Polisen

Tilläggsprotokoll

till 5000-K345268-21

Arkiv/Åkl. ex

Åklnr
AM-47157-21

Signerat av
Mattias Karlsson

Signerat datum
2021-09-16 11:57

Enhet
Polisregion Stockholm, Utredning 15 Rgn Sthlm

Handläggare (Protokollförare)
Mattias Karlsson

Datum
2021-09-16

Bitr. handläggare
Eva Söderström

Undersökningsledare
Joakim Hall

Polisens diarienummer
5000-K345268-21

Personer i ärendet

Förtursmål	Beslag	Målsägande vill bli underrättad om tidpunkt för huvudförhandlingen
Annat förtursmål		Nej
Ersättningsyrkanden		**Tolk krävs**

Notering

Innehållsförteckning

Operation Trojan Shield
Technical Details

Last Updated: August 31, 2021

TABLE OF CONTENTS

1. OVERVIEW OF SERVERS AND NETWORKING

Operation Trojan Shield (OTS) required servers at several levels to provide data transfer, storage, and review.

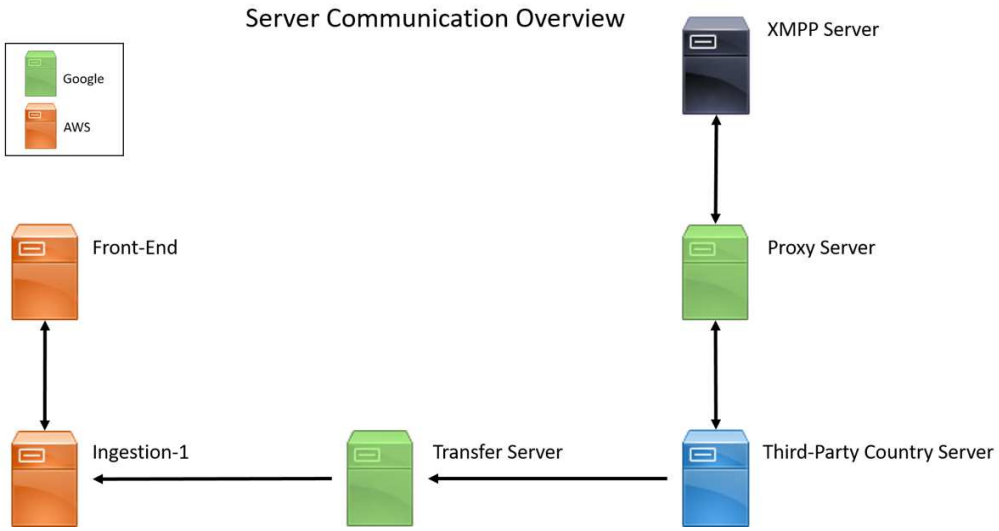


Diagram 1.A

1.1 XMPP Server

XMPP is an open standard protocol for instant messaging, which can facilitate multi-party chats, supporting multimedia such as voice, images, and video. Anyone is free to run their own XMPP server and network and build upon the framework that is provided. The ANØM app included a feature whereby when a message was sent, the app, via XMPP server, automatically sent a 'blind carbon copy' of the message to a ghost user ID, 'bot'. This process was not visible to the sending user, nor were they aware that this process occurred.

1.2 Third-Party Country Server

A server was provisioned by law enforcement from a third-party country in October of 2019 for collecting the blind carbon copy messages sent to the ‘bot’ ghost user as referenced in Appendix A.8 - 59. This server was owned, operated, and maintained by the third-party country until the end of the operation.

1.3 Google Servers

A covert Google Cloud account was provisioned for the creation of Google Compute Engines, a service that Google Cloud provides to create and run virtual machines within Google Cloud, needed for transferring data and providing a proxy server for the third-party country's server.

1.3.1 Transfer

The transfer server was a configured Google Compute Engine. Its only purpose was to transfer data received from the third-party country's server, into the Ingestion-1 (I1) server within AWS GovCloud.

1.3.2 Proxy

The proxy server was provisioned as a Google Compute Engine. The server acted as a relay of all messages between the ANØM network (XMPP server) and the third-party country's collection server. This provided private communications to the ANØM network from the third-party server.

1.4 AWS GovCloud

Within AWS GovCloud West, two Elastic Compute Cloud (EC2) servers were provisioned, operated and maintained for OTS.

1.4.1 Security

The AWS environment is scanned on a monthly basis for vulnerabilities and other misconfigurations that could pose as threats. Each EC2 server within AWS GovCloud runs a hardened version of the CentOS operating system. Both EC2 servers are also scanned and patched for vulnerabilities. The vulnerability scanning and patching is performed in accordance with Security Technical Implementation Guides (STIGs) that the Defense Information Systems Agency (DISA) publishes.

1.4.2 Front-End

The front-end server was designated as the web server for the Hola iBot review application. More details on the Hola iBot review application can be found in section 10.

1.4.3 Ingestion-1 (I1)

Ingestion-1 (I1) was the destination of new data packages from the third-party country's server for processing and review. It also operates as a database and file server for the display and review of data.

1.5 Messaging Network Activity

As mentioned in Appendix A.8 - 59, the data collected from the platform was received as "blind carbon copy" messages. Diagram 1.B shows an overview of the networking data flow for the collection of these messages.

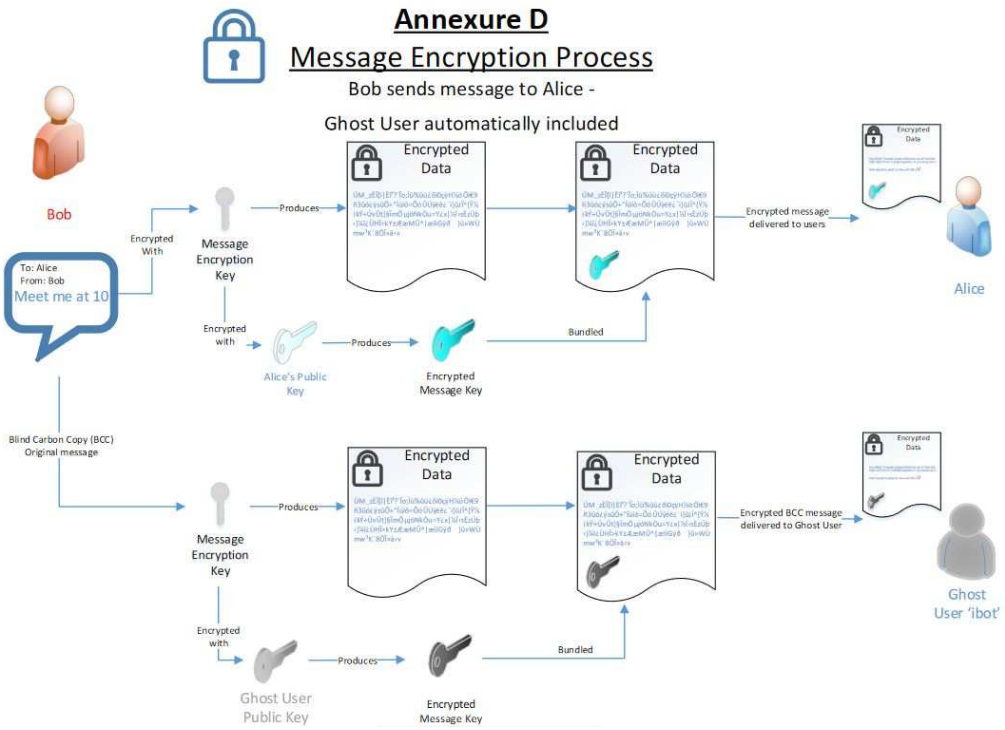


Diagram 1.B

2. NEW DATA PACKAGE CREATION

A mutual legal assistance treaty (MLAT) allowed for transfer of data every Monday, Wednesday, and Friday from the third-party country. A process to obtain only new data was developed and provided for use by the third-party country. The following is a list of data that was processed into the new data packages to be sent:

- MySQL Database Items:
  - Messages
    - A Unique Message ID
      - UUID that uniquely identifies an individual message
    - Sender
      - The User ID (JID) of the sending user
    - Receiver
      - The User ID (JID) or Group ID (GID) of the receiving entity
    - Location information
      - Latitude and Longitude would be attached to each message (if the device provides it).
      - If location information is provided, it will be encrypted.
    - Mobile Country Code (MCC)
      - The MCC that the message was sent from
    - Time
      - The time the message was sent according to section 8.6.1

        - Audio Pitch Adjustment
            - If the message was a Push-To-Talk (ptt) message, a value was sent along with the attachment that provides a way to reverse the pitch adjustment performed on the app as referenced in Appendix A.7.1.4 – 57)
        - Type
            - Each message could be one of the following:
                - text, forwarded, contact
                    - Designates that the 'content' field will contain encrypted text
                - ptt, video, image, note
                    - Designates that the 'content' field will contain the path of the file sent by the sender with the file name [Unique Message ID].[attachment extension]
        - Content
            - If the message type was 'text', 'contact', or 'forwarded', the content was encrypted
            - If the message type was 'ptt', 'video', 'image', or 'note', the field had the plaintext path of the attachment sent
    - Roster
        - Device User ID (JID)
            - A unique identifier for a device user
        - Nickname
            - Encrypted, user-entered nickname
    - Group Information
        - Group ID (GID)
            - A unique identifier for a group of devices
        - Group Name
            - Encrypted, user-entered name of the group
- Files
    - Audio, Image, Video, and Note files were named as the following: [Unique Message ID].[attachment extension]

2.1 Encryption of Data

The data received (attachments, content/text messages, location information, nicknames, and group names) was encrypted using the end-to-end encryption scheme as described in the Extensible Messaging and Presence Protocol (XMPP). On the third-party country's server, the data was decrypted in temporary storage (RAM) upon receipt, and the data was encrypted again using a combination of RSA asymmetric encryption and AES symmetric encryption before being written to disk. No plaintext data was stored on the third-party country's server. The private key for decrypting (in regards to RSA) was stored on the I1 server within AWS GovCloud.

