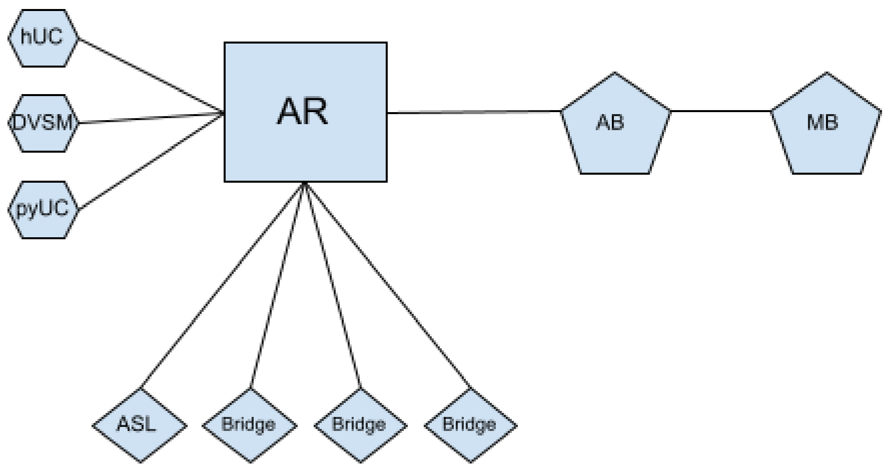
The DVSwitch Analog_Reflector and DVSwitch HTML Client (hUC)

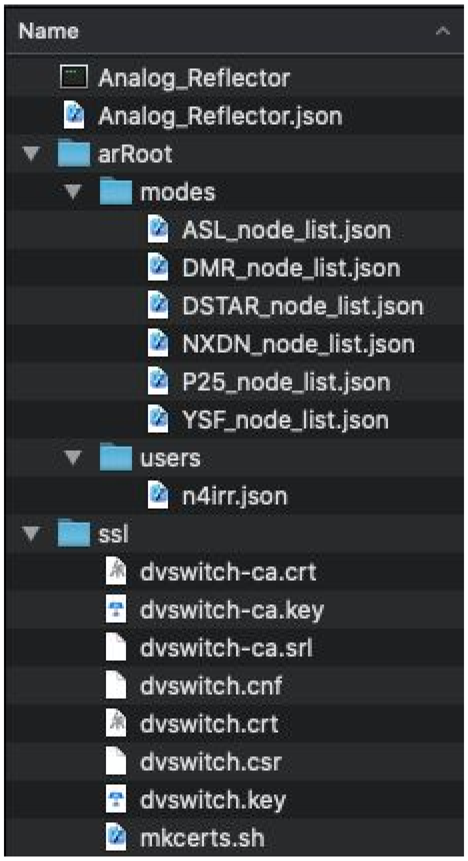
Document Version 0.7

- Introduction
  - Welcome to the DVSwitch Analog_Reflector and HTML client. This new component in the DVSwitch suite of applications is a significant step forward as it adds new ways to combine existing components and adds a new client to the suite. The app serves multiple use cases being a reflector (or room if you are a YSF fan), a proxy of the hUC web application and a bridge which preserves metadata between dissimilar networks. It supports multiple clients at the same time and allows you to share your DVSwitch Server with other hams. If you do allow others to access your DVSwitch Server, you must understand this is a single resource. All users can receive, transmit and tune a SINGLE reflector. This is not multi-user where each user gets their own instance.
  - Features
    - Analog_Reflector is the server component for the hUC HTML client.
    - A Reflector for hUC and DVSM/pyUC clients
    - A Bridge for Analog_Bridge and AllStar sources



  - There is a lot here. So, sit back and take a deep breath. Start with a minimal install and grow it over time to add connection types and capabilities. We see two main types of users of the app, those who are looking for a way of accessing a digital/analog network from the web (or mobile), and those who are looking to add a more versatile bridge to their toolbox. Of course, there will be those of you who want both of these as well.
- Analog_Reflector (the server)
  - Installation
    - What could be easier? apt-get install Analog_Reflector
  - Directory structure

- The Analog_Reflector application is located in /opt/Analog_Reflector. The executable and configuration file Analog_Reflector.json are located in that directory and other support files are located below.



```
Name
    Analog_Reflector
    Analog_Reflector.json
  ▼ arRoot
    ▼ modes
        ASL_node_list.json
        DMR_node_list.json
        DSTAR_node_list.json
        NXDN_node_list.json
        P25_node_list.json
        YSF_node_list.json
    ▼ users
        n4irr.json
  ▼ ssl
      dvswitch-ca.crt
      dvswitch-ca.key
      dvswitch-ca.srl
      dvswitch.cnf
      dvswitch.crt
      dvswitch.csr
      dvswitch.key
      mkcerts.sh
```

- *arRoot* contains two subdirectories; *modes* and *users*. The *modes* directory contains one file per mode. Each file contains a list of display names and talk groups. Edit this file to add your own favorite places. The *users* directory contains one file for each registered user. When a user logs into the reflector values from their user.json will be used to authenticate and grant rights as needed. To add a new user you should use the Analog_Reflector "user" command (see below).
- The *ssl* directory contains the files needed to create a secure connection between the server and hUC clients. The script *mkcerts.sh* in the ssl directory is used to create new self signed certificates (see below).

