

WICKRIO CLIENT REST API

Prepared for: Wickr Inc, Engineering Group

Prepared by: Paul Cushman, Project Lead

Date: April 13, 2018

Version: 4.35

This document describes the REST APIs that are currently supported by the WickrIO Client. Each entry in this table identifies the action the API performs, the type of HTTP request and the URL used. Each API will be described in detail later in this document.

|  |  |  |
| --- | --- | --- |
| **API** | **HTTP** | **URL** |
| Send Message | POST | https://<host>:<port>/Apps/<API Key>/Messages |
| Set Message URL Callback | POST | https://<host>:<port>/Apps/<API Key>/MsgRecvCallback?callbackurl=<url> |
| Get Message URL Callback | GET | https://<host>:<port>/Apps/<API Key>/MsgRecvCallback |
| Delete message URL Callback | DELETE | https://<host>:<port>/Apps/<API Key>/MsgRecvCallback |
| Set Email Callback | POST | https://<host>:<port>/Apps/<API Key>/MsgRecvEmail?server=[<](http://secure.emailsrvr.com/)outgoing server>  &port=<out port>  &type=<smtp|ssl|tls>  &account=<account>  &password=<password>  &recipient=<recipient>  &sender=<sender>  &subject=<subject> |
| Get Email Callback | GET | https://<host>:<port>/Apps/<API Key>/MsgRecvEmail |
| Delete Email Callback | DELETE | https://<host>:<port>/Apps/<API Key>/MsgRecvEmail |
| Get Received Messages | GET | https://<host>:<port>/Apps/<API Key>/Messages?start=<index>&count=<number> |
| Get Statistics | GET | https://<host>:<port>/Apps/<API Key>/Statistics |
| Clear Statistics | DELETE | https://<host>:<port>/Apps/<API Key>/Statistics |
| Create Secure Room | POST | https://<host>:<port>/Apps/<API Key>/Rooms |
| Get Room | GET | https://<host>:<port>/Apps/<API Key>/Rooms/<vGroupID> |
| Get Rooms | GET | https://<host>:<port>/Apps/<API Key>/Rooms |
| Delete Room | DELETE | https://<host>:<port>/Apps/<API Key>/Rooms/<vGroupID> |
| Leave Room | DELETE | https://<host>:<port>/Apps/<API Key>/Rooms/<vGroupID>&reason=leave |
| Modify Room | POST | https://<host>:<port>/Apps/<API Key>/Rooms/<vGroupID> |
| Create Group Conversation | POST | https://<host>:<port>/Apps/<API Key>/GroupConvo |
| Get Group Conversations | GET | https://<host>:<port>/Apps/<API Key>/GroupConvo |
| Get Group Conversation | GET | https://<host>:<port>/Apps/<API Key>/GroupConvo /<vGroupID> |
| Delete Group Conversation | DELETE | https://<host>:<port>/Apps/<API Key>/GroupConvo/< vGroupID > |

# AUTHENTICATION

This version of the WickrIO supports two types of authentication: basic and wickr-token. Both forms will use the "Authorization" HTTP header to send the necessary authentication information to the WickrIO server.

If the proper authentication information is not presented to the WickrIO server then an HTTP 401 response will be sent.

## Basic Authentication

When using basic authentication an encoded string of the user name and password will be sent to the WickrIO server.  The following steps should be performed for basic authentication:

1. Build a string of the form username:password
2. Base64 encode the string
3. Supply an "Authorization" header with content "Basic " followed by the encoded string.  For example, the string "fred:fred" encodes to "ZnJIZDpmcmVk" in base 64, so you would send the string "Basic ZnJlZDpmcmVk" in the "Authorization" HTTP header.

This is the simplest form of authentication but is the least secure.

The credentials used for authentication must be associated with a console user account that is authorized to access/modify the specific client. If the client credentials authenticate successfully but is NOT associated with a console user that can access the specific client, then an HTTP 403 (Forbidden) response will be generated.

