

Firewall Streaming

- [Example of TURN server usage](#)
- [The code of the example](#)
- [Analyzing the code](#)

Example of TURN server usage

The example of firewall traversal video streaming and playback using TURN server. It works in all browsers except Microsoft Legacy Edge because this browser do not support TURN over TCP.

Firewall Traversal Streaming

Local



test Stop

Player



test Stop

PUBLISHING

WCS Server

Turn Server

Username of turn server

Credential of turn server

Force relay ☒

ESTABLISHED

PLAYING

The code of the example

The source code of the example is on WCS server by this path:

`/usr/local/FlashphonerWebCallServer/client2/examples/demo/streaming/firewall-traversal-streaming/`

firewall-traversal-streaming.css - CSS style file

firewall-traversal-streaming.html - the example page

firewall-traversal-streaming.js - script for example to work

The example may be tested at this URL:

<https://host:8888/client2/examples/demo/streaming/firewall-traversal-streaming/firewall-traversal-streaming.html>

where host is your WCS server address.

Analyzing the code

To analyze the code get firewall-traversal-streaming.js file version with hash `ecbadc3` that can be found [here](#) and is available to download in build `2.0.212`.

1. API initializing.

Flashphoner.init() [code](#)

```
Flashphoner.init();
```

2. Connection to the server.

Flashphoner.createSession() [code](#)

These parameters are passed to createSession() method:

- WCS server URL
- TURN server URL
- TURN server user credentials

```
var options = {
  urlServer: url,
  mediaOptions: {
    "iceServers": [
      {
        'url': $('#urlTurnServer').val(),
        'username': $('#usernameTurnServer').val(),
        'credential': $('#credentialTurnServer').val()
      }
    ]
  }
};
if ($("#forceRelay").is(':checked')) {
  options.mediaOptions.iceTransportPolicy = "relay";
}
Flashphoner.createSession(options).on(SESSION_STATUS.ESTABLISHED, function (session) {
  ...
});
```

3. Receiving the event confirming successful connection

ConnectionStatusEvent ESTABLISHED [code](#)

```
Flashphoner.createSession(options).on(SESSION_STATUS.ESTABLISHED, function (session) {
  setStatus("#connectStatus", session.status());
  onConnected(session);
}).on(SESSION_STATUS.DISCONNECTED, function () {
  ...
}).on(SESSION_STATUS.FAILED, function () {
  ...
});
```

4. Video streaming

session.createStream(), stream.publish() [code](#)

These parameters are passed to createStream() method:

- streamName - the stream name
- localVideo - <div> element to display preview stream.

```

session.createStream({
  name: streamName,
  display: localVideo,
  cacheLocalResources: true,
  receiveVideo: false,
  receiveAudio: false
  ...
}).publish();

```

5.Receiving the event confirming successful streaming

StreamStatusEvent PUBLISHING [code](#)

```

session.createStream({
  ...
}).on(STREAM_STATUS.PUBLISHING, function (stream) {
  setStatus("#publishStatus", STREAM_STATUS.PUBLISHING);
  onPublishing(stream);
}).on(STREAM_STATUS.UNPUBLISHED, function () {
  ...
}).on(STREAM_STATUS.FAILED, function () {
  ...
}).publish();

```

6. Stream playback

session.createStream(), stream.play() [code](#)

These parameters are passed to createStream() method:

- streamName - the stream name (including the stream published on step above)
- remoteVideo - <div> element to display stream playback.

```

session.createStream({
  name: streamName,
  display: remoteVideo
  ...
}).play();

```

7.Receiving the event confirming successful stream playback

StreamStatusEvent PLAYING [code](#)

```

session.createStream({
  ...
}).on(STREAM_STATUS.PLAYING, function (stream) {
  document.getElementById(stream.id()).addEventListener('resize', function (event) {
    resizeVideo(event.target);
  });
  setStatus("#playStatus", stream.status());
  onPlaying(stream);
}).on(STREAM_STATUS.STOPPED, function () {
  ...
}).on(STREAM_STATUS.FAILED, function () {
  ...
}).play();

```

8. Stream playback stop

stream.stop() [code](#)

```
function onPlaying(stream) {
    $("#playBtn").text("Stop").off('click').click(function () {
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
}
```

9.Receiving the event confirming successful playback stop

StreamStatusEvent STOPPED [code](#)

```
session.createStream({
    name: streamName,
    display: remoteVideo
}).on(STREAM_STATUS.PLAYING, function (stream) {
    ...
}).on(STREAM_STATUS.STOPPED, function () {
    setStatus("#playStatus", STREAM_STATUS.STOPPED);
    onStopped();
}).on(STREAM_STATUS.FAILED, function () {
    ...
}).play();
```

10. Streaming stop

stream.stop() [code](#)

```
function onPublishing(stream) {
    $("#publishBtn").text("Stop").off('click').click(function () {
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
}
```

11.Receiving the event confirming successful streaming stop

StreamStatusEvent UNPUBLISHED [code](#)

```
session.createStream({
    ...
}).on(STREAM_STATUS.PUBLISHING, function (stream) {
    ...
}).on(STREAM_STATUS.UNPUBLISHED, function () {
    setStatus("#publishStatus", STREAM_STATUS.UNPUBLISHED);
    onUnpublished();
}).on(STREAM_STATUS.FAILED, function () {
    ...
}).publish();
```