2.2 How the Difference-Dump of the Database was Created

The process to obtain new data included finding only new data in the MySQL database (see above for list of entities obtained from the database) between the last full database dump and the new/current full database dump, internally called a "difference-dump". The process started by creating a new MySQL database dump. A program used the database dump taken from the previous transfer and the new database dump as input. The program performed a difference between the previous database dump

and the current, which generated a MySQL dump with only new data. Reference Diagram 2.A for a visual representation of the process.

2.3 How the List of Attachments is Determined Based on the Difference-Dump

The process used the difference-dump to gather the new attachments by referencing the 'content' field of all messages that have paths. See section 2 for more information about where paths are stored as entities. As mentioned previously, the file names were placed in the 'content' field if the type was not 'text', 'contact', or 'forwarded'.

2.4 How Attachments were Copied from File Storage

Attachments were stored in an encrypted state directly on the server, and the program that retrieved the new attachments utilized the path from the 'content' field as referenced in section 2.3, and copied the attachments to a temporary storage location for packaging.

2.5 Creation of the Encrypted Package

In order to package the MySQL dump with new data and attachments, the files were compressed into an archive file with the extension '.tar.gz'. The archive file was encrypted with GnuPG (GPG) asymmetric encryption, which appended the extension '.gpg' to the end of the archive. After the encrypted package was properly constructed, it was hashed using the MD5 hashing algorithm. A hash function is a mathematical process whereby the data input, or 'message', is processed into a 'message digest' or hash value of fixed length. It is commonly represented as hexadecimal characters which consist of the numbers 0 to 9, and letters A-F. Hash functions are often used for data verification to identify if data has been altered, due to the very high improbability that two different input 'messages' will result in the same hash value. The MD5 hash was saved to a file to be transferred. Diagram 2.A demonstrates an overview of the packaging process.

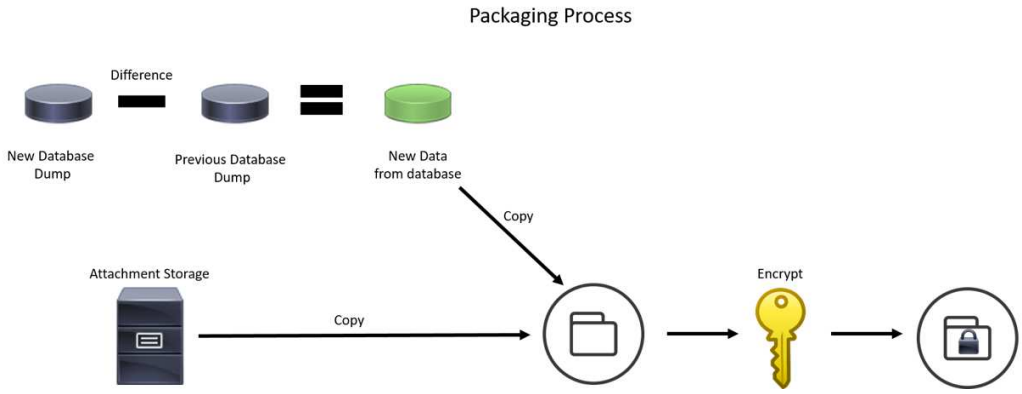


Diagram 2.A

3. TRANSFER FROM THIRD-PARTY COUNTRY SERVER TO TRANSFER SERVER

A program was manually run by the third-party country that performed the new-data processing as described in section 2.

3.1 Program that Transfers Data to Transfer Server

Each Monday, Wednesday, and Friday, a program was run by the third-party country that packaged and sent the new data. The program sent the MD5 hash, followed by the encrypted package of new data to the covert Google Compute Engine transfer server as referenced in section 1.3.1.

4. TRANSFER FROM TRANSFER SERVER TO AWS GOVCLOUD

The transfer server automatically forwarded the data received into the I1 server located in AWS GovCloud after checking the data integrity (verifying the hash).

4.1 Automated Program to Transfer Data

As the transfer server received new encrypted packages, it checked the inbox directory where data was uploaded for any files ending in '.gpg'. This indicated that a new package of data was ready to be transferred into I1 located in AWS GovCloud. The program then calculated the MD5 hash of the '.gpg' file, and it was compared to the result to the MD5 hash uploaded by the third-party country's server. If the hashes matched, this indicated a successful transfer, and the encrypted package was forwarded into I1 within AWS GovCloud.

5. ENCRYPTED PACKAGE PROCESSING

Each encrypted package was transferred to a designated location on I1 within AWS GovCloud. An automated task would check the location for new data. If new data was found in the location, the ingestion process would begin on the server.

5.1 Decrypt GPG Package

First, the encrypted package of MySQL data (database) and attachment files were decrypted so that the archive of data (database and message attachments) would be available. As mentioned before, GPG asymmetric encryption was used to protect the archive of data, so a private key, only available on the I1 server, was used to decrypt the package.

5.2 Extract New MySQL Database Dump and Attachments from Archive

After decryption, the archive was extracted to the file system, making the MySQL data (database) and attachments available for continued processing.

5.3 Ingest Database Dump Into Temporary Database

The I1 server utilized two databases for processing of incoming data. One of these databases, iBot_enc, was the temporary storage for new data received from the third-party country's server. This is where the new MySQL data was temporarily stored for processing. The new data needed to be decrypted, normalized, and processed into the other database so that the front-end web application, Hola iBot, had the ability to display the data.

6. DECRYPTION PROCESS

In order to decrypt and normalize the incoming data, the database dump was ingested into a database on the I1 server, as mentioned in section 5.3.

6.1 Decryption of Encrypted Fields Within Database

As mentioned in section 2, the following fields within the database dump were encrypted:

- Messages

    - Content (if type is ‘text’, ‘contact’, or ‘forwarded’)
    - Location information (if provided)
- Roster
    - Device user display name (reference section 8.3)
        - Also referred to as nickname, username, alias
- Group Info
    - Group name

The decryption process decrypted all non-attachment messages’ ‘content’ field in-place (replacing the encrypted content with the decrypted content), utilizing the private key associated with the encryption process followed by the third-party country’s server as mentioned in section 2.1.

6.1.1 Update Attachment Paths to be Correct

When receiving the encrypted attachment message files on the third-party country’s server, the entire plaintext path of the file was stored in the ‘content’ field of the database if the type was an attachment. This path was relative to the third-party country’s server, but this field in the database was used similarly on the I1 server to serve attachments for review on the front-end web server. In order to make the attachments accessible, the path in the ‘content’ field was updated to the path they would be placed on the I1 server following decryption and conversion (if needed).

6.2 Decryption of All Attachment Files

After the paths in the ‘content’ field were updated correctly, the processing code decrypted all new attachment files using the private key associated with the encryption process followed by the third-party country’s server as mentioned in section 2.1. The attachment decryption process placed the decrypted files in the correct location to be accessed by the front-end web server for view in the Hola iBot web application.

7. AUDIO FILE PROCESSING

The audio files received from the third-party country’s server needed additional processing for them to become available for listening within the Hola iBot web application.

7.1 Convert Audio Files to Usable Format

While the decryption process runs, if an attachment was an audio file (indicating a push-to-talk message), the audio file would be converted from the Ogg Opus (OPUS) File Format to a Waveform Audio (WAV) File Format. This is due to browsers not supporting OPUS files, while WAV files can be played natively in most modern web browsers.

7.2 Pitch Change Audio Files if Needed

As mentioned in Appendix A.7.1.4 – 57, the ANØM application provided the capability of adjusting the pitch of the message recorded. The user had the ability to shift the pitch of PTT messages up ('Helium’) or down (‘Jellyfish’). With each pitch-adjusted PTT message sent, a value that represented the adjusted frequency was also sent. This value was used to reverse the pitch, which would provide an adjusted audio file. While the pitch-adjusted audio file is made available to users of the Hola iBot review platform, the original audio file is also retained. While the original audio file is something that was retained, it's currently not available for retrieval. A process to retrieve the original audio files has not been developed.

8. DATABASE OVERVIEW

The remainder of the ingestion process after the processes described in sections 6 and 7 involves reorganizing the data into a secondary database (iBot_dec). The iBot_dec database is the database used by the front-end Hola iBot web application. The following section describes the layout of the primary iBot_dec tables, as well as relationships within. While not all tables within iBot_dec are described in this section, they will be introduced as needed in later sections.

8.1 Cases/Syndicates

The iBot_dec database's central component for organization is "cases" or "syndicates". Within the database, cases have a case ID and a name. The case ID is a unique identifier for the case, and the name provides a space for administrators to enter an easy-to-remember identifier for the case. Cases were predominantly named after randomly chosen Roman and Greek Gods and Goddesses. Within the database, there is a table named 'cases', where descriptive information is stored regarding cases.

8.1.1 Groupings of Device Users (JIDs)

Cases provide the means to organize device users (JIDs) that frequently talk to each other into groups. They also provide organization for review platform users.

