Wickr Enterprise Compliance Service

Installation, Maintenance, and Examples

Version 4.0, May 11 2022

Table of Contents

1. Introduction

The Wickr Compliance Service allows a network administrator to record communications within a single network. Multiple bots can be setup and configured to capture messages and files across multiple networks. In the current iteration you need one bot per network, and it **must** be done at network creation. Bots cannot be added to existing networks at this time.

2. Installation

The Wickr Compliance Service requires the Wickr IO platform as it is now an integration within that framework. Wickr IO manages the server-side portions of the setup process while the Network Dashboard in your Enterprise deploy manages the bot users.

INFO: You can complete the server-side setup before completing the Base Deploy.

2.1. Server Setup

The following table describes the recommended server resources.

Table 1. Server Requirements

Resource	Recommendation
OS	Ubuntu or CentOS 7/8
CPU	2+ Cores
RAM	8GB+
Disk Space	100GB+
Ports	**TCP**: 443, Egress

Disk space is flexible and depends on the number of users and traffic.

If you're not rotating the collected information, we'd suggest 5-10GB per user to start. If you're exporting the information to another source besides syslog please use your best judgement after reviewing the usage on your external system.

2.1.1. Compliance Dependencies

The only requirement for Compliance is Docker. The following commands will install the Docker repository and Docker, give the `ubuntu` user permissions to interact with those services, and create a local directory to save data.


```
sudo apt install \
  ca-certificates \
  curl \
  gnupg \
  lsb-release

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg

sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"

sudo apt update

sudo apt install docker-ce

sudo systemctl enable docker
sudo systemctl start docker

sudo usermod -aG docker ubuntu ①

sudo mkdir /opt/WickrIO
```

① Replace with your own username if different.

The instructions to install Docker on a CentOS server are here: Docker CE Install

2.1.2. Wickr IO Docker Image

WickrIO is publicly listed on DockerHub named `wickr/bot-enterprise`. You can pull this using the following command:

```
$ docker pull wickr/bot-enterprise:5.92.07.01
```

2.1.3. Starting Wickr IO

The following command will start the Wickr IO container in the background with a persistent volume and will restart if the process dies. The first time Wickr IO is launched you'll be presented with a license agreement for this bot. You'll also be shown a quick start for the broadcast bot which can be ignored.

If this is an upgrade to an existing Compliance install, please see the [Compliance Upgrades] section! Only follow the below sections if this is a brand new compliance bot.

```
$ docker run -v /opt/WickrIO:/opt/WickrIO -d --name wickr-compliance --restart=always
-ti wickr/bot-enterprise:5.92.07.01
```

The compliance service can be used with a proxy by adding flags to the run command above. Add the following environment variables to route appropriately:

```
$ docker run -v /opt/WickrIO:/opt/WickrIO -d --name wickr-compliance --restart=always
-e http_proxy=http://proxy:port -e https_proxy=https://proxy:port -ti wickr/bot-
enterprise:5.92.07.01
```

Once this is running in the background you can attach to the container and begin:

```
$ docker ps
CONTAINER ID   IMAGE                              COMMAND        CREATED      STATUS
11d5626f74b9   wickr/bot-enterprise:5.92.07.01   "./start.sh"   2 days ago   Up 2 days

$ docker attach wickr-compliance
Enter command:
```

To exit the foreground mode use the following key combination: `Ctrl + P` and then `Ctrl + Q`.

3. Initial Network Configuration

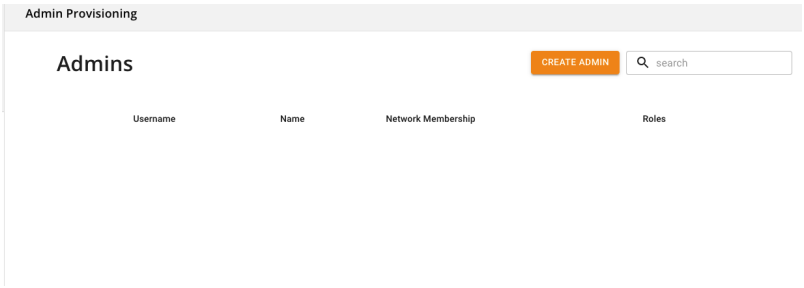
Once the `bot-enterprise` container is running and have a working deployment of Wickr Enterprise you can begin adding a bot user and completing the setup process. Setup is comprised of four steps:

- Adding additional administrators
- Naming the network
- Adding a compliance bot
- Adding users

3.1. Network Dashboard Setup

3.1.1. Welcome to Wickr

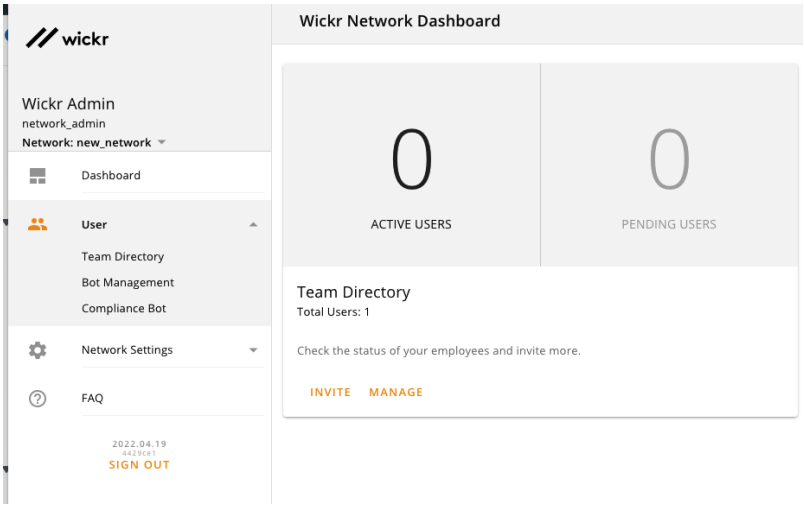
When you first log into Wickr Enterprise you'll see a screen similar to the one below:



Click `Create Admin` to create a network admin account with a new network.

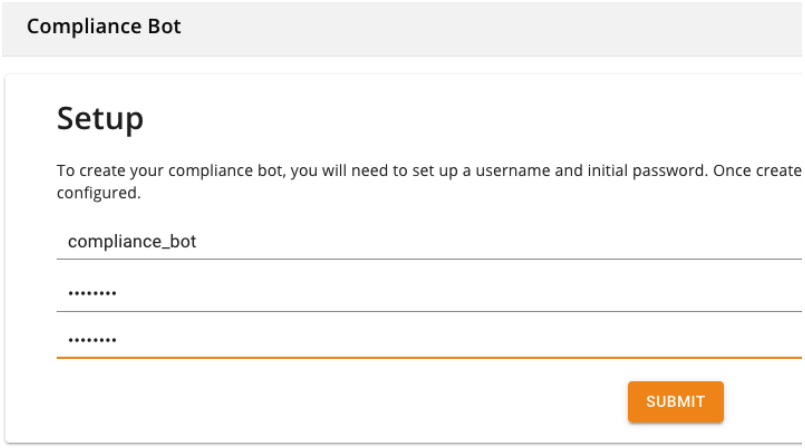
3.1.2. Using Network Admin

Once your network admin account has been created, click `Sign Out` and then sign in under the new account.



Once logged in, select `User` - `Compliance Bot`

3.1.3. Creating a Compliance Bot User



You'll need to enter the following:

- Username

This username is unique for each bot. We recommend matching the network name in some fashion in the event you use multiple bots.