- Configuration
  - Analog_Reflector uses a json file to configure the operation of the reflector. The json file (*Analog_Reflector.json*) contains all of the information needed to connect to an instance of Analog_Bridge and any other bridges that you may want.
  - *Analog_Reflector.json*

- *nodeName* - A human readable name for your reflector. This name is used on the Manage screen of the hUC client as well as part of the mqtt topic "scheme". MQTT is discussed here.
- *dmrID* - The default DMR ID used if no other metadata can be found
- *abAddress* - IP Address or FQDN of the Analog_Bridge instance which is being controlled
- usrpTxPort - Transmit data to Analog_Bridge on this port. In order to run Analog_Reflector without an instance of Analog_Bridge, set the usrpTxPort to zero. You would not need Analog_Bridge if you are setting up a stand alone AllStar client or if you did not want your digital bridge to be steerable (bridge mode).
- *usrpRxPort* - Receive data from Analog_Bridge on this port
- *mobilePort* - Port to listen for DVSM/pyUC client connections
- *wsPort* - TCP port to listen for hUC (websocket) connections. By default this is set to port 443 which is the well known port for HTTPS traffic. If you define a different port, make sure to adjust the URL you use in the browser to this port number. For example, if you set this port to 8090, you would have a url like https://mydomain.net:8090
- keyDir - Location of the certificate files. See here for more on what goes in the directory.
- *siteroot* - Base directory for the HTML server. No access to files outside this directory will be allowed for any HTML client. Your mode and user json files will live below this root directory.
- *logFileName* - Your log file name and location
- *logLevel* - Log levels 1-6 (DEBUG - FATAL)
- *disallowUnknownClients* - Do not allow unknown clients to connect. If this is "true", any unknown client connection will be sent a disconnect command.
- *brokerURL* - MQTT broker URL. See here
- rightsMask - A rights value applied to any bridge that does not have an explicit rights defined. Default is TRANSMIT and MUTE (5).
- autoMute - Defines if the reflector will automatically mute analog and digital bridges when a mode change happens. In order to prevent inadvertent spamming of networks, we have implemented an autoMute rule in the reflector. This rule will mute all digital

bridges when an analog mode is selected and mute all analog bridges when a digital mode is selected. This ensures that traffic from these bridges do not leak into a network that may not want it. You can turn off this rule (PLEASE BE CAREFUL) by setting the autoMute flag to false. This should only be done by static bridges where all bridge endpoints agree to the bridge traffic.

- clientDebugMode - turn this on to see Analog_Reflector log traffic in the hUC client. DVSwitch developers may request you do this to help us debug your setup.
- clientModes - an array of allowed client modes for the hUC client. These modes will populate the modes menu on the main hUC screen. Remove any mode from the array that you do not want to appear. If you do not have an Analog_Bridge instance defined (usrpTxPort = 0), the first entry in this array will be the client startup mode. Any mode that has a vertical bar separating its value will be interpreted as having a display name | mode name. This is useful for having a human readable mode name and a easly parsed mode identifier.
- *bridges*
    - May have multiple Allstar and Digital (Analog_Bridge) bridges
    - All metadata is preserved across the reflector
    - The first AllStar node is special (it is “controlled”)
    - *asl*
        - *node* - Your node number
        - *address* - Ip Address of your AllStar node
        - *rxPort* - Port to listen for USRP packets
        - *txPort* - Port to send USRP packets
        - *rights* - Standard ACL rights
        - *amiUserName* - See manager.conf
        - *amiSecret* - See manager.conf
    - *ab*
        - *name* - Your bridge name
        - *address* - The IP address/FQDN of your Analog_Bridge instance
        - *rxPort* - Port to listen for USRP packets
        - *txPort* - Port to send USRP packets

        - If you are adding Analog_Reflector to an existing DVSwitch Server installation You will want to either set the ***mobilePort*** to either a unique port (and then expose this new port outside your firewall) or edit your Analog_Bridge.ini file to have a new rxPort/txPort and let Analog_Reflector become your new external port on the port that used to belong to Analog_Bridge.
    - Certificates
        - All modern browsers require a secure connection between the server and the client application before they will allow streaming audio and metadata to be passed. Secure connections need a certificate to manage the private data.
        - There are two methods to create/use certificates
            - We provide a script "*mkcerts.sh*" which is located in the ssl directory. This script will create a self signed certificate for your site. Self signed certificates are limiting but good enough for most of what we want to do here. The script:
                - Creates a self signed certificate for your site
                - The certificate will find your public IP address and fill in the name, IP, country, state
                - 3 files are created: key, certificate and authority
            - Using a certificate from an authority. If you are looking for a better experience, you can use a certificate that has been verified by an authority. If you already have one, they can be used or if you do not, you can get one from a variety of sources (for free)
                - https://letsencrypt.org/
                - Once you have the certificate files, you must place them in the ssl directory with the correct file names:
                    - dvswitch.crt - The certificate file
                    - dvswitch-ca.crt - The certificate authority
                    - dvswitch.key - Your private key file
                - If you already have these files from another SSL web server, you can symlink to these names so that as they get updated by your automation scripts, your Analog_Reflector files are updated too.
            - Special considerations for iOS devices
                - iOS requires the certificate authority be imported if using a self signed certificate
                - When in the hUC client on Safari
                    - Download the root certificate
                    - The iOS device will inform you that the certificate must be xxx