8.2 Review Platform Users

Upon obtaining a CJIS Law Enforcement Enterprise Portal (LEEP) account, FBI employees and other individuals from law enforcement agencies monitoring data within the platform will need additional access permissions to be granted before accessing the review platform. These permissions are granted through the creation of an account that exists within the iBot_dec database.

Three main types of users exist within the database:

- General Users
- Super Users
- Admins

Within the iBot_dec database, there is a table named 'users', where descriptive information is stored regarding review platform users.

8.2.1 General Users

Users are FBI and other law enforcement agency investigative and support personnel working on Operation Trojan Shield investigations. For the specific investigation for which they have been granted access, they can access the products pages. Products are described more in the database in section 8.5. This is where they can review, mark, and submit notes on products. General Users are also able to access and use the features on the roster. Reference section 8.3 for more information about the roster.

8.2.3 Super Users

Super Users are FBI supervisors responsible for Operation Trojan Shield investigations. They can access the communications data related to their investigations as General Users can. Super Users can also create and grant General Users access to investigations under their supervision.

8.2.4 Admins

Administrators are FBI personnel responsible for the operation, maintenance, and oversight of the Hola iBot review platform. They vet access requests to ensure users have adequate justifications for accessing Hola iBot and grant review platform users access to the system. Administrators can also grant access to

investigations for any General Users or Super Users. Additionally, they can access systems to create new cases/syndicates.

8.3 The Roster

The Roster is the location of all device user (JID) information. It is a unique list of every user of the ANØM platform that data has been received for. Within the iBot_dec database, there is a table named 'roster', where the following information is stored regarding device users:

- Unique device user identifier (JID)
  - Each user on ANØM is identified by a unique user identifier (user ID), known technically as a Jabber Identifier (JID). Initially, this user ID took the form of 6 alpha-numeric characters (for example 'aab2c3') and was automatically generated based on the handset International Mobile Equipment Identity (IMEI) number (Appendix A.3 – 19 and A.7.1.1 – 46).
- Device user display name
  - The user of the handset could set a 'display name' (also known as a nickname, username, alias, or handle), which could be changed whenever the user desires. This would not change their user ID (JID) on the platform (Appendix A.7.1.2 – 47)
- Real Name
  - If discovered through reviewing content of a device user, there is a space for entering a true identity for a user.
- Bio
  - The review platform allows for entering in any additional information gathered about a device user that may be useful to include on their JID profile. The JID profile is discussed further in section 10.7.
- Location
  - If discovered through reviewing content of a device user or other users, there is a space for review platform users to enter the assumed location of a device user.

8.4 Groups

As referenced in Appendix A.7.1.4 – 51, A device user could also send end-to-end encrypted messages to a group of ANØM users, and each user in that group was able to see the other members of the group. Groups were also given names that the device users would define. More information about groups can be found in the Appendix, section A.7.1.4 – 51.

The iBot_dec database contained a table named 'groups', which has the following information:

- Group ID (GID)
  - The GID is a string of alphanumeric characters that uniquely identify a group.
  - GIDs are assigned similar to JIDs, at the XMPP server.
- Name
  - The user-created label or name for a group
- Inserted
  - The date of the new data package that initially contained information on the group

For information about how device users are linked to groups, see section 8.7.4.

8.4.1 Lack of Group Information

Throughout the reception of data packages from the third-party country, group information was not always collected. This would cause a lack of information about groups within the database.

8.5 Products

Products are synonymous with conversations contained within each new data package. A single product represents a conversation between two or more device users assigned to a case, over the period of time between the previous data package and the data package the current product corresponds to.

8.5.1 Products in Detail

A new unique set of products are created with each new data package received from the third-party country for each case. More information about how products are created can be found in section 9.

There is a table within the iBot_dec database named ‘products’. This table contains the following information:

- Case ID
  - Each product that is created has a one-to-one relationship with a case.
  - The case/syndicate unique identifier that the product belongs to is stored within the product table, represented as a positive integer.
- Product ID (PID)
  - A unique identifier for the product, represented as a positive integer.
- Product Note
  - Review platform users can add notes to products.
  - This field is where the notes are added.
- Date Created
  - New data packages are received from the third-party country every Monday, Wednesday, and Friday.
  - Since new sets of products are created each time a new data package is received, the date of which the product was generated is also saved, which corresponds to the date the new data package was created.
- MCC String
  - A unique list of the different Mobile Country Codes used by the devices within the product
- JID String
  - An alphabetized list of device user unique identifiers (JIDs) that sent or received messages within the product
- Message Count
  - The number of messages that correspond to the product
- JID 1
  - The first sender of a message within the product
- JID 2
  - The other participant or JID in the product if the product is not a group product
- Group ID (GID)
  - The group unique identifier if the product is a group product

- Is Duplicate
    - This value will be either 1 or 0 (indicating whether it is a duplicate (1) or it is not a duplicate (0)).
    - As said before, each product has a one-to-one relationship with a case.
        - If one or more of the JIDs within a product do not all belong to the same case or syndicate (I.E. JID 123456 belongs to case Alpha and JID 789100 belongs to case Bravo), a product will be created for each case.
        - The messages and participants of the product will all be the same, but there will be a product created for each case a JID within the product belongs to.
        - See section 8.7.3 for more information about JID to Case relationships within the database and section 9 for more information about the ingestion process.
- Known Group
    - This value will be either 1 or 0 (indicating whether our database has information about the group (1) or not (0)).
    - As previously mentioned in section 8.4.1, group information was not always collected on the third-party country's server.
        - This would prevent the Hola iBot review platform from displaying the group identifier, the group name, and potentially **all** of the group members.
        - However, the products are formed directly from the messages, and the members of the conversation/product are determined by the senders and receivers of messages.
        - Thus, if the database does not contain group information, the JID String will still reflect the members of the conversation/product.

8.6 Messages

The MySQL difference-dump within the new data package received from the third-party country contained a single table for messages. Details on the ingestion process for messages are provided in section 9. There are four tables in the iBot_dec database that contain message content and metadata associated:

- 'text_message'
    - This table contains decrypted text messages, contact messages, forwarded messages, and text components of note messages.
    - This table also contains the 'from note' field, indicating that the text in the 'content' field is derived from a note message type.
        - Reference 9.3.2.2.2 for more information about how note message types are ingested to the database.
- 'ptt_message'
    - This table contains the paths (on the I1 server) of decrypted, converted, and pitch-adjusted (if needed) push-to-talk messages.
- 'video_message'
    - This table contains the paths (on the I1 server) of decrypted video messages.
- 'image_message'
    - This table contains the paths (on the I1 server) of decrypted image messages and image components of note messages.

    - This table also contains the 'from note' field, indicating that the image that can be found at the path within the 'content' field is derived from a note message type.
        - Reference section 9.3.2.2.2 for more information about how note message types are ingested to the database.

All four tables contain the following metadata fields:

- Message ID (MID)
    - A unique identifier for a message represented as a positive integer
    - The MID is shared across all four tables
        - Ex. if MID 5 appears in ptt_message, that is the only place the MID will appear
- Original Message ID
    - Another unique identifier for a message, which is inherited from the new data packages.
    - A unique string of characters, represented as a Universally Unique Identifier (UUID).
    - This message ID was stored with each message the third-party country's server received as a 'blind carbon copy'.
    - This message ID is described further in Appendix A.8.2 – 64.a.
    - The MID was created within the iBot_dec database to provide a uniform standard of primary keys within the database.
        - I.E. PIDs as positive integers, case IDs as positive integers, etc.
- Inserted Date
    - The inserted date corresponds to the date in which the data package containing the message was ingested to the database.
- Mobile Country Code (MCC)
    - The MCC of the sending device at the time of the sent message
- Is Group
    - This value is either 1 or 0 (indicating whether the message was sent to a group (1) or not (0)).
    - If the message is a group message (value of 1), the receiver field contains the group ID (GID) that the message was sent to.
- Sender
    - The sending device user's ID (JID)
- Time
    - When the message was sent according to the timeframe in 8.6.1.
    - For example, a message sent on June 6, 2021 at 10:00 pm Pacific Time, the database would have the time and date as 5:00am on June 7, 2021 UTC.
- Location Information
    - If location information (latitude and longitude) was received with a message, the latitude and longitude would be stored with each message in each of the four tables.

8.6.1 Message Time Zone Discrepancy

Due to a misconfiguration on the third-party country's server, the time zones of messages varied throughout the collection period. The following timeframes provide insight into the correct conversions of timestamps for messages:

- Timeframe 1

    - October 1, 2019 – October 27, 2019: UTC – 3
- Timeframe 2
    - October 27, 2019 – March 29, 2020: UTC – 2
- Timeframe 3
    - March 29, 2020 – September 18, 2020: UTC – 3
- Timeframe 4
    - September 18, 2020 – End of operation: UTC

8.7 Relationships

The following sections define the relationships within iBot_dec that correspond to all tables defined in sections 8.1-8.6. The tables defined in section 8.7 all correspond to many-to-many relationships, and they follow a naming convention of this example:

User **assigned_to** *case*

Where the italicized are single-keyed tables (unique entity of the user table is uniquely identified by the user_id), and the bold indicates a composite-keyed table (unique entity of the assigned_to table is uniquely identified by both a user ID and case ID).