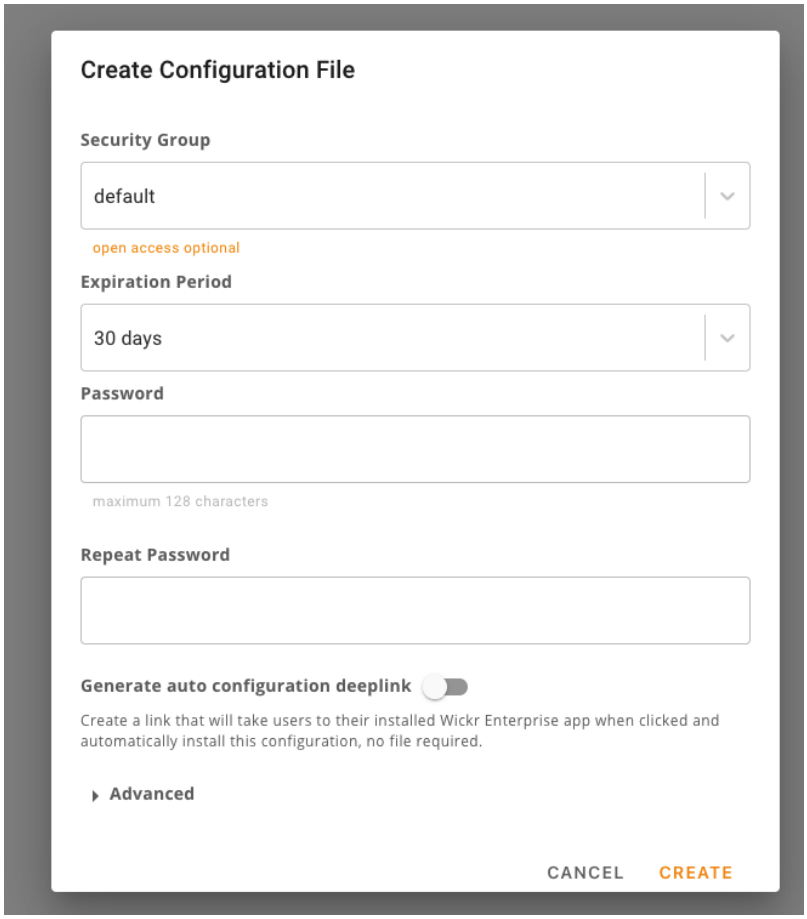
- Initial Password

  This password is only used once during the bot setup process, so it's safe to make this generic.

Click `Submit` once ready.

3.1.4. Configuration File

Now that you have a bot user, it's time to create the configuration file it will use to connect to your Wickr Enterprise deployment. Click `Create New Config`



On this page you'll need to enter another password used solely for the configuration file and an expiration time. The expiration time should be low, only allowing enough time to go through the compliance setup process. `1 day` is usually sufficient.

Click `Create`, then `Download Configuration File` to download the `conf.wickr` file.

3.2. Server-side Bot Setup

Once you have the `conf.wickr` file, you'll need to upload it to the compliance server to begin the server setup.

An example setup is below, with annotations underneath:

Compliance Setup


```
$ sudo mv conf.wickr /opt/WickrIO
$ docker attach wickr-compliance

Enter command: add
Enter the user name: compliancebot ①
Enter the password: ******** ②
Enter config file: /opt/WickrIO/conf.wickr
Enter the config file password: ******** ③
Creating user:  "compliancebot"

**********************************************************************
**** GENERATED PASSWORD
**** DO NOT LOSE THIS PASSWORD, YOU WILL NEED TO ENTER IT EVERY TIME
**** TO START THE BOT
****
 "Ft1MAC0FUaE1DiS67J8WN8nt" ④
**********************************************************************
Begin registration with password.

Successfully created user
**********************************************************************
**** USER SIGNING KEY
**** You will need this to enter into the console for the Bot
****
"00040002f9abe40c27c8b29c14cfa014db29c02c53b72bcb11b5e6657414b89101574f531c772fae84966
0defc4dddb3115eb34acba654f8d5eda00edbcb75da08f398d901e4a2f95cdd9f519f236b3f541b6af282f
ac0d89f3524da0cb79d202f37212ec8b08ab319e84da6c0e6214205b95cbff43a7dc3a1ba5e30e434a117b
884f51354fc" ⑤
**********************************************************************
Successfully logged in as new user!
Our work is done here, logging off!
Return code from provision is: 0

The autologin capability allows you to start a bot without having to enter the
password, after the initial login.
NOTE: The bot client's password is NOT saved to disk.

Do you want to use autologin? (default: yes):yes ⑥

These integrations are local:
  - wickrio-calendar-bot
  - wickrio-example-app
  - wickrio-zendesk-bot
  - wickrio-file-bot
  - wickrio-compliance-bot ⑦
  - wickrio_web_interface
  - wickrio-broadcast-bot
  - core_bot
  - wickrio-monitor-bot
  - hubot
  - wickrio-user-engagement-bot
```

```
  - wickrio-hello-world-bot

These integrations are from the NPM registry:
  - wickrio-alias-bot

Please enter one of:
  - The full integration name from the list above
  - The word "search" to search the NPM registry for an integration
  - The word "import" to import an integration
  - The word "quit" to cancel adding the bot

Enter the bot integration to use:wickrio-compliance-bot
************************************************************************
Begin setup of wickrio-compliance-bot software for compliance-bot
Copying wickrio-compliance-bot software

Installing wickrio-compliance-bot software
Installing
Installing
```

