NGINX Load Balancer

This section describes the configurations required at WRTC gateway for load balancer and also illustrates a sample NGINX load balancer configuration for reference.

Figure 1: NGINX Deployment in Cloud

In cloud deployment, NGINX load balancer is configured with domain name of WRTC server, as upstream configuration. DNS server in cloud is updated with mapping of several WRTC server instances to domain name, and resolves the DNS entry by round-robin algorithm to support load sharing.

WRTC Load Balancer Configuration

Perform the following steps to edit `startup.json` file to support WRTC load balancer configuration:

1. Login as wrtc user to perform the configuration.
2. Navigate to the following location and open the `startup.json` file in `vi` editor:

   ```
   cd /opt/sonus/wrtc
   ```

3. Perform the following steps to configure WRTC load balancer to share traffic to WRTC upstream servers:
   a. Set the `IsSSLTerminationOn` field to true if you want to configure load balancer to offload SSL from all application nodes. By default, this field is set to false.
   b. If `IsSSLTerminationOn` is set to true, enter the termination port in the `SSLTerminationPort` field. By default, port value is set to 8088, you can enter any port number other than port 443.
4. Set `IsEphemeralOn` field to true, if the deployed TURN server supports ephemeral user ID generation like Google TURN server. By default, this field is set to false.

Enterprise, Enterprise URLs, and all callback URLs of OAuth providers, must carry domain name or IP of load balancer, you can update the same in Cluster Configuration tables or `startup.json` if running without EMS.

NGINX Configuration to Share Traffic to WRTC Upstream Server

The following is the configuration to share traffic to WRTC upstream servers, if using NGINX load balancer.
nginx.conf

#user nobody;
worker_processes 1;
#error_log logs/error.log;
#error_log logs/error.log notice;
#error_log logs/error.log info;
#pid logs/nginx.pid;
events {
  worker_connections 1024;
}
http {
  include mime.types;
  default_type application/octet-stream;
  fastcgi_buffers 8 16k;
  fastcgi_buffer_size 32k;
  upstream ws_rack{
    # ip_hash;
    server 10.54.164.233:9080;
    server 10.54.164.234:9080;
  }
  upstream http_rack_443{
    server 10.54.164.233:443;
    server 10.54.164.234:443;
  }
  upstream http_rack_80{
    server 10.54.164.233:80;
    server 10.54.164.234:80;
  }
  upstream http_rack_8081{
    server 10.54.164.233:8081;
    server 10.54.164.234:8081;
  }
  server {
    listen 9080 ssl;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
    ssl on;
    ssl_certificate /opt/nginx/ssl/server.crt;
    ssl_certificate_key /opt/nginx/ssl/server.key;
    ssl_session_timeout 5m;
    ssl_protocols SSLv2 SSLv3 TLSv1;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    #ssl_verify_client off;
    location / {
      #root html;
      # index index.html index.htm;
      proxy_pass http://ws_rack;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header Host $host;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      # WebSocket support
      proxy_http_version 1.1;
      proxy_set_header Upgrade $http_upgrade;
      proxy_set_header Connection "upgrade";
      proxy_set_header Host $host;
      proxy_read_timeout 86400;
      proxy_redirect off;
      proxy_buffers 8 32k;
      proxy_buffer-size 64k;
      # proxy.ssl_session_reuse off;
      # proxy_set_header X_FORWARDED_PROTO https;
    }
  }
  listen 443 ssl;
  server_name 10.160.139.26;
  large_client_header_buffers 8 32k;
  ssl on;
ssl_certificate /opt/nginx/ssl/server.crt;
ssl_certificate_key /opt/nginx/ssl/server.key;
ssl_session_timeout 5m;
ssl_protocols SSLv2 SSLv3 TLSv1;
ssl_ciphers HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers on;
#ssl_verify_client off;
tcp_nopush on;
location / {
    #root html;
    index index.html index.htm;
    proxy_pass http://http_rack_443/;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_buffers 8 16k;
    proxy_buffer_size 32k;
    fastcgi_buffers 8 16k;
    fastcgi_buffer_size 32k;
    proxy_read_timeout 300;
    proxy_connect_timeout 300;
    tcp_nopush on;
}
}
server {
    listen 80;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
    #ssl_verify_client off;
tcp_nopush on;
location / {
    #root html;
    index index.html index.htm;
    proxy_pass http://http_rack_80/;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_buffers 8 16k;
    proxy_buffer_size 32k;
    fastcgi_buffers 8 16k;
    fastcgi_buffer_size 32k;
    proxy_read_timeout 300;
    proxy_connect_timeout 300;
    tcp_nopush on;
}
}
server {
    listen 8081 ssl;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
    ssl on;
    ssl_certificate /opt/nginx/ssl/server.crt;
    ssl_certificate_key /opt/nginx/ssl/server.key;
    ssl_session_timeout 5m;
    ssl_protocols SSLv2 SSLv3 TLSv1;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    #ssl_verify_client off;
tcp_nopush on;
location / {
    #root html;
    index index.html index.htm;
    proxy_pass http://http_rack_8081/;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_buffers 8 16k;
    proxy_buffer_size 32k;
    fastcgi_buffers 8 16k;
    fastcgi_buffer_size 32k;
proxy_read_timeout 300;
proxy_connect_timeout 300;
tcp_nopush on;
Perform the following tasks to configure NGINX to share traffic to WRTC upstream servers:

1. Copy the preceding nginx.conf file in to `<nginx_install>/conf directory.
2. Update the server IPs in both http and websocket racks of WRTC nodes.
3. Update `server_name` in each lister server configuration to eth0 IP of NGINX node.
4. NGINX acts as SSL termination endpoint to WRTC browser endpoint, so you need to configure certificate and key in each http and websocket endpoint. To generate self-signed certificates using open SSL libraries, refer to [How To Set Up Nginx Load Balancing with SSL Termination](#).