8.7.1 Product to Message Relationship

There is a table within the iBot_dec database named 'has_message'. This table contains the following information:

- Product ID (PID)
    - A unique identifier for the product, represented as a positive integer, that the message in the same row of values is related to.
- Message ID (MID)
    - A unique identifier for the message, represented as a positive integer, that the product in the same row of values is related to.
- Type
    - The type of the message that is referenced by Message ID in the same row of values.
- Marking
    - Each message allows the marking of 'Not Marked', 'Pert', 'Not Pert', or 'Priv'.
- Original Message ID
    - The unique string of characters, represented as a Universally Unique Identifier (UUID), that uniquely identifies a message, inherited from the original data package from which the message corresponds.

8.7.2 User to Case Access Relationship

Review platform users are granted explicit access to cases by administrators. The relationships for review platform users to cases is managed by a table named 'assigned_to'. The 'assigned_to' table contains the following attributes:

- LEEP Username
    - The username of the Hola iBot user that has access to the case identified in the same row of values.
- Case ID

- The case ID of the case that the Hola iBot user in the same row of values has access to

8.7.3 JID to Case relationship

Device users (JIDs) are assigned to cases to group them amongst other device users that frequently communicate together, as mentioned in 8.1.1. Since many device users can correspond to many cases, the 'belongs_to' table allows for these relationships. The 'belongs_to' table contains the following attributes:

- Device User ID (JID)
  - The JID of the user that is related to the case ID within the same row of values
- Case ID
  - The case ID that the corresponding JID is related to.
- Date Added
  - The date the relation was created in the iBot_dec database.

8.7.4 JID to Group Relationship

Device users (JIDs) are added to groups by group administrators within the ANØM application. JIDs can be members of many groups, and groups can have many JIDs that are members. Thus, the 'member_of' table allows for this relationship. The 'member_of' table contains the following attributes:

- Device User ID (JID)
  - The JID of the user that is a member of the group identified in the same row of values.
- Group ID (GID)
  - The GID of the group that the corresponding JID is a member of.

9. INGESTION PROCESS

Following the decryption of all encrypted content fields, roster nicknames, group names, and attachments, and the adjustment of paths within content fields of messages that are of type ptt, video, image, or note, the ingestion process will continue by following the steps defined within this section (section 9). This section describes the process of transitioning data on I1 from the iBot_enc database (where the new data packages were temporarily stored) to the iBot_dec database, which permits the review platform code access to the new data.

The prerequisite to begin the ingestion process is sections 5-7:

- Decrypt the encrypted new data package
- Extract the archive of data
  - Attachment files
  - MySQL difference-dump
    - As mentioned in section 2.2, the difference-dump only contains data that is new or different.
- Ingest the MySQL difference-dump into the iBot_enc temporary database
- Decrypt all encrypted fields within the database
- Fix paths of all fields in the database that reference attachment files
- Decrypt all attachment files
- Convert and pitch-adjust audio, if necessary

9.1 Insert new Roster Information

The first data inserted to the iBot_dec database is the roster information. The following is the process for adding new roster information:

1. Load all new roster data from iBot_enc (new MySQL data)
2. For each new roster entry:
    a. Check to see if the device user ID (JID) already exists in the iBot_dec 'roster' table
    b. If the device user already exists, this indicates that the user changed their Device User Display Name.
        i. If the new display name does not match the old display name, the new display name is updated in the 'roster' table of iBot_dec.
    c. If the device user doesn't exist, this indicates a completely new user. The user is added to the 'roster' table of iBot_dec, and a relationship is added for the device user to correspond to the "UNKNOWN" case.
        i. The "UNKNOWN" case is a default case for all new device users (JIDs) and for devices to remain until a case or syndicate is found to place them.

Any errors updating the device user display name or inserting the user into the 'roster' table are logged.

9.2 Insert new Groups

9.2.1 Insert New Group Information

New information regarding groups on the ANØM platform is inserted after new roster information. The following is the process for adding new group information:

1. Load all new group data from iBot_enc (new MySQL data)
2. For each new group from the new data:
    a. The group ID (GID), group name, and the inserted date is inserted into the 'groups' table within the iBot_dec database.

9.2.2 Insert Group Membership Updates

Members were added to groups on the ANØM platform by administrators of groups, and these updates were received in the MySQL difference-dump. The following is the process of updating the group membership relation:

1. Load all new group membership data from iBot_enc (new MySQL data)
2. For each new group ID (GID) to device user ID (JID) relationship:
    a. The GID and JID are inserted into the 'member_of' table within the iBot_dec database.

9.3 Message Ingestion

After changes to the roster and groups are updated within the database, the process to ingest messages starts. This is the primary ingestion mechanism that allows for data to be reviewed in the form of products (conversations). Message ingestion follows the process described in this section (9.3).

9.3.1 Initialization

Before ingesting new data, some components are initialized.

9.3.1.1 Blacklist

The blacklist is a line-delimited text file separating JIDs. This file contains JIDs whose messages sent are blacklisted from being ingested to the iBot_dec database. The blacklisting feature will be further described in a future addendum.

9.3.1.2 Conversation Temporary Data Structure

The data structure to hold all unique conversations and the messages that correspond to them is initialized. This will be the data used for forming products after cycling through each new message.

9.3.1.3 Case to JID Temporary Data Structure

This is a data structure that maintains a list of JIDs that correspond to each case within the database. This data structure will be later used for forming products correctly. It is formed from data residing within the 'belongs_to' table.

9.3.1.4 Retrieve Last Message ID (MID)

Getting the last inserted Message ID (MID) provides the starting Message ID (MID) for the new data, as MID is an ascending positive integer.

9.3.1.5 Load New Messages

The new messages are loaded from iBot_enc (new MySQL data) to be inserted.

9.3.2 Insert Messages

For each message, the remainder of section 9.3.2 is performed.

9.3.2.1 Metadata Insertion

Within the iBot_dec database, a table named 'metadata' exists that contains the data that has common attributes from all four message tables as referenced in section 8.6. The metadata table has the following attributes:

- Message ID (MID)
- Original Message ID
- Time
- Sender
- Receiver
- Mobile Country Code (MCC)
- Is Group
- Type
- Location Information

The attributes above have all been defined in section 8.6 under the metadata section.

Regardless of message type, all of the metadata attributes are inserted for each message.

9.3.2.2 Insert by Type

The ingestion process differs by type of message.

9.3.2.2.1 Text Messages, Contact Messages, and Forwarded Messages

If the message type is 'text', 'contact', or 'forwarded', the 'content' field from the iBot_enc messages table is copied into the 'content' field in the iBot_dec 'text_message' table. The value in the 'content' field was already decrypted by the steps in section 6.1.

9.3.2.2.2 Note Messages

If the message type was 'note', the note attachment would be decrypted. After being decrypted, note attachments are revealed as compressed ZIP files that contain text and image files. These files are parceled out into the 'text_message' and 'image_message' tables respectively.

First, the ZIP files are extracted for processing. For each extracted file, a determination whether the file is a text file or image file is made, and if it is a text file, the text would be read and inserted to the 'content' field of the 'text_message' table. However, if it was an image file, the image file would be copied to the directory which images are served to the front-end review web application, and an entry of data would be inserted to the 'image_message' table.

In all entries to 'text_message' or 'image_message', the 'from note' attribute would be updated to a value of 1 to indicate the text/image is a component from a note.

9.3.2.2.3 Push-to-Talk, Image, and Video Messages

If the message type is 'ptt', 'image', or 'video', the 'content' field would be copied directly from the iBot_enc messages table into the iBot_dec corresponding message table based on the type. The paths in the 'content' field were fixed in-place and updated to the new paths which the front-end review web application utilizes. More information on the paths being updated can be found in section 6.1.1.

9.3.4 Assigning Messages to Products

After the message metadata and type-based table insertion have completed, the message is assigned to a conversation, utilizing the temporary data structure initialized in section 9.3.1.2.

This data structure contains a nested unique list of conversations, each of which have the members (JIDs) of the conversation, the mobile country codes (MCCs) that messages were sent from, the message ID (MID), and the message type ('text', 'ptt', 'video', etc).

After assigning each message to a conversation, the loop over the new messages is complete, and the usage of the temporary iBot_enc database concludes.

9.4 Product Ingestion

After the conversation data is finished populating, the products are ready to be created for the current new data package. The remainder of section 9.4 processes each unique conversation in the temporary data structure that had message(s).

9.4.1 Product Creation Per Case

For each case, the steps under 9.4.1.1 are executed on the conversation.

9.4.1.1 If JID in conversation belongs to case

If a device user that is a member of the current conversation belongs to the current case, 9.4.1.1.1-9.4.1.1.3 are completed.

9.4.1.1.1 Generate Product for Case

If 9.4.1 and 9.4.1.1 are satisfied for the current conversation, a product will be created for the current case. The following values are inserted based on temporary data structures and the current status of the ingestion code:

- Case ID
- Date Created
- Mobile Country Code (MCC) String
- Device User ID (JID) String
- Count
- Group ID (GID) – if necessary
- Known Group – If group information was found during product creation (by querying the 'groups' table), this will be set to 1

9.4.1.1.2 Create Message and JID Relation to Product

As mentioned in 8.7.1, messages correspond to products through the use of the 'has_message' table. After creating the product for the current case, the messages within the temporary conversation data structure are all associated with the product by inserting their message IDs (MIDs) into 'has_message', along with the product ID (PID) that was inserted.

There is a table within the iBot_dec database named 'part_of' that links device user IDs (JIDs) to product IDs (PIDs). This creates a table that informs the system that a specific device user sent messages that appear in a product. This table has the following attributes:

- Device User ID (JID)
- Product ID (PID)

This table is also be populated upon creation of a new product.