① This is the username from Creating a Compliance Bot User

② This is the password from Creating a Compliance Bot User

③ This is the password from Configuration File

④ This is the password the bot will use going forward. It's very important to save this!

⑤ This is the text to copy into the bottom empty space in Configuration File

⑥ This will have your compliance bot user login automatically if disconnected.

⑦ If this isn't the default option, please enter `wickrio-compliance-bot`.

3.2.1. Configuring the Compliance Integration

As of version 5.56, the Compliance integration has these extra options:

- Logs saved to alternate locations
- Output file prefix
- Size based log rotation
- Attachments saved to alternate location
- Automatic service restart

Compliance Configuration Options

```
Begin configuration of wickrio-compliance-bot software for compliance-bot
 Use new file save process [yes/no]: (no) :yes ①
 Please specify the directory to save message data: (/opt/WickrIO/clients/compliance-
bot/integration/wickrio-compliance-bot/messages) : ②
 Please add a prefix for message data files: (receivedMessages) : ③
 Please enter the maximum size in bytes for each messages file [1GB = 1073741824]:
(1073741824) : ④
 Please specify the directory to save attachment data:
(/opt/WickrIO/clients/compliance-bot/integration/wickrio-compliance-bot/attachments) :
⑤
 Memory can reach 100% if the bot isn't restarted periodically. Please enter a time in
minutes to restart the service [24hrs = 1440]: (1440) : ⑥
Finished Configuring!

Integration files written to:
/opt/WickrIO/clients/compliance-bot/integration/wickrio-compliance-bot ⑦

End of setup of wickrio-compliance-bot software for compliance-bot
**********************************************************************
Successfully added record to the database!
Enter command:
```

① Entering `No` here will use the previous settings and options

② The directory specified here must be writable and in the current mount point

③ This sets a prefix on log files that are rotated

④ This sets the size limit on log files

⑤ The directory specified here must be writable and in the current mount point

⑥ The integration will restart at a regular interval set here.

⑦ This directory will house all configuration and necessary files for the compliance service

3.2.2. Starting the compliance bot

All that's left is starting the bot you just configured! Use the `start` command as described here:

```
Enter command: start
Do you really want to start the client with the name compliancebot: y
Enter password for this client: ********** ①
```

① This is the password highlighted as #4 in the section above!

3.2.3. Other commands

The other available commands besides `add` and `start` are below:

- `list`

The list command will show all active or inactive bots.

- `delete <bot number>`

  Delete will let you remove a bot from your system. If you have more than one, use the number shown from the `list` command to specify which you'd like to delete.

- `pause <bot number>`

  Pause will let you temporarily stop a running bot.

- `restart <bot number>`

  Restart will let you restart a bot.

4. Compliance Data Location

The compliance bot will save messages and output by default to:

`/opt/WickrIO/clients/<bot_name>/integrations/compliance_bot/receivedMessages.log`

Where `<bot_name>` is the username entered during Creating a Compliance Bot User.

5. Compliance Container Upgrades

If already running the compliance bot within a container, the upgrade process is very simple and will not result in any downtime. Any messages received while the bot is offline will be downloaded once the new version is up and running.

The first step is to pause the compliance bot:

```
$ docker attach wickr-compliance

Continue to see welcome message on startup? (default: yes):
Current list of clients:
 client[0] compliance, State=Running, Integration=compliance_bot ①

pause 0
```

To exit the foreground mode use the following key combination: `Ctrl + P` and then `Ctrl + Q`.