## Wickr-Token Authentication

With the Wickr-Token form of authentication a SHA256 encrypted string will be sent in the "Authorization" HTTP header. The string will be an encrypted version of the username, using a key that is generated on the WickrIO server.  The key will be sent from the WickrIO server to an email address that is configured locally on the WickrIO server.  This key will be re-generated when authentication fails.  The key will also be re-generated after a period of 20 minutes of inactivity (not implemented yet).

Details of this form of authentication will be added. TBD.

# SEND MESSAGE API

This API is used to send a message to one or more Wickr clients. To have a specific WickrIO client send a message, send an HTTP POST message to the following URI:

https://<host>:<port>/Apps/<API Key>/Messages

The body of the POST request is JSON and the contents of that JSON will determine the type of Wickr message and the destination(s) of the message. You can send a message to one or more 1-to-1 recipients, a secure room or a group conversation. To send to a secure room or a group conversation, the room will have to been created before sending any messages.  You will also need the vGroupID associated with the secure room or group conversation.

The send message API also supports sending files.

There are several JSON fields used by this API, and depending on the type of message or file being sent. The following table lists all of the JSON fields that are supported by the Send Message API.

|  |  |
| --- | --- |
| **Key** | **Description** |
| message | String value that is the message to be sent, when sending a Wickr message. Not used when sending a file. |
| users | This is an array of users. These are the users to be sent to. Not used when sending to a secure room or group conversation. These are the wickr ID values associated with the user to send to. |
| bor | The burn-on-read value to use when sending the message |
| ttl | The time-to-live value to use when sending the message (referred to as Expiration Time in clients) |
| vgroupid | The vgroupid to use when sending to a secure room or group conversation. |
| runtime | Used to specify when the message(s) are to be sent. The format of this value is [**ISO 8601**](http://www.iso.org/iso/catalogue_detail?csnumber=40874) extended format: either yyyy-MM-dd for dates or yyyy-MM-ddTHH:mm:ss (e.g. 2017-07-24T15:46:29), or with a time-zone suffix (Z for UTC otherwise an offset as [+|-]HH:mm) where appropriate for combined dates and times. |
| attachment | This is a JSON object that identifies a file to be sent. The message value will be disregarded if the attachment field is used. |
| attachments | Identifies files to be sent. Currently only supports one file. Future implementation will support sending multiple files. |

When sending files, the location of the files either needs to be located on the WickrIO system or addressable and downloadable using a specific URL.

To send a message to one or more recipients, the HTTP POST message should contain a JSON body with the following format:

{

"message”: "Welcome to Wickr! This message will self-destruct in 5 seconds.",

"users" [

{ "name" : "username001" },

{ "name" : "username002" }

]

}

Figure 1: Send message JSON body

The "users" field may contain an array of 1 or more users to send the message to.  The message will be sent to each user on a separate 1-to-1 conversation. So, if the POST message contains 5 users then 5 messages will be sent, using the text from the "message" field.

When sending a message, you can also set the specific burn on read (BOR) value for the message.  The following format shows how to set the BOR value to 10 seconds:

{

"message": "Welcome to Wickr! This message will self-destruct in 5 seconds.",

"users" [ { “name": "username002" } ],

"bor": 10

}

Figure 2: Send message JSON body with BOR

If you want to send a message to a secure room or a group conversation you will need to get the vGroupID associated with the room. The vGroupID will be returned when you create the room/conversation using the appropriate API. Also, the get rooms API will return a list of known rooms that you can send to, the vGroupID is contained in the response. To send to a secure room or group ID the following is an example:

{

"message": "Welcome to Wickr! This message will self-destruct in 5 seconds.",

"vgroupid": "S8a97892379289bca979293709822718928392837492837492834"

}

Figure 3: Send message to Secure Room JSON body

The client also supports the ability to send files. Files must be located at a URL that can be downloaded by the client or on the actual WickrIO system where the WickrIO client can access the file. The following is an example of sending a file to a single user, where the file is located on the WickrIO system:

{

"users" [ { "name" : "username001" } ],

"attachment": { "filename" : "/home/ubuntu/pictures/picturesent.jpg" }

}

Figure 4: Send file JSON body

In this example, the file to be sent is already on the WickrIO client. The filename identifies the full path to the file. The Send Message API will respond with a 202 (Accepted) response, unless there is an error.

