FreeBSD Administration - Support #718

Install a Firefox Accounts Server on FreeBSD

01/09/2016 09:27 PM - Daniel Curtis

Status:	Suspended	**Start date:**	01/09/2016
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:	Sync Server	**Estimated time:**	2.00 hours
Target version:	FreeBSD 10	**Spent time:**	13.00 hours

Description

This is a guide on how I installed the Firefox Auth, Content, and Sync components to form the Firefox Accounts Server on FreeBSD 10.

Prepare the Environment

- Make sure the system is up to date:

```
pkg update && pkg upgrade -y
```

- Install a few dependencies:

```
pkg install portmaster sudo bash git gmp graphicsmagick redis gmake postfix python2 py27-virtu
alenv sqlite py27-sqlite3 gcc48 scrypt
```

- Start and enable postfix and disable sendmail at boot:

```
echo 'postfix_enable="YES"' >> /etc/rc.conf
echo 'sendmail_enable="NONE"' >> /etc/rc.conf
newaliases
service postfix start
```

- Install node4 and npm2 from ports:

```
portmaster www/node4 www/npm2
```

- Install a few node modules globally:

```
npm install -g pm2 bower grunt
```

- Add the Firefox Accounts user:

```
mkdir -p /usr/local/www/fxa
pw groupadd fxa
pw adduser -n fxa -g fxa -d /usr/local/www/fxa -s /bin/sh -c "Firefox Accounts"
chown fxa:fxa /usr/local/www/fxa
```

Memcached Server

- Install memcached:

```
pkg install memcached
```

- Start and enable memcached at boot:

```
echo 'memcached_enable="YES"' >> /etc/rc.conf
service memcached start
```

MySQL Database

- Install MariaDB:

```
pkg install mariadb101-{client,server}
```

- Start and enable MariaDB at boot:

```
echo 'mysql_enable="YES"' >> /etc/rc.conf
service mysql-server start
```

- Secure the mysql installation:

```
mysql_secure_installation
```

- Log into the MySQL console:

```
mysql -u root -p
```

    - Create the **fxauser** user with the SuperSecretPassword password and the **fxadb** database:

```
CREATE USER 'fxauser'@'localhost' IDENTIFIED BY 'SuperSecretPassword';
CREATE DATABASE IF NOT EXISTS  `fxadb` CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON `fxadb`.* TO 'fxauser'@'localhost';
```

    - Create the **fxasyncuser** user with the SuperDuperPassword password and the **fxasyncdb** database:

```
CREATE USER 'fxasyncuser'@'localhost' IDENTIFIED BY 'SuperDuperPassword';
CREATE DATABASE IF NOT EXISTS  `fxasyncdb` CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON `fxasyncdb`.* TO 'fxasyncuser'@'localhost';
```

    - Create the **fxaprofileuser** user with the SuperSuperPassword password and the **fxaprofiledb** database:

```
CREATE USER 'fxaoauthuser'@'localhost' IDENTIFIED BY 'SuperOauthPassword';
CREATE DATABASE IF NOT EXISTS  `fxaoauthdb` CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON `fxaoauthdb`.* TO 'fxaoauthuser'@'localhost';
```

    - Create the **fxaprofileuser** user with the SuperSuperPassword password and the **fxaprofiledb** database:

```
CREATE USER 'fxaprofileuser'@'localhost' IDENTIFIED BY 'SuperSuperPassword';
CREATE DATABASE IF NOT EXISTS  `fxaprofiledb` CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON `fxaprofiledb`.* TO 'fxaprofileuser'@'localhost';
```

- Exit the mysql console:

```
flush privileges;
exit
```

Install Accounts Server

- Switch to the fxa directory:

```
cd /usr/local/www/fxa
```

- Download the firefox auth server from GitHub:

```
sudo -u fxa git clone https://github.com/mozilla/fxa-auth-server.git
cd fxa-auth-server
```

- Install the auth server:

```
sudo -u fxa npm install
```

- And test the auth server:

```
sudo -u fxa npm start
```

NOTE: Press Ctrl+C to stop the test server.