Next is to stop the current container:

```
$ docker stop wickr-compliance
```

From here you'll need to rename (or remove) the old container. Renaming is safest if you decide you need to roll back, but removing the container will save on disk space in the long run.


```
$ docker rename wickr-compliance old-compliance-bot
```

OR

```
$ docker rm wickr-compliance
```

Once the old container is out of the way you can start the new one:

```
$ docker run -v /opt/WickrIO:/opt/WickrIO -d --name wickr-compliance --restart=always
-ti wickr/bot-enterprise:5.92.07.01
```

Now you can attach to continue the upgrade. Rename the bot to match the newer integration:

```
$ docker attach wickr-compliance

Continue to see welcome message on startup? (default: yes):
Current list of clients:
 client[0] compliance, State=Paused, Integration=compliance_bot

Enter command:rename 0
Enter the new name:wickrio-compliance-bot
Confirm that you want to change the integration name
From compliance_bot to wickrio-compliance-bot
Do you want change the integration name? (default: yes):
```

Once the integration has been renamed it can be upgraded from NPM:

```
Enter command:list
Current list of clients:
#  Name                        Status  Integration            Version   Events
Misc
================================================================================
=
0  wickr_entbeta_compliance_bot  Paused  wickrio-compliance-bot  0.1000.0  2357
Needs Upgrade!
```

From here we'll upgrade the integration code:

```
Enter command:upgrade 0
Upgrading from version 0.1000.0 to version 5.44.9
Okay to proceed? (default: yes):yes
Searching NPM registry
Searching NPM registry
Searching NPM registry
Searching NPM registry
Searching NPM registry
Searching NPM registry
Copying wickrio-compliance-bot from the NPM registry
Upgrading wickrio-compliance-bot software
Installing wickrio-compliance-bot software
Installing
Begin configuration of wickrio-compliance-bot software
```

At this point the bot is ready to go! Use the `list` command to make sure the bot is running. If it isn't you can start it:

```
Current list of clients:
 client[0] compliance, State=Paused, Integration=compliance_bot

start 0
```

Now the log can be tailed to make sure new messages are captured.

```
$ tail /opt/WickrIO/clients/compliance/integration/wickrio-compliance-
bot/receivedMessages.log

{"id":"d59ce4c0e16c11e98dd1cb32338ed079","message":"test","msg_ts":"1569619316.805649"
,"msgtype":1000,"receiver":"user1","sender":"user3","time":"9/22/19 9:21 PM"
,"vgroupid":"083983510e793950fabd97774979089ed550b02c00b63f8c52bc525c4afbb9a4"}
```

If you see new messages you're good to go! If you're not seeing new messages you'll need to make sure your bot is in the `Running` state. You may need to stop and start the bot again to provide the password. If you see any issues please don't hesitate to reach out to Wickr Support!

If your bot is offline it will still keep the messages server-side. It will download any new messages once it's back up and running, meaning no messages will be lost.

Appendix A: Compliance Message Description

Table 2. Message Fields

Field	Description
id	A unique id for each message
msg_ts	Timestamp in microseconds based on server time

Field	Description
msgtype	Identifies the type of message
sender	The Wickr ID of the sender
receiver	The Wickr ID of the recipient
time	Human readable time the message was sent
vgroupid	Identifies the conversation the message was sent in

The `msgtype` value will describe the type of message being sent:

Message Type	msgtype value
Text message	1000
File transfer	6000
Verification message	3000
Calling message	7000
Location	8000
Edit Message	9000
Create room	4001
Modify room members	4002
Leave room	4003
Modify room parameters	4004
Delete room	4005
Delete message	4011
Message Attributes Msg	4012

Appendix B: Compliance Output Examples

B.1. Normal Messages

B.1.1. Text message

```
{
  "message": "test",
  "message_id": "fe280cc0955411ea87a737e4d348fad3",
  "msg_ts": "1589400286.204925",
  "msgtype": 1000,
  "receiver": "user001",
  "sender": "user004",
  "time": "5/10/20 8:04 PM",
  "vgroupid": "9f57b455651f910edb2f2174b8b3a0f6079d0c1f5c0300f89c950fff7b024191"
}
```