Currently any files sent will remain on the WickrIO operating system. It is up to the Administrators to remove any files uploaded to the server after sending via Wickr.

# RECEIVED MESSAGE

The WickrIO client has the capability to receive messages and then forward these messages to a third-party application or to a specific email.

Message callbacks are used to identify where the WickrIO client should send incoming Wickr messages. Currently there are two types of message callbacks supported, URL and Email. Only one can be setup at a time. If you want to change to a different type of callback then you will need to delete the current callback setup, and then create the new one. To change the settings of the same type you can just send a POST command for the associated type. Messages received by the WIckrIO client will be sent to the configured URL or Email callback in a JSON format. The format of the received messages and meta data will be described later.

NOTE: If a callback destination is configured, the WickrIO client will queue these messages to be sent to that destination.  These messages will remain in this queue until they have been successfully posted to the desired destination.

## URL Callbacks

This type of callback will define a URL that the client will connect to when a message is received, the received message will be sent to the URL. It is assumed that there is an application consuming messages that are pushed to this URL. For example, if you run a process on the same machine as the WickrIO client you can use a URL like the following:

http://localhost:4100

Wickr can supply a sample program that accepts messages on a specific port and posts to the syslog.

### Set URL Callback API

You will need to configure the specific URL that the WickrIO client will send incoming messages to. To set the URL callback value, send an HTTP POST request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvCallback?callbackurl=<url>

The client will respond back with a success or failure response.  Any messages received after this call is performed will be sent to the defined URL.

NOTE: this request will fail if an Email callback has been defined. Please delete the Email callback before setting the URL callback.

### Get URL Callback API

To get the currently set URL callback send an HTTP GET request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvCallback

The client will respond with either a success (200) or failure response. If there is a URL callback defined the following is the format of the body:

{

"callbackurl": "https://localhost:4008"

}

Figure 5: Get MsgRecvCallback response body

### Delete URL Callback

If a URL callback is no longer needed or you need to switch to an email callback then the delete URL callback API should be used. To delete a URL callback, send an HTTP DELETE request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvCallback

The client will respond with a success or failure response.

## Email Callbacks

This type of callback will define an email address to send received messages to. Any message received by the client will be forwarded to the defined email address.

### Set Email Callback

To perform the set Email callback API, send an HTTP POST request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvEmail?server=[<](http://secure.emailsrvr.com)outgoing server>&port=<out port>&type=<smtp|ssl|tls>&account=<account>&password=<password>&recipient=<recipient>&sender=<sender>&subject=<subject>

Where:

|  |  |
| --- | --- |
| **Argument** | **Description** |
| server | The host address of the email outgoing server |
| port | The port on the email outgoing server to send the emails to |
| type | The type of port, one of: smtp, ssl or tls |
| account | The account name to use when logging into the email server |
| password | The password associated with the email account |
| recipient | The email address of the recipient for the messages |
| sender | The email address to use when sending messages |
| subject | The subject text to use when sending messages |

The client will respond back with a success or failure response.  Any messages received after this call is performed will be sent to the defined recipient email address using the defined email credentials.

NOTE: this request will fail if an URL callback has been defined. Please delete the URL callback first.

WARNING: This API will change in the near future. The email server information will likely be defined in a more central location, per WickrIO server as opposed to per WickrIO client.

### Get Email Callback

To get the currently set Email callback send an HTTP GET request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvEmail

The client will respond with either a success (200) or failure response. If there is an Email callback defined the following is the format of the body:

{

"callbackemail”: {

"message\_setup”: {

"recipient”: "recipientemail",

"sender”: "senderemail",

"subject”: "Message subject"

},

"server\_setup”: {

"account”: "serveraccount",

"password”: "accountpassword",

"server”: "serverhost",

"port”: <portnumber>,

"type”: "<smtp|ssl|tls>"

}

}

}

Figure 6: Get email callback response body

WARNING: If and when the email server setup moves to a more central location that portion of the response will move as well.