- Edit the development environment config file:

```
vi .env.dev
```

- And adjust the config file accordingly:

```
MYSQL_USER=fxauser
MYSQL_PASSWORD=SuperSecretPassword
MYSQL_DATABASE=fxadb
MYSQL_HOST=localhost
MYSQL_PORT=3306
MYSQL_SLAVE_USER=fxauser
MYSQL_SLAVE_PASSWORD=SuperSecretPassword
MYSQL_SLAVE_DATABASE=fxadb
MYSQL_SLAVE_HOST=localhost
MYSQL_SLAVE_PORT=3306
PUBLIC_URL=https://api.accounts.example.com
CONTENT_SERVER_URL=https://accounts.example.com
CUSTOMS_SERVER_URL=none
OAUTH_URL=https://oauth.accounts.example.com
LOCKOUT_ENABLED=true
LOG_FORMAT=pretty
LOG_LEVEL=info
RESEND_BLACKOUT_PERIOD=0
SIGNIN_CONFIRMATION_ENABLED=false
SIGNIN_CONFIRMATION_RATE=1
SMTP_HOST=mail.example.com
SMTP_PORT=25
SMTP_SECURE=false
SMTP_USER=no-reply@example.com
```

```
SMTP_PASS=SuperSecretMailPassword
SMTP_SENDER='Firefox Accounts <no-reply@example.com>'
SNS_TOPIC_ARN=disabled
STATSD_SAMPLE_RATE=1
TRUSTED_JKUS=http://127.0.0.1:8080/.well-known/public-keys,http://127.0.0.1:10139/.well-kn
own/public-keys
VERIFICATION_REMINDER_RATE=1
VERIFIER_VERSION=0
SIGNIN_CONFIRMATION_FORCE_EMAIL_REGEX=/.+@example\.com$/
```

- **NOTE**: Copy the development environment file over to the production environment file to work with when going into production:

```
cp .env.dev .env.prod
```

- Start the server in dev MySQL store mode:

```
sudo -u fxa npm run start-mysql
```

NOTE: Press Ctrl+C to stop the test server.

- A persistent deployment will require pm2:

```
sudo -u fxa pm2 start npm --name fxa-auth -- run start-mysql
```

Auth Server Init Script

- Create a firefox auth server init script:

```
vi /usr/local/etc/rc.d/fxa-auth
```

- and add the following

```
#!/bin/sh

# PROVIDE: fxa-auth
# KEYWORD: shutdown

. /etc/rc.subr

: ${fxa_auth_path="/usr/local/www/fxa/fxa-auth-server"}
: ${fxa_auth_env="dev"}

name="fxa_auth"
start_cmd="${name}_start"
stop_cmd="${name}_stop"

fxa_auth_start() {
   echo "Firefox auth server starting"
   su - fxa -c "cd ${fxa_auth_path}; /usr/local/bin/pm2 start npm --name ${name} --env ${f
xa_auth_env} -- run start-mysql; exit"
}

fxa_auth_stop() {
   echo "Firefox auth server stopping"
   su - fxa -c "/usr/local/bin/pm2 delete ${name}; exit"
}

run_rc_command "$1"
```

- And make it executable:

```
chmod +x /usr/local/etc/rc.d/fxa-auth
```

- Start and enable firefox auth server at boot

```
echo 'fxa_auth_enable="YES"' >> /etc/rc.conf
service fxa-auth start
```

  - **NOTE**: If switching the Firefox Account server to production, use the following to start pm2 in the prod environment:

```
echo 'fxa_auth_env="prod"' >> /etc/rc.conf
service fxa-auth restart
```

Firefox Content Server

- Switch to the fxa directory:

```
cd /usr/local/www/fxa
```

- Download the firefox content server from GitHub:

```
sudo -u fxa git clone https://github.com/mozilla/fxa-content-server.git
cd fxa-content-server
```

- Generate a strong secret and copy the contents over to the secret parameter in the syncserver config:

```
head -c 20 /dev/urandom | shasum db8a203aed5fe3e4594d4b75990acb76242efd35 -
```