9.4.1.1.3 Assign default marking to product

There is a table within the iBot_dec database named 'has_marking' that links text-based markings to products. Products can be marked within the Hola iBot review platform by users of Hola iBot. The has_marking table has the following attributes:

- Product ID (PID)
- Marking
    - A text-based marking that can be assigned to a product.
    - Examples include "Not Reviewed (default)", "Pertinent", "Not Pertinent".

The default product marking of 'Not Reviewed' is inserted to the 'has_marking' table for the currently generated product.

9.5 Post-Ingestion Exports

Encrypted exports were automatically sent over Secure File Transfer Protocol (SFTP) to countries that required immediate review of new data received from data packages. These encrypted exports contained a JavaScript Object Notation (JSON) file for each case the requested country has been given access to. This JSON file contains all product data for the previously ingested data package.

10. WEB APPLICATION/REVIEW PLATFORM

Hola iBot is the custom web application built and operated by San Diego FBI to serve as the review platform of all data received from the ANØM platform.

10.1 User Access Controls within Hola iBot

The review platform user roles defined in section 8.2 are roles of the users of Hola iBot. As stated before, the users in section 8.2 must all be provisioned accounts within the Law Enforcement Enterprise Portal (LEEP).

10.1.1 Law Enforcement Enterprise Portal (LEEP) Access Control

The Law Enforcement Enterprise Portal (LEEP) is a portal that law enforcement agencies (including international) and other intelligence groups can be provisioned accounts. Once the applying user is vetted by the provisioning department, the user will have access to the list of applications/tools, one of which is the Hola iBot review portal.

LEEP also follows common security practices for authenticating users, such as two-factor authentication (2FA), security pictures, and security questions.

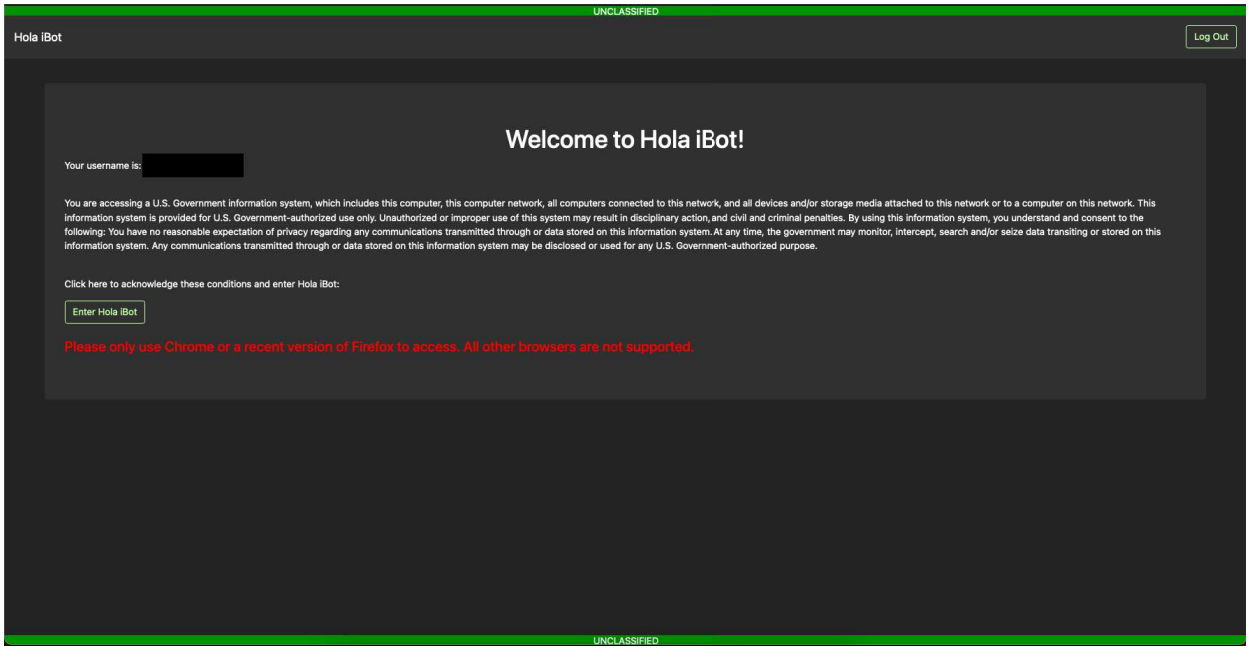
10.1.2 Hola iBot Access Control

After a user authenticates through the login process that LEEP administers, the user will have the ability to visit the Hola iBot application by clicking the application in the list as shown in the screenshot below.

Screenshot 10.A

After clicking the Hola iBot application, a single-sign-on (SSO) request using the Security Assertion Markup Language (SAML) is sent to the front-end web application server. SAML is a protocol that allows identity providers to pass authenticated credentials to another service, which is how Hola iBot authenticates users.

The review platform user is only granted access to the system if they exist within the 'users' table cited in section 8.2. If they are granted permission to the application, they will be greeted with the warning banner shown in screenshot 10.B.

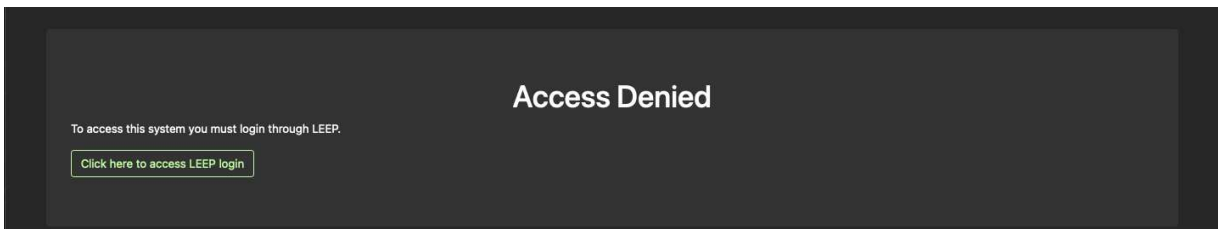


Screenshot 10.B

The text of the banner is the following:

> *You are accessing a U.S. Government information system, which includes this computer, this computer network, all computers connected to this network, and all devices and/or storage media attached to this network or to a computer on this network. This information system is provided for U.S. Government-authorized use only. Unauthorized or improper use of this system may result in disciplinary action, and civil and criminal penalties. By using this information system, you understand and consent to the following: You have no reasonable expectation of privacy regarding any communications transmitted through or data stored on this information system. At any time, the government may monitor, intercept, search and/or seize data transiting or stored on this information system. Any communications transmitted through or data stored on this information system may be disclosed or used for any U.S. Government-authorized purpose.*

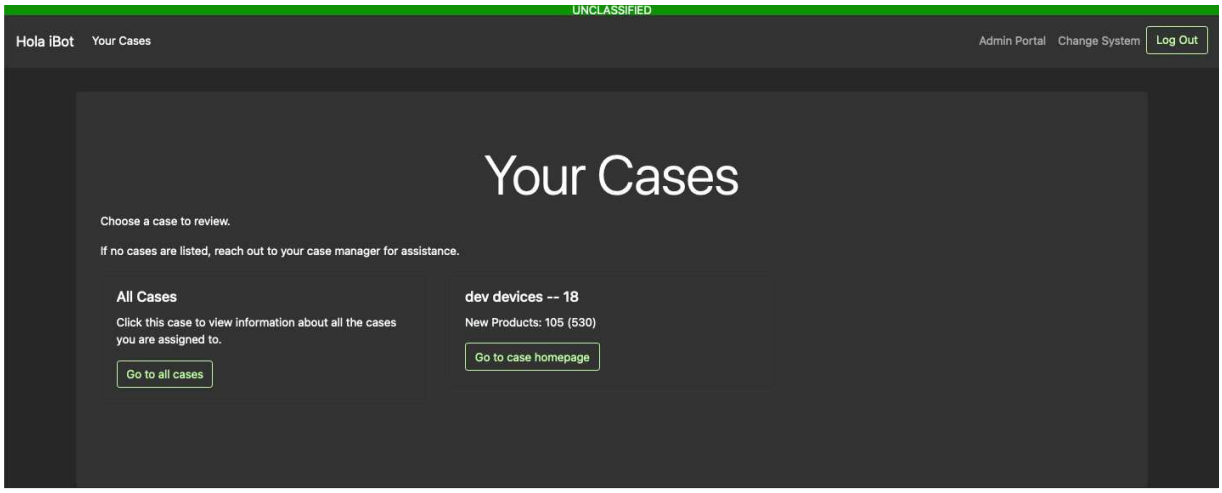
If the user does not exist in the database, they are presented with a generic invalid access page. This page can be viewed in Screenshot 10.C



Screenshot 10.C

10.2 Cases in Hola iBot

Assuming the user has been granted access to the system through LEEP authentication and Hola iBot explicit database access, the user is navigated to their homepage, where a list of cases they have been given explicit access will be displayed. Screenshot 10.D portrays the homepage of a user with access to a single case, named “dev devices” with a case ID of 18.



