PRESALES DEC 2017

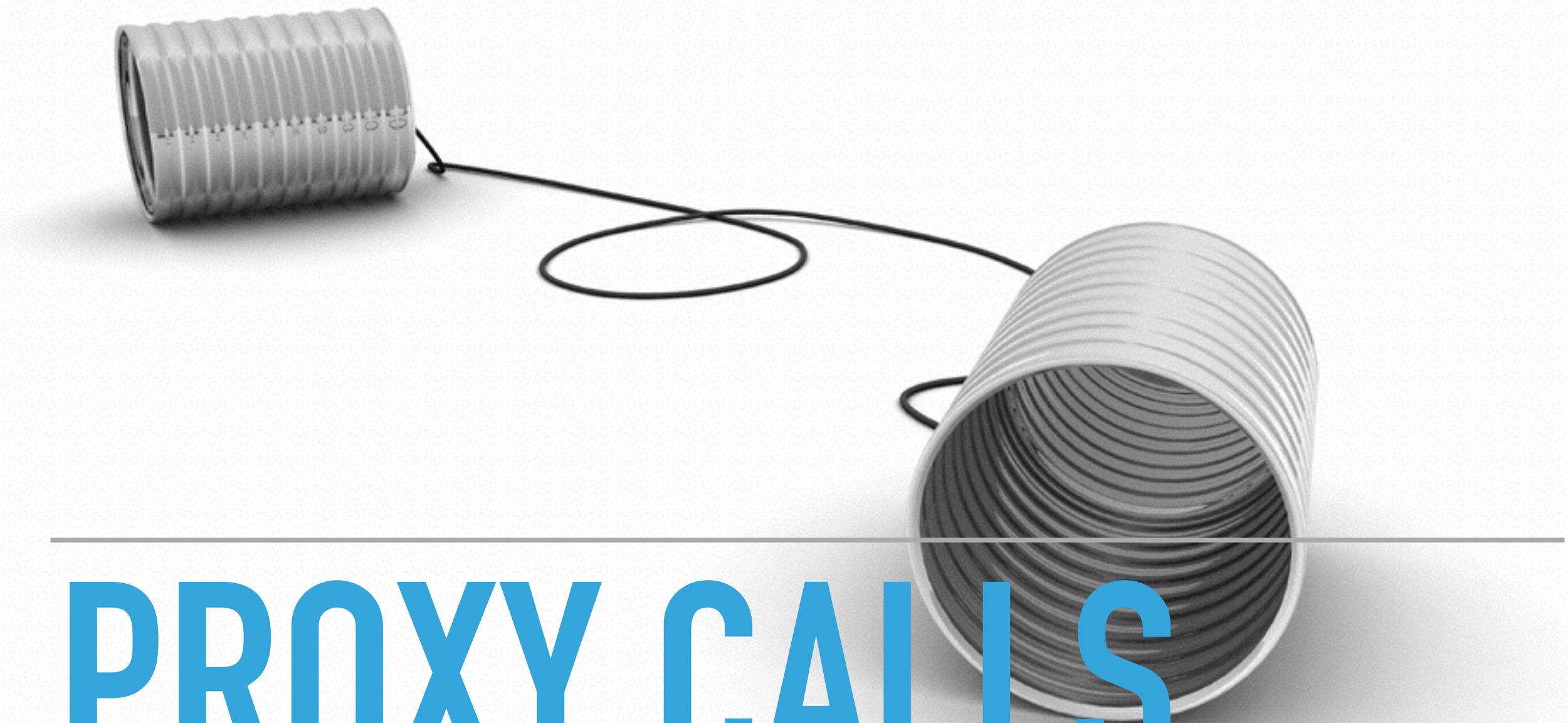
VOICE WORKSHOP

SAM MACHIN

DEVELOPER ADVOCATE

SAM.MACHIN@NEXMO.COM

@SAMMACHIN



PROXY CALLS

VAPI IS PRIMARILY ANN APP TO PERSON OR PERSON TO APP PLATFORM.