  NOTE: Make sure to copy the output

- Create content server config file:

```
sudo -u fxa vi server/config/content.json
```

  - And modify the following values:

```
{
  "public_url": "https://accounts.example.com",
  "fxaccount_url": "https://api.accounts.example.com",
  "oauth_client_id": "98e6508e88680e1a",
  "oauth_url": "https://oauth.accounts.example.com",
  "profile_url": "https://profile.accounts.example.com",
  "profile_images_url": "https://image.accounts.example.com",
  "sync_tokenserver_url": "https://sync.accounts.gnetsolutions.net/token",
  "client_sessions": {
    "cookie_name": "session",
    "secret": "8fe72cba641d5c4afbf54127a0fc7bb2cc6618d0",
    "duration": 86400000
  },
  "env": "development",
  "use_https": false,
  "static_max_age" : 0,
  "route_log_format": "dev_fxa",
```

```
  "logging": {
    "fmt": "pretty",
    "level": "debug"
  },
  "static_directory": "app",
  "allowed_parent_origins": ["/"],
  "csp": {
    "enabled": true,
    "reportUri": "/_/csp-violation"
  }
}
```

- **NOTE**: Create a production config from the development copy to work with when going into production, remember to change the env parameter to production as well:

```
cp server/config/content.json server/config/production.json
```

- Install the content server:

```
sudo -u fxa npm install
```

- Build the assets:

```
bower install
grunt build
```

- Test the content server:

```
sudo -u fxa npm run start-remote
```

NOTE: Press Ctrl+C to stop the test server.

- A persistent deployment will require pm2:

```
sudo -u fxa pm2 start npm --name fxa-content -- run start-remote
```

Content Server Init Script

- Create a firefox content server init script:

```
vi /usr/local/etc/rc.d/fxa-content
```

- and add the following

```
#!/bin/sh

# PROVIDE: fxa-content
# KEYWORD: shutdown

. /etc/rc.subr

: ${fxa_content_path="/usr/local/www/fxa/fxa-content-server"}
: ${fxa_content_env="dev"}

name="fxa_content"
start_cmd="${name}_start"
stop_cmd="${name}_stop"
```

```
fxa_content_start() {
   echo "Firefox content server starting"
   su - fxa -c "cd ${fxa_content_path}; /usr/local/bin/pm2 start npm --name ${name} --env
${fxa_content_env} -- run start-remote; exit"
}

fxa_content_stop() {
   echo "Firefox content server stopping"
   su - fxa -c "/usr/local/bin/pm2 delete ${name}; exit"
}

run_rc_command "$1"
```

- And make it executable:

```
chmod +x /usr/local/etc/rc.d/fxa-content
```

- Start and enable firefox content server at boot

```
echo 'fxa_content_enable="YES"' >> /etc/rc.conf
service fxa-content start
```

  - **NOTE**: If switching the Firefox Account server to production, use the following to start pm2 in the prod environment:

```
echo 'fxa_content_env="prod"' >> /etc/rc.conf
service fxa-content restart
```

Install OAuth Server

- Switch to the fxa directory:

```
cd /usr/local/www/fxa
```

- Download the firefox oauth server from GitHub:

```
sudo -u fxa git clone https://github.com/mozilla/fxa-oauth-server.git
cd fxa-oauth-server
```

- Edit the development oauth server config:

```
sudo -u fxa vi config/dev.json
```

  - And modify the following parameters:

```
  "browserid": {
    "issuer": "sync.accounts.example.com",
  ...
  },
  "contentUrl": "http://accounts.example.com/oauth/",
  "db": {
    "driver": "mysql"
  },
  "mysql": {
```

```
    "user": "fxaoauthuser",
    "password": "SuperOauthPassword",
    "database": "fxaoauthdb",
    "host": "localhost",
    "port": "3306"
  },
```

- Install the oauth server:

```
sudo -u fxa npm install
```

- Test the oauth server:

```
sudo -u fxa npm start
```

NOTE: Press Ctrl+C to stop the test server.