B.1.2. File Transfer


```
{
  "file": {
    "filename": "wickr_logs.tar.gz",
    "guid": "A4F70D32-893C-41B6-9345-757386D5B301",
    "localfilename":
"/opt/WickrIO/clients/compliancebot/attachments/attachment_20200513200545427_wickr_log
s.tar.gz"
  },
  "message_id": "216db040955511eaabef733b28863703",
  "msg_ts": "1589400345.381621",
  "msgtype": 6000,
  "receiver": "user002",
  "sender": "user007",
  "time": "5/10/20 8:05 PM",
  "vgroupid": "9f57b455651f910edb2f2174b8b3a0f6079d0c1f5c0300f89c950fff7b024191"
}
```


B.1.3. Verification message


```
{
    "id":"4d5970e0fd6b11e8aa01ed4074b0abb4",
    "keyverify":
    {
        "hash":"87c6303a4ea684e2f696116406d5fd6a00a0a2ec4da147069e3d42844
⋯.f246a5039f7d8fa77a0",
        "key":"0062f469e842f29e52e2ce0220da1b54e13429247ec31eb57a28ceb6a4e10d8f47",
        "msgtype":1
    },
    "msg_ts":"1544549793.583461",
    "msgtype":3000,
    "sender":"user004",
    "time": "5/10/20 8:07 PM",
    "vgroupid":"5d3255d71800f350a41267c7cd0edd8c8dec35dcd385dd3619f48eb22c9393d2"
}
```


Verification messages have an additional nested `msgtype`:

Verification Message Type	msgtype value
Verification request	1
Verification response *and* request	2
Verification acceptance	3
Verification rejection	4
"Not Now" response	5

B.2. Control Messages

Control messages are used to setup and configure both Rooms and Conversations. The messages are required to re-construct the list of users involved in specific Rooms and Conversations.

Like Verification, these messages have additional metadata to describe what they did. The `control` object can contain:

- A `bor` field. This is the Burn on Read time in seconds.
- A `ttl` field. This is the Expiration time in seconds.
- A `changemask` field is a number value created from adding the following flag values:

Message Type	`msgtype` value
Masters field (Moderators)	1
TTL (Expiration)	2
Title field (Room name)	4
Description	8
Meeting ID Key	16
Burn on Read	32

In the example below you'll see a `changemask` value of 47. This is equal to the sum of Masters (1), TTL (2), Title (4), Description (8), and Burn on Read (32).

B.2.1. Create Room message

```
{
    "control":{
        "bor":0,
        "changemask":47,
        "description":"",
        "masters":["user001", "user002"],
        "members":["user001", "user002", "user003"],
        "msgtype":4001,
        "title":"Creating a room",
        "ttl":2592000
    },
    "id":"be452b00f89711e883588d1e7a946847",
    "msg_ts":"1544019125.75323",
    "msgtype":4001,
    "sender":"user002",
    "time": "5/10/20 6:17 PM",
    "vgroupid":"S58a15186365d2125a9b417e71b99bcb29e3770078e157e953cfbe28443eb750"
}
```


B.2.2. Modify Room Members message

The `addeduser` and `deletedusers` array will show who was added and removed.


```
{
    "control":{
        "addedusers":[],
        "deletedusers":["testuser"],
        "msgtype":4002
    },
    "id":"d34058a0f89711e88760d7c8037ea946",
    "msg_ts":"1544019160.275884",
    "msgtype":4002,
    "sender":"user002",
    "time": "5/09/20 3:22 PM",
    "vgroupid":"S58a15186365d2125a9b417e71b99bcb29e3770078e157e953cfbe28443eb750"
}
```


B.2.3. Leave Room message

The `sender` in a Leave Message will show who left.


```
{
    "id":"f660fe80f89711e887d86d198d2ef374",
    "msg_ts":"1544019219.210107",
    "msgtype":4003,
    "sender":"user002",
    "time":"5/9/20 6:14 PM",
    "vgroupid":"S58a15186365d2125a9b417e71b99bcb29e3770078e157e953cfbe28443eb750"
}
```