### Delete Email Callback

If an Email callback is no longer needed or you need to switch to a URL callback then the delete Email callback API should be used. To delete an Email callback, send an HTTP DELETE request using the following URI:

https://<host>:<port>/Apps/<API Key>/MsgRecvEmail

The client will respond with a success or failure response.

## Callback Message Format

There are different types of messages that will be delivered to the configured message destination, URL or email callbacks. This section will describe each of these formats. All of the formats are using JSON.

### One-to-one messages

The following shows a normal one-to-one message format. All text-based messages will have the msgtype of 1000.

{

"id":"3960e020ca4211e799802f2894564caa",

"message":"This is a typical 1:1 message",

"msg\_ts":"1510777143.738976",

"msgtype":1000,

"receiver":"pwcuser001",

"sender":"pwcuser003",

"time":"11/15/17 3:19 PM",

"vgroupid":"fb6e21630c05fde50ae39113c3626018712cf2c374b4a80eba4d28ced9419c07"

}

### Group and Secure Room Conversation messages

The following shows a normal group and secure room conversation message format. The WickrIO client does not currently track the list of clients associated with group conversations, so the list of destination clients will not be included.

{

"id":"76775de0ca4211e7bddcafd7007db1d1",

"message": "Typical message in a secure room",

"msg\_ts":"1510777246.227505",

"msgtype":1000,

"sender":"pwcuser003",

"time":"11/15/17 3:20 PM",

"vgroupid":"S3042f1bd04491c6f3732a871e27ab516a8d1534cc1e2d25c4e4869ce72e8541"

}

### File Transfer messages

The following shows the format of a file transfer message. The msgtype for files is 6000. Files received by the WickrIO client will be decrypted and remain on the WickrIO client until removed by your software.

{

"file": {

"filename": "picture.jpeg",

"guid": "AD20D048-9B60-4F32-A691-2D4BE4152E58",

"localfilename": "/opt/WickrIO/clients/compliancebot01/attachments/attachment\_20171116111610865\_picture.jpeg"

},

"id": "91a189c0cae911e79ec4eb19a763225b",

"msg\_ts": "1510849017.756174",

"msgtype": 6000,

"sender": "pwcuser003",

"time": "11/16/17 11:16 AM",

"vgroupid": "S3042f1bd04491c6f3732a871e27ab516a8d1534cc1e2d25c4e4869ce72e8541"

}

As of version 4.35, files sent for screen shots will be identified by a “isscreenshot” key value pair, in the “file” object. This is a Boolean value, where true identifies the file as a screenshot. If the “isscreenshot” key is not found then the file is not a screen shot.

### Wickr Control messages

Wickr control messages used to setup and configure the conversations will also be sent to the callback destination. These messages are useful to reproduce the conversations, specifically which clients are associated with the specific conversation. Details of these message types will be included later.

# STATISTICS APIs

Each WickrIO client maintains statistics associated with the number of messages sent and received. The statistics will also include information about send and receive errors as well as the number of pending messages. This section describes the APIs associated with statistics.

## Get Statistics API

This API will retrieve statistics that are saved on the client.  These are a little different than the statistics that can be retrieved via the main console interface. Here is the HTTP GET used to get client statistics:

https://<host>:<port>/Apps/<API Key>/Statistics

The following is the type of output received in the response. Some of these statistics may be missing if they have not been changed.