OAuth Server Init Script

- Create a firefox oauth server init script:

```
vi /usr/local/etc/rc.d/fxa-oauth
```

  - and add the following

```
#!/bin/sh

# PROVIDE: fxa-oauth
# KEYWORD: shutdown

. /etc/rc.subr

: ${fxa_oauth_path="/usr/local/www/fxa/fxa-oauth-server"}

name="fxa_oauth"
start_cmd="${name}_start"
stop_cmd="${name}_stop"

fxa_oauth_start() {
   echo "Firefox oauth server starting"
   su - fxa -c "cd ${fxa_oauth_path}; /usr/local/bin/pm2 start npm --name ${name} -- start
; exit"
}

fxa_oauth_stop() {
   echo "Firefox oauth server stopping"
   su - fxa -c "/usr/local/bin/pm2 delete ${name}; exit"
}

run_rc_command "$1"
```

- And make it executable:

```
chmod +x /usr/local/etc/rc.d/fxa-oauth
```

- Start and enable firefox oauth server at boot

```
echo 'fxa_oauth_enable="YES"' >> /etc/rc.conf
service fxa-oauth start
```

Install Profile Server

- Switch to the fxa directory:

```
cd /usr/local/www/fxa
```

- Download the firefox profile server from GitHub:

```
sudo -u fxa git clone https://github.com/mozilla/fxa-profile-server.git
cd fxa-profile-server
```

- Edit the development profile server config:

```
sudo -u fxa vi config/dev.json
```

  - And adjust the following values:

```
{
  "authServer": {
    "url": "https://api.accounts.example.com"
  },
  "db": {
    "driver": "mysql"
  },
  "logging": {
    "fmt": "pretty",
    "level": "all",
    "debug": true
  },
  "img": {
    "driver": "local"
  },
  "mysql": {
    "user": "fxaprofileuser",
    "password": "SuperSuperPassword",
    "database": "fxaprofiledb",
    "host": "localhost",
    "port": "3306"
  },
  "oauth": {
    "url": "https://oauth.accounts.example.com/v1"
  },
  "customsUrl": "none",
  "publicUrl": "https://profile.accounts.example.com"
}
```

- Install the profile server:

```
sudo -u fxa npm install
```

- Test the profile server:

```
sudo -u fxa npm start
```

NOTE: Press Ctrl+C to stop the test server.

Profile Server Init Script

- Create a firefox profile server init script:

```
vi /usr/local/etc/rc.d/fxa-profile
```

  - and add the following

```
#!/bin/sh

# PROVIDE: fxa-profile
# KEYWORD: shutdown

. /etc/rc.subr

: ${fxa_profile_path="/usr/local/www/fxa/fxa-profile-server"}

name="fxa_profile"
start_cmd="${name}_start"
stop_cmd="${name}_stop"

fxa_profile_start() {
   echo "Firefox profile server starting"
   su - fxa -c "cd ${fxa_profile_path}; /usr/local/bin/pm2 start npm --name ${name} -- sta
rt; exit"
}

fxa_profile_stop() {
   echo "Firefox profile server stopping"
   su - fxa -c "/usr/local/bin/pm2 delete ${name}; exit"
}

run_rc_command "$1"
```

- And make it executable:

```
chmod +x /usr/local/etc/rc.d/fxa-profile
```

- Start and enable firefox profile server at boot

```
echo 'fxa_profile_enable="YES"' >> /etc/rc.conf
service fxa-profile start
```

Firefox Sync Server

- Switch to the fxa directory:

```
cd /usr/local/www/fxa
```

- Get the latest version of the syncserver:

```
sudo -u fxa git clone https://github.com/mozilla-services/syncserver.git
```

```
cd syncserver
```

- Build the Sync Server:

```
sudo -u fxa gmake build
```

- Generate a strong secret and copy the contents over to the secret parameter in the syncserver config:

```
head -c 20 /dev/urandom | shasum db8a203aed5fe3e4594d4b75990acb76242efd35 -
```

NOTE: Make sure to copy the output

- Edit the syncserver config file:

```
sudo -u fxa vi syncserver.ini
```

  - And modify the following values:

```
[syncserver]
public_url = https://sync.accounts.example.com/
sqluri = pymysql://fxasyncuser:SuperDuperPassword@localhost/fxasyncdb
secret = e48ee2c1a880c31100b5e3217a438f6c2d115b04
[browserid]
backend = tokenserver.verifiers.LocalVerifier
audiences = https://sync.accounts.example.com/
```

- Test run the syncserver:

```
sudo -u fxa gmake serve
```

NOTE: Press Ctrl+C to stop the test server.