- Therefore in order to make person to person calls we actually combine P2A with an A2P call flows
- Depending on the use case CLI is an important consideration.
- Bear in mind that in most scenarios at least one party will be assuming its just a regular P2P call so certain conventions need to be observed.
- In P2P calls, latency is also a consideration.

CONNECT

- The simplest form of a proxy call is to return a CONNECT action to an incoming call

```
[
    {
        "action": "connect",
        "endpoint": [
            {
                "number": "447973994474",
                "type": "phone"
            }
        ]
    }
]
```

PHP CONNECT APP

- Edit the index.php file in PHP_Connect so that the call is connected to your mobile number, also edit the appid param to be your application uuid

- Open that folder in your terminal and then run

```
composer install

php -S localhost:8000 index.php
```

- Get the person next to you to call your nexmo number

TASK ONE

- Edit the application so that your Nexmo number is used as the CLI for the outgoing call

https://developer.nexmo.com/api/voice/ncco#connect

```
$ncco = array([
        'action'=>  'connect',
        'from' => ‘447520616161',
        'endpoint'=> array([
            'type' => 'phone',
            'number' => '447973994474'
            ])
        ]);
```

TASK TWO

- Edit the application so that the caller hears a message before the call is connected

https://developer.nexmo.com/api/voice/ncco#talk

```
$ncco = array([
       'action' => 'talk',
       'text' => 'Your call is being connected, please wait'],
       ['action'=>  'connect',
       'from' => '447520616161',
       'endpoint'=> array([
           'type' => 'phone',
           'number' => '447973994474'
           ])
       ]);
```

TASK THREE

- Edit the application so that the call is recorded and the recording event is sent to your /event endpoint

https://developer.nexmo.com/api/voice/ncco#record

```
$ncco = array([
    'action' => 'record',
    'eventUrl'=> ['https://sammachin.ngrok.io/event']],
    ['action' => 'talk',
    'text' => 'Your call is being connected, please wait'],
    ['action'=>  'connect',
    'from' => '447520616161',
    'endpoint'=> array([
        'type' => 'phone',
        'number' => '447973994474'
        ])
    ]);
```

TASK FOUR

- Edit the application to fetch the recording and save it to a /recordings folder

https://developer.nexmo.com/voice/voice-api/guides/record-calls-and-conversations/

```
$event = $request->getParsedBody();
    if (array_key_exists('recording_url', $event)) {
        $keypair = new \Nexmo\Client\Credentials\Keypair($app->privatekey, $app->appid);
        $client = new \Nexmo\Client($keypair);
        $data = $client->get($event['recording_url']);
        $file = fopen("recordings/".$event['recording_uuid'].".mp3","w");
        echo fwrite($file, $data->getBody());
        fclose($file);
    }
    error_log("EVENT: ".json_encode($event));
```

TASK FIVE

- Create a new /call url to make an outbound call to a number passed in the url and connect that to your number.

https://developer.nexmo.com/api/voice#create-an-outbound-call

https://github.com/nexmo/nexmo-php#usage

```
$app->get('/call', function (Request $request, Response $response)  use ($app){

    $keypair = new \Nexmo\Client\Credentials\Keypair($app->privatekey, $app->appid);
    $client = new \Nexmo\Client($keypair);
    $call = new Call();
    $call->setTo($request->getQueryParam('to', $default = null))
         ->setFrom('447520616161')
         ->setWebhook(Call::WEBHOOK_ANSWER, 'https://sammachin.ngrok.io/ncco')
         ->setWebhook(Call::WEBHOOK_EVENT, 'https://sammachin.ngrok.io/event');
    $client->calls()->create($call);
    return $response->write("ok");
});
```