            - Goto iOS Settings -> General -> Profile and install the root certificate
            - Goto Settings -> General -> About -> Certificate Trust Settings -> turn on the certificate

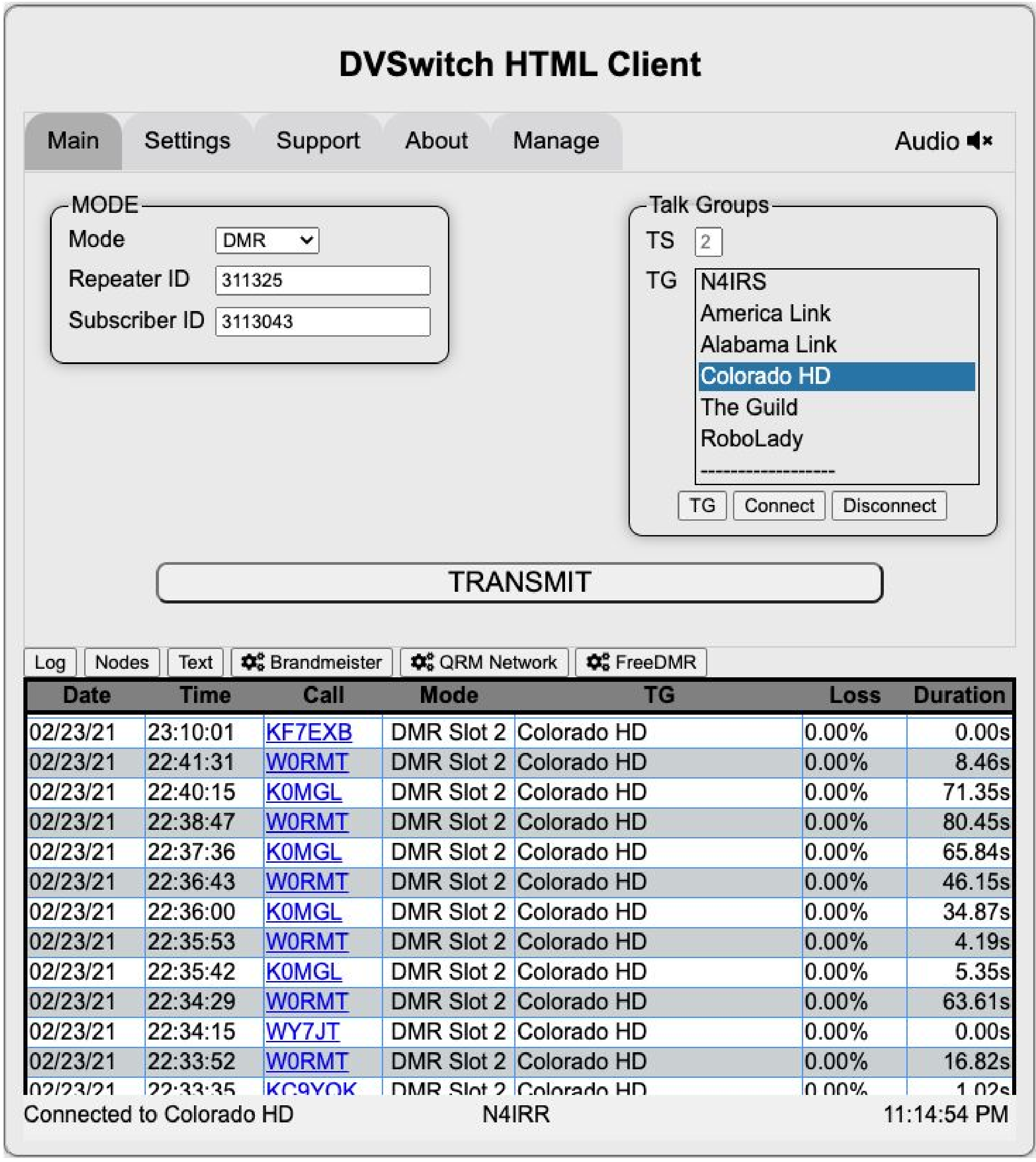
- Users
    - Creating a new user is done by using the Analog_Reflector application with the “user” command line argument. You will need several pieces of information before you create a new user.
        - Callsign used to login to Analog_Reflector
        - dmrID
        - rptID
        - Password (case sensitive)
        - rights
        - Rights are bits in a word where
            - 1 = Allow Transmit
            - 2 = Allow Tune
            - 4 = Mute
            - 8 = SYSOP
        - For example, most normal users will need to transmit and tune. Set the user rights to 3 (1+2). Your other two most common values will be to allow transmit without tune (1) and to become a SYSOP, (11).
        - A file in the arRoot/users directory will be created with the filename *callsign.json*
        - *Example:*
        - *./Analog_Reflector user KC4DEA 1234567 123456789 secret 3*
    - Editing an existing user
        - Edit the *callsign.json* file in arRoot/users Note: you can not edit the password.
        - If the user is logged in, kick them from the manage tab of the HTML client.
    - N0CALL
        - The user N0CALL (N zero CALL) is shipped by default with Analog_Reflector. This user has receive only rights and can be used to allow non-hams or other unauthenticated users to access your server. The hUC app will make it easy to use this by providing the default login dialogs with the username and password already filled in.
- Editing the modes menu
    - Analog_Reflector allows you to maintain a set of mode specific favorite talk groups and/or macros. Each mode is displayed in the talk group list of the client application when that mode is activated.

