Websocket traffic proxying for WebRTC publishing/playing

Sometimes, it is necessary to hide Websocket WCS port behind proxy server, for example, for security reasons. A reverse proxy setup based on nginx and correcponding WCS setup examples are described below.

Reverse proxy setup with basic authentication for Websocket 1. Enable basic authentication by login and password in nginx settings

```
auth_basic "Restricted Area";
auth_basic_user_file /etc/nginx/.htpasswd;
```

2. Configure nginx to listen HTTPS (WebRTC publishing and playback work only through secure connection in the most browsers)

```
server {
  listen 443 ssl;
  ssl_certificate /etc/pki/tls/yourdomain/yourdomain.crt;
  ssl_certificate_key /etc/pki/tls/yourdomain/yourdomain.key;
  server_name wcs.yourdomain.com;
  server_tokens off;
  client_max_body_size 500m;
  proxy_read_timeout 10m;

  root         /usr/share/nginx/html;
  ...
}
```

3. Configure proxy to WCS Websocket port (suppose nginx to be installed on the same server as WCS)

```
location /wss {
   proxy_set_header Host $host;
   proxy_pass https://localhost:8443;
   proxy_http_version 1.1;
   proxy_set_header Upgrade $http_upgrade;
   proxy_set_header Connection "upgrade";
   proxy_read_timeout 86400;
}
```

4. Restart nginx 5. Use the following Websocket URL to connect from browser

```
wss://login:password@wcs.yourdomain.com:443/wss
```

??? example "Full nginx configuration file"

```
http {
   log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                     '$status $body_bytes_sent "$http_referer" '
                     '"$http_user_agent" "$http_x_forwarded_for"';

   access_log  /var/log/nginx/access.log  main;

   sendfile            on;
   tcp_nopush          on;
   tcp_nodelay         on;
   keepalive_timeout   65;
   types_hash_max_size 2048;

   include             /etc/nginx/mime.types;
   default_type        application/octet-stream;

   auth_basic "Restricted Area";
   auth_basic_user_file /etc/nginx/.htpasswd;

   include /etc/nginx/conf.d/*.conf;

   server {
       listen 443 ssl;
       ssl_certificate /etc/pki/tls/yourdomain/yourdomain.crt;
       ssl_certificate_key /etc/pki/tls/yourdomain/yourdomain.key;
       server_name wcs.yourdomain.com;
       server_tokens off;
       client_max_body_size 500m;
       proxy_read_timeout 10m;

       include /etc/nginx/default.d/*.conf;

       location / {
       }

       location /wss {
           proxy_set_header Host $host;
           proxy_pass https://localhost:8443;
           proxy_http_version 1.1;
           proxy_set_header Upgrade $http_upgrade;
           proxy_set_header Connection "upgrade";
           proxy_read_timeout 86400;
       }

       error_page 404 /404.html;
           location = /40x.html {
       }

       error_page 500 502 503 504 /50x.html;
           location = /50x.html {
       }

   }
}
```

Reverse proxy setup with passing authentication token in cookies Authentication parameters passing in URL is deprecated. However, browsers still does not support a ways to pass a custom headers (including Authorization header) when establishing Websocket connection. In this case, passing authentication token in cookies with cookie checking on nginx side may be alternative.
Client code A client should set a cookie with authentication token before establishing websocket connection:

```
setCookie("AUTH", token, {secure: true, 'max-age': 3600});
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED,
function (session) {
  ...
});
```

A sample code to set or change cookies in browser

```
function setCookie(name, value, options = {}) {
  options = {
    path: '/',
    ...options
  };

  if (options.expires instanceof Date) {
    options.expires = options.expires.toUTCString();
  }

  let updatedCookie = encodeURIComponent(name) + "=" +
encodeURIComponent(value);

  for (let optionKey in options) {
    updatedCookie += "; " + optionKey;
    let optionValue = options[optionKey];
    if (optionValue !== true) {
      updatedCookie += "=" + optionValue;
    }
  }

  document.cookie = updatedCookie;
}
```

