5S -b aaa.st16.inno-sne.ru
```

aaa.st16.inno-sne.ru (before)

aaa.st16.inno-sne.ru (after)

```
ahmedelkashef@Su...
Lifting the server siege...
Transactions:      3321 hits
Availability:      96.01 %
Elapsed time:      59.11 secs
Data transferred:  0.41 MB
Response time:     2.27 secs
Transaction rate:  56.18 trans/sec
Throughput:        0.01 MB/sec
Concurrency:       127.33
Successful transactions: 3321
Failed transactions: 138
Longest transaction: 34.03
Shortest transaction: 0.36
ahmedelkashef@Supertramp:~$
```

```
ahmedelkashef@Su...
Lifting the server siege...
Transactions:      3668 hits
Availability:      95.85 %
Elapsed time:      59.37 secs
Data transferred:  0.46 MB
Response time:     1.67 secs
Transaction rate:  61.78 trans/sec
Throughput:        0.01 MB/sec
Concurrency:       103.17
Successful transactions: 3668
Failed transactions: 159
Longest transaction: 22.71
Shortest transaction: 0.31
ahmedelkashef@Supertramp:~$
```

bbb.st16.inno-sne.ru (before)

bbb.st16.inno-sne.ru (after)

```
ahmedelkashef@Su...
Lifting the server siege...
Transactions:          31 hits
Availability:          37.80 %
Elapsed time:          59.08 secs
Data transferred:      2.26 MB
Response time:         7.19 secs
Transaction rate:      0.52 trans/sec
Throughput:            0.04 MB/sec
Concurrency:           3.77
Successful transactions: 31
Failed transactions:   51
Longest transaction:   56.14
Shortest transaction:  0.00
ahmedelkashef@Supertramp:~$
```

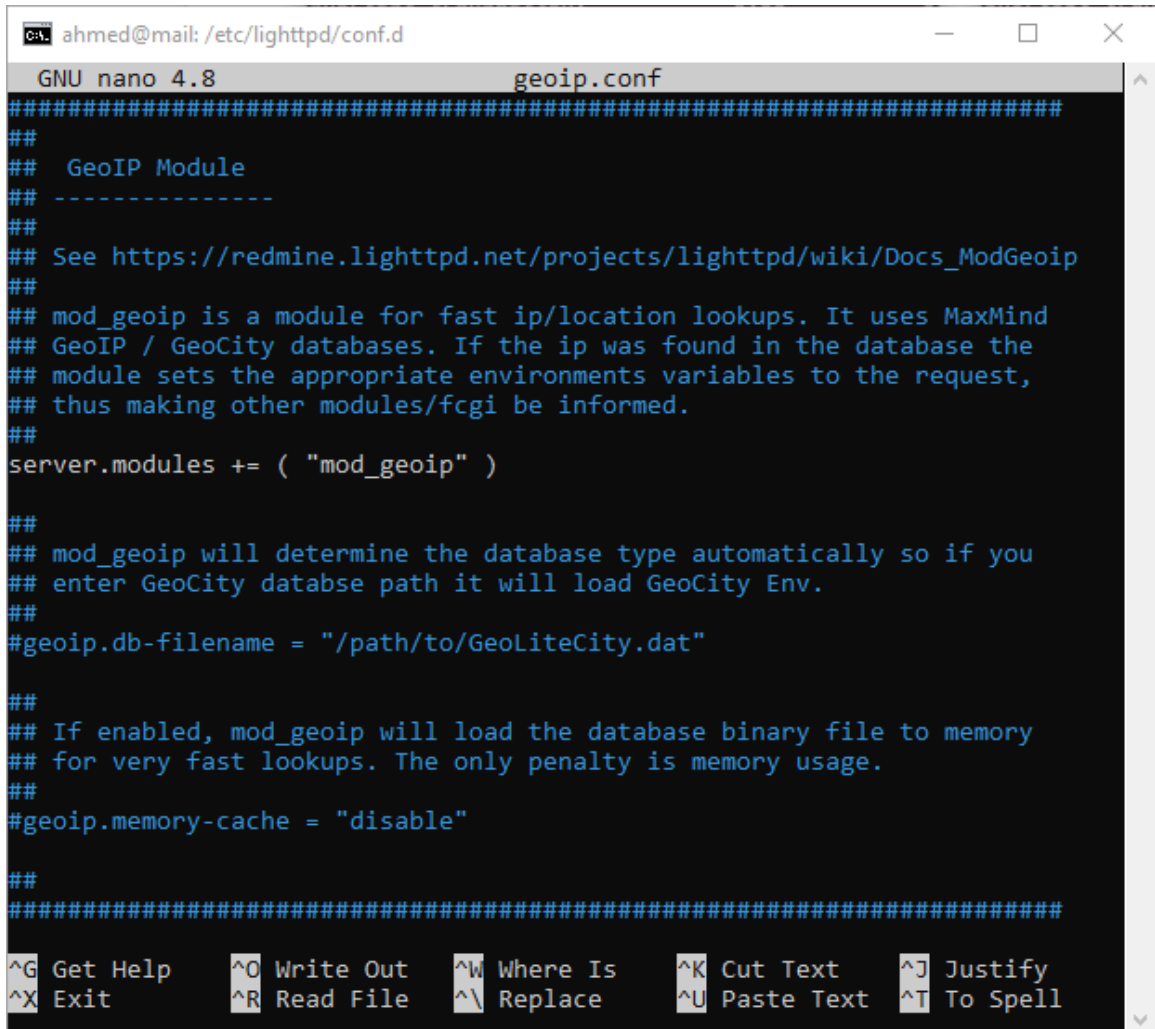
```
ahmedelkashef@Su...
Lifting the server siege...
Transactions:          150 hits
Availability:          69.44 %
Elapsed time:          59.66 secs
Data transferred:      28.64 MB
Response time:         22.08 secs
Transaction rate:      2.51 trans/sec
Throughput:            0.48 MB/sec
Concurrency:           55.52
Successful transactions: 150
Failed transactions:   66
Longest transaction:   59.06
Shortest transaction:  0.00
ahmedelkashef@Supertramp:~$
```

As it is clear from the results, the server was now having a slightly better improvement in the response time, and an increase in the successful transactions in the case of static pages. However, the true improvement is noticed when the power of cache is in use with the dynamic page. Since the server was able to do more transactions, is more available and transfers a bigger number of data with higher concurrency. Also the number of successful transactions has increased significantly as a result.

• Task 3 - GeoIP

Previously, we have configured our server to use GeoIP libraries in the building and compilation process.

By examining the conf.d directory, we find a geoip.conf file, we examine the contents of that file:



```
ahmed@mail: /etc/lighttpd/conf.d
GNU nano 4.8 geoip.conf
#####
##
##  GeoIP Module
##  -----
##
##  See https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModGeoip
##
##  mod_geoip is a module for fast ip/location lookups. It uses MaxMind
##  GeoIP / GeoCity databases. If the ip was found in the database the
##  module sets the appropriate environments variables to the request,
##  thus making other modules/cgi be informed.
##
server.modules += ( "mod_geoip" )

##
##  mod_geoip will determine the database type automatically so if you
##  enter GeoCity database path it will load GeoCity Env.
##
#geoip.db-filename = "/path/to/GeoLiteCity.dat"

##
##  If enabled, mod_geoip will load the database binary file to memory
##  for very fast lookups. The only penalty is memory usage.
##
#geoip.memory-cache = "disable"

##
#####
^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify
^X Exit          ^R Read File    ^\ Replace      ^U Paste Text   ^T To Spell
```

From the documentation, we understand that the server module “mod_geoip” has been deprecated and the currently supported module is “mod_maxminddb”

We need to download the GeoLite2 database file, which If the IP was found in the database, the module sets the appropriate environment variables to the request, thus making this information available to other modules/cgi.

In order to acquire that GeoLite2 databases files, we [signup](#) and download them and place them in the conf.d/ directory