Nginx

- Install nginx:

```
pkg install nginx openssl
```

- Generate dhparam file:

```
openssl dhparam -out /usr/local/etc/nginx/dhparam.pem 4096
```

- Start and enable nginx at boot:

```
echo 'nginx_enable="YES"' >> /etc/rc.conf
service nginx start
```

- Create a configuration directory to make managing individual server blocks easier

```
mkdir /usr/local/etc/nginx/conf.d
```

- Edit the main nginx config file:

```
vi /usr/local/etc/nginx/nginx.conf
```

  - And strip down the config file and add the include statement at the end to make it easier to handle various server blocks:

```
load_module /usr/local/libexec/nginx/ngx_mail_module.so;
load_module /usr/local/libexec/nginx/ngx_stream_module.so;

worker_processes  1;
error_log  /var/log/nginx-error.log;

events {
  worker_connections  1024;
}

http {
  include       mime.types;
  default_type  application/octet-stream;
  sendfile        on;
  keepalive_timeout  65;

  include /usr/local/etc/nginx/conf.d/*.conf;
}
```

uWSGI

- Install uwsgi:

```
pkg install uwsgi
```

- Start and enable uwsgi at boot with additional arguments:

```
echo 'uwsgi_enable="YES"' >> /etc/rc.conf
echo 'uwsgi_flags="-M -L --manage-script-name --mount /=/usr/local/www/fxa/syncserver/syncserv
er.wsgi"' >> /etc/rc.conf
service uwsgi start
```

NOTE: Pay attention to the **/=** preceding the actual path of the syncserver.wsgi file.

Syncserver Nginx Config

- Add a **sync.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/sync.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  sync.accounts.example.com;
  access_log   /var/log/sync.accounts.example.com-access.log;
  error_log    /var/log/sync.accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/sync.accounts.example.com/fullchain.pem
;
```

```
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/sync.accounts.example.com/privkey.p
em;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    include uwsgi_params;
    uwsgi_pass unix:/tmp/uwsgi.sock;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

Content Server Nginx Config

- Add a **accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  accounts.example.com;
  access_log   /var/log/accounts.example.com-access.log;
  error_log    /var/log/accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/accounts.example.com/fullchain.pem;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/accounts.example.com/privkey.pem;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
```

```
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:3030/;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

Auth Server Nginx Config

- Add a **api.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/api.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  api.accounts.example.com;
  access_log   /var/log/api.accounts.example.com-access.log;
  error_log    /var/log/api.accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/api.accounts.example.com/fullchain.pem;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/api.accounts.example.com/privkey.pe
m;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:9000/;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

OAuth Server Nginx Config

- Add a **oauth.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/oauth.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
  listen 443 ssl;
#  server_name  oauth.accounts.example.com;
  access_log   /var/log/oauth.accounts.example.com-access.log;
  error_log    /var/log/oauth.accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/oauth.accounts.example.com/fullchain.pe
m;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/oauth.accounts.example.com/privkey.
pem;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:9010/;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

Profile Server Nginx Config

- Add a **profile.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/profile.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  profile.accounts.example.com;
  access_log   /var/log/profile.accounts.example.com-access.log;
  error_log    /var/log/profile.accounts.example.com-error.log;
```

```
#  ssl_certificate /usr/local/etc/letsencrypt/live/profile.accounts.example.com/fullchain.
pem;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/profile.accounts.example.com/privke
y.pem;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:1111/;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

Profile Image Server Nginx Config

- Add a **image.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/image.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  image.accounts.example.com;
  access_log   /var/log/image.accounts.example.com-access.log;
  error_log    /var/log/image.accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/image.accounts.example.com/fullchain.pe
m;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/image.accounts.example.com/privkey.
pem;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
```