B.2.4. Modify Room Parameters message


```
{
    "control":{
        "bor":0,
        "changemask":47,
        "description":"change description",
        "masters":["user001"],
        "members":["user001","user002"],
        "msgtype":4004,
        "title":"Creating a room",
        "ttl":2592000
    },
    "id":"db805750f89711e8a01ab328ac0b2f04",
    "msg_ts":"1544019174.117057",
    "msgtype":4004,
    "sender":"user002",
    "time":"5/11/20 4:53 PM",
    "vgroupid":"S58a15186365d2125a9b417e71b99bcb29e3770078e157e953cfbe28443eb750"
}
```


B.2.5. Modify Saved Item in Room

```
{
    "control":{
        "bor":0,
        "changemask":64,
        "description":"",
        "filevaultinfo":{
            "filehash":"4eab763c5f3211ca93966a···d3e461c27f5432c1",
            "guid":"D094F147-0490-4A14-9A65-6F23C896A8B4",
            "key":"00f8058d88e6b7520849bbfd8e6b5cc12d35a70c856dde44d64e2e331fc50ce700"
        },
        "masters":["user002","user001"],
        "members":["user002","user001"],
        "msgtype":4004,
        "title":"Test",
        "ttl":2592000
    },
    "message_id":"c3cb62e08dff11eab641a549111a64cb",
    "msg_ts":"1588594022.928361",
    "msgtype":4004,
    "sender":"user001",
    "time":"5/4/20 5:07 AM",
    "vgroupid":"S243f2ec645d3961bdd531f51f3244205d292b8d0fbd41802827746271d31d41"
}
```


B.2.6. Delete Room message


```
{
    "id":"06364710f89811e899418b6723464a0c",
    "msg_ts":"1544019245.773676",
    "msgtype":4005,
    "sender":"user002",
    "time":"5/7/20 4:33 PM",
    "vgroupid":"S7879eb406958d83b991a5f2acb29e5ad8565a4faa41e1c5cbd7004c5586ddd5"
}
```


B.2.7. Delete or Recall Message

The `isrecall` field will show if the message was deleted or recalled.


```
{
    "control":{
        "isrecall":false,
        "msgid":"4ab537d0f85d11e88c2225680208f9ff",
        "msgtype":4011
    },
    "id":"bea7ad10f89811e8822887c76561d99d",
    "msg_ts":"1544019555.217633",
    "msgtype":4011,
    "sender":"user002",
    "time":"5/10/20 5:11 PM",
    "vgroupid":"3f13df0d8f267812d7e743a518fcfb6dacf6fd0824e16a83a4d2a06d32cf8d9c"
}
```


B.2.8. Message Attribute Change

These control messages show when a message is starred or unstarred.


```
{
    "control":{
        "attributes":[
            {
                "isstarred":true,
                "msgid":"93cf81608bbd11ea9a16b7554e31eed2"
            }
        ],
        "msgtype":4012
    },
    "message_id":"b0d284d08be311eaaf7f51713016a94c",
    "msg_ts":"1588362062.864376",
    "msgtype":4012,
    "receiver":"user123",
    "sender":"user100",
    "time":"5/1/20 12:41 PM",
    "vgroupid":"4ebf561eb2214c4e6f924d09e37bf80b6f9b85cb96b72badb03753d9ed26f7f4"
}
```


B.3. Calling messages

Calls can have four `status` values:

Call Status	`status` **value**
Call Starting	0
Call Completed	1
Call Missed	2

Call Status	`status` value
Call Cancelled	3

B.3.1. Call Start

Call Start Example


```
{
    "call":{
        "calluri":"3.86.149.242:16398",
        "calluriipv6":"[2600:1f18:2741:9e01:e0cb:6847:bb1d:415d]:16398",
        "meetingid":0,
        "participants":[
            "8995950dad747c24fd7c9e2da68edeb6c2c0ac6cb96d405c0c4c58e09ff46969",
            "ae182b20ce36a94c613af5a2964bbd616de662d3f8487dc39fa9400e739a3049"
        ],
        "status":0,
        "version":2,
        "versioncheck":true
    },
    "message_id":"4d421aa08be811eaaf7a493af2765a5b",
    "msg_ts":"1588364043.307106",
    "msgtype":7000,
    "receiver":"user5",
    "sender":"user100",
    "time":"5/1/20 1:14 PM",
    "vgroupid":"4ebf561eb2214c4e6f924d09e37bf80b6f9b85cb96b72badb03753d9ed26f7f4"
}
```