⚠️ Upstream REST port value must be changed to match with the "SSLTerminationPort" value in `startup.json`, instead of the default port 443.

**NGINX Configuration for End to End Encryption**

The following is the configuration for enabling end to end encryption, if using NGINX load balancer. In this case, user communicate with NGINX over HTTPS. NGINX decrypts the requests when received, and then encrypts the request before sending to the backend servers (upstream servers).

```
# nginx.conf

#user nobody;
worker_processes 1;
#error_log logs/error.log;
#error_log logs/error.log notice;
#error_log logs/error.log info;
#pid logs/nginx.pid;
events {
  worker_connections 1024;
}
http {
  include mime.types;
default_type application/octet-stream;
fastcgi_buffers 8 16k;
fastcgi_buffer_size 32k;
#log_format upstreamlog '
    [$time_local] $remote_addr - $remote_user - $server_name to: $upstream_addr:
    $request upstream_response_time $upstream_response_time msec $msec request_time $request_time' '
    $upstream_addr ';
  upstream ws_rack{
    #ip_hash;
    server 10.160.139.24:9080;
    server 10.160.139.27:9080;
  }
  upstream http_rack_443{
    server 10.160.139.24:443;
    server 10.160.139.27:443;
  }
  upstream http_rack_80{
    server 10.160.139.24:80;
    server 10.160.139.27:80;
  }
  upstream http_rack_8081{
    server 10.160.139.24:8081;
    server 10.160.139.27:8081;
  }
  server {
    listen 9080 ssl;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
  }
```
ssl on;
ssl_certificate /opt/nginx/ssl/server.crt;
ssl_certificate_key /opt/nginx/ssl/server.key;
ssl_session_timeout 5m;
ssl_protocols SSLv2 SSLv3 TLSv1;
ssl_ciphers HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers on;
#ssl_verify_client off;
location / {
    #root html;
    #index index.html index.htm;
    proxy_pass https://ws_rack;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    # WebSocket support
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_read_timeout 86400;
    proxy_redirect off;
    proxy_buffers 8 32k;
    proxy_buffer_size 64k;
    #proxy_ssl_session_reuse off;
    #proxy_set_header X_FORWARDED_PROTO https;
    }
}
server {
    listen 443 ssl;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
    ssl on;
    ssl_certificate /opt/nginx/ssl/server.crt;
    ssl_certificate_key /opt/nginx/ssl/server.key;
    ssl_session_timeout 5m;
    ssl_protocols SSLv2 SSLv3 TLSv1;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;
    #ssl_verify_client off;
    tcp_nopush on;
    location / {
        #root html;
        #index index.html index.htm;
        proxy_pass https://http_rack_443/;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_buffers 8 16k;
        proxy_buffer_size 32k;
        fastcgi_buffers 8 16k;
        fastcgi_buffers_size 32k;
        proxy_read_timeout 300;
        proxy_connect_timeout 300;
        tcp_nopush on;
    }
}
server {
    listen 80;
    server_name 10.160.139.26;
    large_client_header_buffers 8 32k;
    #ssl_verify_client off;
    tcp_nopush on;
    location / {
        #root html;
        index index.html index.htm;
        proxy_pass http://http_rack_80/;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header Host $host;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_buffers 8 16k;
proxy_buffer_size 32k;
favicon_buffers 8 16k;
favicon_buffer_size 32k;
proxy_read_timeout 300;
proxy_connect_timeout 300;
tcp_nopush on;
}
}
server {
listen 8081 ssl;
server_name 10.160.139.26;
large_client_header_buffers 8 32k;
#access_log /opt/nginx/access1.log upstreamlog;
ssl on;
ssl_certificate /opt/nginx/ssl/server.crt;
ssl_certificate_key /opt/nginx/ssl/server.key;
ssl_session_timeout 5m;
ssl_protocols SSLv2 SSLv3 TLSv1;
proxy_ssl_protocols SSLv3;
ssl_ciphers HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers on;
#ssl_verify_client off;
tcp_nopush on;
location / {
#root html;
index index.html index.htm;
proxy_pass https://http_rack_8081/;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header Host $host;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_buffers 8 16k;
proxy_buffer_size 32k;
favicon_buffers 8 16k;
favicon_buffer_size 32k;
proxy_read_timeout 300;
proxy_connect_timeout 300;
tcp_nopush on;
For information on installing NGINX, refer to http://wiki.nginx.org/Main.