        You should edit the mode list (with your favorite editor) to customize the client's favorite items to your tastes.

    - Mode json file name construction
        - Each mode menu file name is constructed by combining the mode name (DMR, YSF, P25, etc) and _node.list.json. All 5 digital modes and AllStar menus are shipped with the reflector.
    - Mode menu items
        - disp - The text to display in the menu
        - tg - The string to send to AB when "connected"
    - Macro buttons
        - Each mode can also display a set of "macro" buttons on the hUC screen. A macro is just a convenient way of showing a command or talk group you would like to access. Most common usage would be network selection commands.
        - Macros are set up by creating an array (called macros) which contain a list of disp and tg items. These are formatted just like the "tgs" items.

- Execution
    - Foreground command line
        - Analog_Reflector {-f file} {-l level} {-d} {-cmd command} {user callsign subscriberID rptID password acl}
            - **-f file** Defines the json configuration file to use. Default is ./Analog_Reflector.json
            - **-l level** Defines the log level override. Normally, Analog_Reflector uses the logLevel in the configuration json file. This option overrides that setting. It is useful for debugging the reflector in the foreground and not having to edit the json file to see additional log information.
            - **-d** Turns on log export to web clients. Use this when instructed by developers. A new log button will appear on any hUC client connected to the reflector. Each log line will be sent to the hUC client. This is useful for remote debugging of the Analog_Reflector.
            - **user** Will create a new user profile in the arRoot/users directory. See here for more information.
            - **-cmd** Will cause this instance of Analog_Reflector to send a MQTT command to another running instance. See here for more information.
    - System unit
        - Use systemctl {start | stop | restart} analog_reflector
- MQTT API

        - Analog_Reflector has the ability to use MQTT for both command and control and for logging. MQTT is a very easy to use messaging system and there are many different clients you can use to interact with the data. To get more information on MQTT, you could start your investigations here: https://mqtt.org/ and here: https://en.wikipedia.org/wiki/MQTT
        - During development I used a free broker on the web. Point your Analog_Reflector brokerURL at mqtt://broker.hivemq.com and then you can use a web based app at http://www.hivemq.com/demos/websocket-client/ . Just hit the connect button, Add New Topic Subscription of dvswitch/# (or to be more exact dvswitch/YOUR_REFLECTOR_NAME/#) and watch the data flow. You can send yourself a text message by changing the publish “topic” to dvswitch/YOUR_NODE_NAME/Analog_Reflector/text and type your message into the “message” window. Select publish and watch your hUC client display the message.
        - Basic scheme is dvswitch/${nodeName}/Analog_Reflector
        - Topics
            - scheme/text - Send text message to all connected clients
            - scheme/command - Execute a command on the server
                - tune tg
                - kick callsign
                - kickAll
                - getAllConnectionIds
                - getAllConnectionObjects
                - getClient clientID
                - authenticate
                - getConfig
                - getReflectorVersion
                - changeLogLevel newLevel
                - refreshAllClients
                - banConnection
                - modifyConnection connectionID rights
                - getConnectedNodes
                - Text message
            - scheme/log - The server publishes all log messages to this topic
- hUC HTML client
    - Included and built into the Analog_Reflector is the hUC client application (**H**TML **U**SRP **C**lient). They come as a single package so installation is simple. The client requires no install on any computer you execute it on, just a compatible browser. Most any modern browser will do. iOS is a little special (thanks Apple) and you should probably stick to Safari even though I see that in the latest iOS 14 releases Chrome and other third party browsers are (now) working.



DVSwitch HTML Client

MODE

Mode	DMR
Repeater ID	311325
Subscriber ID	3113043

Talk Groups

TS 2

TG
- N4IRS
- America Link
- Alabama Link
- Colorado HD
- The Guild
- RoboLady
- ------------------

Date	Time	Call	Mode	TG	Loss	Duration
02/23/21	23:10:01	KF7EXB	DMR Slot 2	Colorado HD	0.00%	0.00s
02/23/21	22:41:31	W0RMT	DMR Slot 2	Colorado HD	0.00%	8.46s
02/23/21	22:40:15	K0MGL	DMR Slot 2	Colorado HD	0.00%	71.35s
02/23/21	22:38:47	W0RMT	DMR Slot 2	Colorado HD	0.00%	80.45s
02/23/21	22:37:36	K0MGL	DMR Slot 2	Colorado HD	0.00%	65.84s
02/23/21	22:36:43	W0RMT	DMR Slot 2	Colorado HD	0.00%	46.15s
02/23/21	22:36:00	K0MGL	DMR Slot 2	Colorado HD	0.00%	34.87s
02/23/21	22:35:53	W0RMT	DMR Slot 2	Colorado HD	0.00%	4.19s
02/23/21	22:35:42	K0MGL	DMR Slot 2	Colorado HD	0.00%	5.35s
02/23/21	22:34:29	W0RMT	DMR Slot 2	Colorado HD	0.00%	63.61s
02/23/21	22:34:15	WY7JT	DMR Slot 2	Colorado HD	0.00%	0.00s
02/23/21	22:33:52	W0RMT	DMR Slot 2	Colorado HD	0.00%	16.82s
02/23/21	22:33:35	KC9YOK	DMR Slot 2	Colorado HD	0.00%	1.02s