Screenshot 10.D

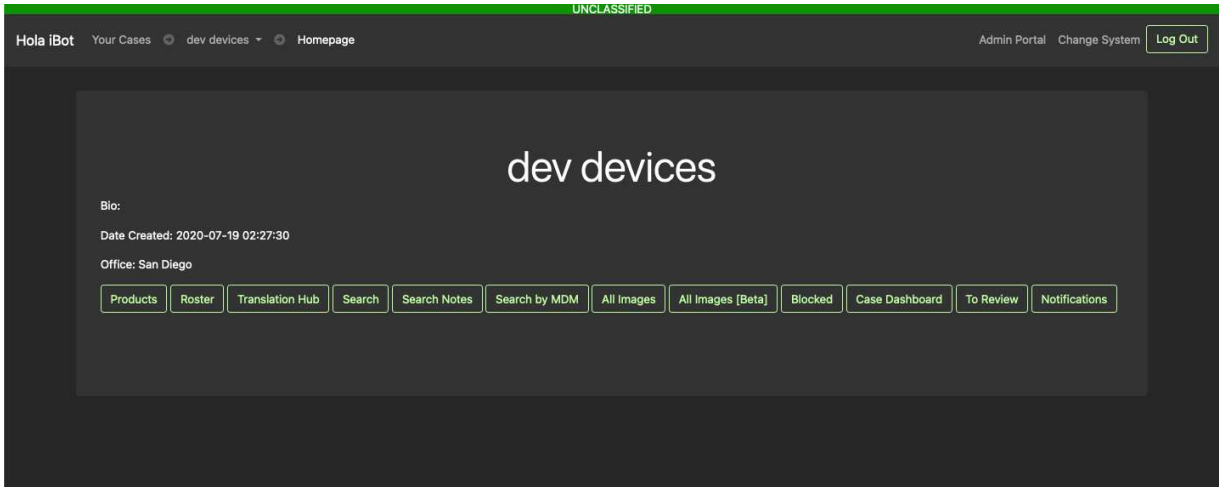
The list of cases displayed on the homepage are displayed based on the ‘assigned_to’ table, as previously described in section 8.7.2.

When visiting a single case, the data in the remaining pages is filtered to only device users (JIDs) that correspond to the case within the ‘belongs_to’ table (JID to case ID relationship), as mentioned in section 8.7.3.

Every user is also shown an option for “All Cases”. When selecting this option, the data in the remaining pages is populated with all data the review platform user has access to.

10.2.1 Case Homepage

After navigating to a case, the case homepage is displayed for the review platform user. Several options are presented after navigating to a homepage for a case. A sample case homepage is displayed in Screenshot 10.E.



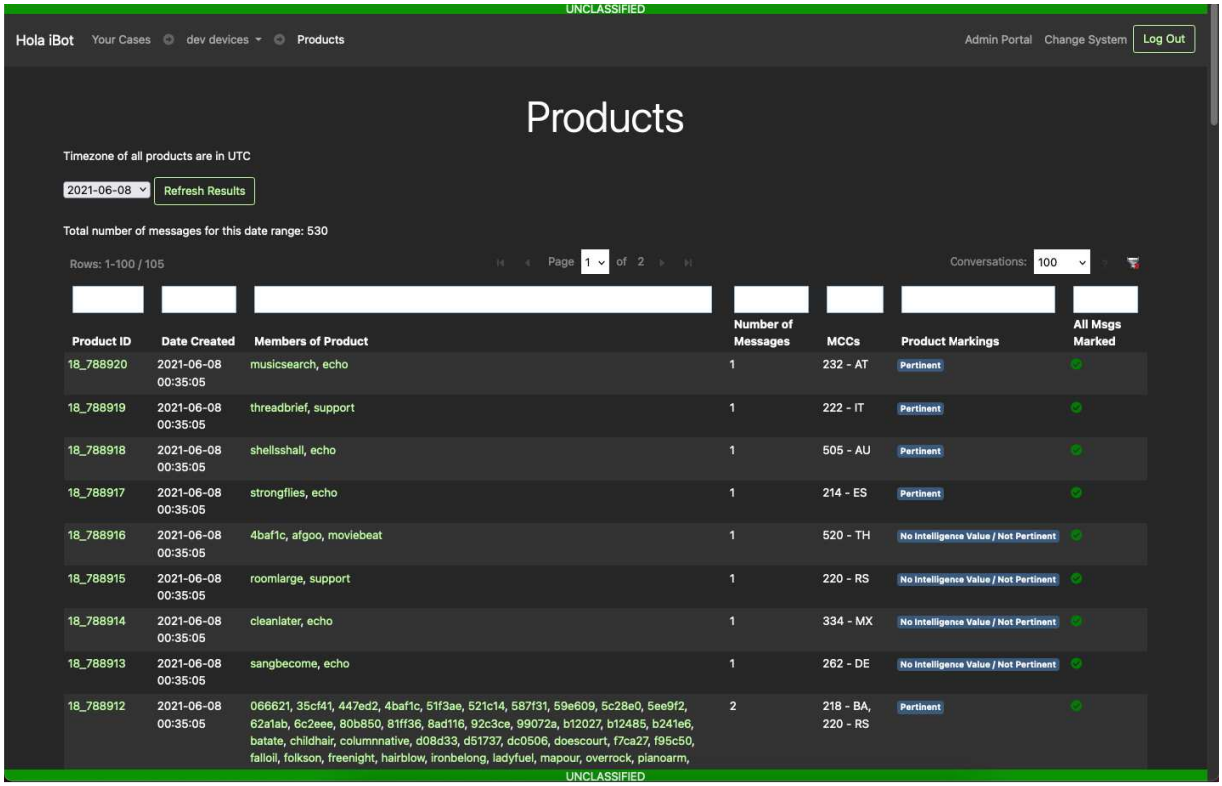
Screenshot 10.E

The remainder of section 10 further describes each of the options displayed on the case homepage.

10.3 Products Page

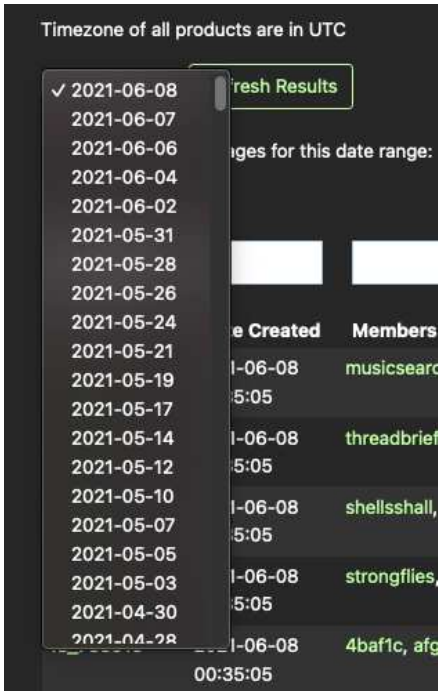
The Products Page is where users navigate to view conversations for the case. See section 8.5 for more information about products and section 9.4 for information about how products are created.

Products are unique conversations for a case that were received for a data package. Screenshot 10.F shows the Product Page for the “dev devices” case.



Screenshot 10.F

The drop-down menu option in the top left populated with “2021-06-08” expands a list of dates that new data packages were received:



Screenshot 10.F.1

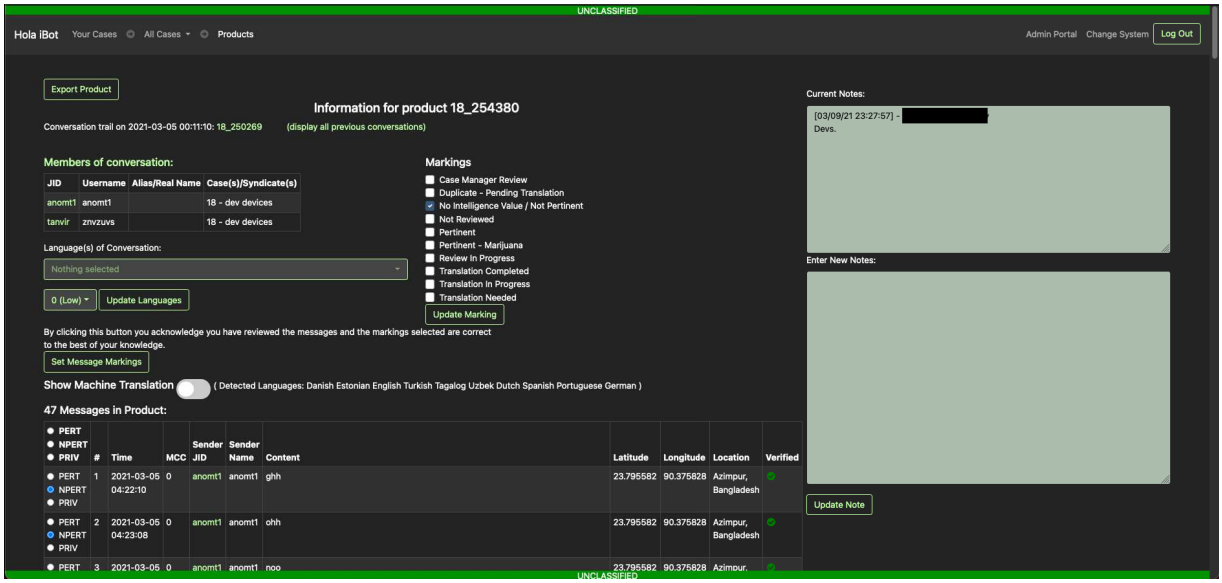
Lists of products are unique, per drop date. As also observed in Screenshot 10.F, the product ID (PID) is not shown under “Product ID” in the list of products. The “Product ID” shown to the user is denoted by case ID, followed by an underscore, followed by the actual product ID (PID). This provides an easier method of universally referencing products and knowing which cases the products belong to.

The rest of the data displayed on this page within the table is gathered from the ‘products’, ‘has_marking’, and ‘has_message’ table.