WEB SOCKETS

THE WEB SOCKET FEATURE ALLOWS YOU TO CONNECT A CALL AUDIO TO A WEB SERVER

- This means that you can access the raw audio stream of the call within your web framework in realtime
- This in turn allows you to connect to AI web services such as transcription, bots or sentiment analysis
- The web socket is just another endpoint like a phone or sip
- If you have a call with multiple participants the web socket will be a mixed stream of the conversation

FORMAT OF WEB SOCKET AUDIO

- Nexmo is always the websocket client
- First message contains JSON with audio format and extra headers
- Subsequent messages are binary
- LPCM Audio 16bit, 16Khz, 20ms,
- Each Message is 320 samples - 640bytes
- Audio can be written to the socket in the same format

CONNECT TO A WEB SOCKET

- Within the NCCO we need to connect the call to the web socket endpoint

```
[
    {
        "action": "connect",
        "endpoint": [
            {
                "uri": “http://sammachin.ngrok.io/socket“,
                "type": "websocket"
            }
        ]
    }
]
```

NODE WEBSOCKET APP

- Edit the ncco.json file in Node_Websocket so that the call is connected to your ngrok user
- Open that folder in your terminal and then run

```
nam install

node ./index.js
```

- Call your Nexmo Number

TASK ONE

- Write the received binary messages back to the web socket to create an echo server

https://www.npmjs.com/package/express-ws

```
ws.on('message', function(msg) {
     if (isBuffer(msg)) {
         ws.send(msg);
        }
        else {
         console.log(msg);
        }
 });
```

TASK TWO

- Write the first 10 sec of received audio to a wav file

https://www.npmjs.com/package/waveheader

```
app.ws('/socket', function(ws, req) {
  console.log("Websocket Connected")
  file = fs.createWriteStream('./output.wav');
  file.write(header(16000 * 30 * 2,{
                    sampleRate: 16000,
                    channels: 1,
                    bitDepth: 16}));
  ws.on('message', function(msg) {
     if (isBuffer(msg)) {
             file.write(msg);
     }
     else {
         console.log(msg);
     }
  });
});
```

TASK THREE

- Downsample the audio to 8kHZ and write to a wav file

http://bit.ly/nodedownsample

```
ws.on('message', function(msg) {
    if (isBuffer(msg)) {
            file.write(convert(msg));
    }
    else {
        console.log(msg);
    }
 });
```

TASK FOUR

- Play a tone when the web socket is connected

https://www.npmjs.com/package/tonegenerator

```
ws.on('message', function(msg) {
    if (isBuffer(msg)) {
            file.write(convert(msg));
    }
    else {
        console.log(msg);
        var tonedata = tone(440, 1, volume = tone.MAX_16, sampleRate = 16000)
        var i,j,sample,chunk = 640;
        for (i=0,j=tonedata.length; i<j; i+=chunk) {
            sample = tonedata.slice(i,i+chunk);
            ws.send(sample);
        }
    }
  });
```

TASK FIVE

- Play the received audio out of the local speaker

https://www.npmjs.com/package/speaker

```
app.ws('/socket', function(ws, req) {
    console.log("Websocket Connected");
    speaker = new Speaker({
                  channels: 1,
                  bitDepth: 16,
                  sampleRate: 16000});
    file = fs.createWriteStream('./output.wav');
    file.write(header(8000 * 30 * 2,{
                  sampleRate: 8000,
                  channels: 1,
                  bitDepth: 16}));
```

TASK FIVE

```
    ws.on('message', function(msg) {
     if (isBuffer(msg)) {
            file.write(convert(msg));
            speaker.write(msg);
     }
     else {
         console.log(msg);
         var tonedata = tone(440, 1, volume = tone.MAX_16, sampleRate = 16000)
         var i,j,sample,chunk = 640;
         for (i=0,j=tonedata.length; i<j; i+=chunk) {
             sample = tonedata.slice(i,i+chunk);
             ws.send(sample);
         }
      }
    });
    ws.on('close', function(ws){
      console.log("Websocket Closed");
      speaker.end();
  })
});
```

THANK YOU

QUESTIONS?

@SAMMACHIN