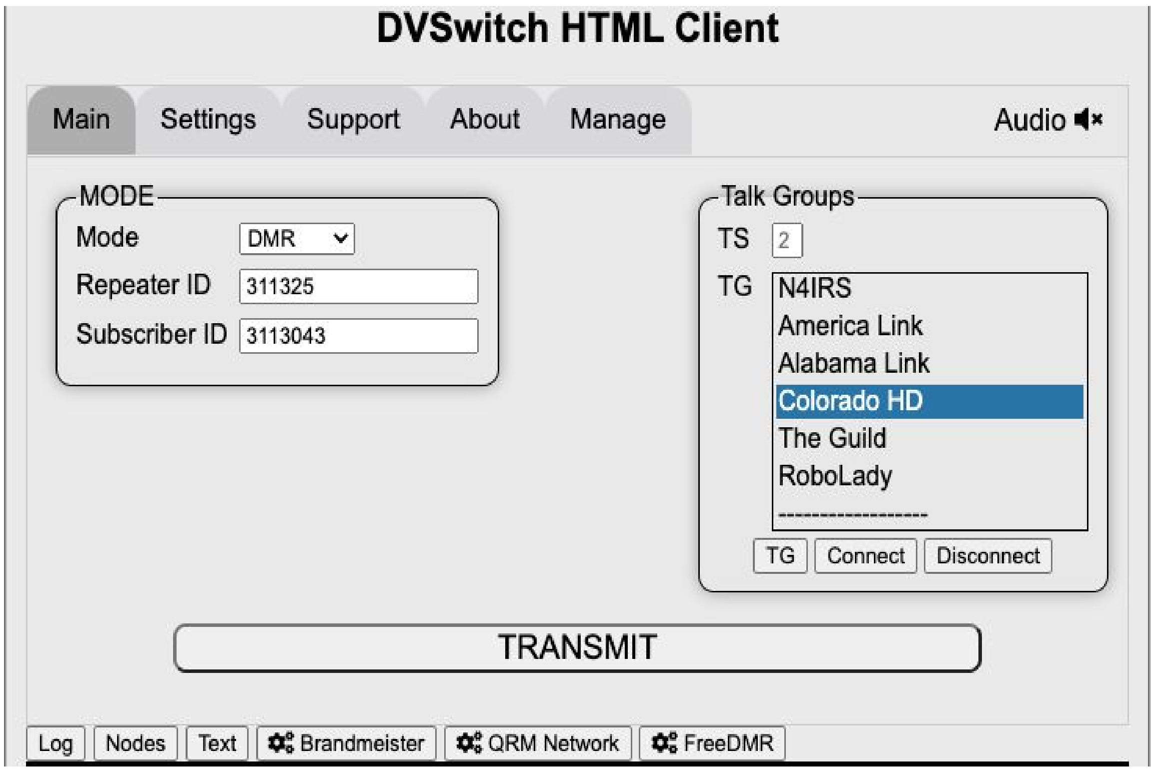
Connected to Colorado HD N4IRR 11:14:54 PM

- The URL
  - When accessing Analog_Reflector please pay special attention to the URL you type in. It must include both HTTPS and a port number unless you are using port 443). The reflector will do an automatic redirect for you to the full path to the hUC app, so you can just type in: https://fqdn:port.
- Logging in
  - The first time you visit the reflector it will ask you to log in. You should have ready your callsign and password for access to the reflector. Because this app will probably be available to you on the open web, we do not want to allow unlicensed people to transmit on the ham bands.

Users must be granted rights by the system administrator for transmit and tune.

- If the user has been granted rights on the system, they should log in with their callsign and password.
- If the user does not have an entry in the access list (ACL) they can log in with their call sign and will be granted receive only rights. They will not be asked for a password as they can only listen to the communications present on the server. This type of access will only be available if the disallowUnknownClients flag is set to false.
- We ship the server with a special call N0CALL (and a password of "passw0rd"). This combination is also used to grant read only access to the server.