```
ahmed@mail: /etc/lighttpd/conf.d
ahmed@mail:/etc/lighttpd/conf.d$ ls
access_log.conf  GeoLite2-ASN.mmdb      rrdtool.conf
auth.conf        GeoLite2-City.mmdb     scgi.conf
cgi.conf         GeoLite2-Country.mmdb  secdownload.conf
cml.conf         magnet.conf            simple_vhost.conf
compress.conf    Makefile               ssi.conf
debug.conf       Makefile.am            status.conf
dirlisting.conf  Makefile.in            trigger_b4_d1.conf
evhost.conf      mime.conf              userdir.conf
expire.conf      mod.template           webdav.conf
fastcgi.conf     mysql_vhost.conf
geoip.conf       proxy.conf
ahmed@mail:/etc/lighttpd/conf.d$
```

And edit the configuration file accordingly to point to the Geolite-City.mmdb file and include the maxminddb module:

```
GNU nano 4.8      geoip.conf      Modified
#####
##
##  MaxmindDB Module
##  -----
##
##  See https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModGeoip
##
##  mod_maxminddb is a module for fast ip/location lookups. It uses MaxMind
##  GeoIP / GeoCity databases. If the ip was found in the database the
##  module sets the appropriate environments variables to the request,
##  thus making other modules/fcgi be informed.
##
server.modules += ( "mod_maxminddb" )
maxminddb.activate = "enable"
maxminddb.db       = "GeoLite2-City.mmdb"
maxminddb.env = (
    "GEOIP_COUNTRY_CODE"    => "country/iso_code",
    "GEOIP_COUNTRY_NAME"    => "country/names/en",
    "GEOIP_CITY_NAME"       => "city/names/en",
    "GEOIP_CITY_LATITUDE"   => "location/latitude",
    "GEOIP_CITY_LONGITUDE"  => "location/longitude",
)
##
##  mod_maxminddb will determine the database type automatically so if you
##  enter GeoCity database path it will load GeoCity Env.
##
^G Get Help      ^O Write Out     ^W Where Is      ^K Cut Text      ^J Justify
^X Exit          ^R Read File     ^\ Replace       ^U Paste Text    ^T To Spell
```

After the setup, we test the program and it appears that the program has a segmentation fault as a result. After some research, it looks like Geolite2 database's structure is now different than what is specified in the documentation of lighttpd here:

https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModMaxMindDB

And the following block:

```
maxminddb.env = (  
    "GEOIP_COUNTRY_CODE"    => "country/iso_code",  
    "GEOIP_COUNTRY_NAME"    => "country/names/en",  
    "GEOIP_CITY_NAME"       => "city/names/en",  
    "GEOIP_CITY_LATITUDE"   => "location/latitude",  
    "GEOIP_CITY_LONGITUDE"  => "location/longitude",  
)
```

Is what maps the database file into the maxminddb environment, which in turn gets passed to the mod_magnet for redirection and control. Since the module for handling the database was not documented enough, I believe that my efforts were not enough to complete the missing gap.

(is only NGINX capable to do this?)

Definitely no, as I showed, it can work with the right and updated modules also in lighttpd using the following modules:

- mod_geopip (or the new mod_mindmaxdb)
- mod_magnet

● References:

1. <https://www.lighttpd.net/>
2. <https://www.lighttpd.net/download/>
3. <https://en.wikipedia.org/wiki/Lighttpd>
4. <https://redmine.lighttpd.net/projects/lighttpd/wiki>
5. <https://redmine.lighttpd.net/projects/lighttpd/wiki/OptionalLibraries>
6. <https://redmine.lighttpd.net/projects/lighttpd/wiki/InstallFromSource>
7. <https://redmine.lighttpd.net/projects/lighttpd/wiki/TutorialConfiguration>
8. <https://git.lighttpd.net/lighttpd/lighttpd1.4/src/branch/master/INSTALL>
9. https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_SSL
10. https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_Performance
11. https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_PerformanceFastCGI
12. <https://redmine.lighttpd.net/projects/lighttpd/wiki/HowToSimpleSSL>
13. <https://www.cyberciti.biz/tips/lighttpd-web-server-awstats-tutorial.html>
14. https://redmine.lighttpd.net/projects/lighttpd/wiki/Docs_ModMaxMindDB
15. https://wiki.alpinelinux.org/wiki/Production_Web_server:_Lighttpd
16. <https://hubpages.com/technology/Configure-Apache2-VirtualHost-and-Bind9-on-Debian>
17. <https://tools.ietf.org/html/rfc5424>