{

"statistics”: {

"message\_count”: 5,

"pending\_messages”: 0,

"sent”: 7,

"received”: 3,

"sent\_errors”: 1,

"recv\_errors”: 1

}

}

Figure 7: Get statistics response JSON

The following table has a description of each of the statistics returned by this API:

|  |  |
| --- | --- |
| Statistics | Description |
| message\_count | The number of incoming messages that are currently on the WickrIO client. |
| pending\_messages | The number of messages that are to be sent from the specific WickrIO client. |
| sent | The number of messages that have been sent by the WickrIO client. |
| received | The number of messages that the WickrIO client has received. |
| sent\_errors | The number of errors that have occurred while trying to send messages. |
| recv\_errors | The number of errors that occurred while receiving messages. |
| pending\_callback\_messages | The number of messages on the callback message queue. These are messages received by the WickrIO client, that are waiting to be send to a callback process. |
| outbox\_sync | The number of outbox sync messages received. These are messages that were sent by another device for this WickrIO client. |

## Clear Statistics API

This API will clear the current statistics that are saved on the client.  Use HTTP DELETE with the following URI to clear the statistics on the specific WickrIO client:

https://<host>:<port>/Apps/<API Key>/Statistics

# SECURE ROOM APIs

This section describes the APIs associated with secure rooms. Using these APIs you can create, modify, get, delete and leave secure rooms that the client is a part of.

## Create Secure Room API

This API will create a new secure room.  To create a secure room, send an HTTP POST command using the following URI:

https://<host>:<port>/Apps/<API Key>/Rooms

The body of this request will contain the information associated with the room. The following is the format of the JSON data for the body of the request:

{

"room”: {

"title" : "Room Title",

"description" : "Description of the room",

"ttl" : 3600,

"bor" : 60,

"members" : [

{ "name" : "username001@wickr.com" },

{ "name" : "username002@wickr.com" }

],

"masters" : [

{ "name" : "username001@wickr.com" },

]

}

}

Figure 8: Create secure room body JSON

The response will either be an error with a description of that error or a successful response with the vGroupId of the newly created secure room. The following is an example of a successful response:

{

"vgroupid": "S0b503ae14cc896aad758ce48f63ac5fae0adccd78ef18cde82563c63b2c7761"

}

Figure 9: Create secure room response JSON

## Get Rooms API

This API will return a list of rooms that are known by the WickrIO client. To get a list of rooms send an HTTP GET to the following URI:

https://<host>:<port>/Apps/<API Key>/Rooms

The WickrIO client will respond with a JSON array of secure rooms. The format of the response will look like the following:

{

"rooms”: [

{

"description”: "Room description",

"masters”: [

{ "name" : "username001" }

],

"members”: [

{ "name" : "username001" },

{ "name" : "username002" }

],

"title”: "Room Title",

"ttl”: "7776000",

"bor”: "0",

"vgroupid”: "S00bf0ca3169bb9e7c3eba13b767bd10fcc8f41a3e34e5c54dab8bflkjdfde"

}

]

}

Figure 10: Get secure rooms response JSON

## Get Secure Room API

This API will return details of a specific secure room or group conversation. Send an HTTP GET using the following URI:

https://<host>:<port>/Apps/<API Key>/Rooms/<vGroupID>

The WickrIO will respond with a JSON structure containing information for the specified conversation. The format of the response will look like the following:

{

"rooms”: [

{

"description”: "Room description",

"masters”: [

{ "name" : "username001" }

],

"members”: [

{ "name" : "username001" },

{ "name" : "username002" }

],

"title”: "Room Title",

"ttl”: "-1",

"vgroupid”: "S00bf0ca3169bb9e7c3eba13b767bd10fcc8f41a3e34e5c54dab8bflkjdfde"

}

]

}

Figure 11: Get secure room response JSON

## Delete Room API

In order to delete a secure room, you will need to have the vGroupID associated with that room. You can use the get rooms API to get the list of rooms known by the WickrIO client, then determine which room to delete. Also, saving the vGroupID returned from the create room API can be used as well.

To delete a secure room, send an HTTP DELETE command using the following URI:

https://<host>:<port>/Apps/<API Key>/Rooms/< vGroupID >?reason=delete

The secure room with the same vGroupID will be deleted. The reason argument does not need to be present, since the delete action is the default action, or if can specify the delete action.