○ Running the app

- 

MODE

Mode DMR

Repeater ID 311325

Subscriber ID 3113043

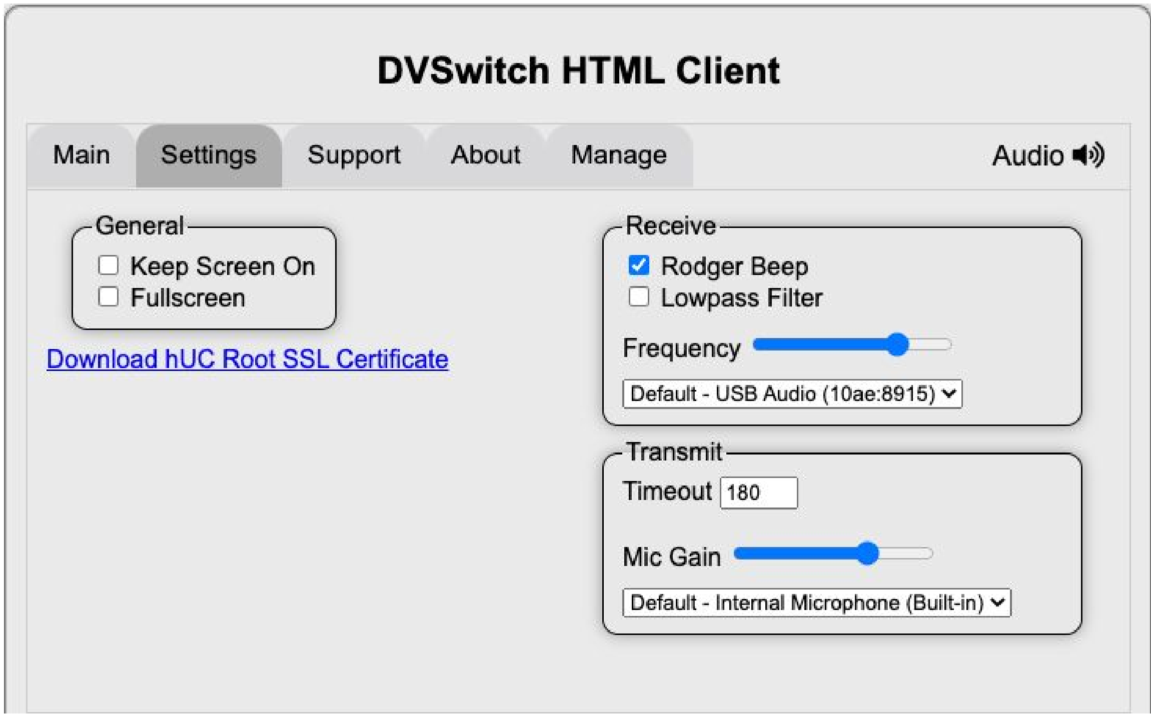
Talk Groups

TS 2

TG N4IRS, America Link, Alabama Link, Colorado HD, The Guild, RoboLady, ------------------

- Audio button enables both receive and transmit audio in the browser. You must interact with this button before you will hear any audio from the reflector.
- Mode Selection is done by using this pull-down menu. The mode and TG menu control the main instance of the attached Analog_Bridge (if there is one).
- Talkgroup Selection is done by selecting the list item and then pressing connect (or double clicking on the item). To disconnect from the last connection, press the "Disconnect" button. These buttons will be disabled if you do not have rights to tune the node.

        - Note that Mode and TG selection may be disabled if the user does not have rights to tune. The TG, Connect, Disconnect and Mode pull down menu will be shadowed.
        - While a transmission is in progress the QRZ picture, mode, call, name and DMR ID is displayed. You can click on the QRZ picture to visit the station's website.
        - The Last Heard list has several views available. **Log** (or last heard) is the stations heard from any bridge or client connected to the reflector. Right click or long press on a log entry to get more information about the station. The **Nodes** view is populated when using the AllStar mode. It shows the nodes connected to your main node. **Text** is used to show a running list of text messages sent to your client from the server or any other client.
        - Mode specific macro buttons can also appear just above the Last Heard list. These buttons will appear with a gear icon and are configured in the json file specific to the current mode.
        - Ad-Hoc talkgroups and menus
            - You may enter your own talk group number by selecting the “TG” button. This will bring up a popup where you can enter a free format string or select from a menu of commands on the pull down menu. The menu is populated only when DVSwitch Server is in the “advanced mode”.
        - Status bar
            - Connected to shows the current talk group/reflector/room/node you are connected to. If the text is truncated because of space, just hover over the item to reveal more information.
            - If the Analog_Reflector server goes offline, the “Connected to” will change to “Server offline” to indicate the new status. hUC will attempt to reconnect by itself when this happens.
            - Your call sign is displayed in the center of the status bar while in idle state. When a station is transmitting the item shows who is transmitting and the destination.
            - Current time is displayed in the bottom right hand corner of the status bar.
    - Settings

- 

DVSwitch HTML Client

Main | Settings | Support | About | Manage | Audio

General
- [ ] Keep Screen On
- [ ] Fullscreen

Download hUC Root SSL Certificate

Receive
- [✓] Rodger Beep
- [ ] Lowpass Filter

Frequency

Default - USB Audio (10ae:8915)