The `participants` array will have the `username_hash` of the users that were on the call. You can get the plaintext usernames from the mysql database with the following query.

```
select username from user_enterprise where username_hash =
'b64e2aa1c3c9edb74f31079c579b5feb22300e6966ac6c1721fd9e4dcfca4dd8'\G;
```

B.3.2. Invite to Call

When a user is invited to an active call it will appear as a new call started. You can use the `invitemsgid` field to match invites to the original call.

```
{
    "call":{
        "calluri":"18.234.76.29:16504",
        "calluriipv6":"[2600:1f18:2741:9e00:3fba:154:6431:df8e]:16504",
        "invitemsgid":"87b5e4a095ef11ea99b03bd5dd4eaa9d",
        "meetingid":0,
        "participants":[
            "ae182b20ce36a94c613af5a2964bbd616de662d3f8487dc39fa9400e739a3049"
        ],
        "startmsgid":"87b5e4a095ef11ea99b03bd5dd4eaa9d",
        "status":0,
        "version":2,
        "versioncheck":true
    },
    "message_id":"ce22f14095ef11eaa1cbdf2a74e88e51",
    "msg_ts":"1589466777.633896",
    "msgtype":7000,
    "receiver":"paulcomptst100",
    "sender":"paulcomptst101",
    "time":"5/14/20 2:32 PM",
    "vgroupid":"7f1a06a35243584436f6df7170e0fc8a784022a66b5e5d168b419b14cecdcdad"
}
```


B.3.3. Call End

Call End Example


```
{
    "call":{
        "duration":38,
        "meetingid":1,
        "startmsgid":"72c9a3b08bea11eab9078f7ca8a1b909",
        "status":1,
        "version":2,
        "versioncheck":true
    },
    "message_id":"8b9d94408bea11ea89aea91887dfff41",
    "msg_ts":"1588365006.918849",
    "msgtype":7000,
    "receiver":"user005",
    "sender":"user100",
    "time":"5/1/20 1:30 PM",
    "vgroupid":"4ebf561eb2214c4e6f924d09e37bf80b6f9b85cb96b72badb03753d9ed26f7f4"
}
```


B.4. Location Messages

There are two types of location messages. The first example is of a static share.

B.4.1. Static Location Share


```
{
    "location":{
        "latitude":40.75017899435506,
        "longitude":-74.99449803034105
    },
    "message_id":"1f88fdc08bec11ea81b689d23fa72c7b",
    "msg_ts":"1588365684.583407",
    "msgtype":8000,
    "receiver":"user003",
    "sender":"user100",
    "time":"5/1/20 8:41 PM",
    "vgroupid":"4ebf561eb2214c4e6f924d09e37bf80b6f9b85cb96b72badb03753d9ed26f7f4"
}
```


B.4.2. Share Location Continuously

This second example is when a user shares their location for a period of time. The `edit` section will show what changed.


```
{
    "edit":{
        "type":"location",
        "shareexpiriation":"",
        "latitude":40.75017899435506,
        "longitude":-74.99449803034105
    },
    "message_id":"1f88fdc08bec11ea81b689d23fa72c7b",
    "msg_ts":"1588365684.583407",
    "msgtype":9000,
    "receiver":"user003",
    "sender":"user100",
    "time":"5/1/20 8:41 PM",
    "vgroupid":"4ebf561eb2214c4e6f924d09e37bf80b6f9b85cb96b72badb03753d9ed26f7f4"
}
```


B.5. Link Previews

When a user shares a link and has "Link Previews" enabled, it will show that here:


```
{
    "edit":{
        "originalmessageid":"11457fa08da211ea881baffab0b42745",
        "text":"https://howdoyoudo.com",
        "type":"text"
    },
    "message_id":"1163e5b08da211eab775a5032a0322ca",
    "msg_ts":"1588553780.419871",
    "msgtype":9000,
    "sender":"user001",
    "time":"5/3/20 5:56 PM",
    "vgroupid":"S243f2ec645d3961bdd531f51f3244205d292b8d0fbd41802827746271d31d41"
}
```