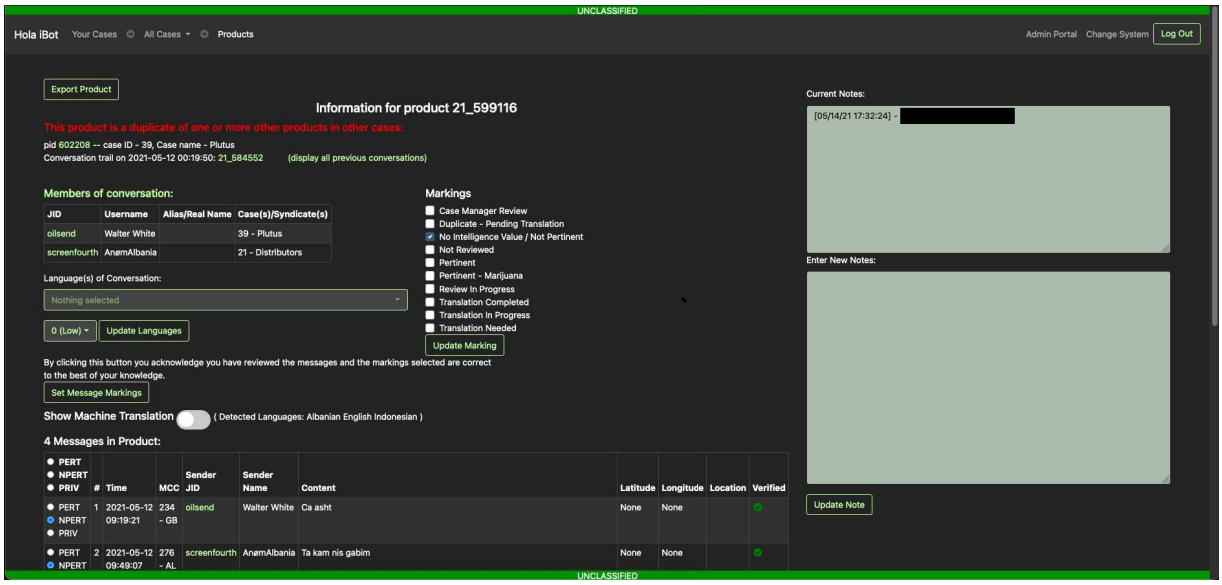
- “Date Created”
  - populated by the ‘date created’ field in the ‘product’ table
- “Members of Product”
  - populated by the ‘JID String’ field in the ‘product’ table
  - The JIDs displayed in this column are all linked to the JID profile.
    - See section 10.7 for more information about the JID profile.
- “Number of Messages”
  - populated by the ‘count’ field in the ‘product’ table
- “MCCs”
  - populated by the ‘MCC String’ field in the ‘product’ table
- “Product Markings”
  - populated by the ‘has_marking’ table, which will relate the product to a text-based marking
- “All Msgs Marked”
  - populated by querying the ‘has_message’ table to test whether all messages have a marking that is not the default

10.4 Product Page

The “Product ID” links will open the product page, which displays the messages for the product, as shown in Screenshot 10.G and 10.G.1.



Screenshot 10.G



Screenshot 10.G.1

The product page displays many different components of information for the review platform user. Referencing Screenshot 10.G and 10.G.1, the product page displays the following information:

- Whether or not the product being referenced is a duplicate of a product within another case
    - Screenshot 10.G.1 shows an example of a duplicate product, which will also provide a link to the corresponding duplicate product.

- The members of the conversation
    - These are compiled during the ingestion process as referenced in section 9.4.1.1.
- The languages that have been added to the product by review platform users
    - This list allows for multiple languages to be added.
    - The languages list corresponds to translation submissions.
        - See section 10.5.1 for more information.
- The markings that have been added to the product
- The message markings that have been set by the review platform users
- Notes that have been created about the product (right pane).
    - Once added, notes cannot be removed, as they will move to the “Current Notes” text area with a timestamp that the note was created and the username of the review platform user that added the note.
    - These notes correspond to the ‘product note’ field in the ‘products’ table.
- All translations that have been added to the product based on the language that was added will appear at the very bottom of the product page.

10.4.1 Group ID Display

If the product corresponds to a group, the following will be displayed if group information is in the iBot_dec database:

Screenshot 10.G.2

10.4.2 Attachment Messages (PTT, Image, Video, Note) in Hola iBot

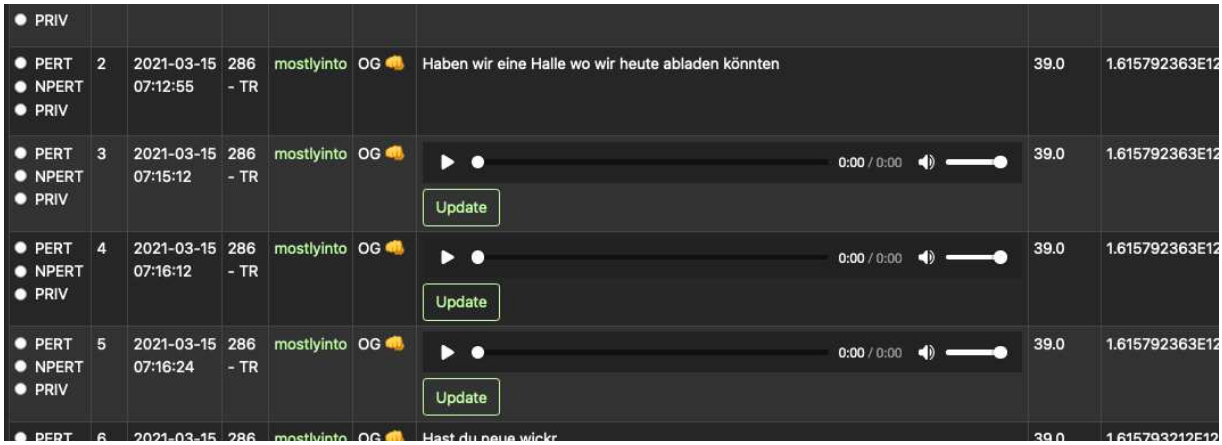
Attachment messages are all displayed inline as text messages are. Attachment messages can also have a marking that review platform users assign. Within the iBot_dec database there is a table named ‘attachment_has_marking’, which allows for marking attachments with text-based descriptions. The ‘attachment_has_marking’ table has the following fields:

- Message ID (MID)
    - The MID that the text-based description corresponds to
- Marking
    - The text-based description of the marking

These attachment markings can be set on the product page by review platform users.

10.4.2.1 Push-to-Talk Messages (PTT) in Hola iBot

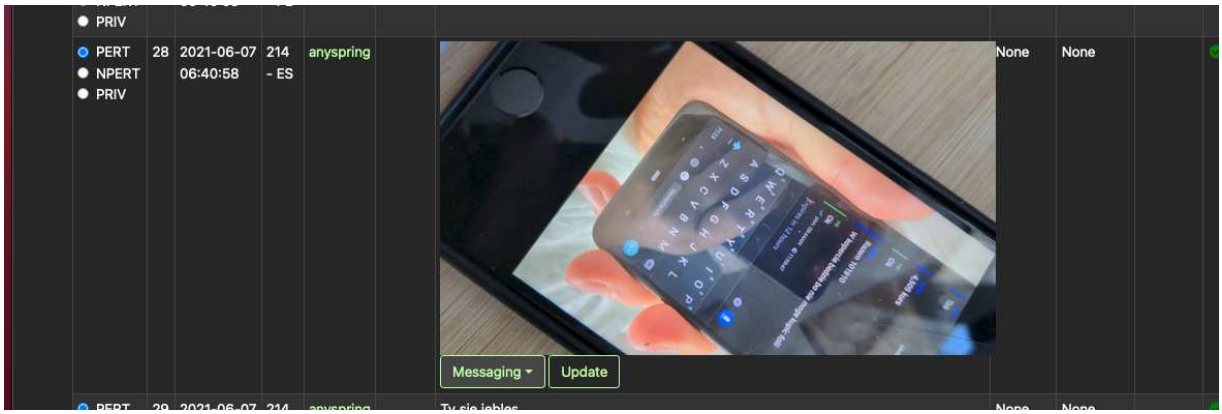
Screenshot 10.G.3 shows an example of a few PTT messages.