Transmit

Timeout 180

Mic Gain

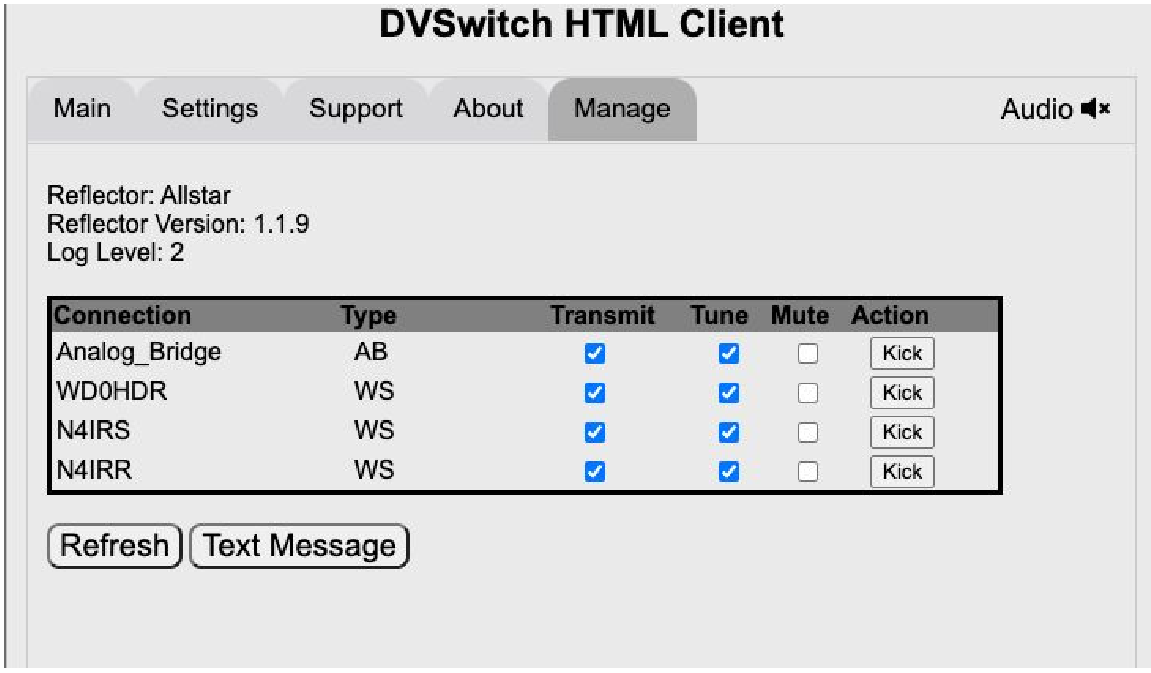
Default - Internal Microphone (Built-in)

- General
    - Keep Screen On - Some browsers support this option to keep the screen from timing out and locking. This is of course a battery drain on mobile devices.
    - Fullscreen - Some browsers allow for going full screen and removing surrounding chrome
- Receive
    - Rodger Beep - Enables/Disables the audible beep at the end of a transmission.
    - Lowpass Filter - controls the inclusion of a low pass filter in the audio path. This is used to remove any high frequency components that may be the result of upsampling of the audio on some devices (thanks Apple). The cutoff frequency is defined by the Frequency slider. Move it to the right to allow higher cutoff frequencies, and to the left to restrict the audio content.
- Transmit
    - Timeout - The transmit timeout timer in seconds. When a timeout occurs the user will hear an audible beep and be transitioned to receive.
    - Mic Gain - As usual each device and hardware combination produces a different audio level on transmit. Use the parrot function to listen to your audio to make sure it is in line with others on that specific mode.
- The devices listed are for display only and can NOT be selected to change your input or output devices. This will be a future addition to the

hUC application, but for now it was deemed valuable to show the user what devices are available to the client app.

- Download hUC Root SSL Certificate - iOS is a "special" player. Apple will not allow self signed certificates to create secure connections without installing a root certificate on the device. Refer to the section in this document covering configuration to learn how to install this certificate and enable it for use.
- Client settings are stored in cookies on the browser. They are not stored on the server so that you may have different settings on each of your devices. For example, you may want different mic gain settings on your phone and your desktop.

- Manage



Reflector: Allstar
Reflector Version: 1.1.9
Log Level: 2

Connection	Type	Transmit	Tune	Mute	Action
Analog_Bridge	AB	✓	✓	☐	Kick
WD0HDR	WS	✓	✓	☐	Kick
N4IRS	WS	✓	✓	☐	Kick
N4IRR	WS	✓	✓	☐	Kick

- The Manage tab will allow the SYSOP to change the access rights of any active client, kick a client off of the reflector and to send "text messages" to all other users.

- Bridging with Analog_Reflector
  - AllStar
    - Features
      - Fully node agile (dial and command your node)
      - Get limited node metadata
      - Text messages sent to the node from the digital side metadata
    - Configuration
      - On the AllStar node
        - Configuring USRP
          - Enable the USRP channel driver
          - If adding to an existing node, create a private node
          - Otherwise, configure the main node section

- Enable the Management interface
    - Analog_Reflector uses the Asterisk Management Interface (AMI) to send commands to your node and to get events back from the node.
    - In manager.conf
        - Make sure the manager interface is enabled.
        - Set the AMI user name as the section name. For example [admin].
        - Set the secret in the section
- On Analog_Reflector
    - Edit the Analog_Reflector.json file and create an AllStar (asl) entry in bridges with the node number, address, rxPort, txPort, rights, amiUserName and amiSecret.

```
"asl":
[
        {
                "node": "41263",
                "address": "192.168.1.88",
                "rxPort": "34001",
                "txPort": "32001",
                "rights": 1,
                "amiUserName": "admin",
                "amiSecret": "secret"
        }
]
```