Cookie may be cleaned when websocket session is closed or failed

```
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED,
function (session) {
  ...
}).on(SESSION_STATUS.DISCONNECTED, function () {
  setCookie("AUTH", "", {'max-age': -1});
  ...
}).on(SESSION_STATUS.FAILED, function () {
  setCookie("AUTH", "", {'max-age': -1});
  ...
});
```

nginx configuration 1. Create a folder to store authentication tokens

```
mkdir -p /var/lib/nginx/tokens
```

and set nginx running user as owner

```
chown -R nginx /var/lib/nginx/token
```

2. Add token checking to nginx configuration file

```
location /wss {
   if (!-f /var/lib/nginx/tokens/$cookie_AUTH) {
      return 403;
   }
   proxy_set_header Host $host;
   proxy_pass https://localhost:8443;
   proxy_http_version 1.1;
   proxy_set_header Upgrade $http_upgrade;
   proxy_set_header Connection "upgrade";
   proxy_read_timeout 86400;
}
```

3. Restart nginx
Usage Before connecting some client, authentication token file should be created

```
touch /var/lib/nginx/tokens/ABCDEF1234565789
chown nginx /var/lib/nginx/tokens/ABCDEF1234565789
```

and the file name should be passed to the client to set it to cookie. A possible way to pass the token are out of the scope.
Known issues For better security, `Origin` header must be checked, and cookie should be applied from allowed domains only.
How to pass a real client IP address to WCS through the reverse proxy If proxy is configured as described above, all the client sessions at WCS point will have IP address 127.0.0.1. This breaks debugging the stream publishing and playing problems because real connection source can not be identified, therefore, this does not allow to [collect debug logs](../Working_with_the_server/CLI_v_2/Connections_management.en.md) by client IP address. To workaround this issue, the parameter is added since build [5.2.743](https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-5.2.743.tar.gz) which allows to pass real client IP address to WCS via HTTP header

```
ws.map_custom_headers=true
ws.ip_forward_header=X-Real-IP
```

By default, proxy server should pass a real client IP address in `X-Real-IP` header. Let's explore nginx and WCS configuration example to pass real client IP address.
nginx configuration 1. Add `X-Client-IP` header creation to Websocket proxy setup

```
location /wss {
   proxy_set_header Host $host;
   proxy_set_header X-Client-IP $remote_addr:$remote_port;
   proxy_pass https://localhost:8443;
   proxy_http_version 1.1;
   proxy_set_header Upgrade $http_upgrade;
   proxy_set_header Connection "upgrade";
   proxy_read_timeout 86400;
}
```

2. Restart nginx
WCS configuration 1. Add the following parameters to [flashphoner.properties] (../Working_with_the_server/Core_settings/Settings_file_flashphoner.properties.en.md) file

```
ws.map_custom_headers=true
ws.ip_forward_header=X-Client-IP
```

2. Restart WCS Then, `sessionId` on WCS side will contain a real client IP address. Also, the header added by proxy server will be passed to backend server in REST hook [/connect] (../REST_Hooks/Four_types_of_REST_methods/Type_1_-_the_connect_method.en.md):

```
{
  "nodeId" : "nziJYH0eu3D08Iu25sXbmwaCgSUuQyGL@192.168.130.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.23.83:65520/127.0.0.1:8443-8ef8fa79-a726-44d3-a20a-
fe27b94bc51f",
  "useWsTunnel" : false,
  "useWsTunnelPacketization2" : false,
  "msePacketizationVersion" : 2,
  "useBase64BinaryEncoding" : false,
  "mediaProviders" : [ "WebRTC", "MSE", "WSPlayer" ],
  "clientVersion" : "0.5.28",
  "clientOSVersion" : "5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36",
  "clientBrowserVersion" : "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36",
  "keepAlive" : false,
  "origin" : "https://wcs.yourdomain.com",
  "X-Client-IP" : "192.168.23.83:65520"
}
```