## Leave Room API

In order to leave a secure room, you will need to have the vGroupID associated with that room. You can use the get rooms API to get the list of rooms known by the WickrIO client, then determine which room to leave. Also, saving the vGroupID returned from the create room API can be used as well.

To leave a secure room or group conversation send an HTTP DELETE command using the following URI:

https://<host>:<port>/Apps/<App ID>/Rooms/< vGroupID >?reason=leave

Note the use of the reason argument that specifies this as a leave, as opposed to a delete room API call. The reason MUST have a value of leave.

The user associated with the WickrIO client will leave the room associated with the specified vGroupID.

## Modify Room API

This API is used to modify some of the settings associated with a secure room. The following secure room attributes can be modified using this API:

* TTL
* BOR
* Description
* Title
* Members
* Moderators

To modify any of these values for a secure room send an HTTP POST command using the following URI:

https://<host>:<port>/Apps/<App ID>/Rooms/< vGroupID >

The body of the request will identify the values to change and the new values to assign. The following JSON is an example of a body for the modify room API which will set the BOR and TTL values:

{

"ttl”: 66000,

"bor”: 300,

}

# GROUP CONVERSATION APIs

This section describes the APIs associated with group conversations. Using these APIs you can create, get or delete group conversations that the client is a part of.

NOTE: These APIs are only available in versions 4.35 and newer of the WickrIO client.

## Create Group Conversation API

This API will create a new group conversation.  To create a group conversation, send an HTTP POST command using the following URI:

https://<host>:<port>/Apps/<API Key>/GroupConvo

The body of this request will contain the information associated with the group conversation. The following is the format of the JSON data for the body of the request:

{

"groupconvo”: {

"members" : [

{ "name" : "username001@wickr.com" },

{ "name" : "username002@wickr.com" }

]

}

}

Figure 12: Create group conversation body JSON

The response will either be an error with a description of that error or a successful response with the vGroupID of the newly created group conversation. The following is an example of a successful response:

{

"vgroupid": "S0b503ae14cc896aad758ce48f63ac5fae0adccd78ef18cde82563c63b2c7761"

}

Figure 13: Create group conversation response JSON

## Get Group Conversations API

This API will return a list of group conversations that are known by the WickrIO client. To get a list of group conversations send an HTTP GET to the following URI:

https://<host>:<port>/Apps/<API Key>/GroupConvo

The WickrIO client will respond with a JSON array of the group conversations. The format of the response will look like the following:

{

"groupconvos”: [

{

"members”: [

{ "name" : "username001" },

{ "name" : "username002" }

],

"ttl”: "7776000",

"bor”: "0",

"vgroupid”: "S00bf0ca3169bb9e7c3eba13b767bd10fcc8f41a3e34e5c54dab8bflkjdfde"

}

]

}

Figure 14: Get group conversations response JSON

## Get Group Conversation API

This API will return details of a specific group conversation. Send an HTTP GET using the following URI:

https://<host>:<port>/Apps/<API Key>/GroupConvo /<vGroupID>

The WickrIO will respond with a JSON structure containing information for the specified conversation. The format of the response will look like the following:

{

"rooms”: [

{

"members”: [

{ "name" : "username001" },

{ "name" : "username002" }

],

"vgroupid”: "S00bf0ca3169bb9e7c3eba13b767bd10fcc8f41a3e34e5c54dab8bflkjdfde"

}

]

}

Figure 15: Get group conversation response JSON

## Delete Group Conversation API

In order to delete a group conversation, you will need to have the vGroupID associated with that conversation. You can use the get group conversations API to get the list of conversations known by the WickrIO client, then determine which conversation to delete. Also, saving the vGroupID returned from the create group conversation API can be used as well.

To delete a group conversation, send an HTTP DELETE command using the following URI:

https://<host>:<port>/Apps/<API Key>/GroupConvo/< vGroupID >

The group conversation with the same vGroupID will be deleted.