- Usage
- Analog_Bridge based bridges
    - Setting up a bridge
    - LOTS OF TEXT GOES HERE
    - Configuring your Analog_Bridge.ini file
        - In the [USRP] section edit the Address, RXPort and TXPort values to point at the corresponding values in the Analog_Reflector configuration file. Remember **RX**Port in AB points to usrp**Tx**Port in the AR json file and **TX**Port in AB points to usrp**Rx**Port in the AR file.
    - Configuring your Analog_Reflector json file
        - Analog_Reflector supports several digital bridges at the same time. To create a new bridge, add an entry to the “ab” array with the fields name, address, rxPort, txPort and rights defined. Just like the discussion above, make sure your rx ports point at Analob_Bridge tx ports and vice versa.
- Automatic Mute between Analog (AllStar) and Digital networks
    - In order to make sure that a user does not bridge analog and digital networks together without knowing it, the reflector will automatically mute

        the mode which is not in use to ensure no traffic is passed without their knowledge. For example, if the user selects the ASL mode from the hUC client, all digital networks (the controlled Analog_Bridge instance and any bridge) will be muted. And, when the user switches back to a digital mode (i.e. DMR) the AllStar connection will be muted.

- Troubleshooting
    - 404 from server
        - If the reflector is running make sure that the siteRoot in the reflector json file is correct and your URL is also correct. Note that this is a HTTPS connection and that you may also need to specify a port. For example **https**://my_domain.org:**8090**/bla/bla/bla
    - Error, configuration json "XXX" not found
        - When running Analog_Reflector you can use the -f *config_name* to specify a configuration json file to use. If the argument is not specified the reflector will use Analog_Reflector.json in the current working directory. You may specify a full path name to the json file you want to use.
    - Fatal error, SSL key and cert files were not found
        - Please make sure your keyDir is correct and that the files dvswitch.crt, dvswitch.key and dvswitch-ca.crt are in the directory.
    - Failed to capture microphone
        - This error can be due to several causes. Make sure you have mic access enabled in your browser.
    - On an iPhone you can not authenticate.
        - Please be aware that passwords are case sensitive. On an iPhone, you may have “auto capitalization” turned on, which would cause the password to not match.
    - Passwords with special characters
        - When you create the password you may want to single quote the whole value in case the user has selected a password which contains special characters (like !). The Linux Bash shell interprets these characters before they get to the app, so quoting them causes the shell to leave them alone.
    - Its json!
        - If you are not used to the format, use an editor that checks your syntax as you type. If you do not do that, use an online json validator to check the files for errors. Even a simple editor like nano can be used to edit json. We suggest you backup the file before editing. Pay attention to the formatting of the file. Every comma, quote, colon, etc are needed. Note: json does not support comments in the file.
- Misc
    - There are two modes on the modes menu you may not be familiar with;
        - STFU (Simple Terminal Functional Update) and Intercom. STFU is used to access the Brandmeister DMR network. It uses their preferred access protocol (Open DMR Terminal Protocol). It does have several

        advantages over the MMDVM/Homebrew protocol including the ability to subscribe and leave a talkgroup without waiting for a station to finish talking. Also, you can subscribe to a talkgroup without having to transmit. For now, DVSwitch is offering the STFU client as a stand alone component as other DMR networks are not supporting this protocol, so both need to be accessible. Configuration of the STFU mode is done in the DVSwitch.ini file using the [STFU] section. Please note, we have experienced that some BM servers may not be configured correctly for this protocol, if yours is one of these, please contact BM for support.
        - Intercom is used to facilitate communication between Analog_Reflector users without transmitting over any ham radio network. It mutes any bridge (digital or analog) and the steerable AB instance. All users connected to the reflector are included in the conversation while in this mode.
    - logrotate is used to archive the old log file. Once the log file has been rotated, logrotate wll send Analog_Reflector a SIGHUP signal which will begin the new log file session. By default, the log files are placed in /var/log/dvswitch.
    - Server restart. Upon exiting the server, all connected clients will be sent an unregister command. DVSM and hUC will attempt to reconnect to the server if instructed, so upon server restart, they should log back in.
    - The server does some simple certificate validation when starting up. It checks for the server IP and hostname/FQDN in the certificate. This ensures that the certificate being used is proper for the operation of the server and that clients will accept it for this ip address.
    - Bluetooth is “supported” so far as both speaker and mic can be routed through the default audio device on your computer/phone. However PTT has not been enabled for BLE or BT Serial type microphones so you have to use the on screen Transmit button.
    - The hUC client does support ad-hoc menus just like DVSM and pyUC. When an ad-hoc menu is loaded, it can be accessed from the “TG” button and then use the pulldown in the modal dialog to access the menu items.
    - The Analog_Reflector can be accessed from outside your firewall. In addition to the UDP port you would forward for DVSM access (just like you did for Analog_Bridge), you should also port forward the TCP port you defined for wsPort in your Analog_Reflector.json configuration file.
    - Mobile operation of the hUC application can be challenging The hUC application is not a “responsive” mobile application. It was not intended to be. At some point we may decide to implement different UI views, but not at this time. Also, many platforms may try to optimize battery usage when the web browser is not the foreground app or the screen is turned off. I have found Apple to perform well in both these situations, some android phones do OK, and Fire tablets are too aggressive on battery optimization. As a browser application we are at the mercy of each vendor to implement their platform as they see fit.