```
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:1112/;
  }

  ## Add well-know location and allow connections from the internet
  location ~ /.well-known {
    allow all;
    root         /usr/local/www/nginx;
  }
}
```

Verifier Server Nginx Config

- Add a **verifier.accounts.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/verifier.accounts.example.com.conf
```

  - Add the following:

```
server {
  listen 80;
#  listen 443 ssl;
  server_name  verifier.accounts.example.com;
  access_log   /var/log/verifier.accounts.example.com-access.log;
  error_log    /var/log/verifier.accounts.example.com-error.log;

#  ssl_certificate /usr/local/etc/letsencrypt/live/verifier.accounts.example.com/fullchain
.pem;
#  ssl_certificate_key /usr/local/etc/letsencrypt/live/verifier.accounts.example.com/privk
ey.pem;

  # Configure Strong SSL
#  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
#  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
#  ssl_session_cache  builtin:1000  shared:SSL:10m;
#  ssl_stapling on;
#  ssl_stapling_verify on;
#  ssl_prefer_server_ciphers on;
#  ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
#  add_header Strict-Transport-Security max-age=63072000;
#  add_header X-Frame-Options SAMEORIGIN;
#  add_header X-Content-Type-Options nosniff;

  location / {
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_redirect off;
    proxy_read_timeout 120;
    proxy_connect_timeout 10;
    proxy_pass http://127.0.0.1:1111/;
  }
```

```
    ## Add well-know location and allow connections from the internet
    location ~ /.well-known {
      allow all;
      root          /usr/local/www/nginx;
    }
  }
```

- Restart nginx:

```
service nginx restart
```

LetsEncrypt

- Install letsencrypt:

```
pkg install py27-certbot
```

- Create a directory for letsencrypt site configs:

```
mkdir /usr/local/etc/letsencrypt/config
```

- Create the **content** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/accounts.example.com.conf
```

  - And add the following:

```
domains = accounts.example.com
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **content** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/accounts.example.com.conf
```

- Create the **auth** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/api.accounts.example.com.conf
```

  - And add the following:

```
domains = api.accounts.example.com
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
```

```
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **auth** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/api.accounts.example.com.conf
```

- Create the **oauth** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/oauth.accounts.example.com.conf
```

  - And add the following:

```
domains = oauth.accounts.example.com
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **oauth** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/oauth.accounts.example.com.conf
```

- Create the **profile** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/profile.accounts.example.com.conf
```

  - And add the following:

```
domains = profile.accounts.example.com
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **profile** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/profile.accounts.example.com.conf
```

- Create the **profile image** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/image.accounts.example.com.conf
```

  - And add the following:

```
domains = image.accounts.example.com
```

```
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **profile image** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/image.accounts.example.com.conf
```

- Create the **sync** server letsencrypt config:

```
vi /usr/local/etc/nginx/config/sync.accounts.example.com.conf
```

  - And add the following:

```
domains = sync.accounts.example.com
rsa-key-size = 4096
server = https://acme-v01.api.letsencrypt.org/directory
email = bob@example.com
text = True
agree-tos
authenticator = webroot
webroot-path = /usr/local/www/nginx
```

- Create the **sync** server SSL key and certificate:

```
certbot certonly -c /usr/local/etc/letsencrypt/config/sync.accounts.example.com.conf
```

- Now edit each of the nginx server block configs and **remove** all the commented out SSL parameters.
- Restart nginx:

```
service nginx restart
```

Connect Firefox

In desktop Firefox, enter “about:config” in the URL bar, search for items containing **fxaccounts**, and edit them to use your self-hosted URLs:

Auth Server

- *identity.fxaccounts.auth.uri*: https://api.accounts.example.com/v1

Content Server

- *identity.fxaccounts.autoconfig.uri*: https://accounts.example.com
- *identity.fxaccounts.remote.signin.uri*: https://accounts.example.com/signin?service=sync&context=fx_desktop_v3
- *identity.fxaccounts.remote.signup.uri*: https://accounts.example.com/signup?service=sync&context=fx_desktop_v3
- *identity.fxaccounts.remote.force_auth.uri*: https://accounts.example.com/force_auth?service=sync&context=fx_desktop_v3
- *identity.fxaccounts.settings.uri*: https://accounts.example.com/settings?service=sync&context=fx_desktop_v3
- *identity.fxaccounts.remote.webchannel.uri*: https://accounts.example.com
- *webchannel.allowObject.urlWhitelist*: https://accounts.example.com

OAuth Server

- *identity.fxaccounts.remote.profile.uri*: https://oauth.accounts.example.com/v1

Profile Server

- *identity.fxaccounts.remote.oauth.uri*: https://profile.accounts.example.com/v1

Sync Server

- To configure desktop Firefox to talk to your new Sync server, go to “about:config”, search for **identity.sync.tokenserver.uri** and change its value to the URL of your server:
    1. *identity.sync.tokenserver.uri*: https://sync.accounts.example.com/token/1.0/sync/1.5

Resources

- https://docs.services.mozilla.com/howtos/run-fxa.html
- https://docs.services.mozilla.com/howtos/run-sync-1.5.html
- https://github.com/mozilla/fxa-auth-server/
- https://github.com/mozilla/fxa-content-server/
- https://github.com/mozilla/fxa-oauth-server/
- https://github.com/mozilla/fxa-profile-server/

History

#1 - 01/10/2016 10:25 AM - Daniel Curtis

- Description updated

- Status changed from New to In Progress

- % Done changed from 0 to 10

#2 - 01/10/2016 11:48 AM - Daniel Curtis

- Description updated

- % Done changed from 10 to 30

#3 - 01/10/2016 12:48 PM - Daniel Curtis

- Description updated

- % Done changed from 30 to 70

#4 - 01/10/2016 01:21 PM - Daniel Curtis

- Description updated

#5 - 01/10/2016 03:47 PM - Daniel Curtis

- Description updated

#6 - 01/10/2016 06:43 PM - Daniel Curtis

- Description updated

#7 - 10/02/2016 12:22 PM - Daniel Curtis

- Description updated

#8 - 10/04/2016 02:49 PM - Daniel Curtis

- Description updated

#9 - 10/04/2016 05:12 PM - Daniel Curtis

- Description updated

#10 - 10/04/2016 05:32 PM - Daniel Curtis

- Description updated

- Target version changed from FreeBSD 9 to FreeBSD 10

#11 - 10/04/2016 07:25 PM - Daniel Curtis

- Description updated

#12 - 10/04/2016 09:23 PM - Daniel Curtis

- Description updated

- % Done changed from 70 to 80

#13 - 10/04/2016 11:14 PM - Daniel Curtis

- Description updated

#14 - 10/05/2016 11:45 AM - Daniel Curtis

- Description updated

#15 - 10/05/2016 11:00 PM - Daniel Curtis

- Description updated

#16 - 10/05/2016 11:06 PM - Daniel Curtis

- Description updated

#17 - 10/05/2016 11:29 PM - Daniel Curtis

- Description updated

#18 - 10/06/2016 11:48 PM - Daniel Curtis

- Description updated

- % Done changed from 80 to 90

#19 - 10/10/2016 10:46 PM - Daniel Curtis

- Description updated

#20 - 10/10/2016 11:11 PM - Daniel Curtis

- Description updated

#21 - 10/13/2016 09:03 PM - Daniel Curtis

- Description updated

#22 - 10/13/2016 10:16 PM - Daniel Curtis

- Description updated

#23 - 10/15/2016 03:40 PM - Daniel Curtis

- Description updated

- % Done changed from 90 to 100

#24 - 10/15/2016 03:51 PM - Daniel Curtis

- Description updated

#25 - 10/15/2016 04:35 PM - Daniel Curtis

- Description updated

#26 - 10/15/2016 09:23 PM - Daniel Curtis

- Description updated

#27 - 06/04/2017 09:46 PM - Daniel Curtis

- Status changed from In Progress to Suspended