Screenshot 10.G.3

10.4.2.2 Image Messages in Hola iBot

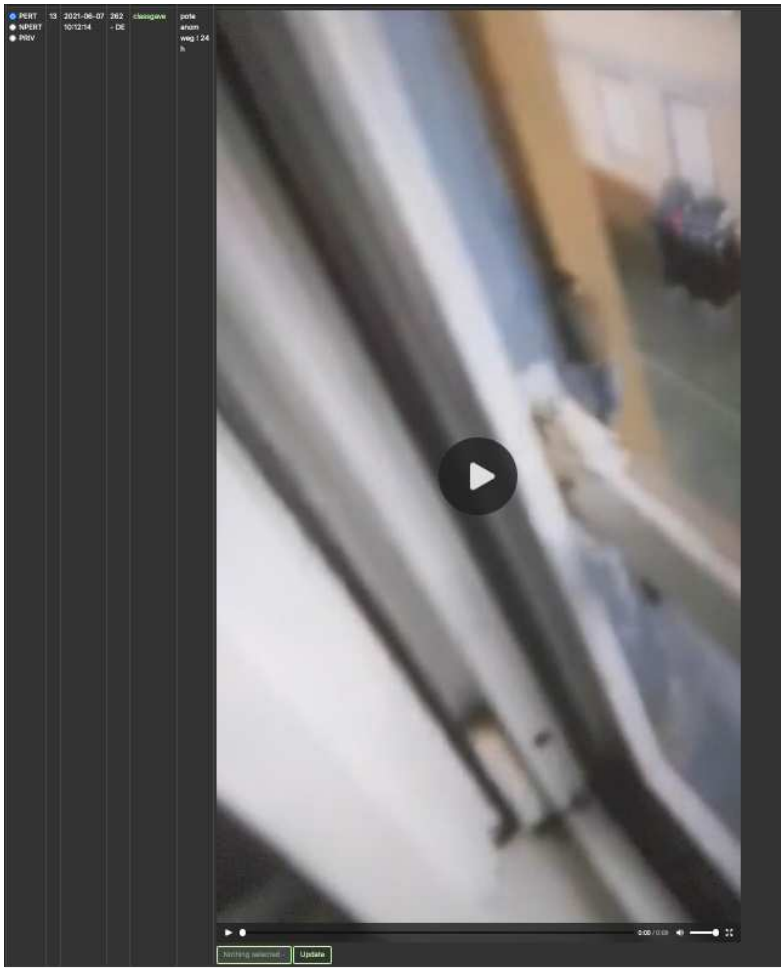
Screenshot 10.G.4 shows an example of an image message displayed inline.



Screenshot 10.G.4

10.4.2.3 Video Messages in Hola iBot

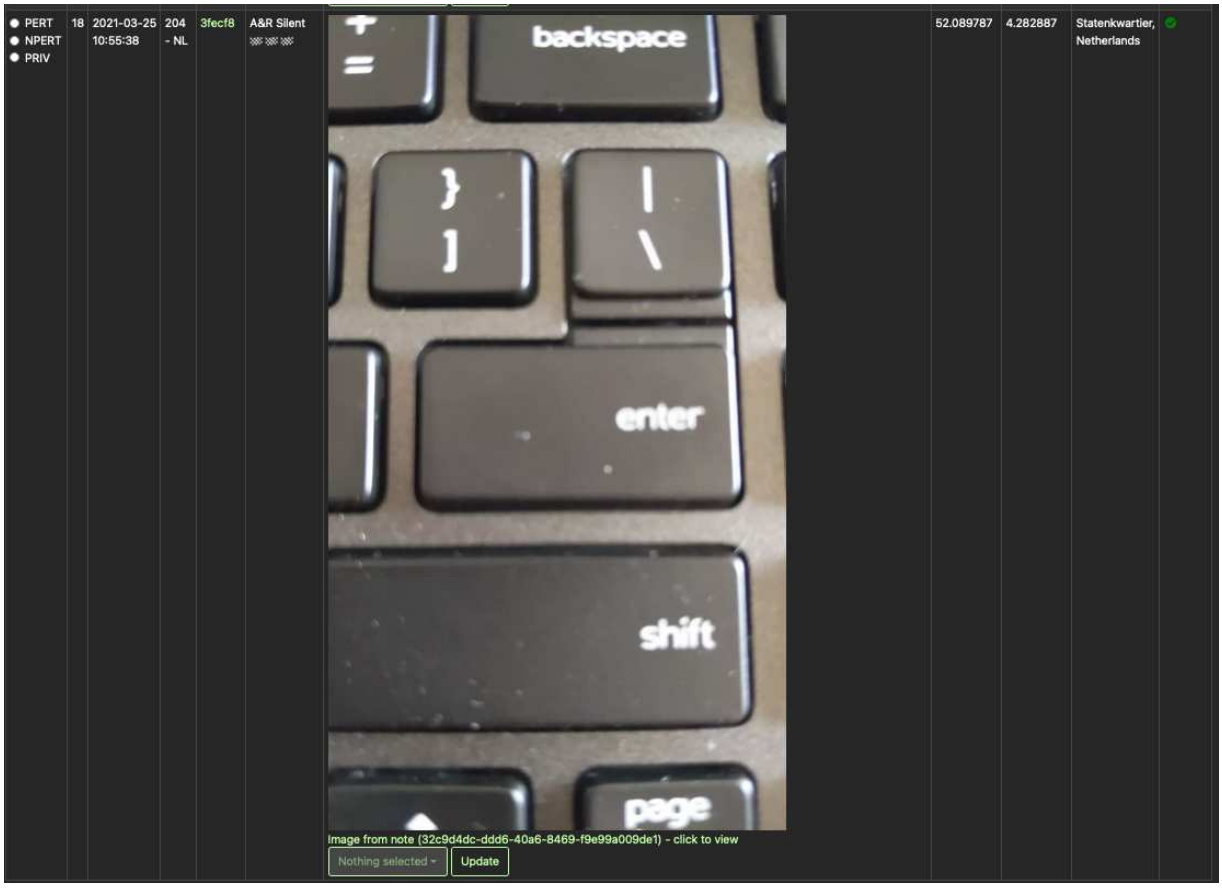
Screenshot 10.G.5 shows an example of a video message in Hola iBot.



Screenshot 10.G.5

10.4.2.4 Note Messages in Hola iBot

Screenshot 10.G.6 shows an example of an image component of a note displayed inline as a message.



Screenshot 10.G.6

The text “Image from note ([Original Message ID]) – click to view” indicates that the image was derived from a note message.

10.5 Translations

If a product has been assigned a language by utilizing the dropdown in Screenshot 10.G and 10.G.1, the product is automatically submitted for translation. Due to the many-to-many relationship that languages have with products, a table named ‘has_language’ exists in the database for creating these relationships. The ‘has_language’ table has the following attributes:

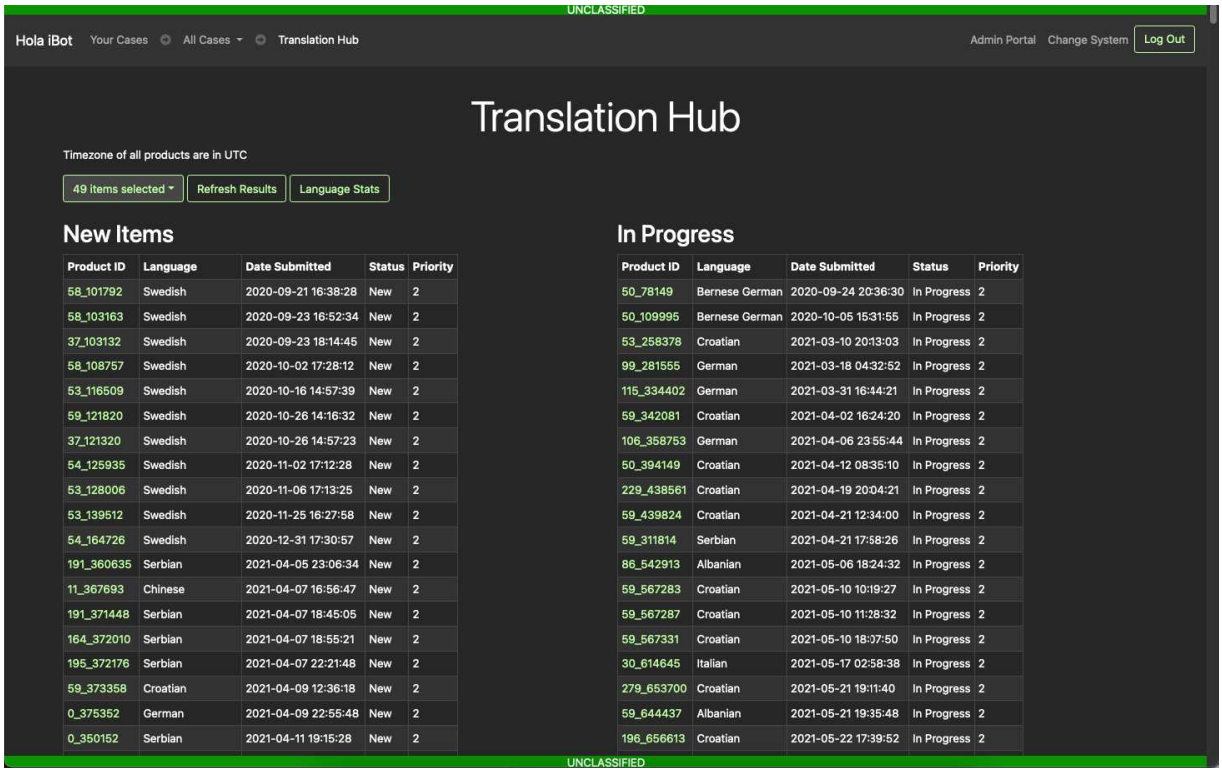
- Language Name
  - The name of the language that is being assigned to the product
  - Examples include “English” and “Spanish”
- Product ID (PID)
  - The PID of the product that the language is being assigned to
- Status
  - This field indicates the status of the translation.
  - This field can be set to either “New”, “In Progress”, or “Completed”.
- Priority
  - This field reflects the priority of the translation request.
- Submitter

  - This is populated with the review platform username that added the language to the product.
- Translation
  - This is the value that linguists will enter for the translation of the product.
- Date Submitted
  - Corresponds to the date the submitter added the language to the product
- Case ID
  - The case ID the product in the row of values corresponds to
- Completed By
  - The review platform username that set the 'status' field to "Completed"

Once a language is added to a product, the product ID, case ID, and language name will be viewable on the Translation Hub page.

10.5.1 Translation Hub

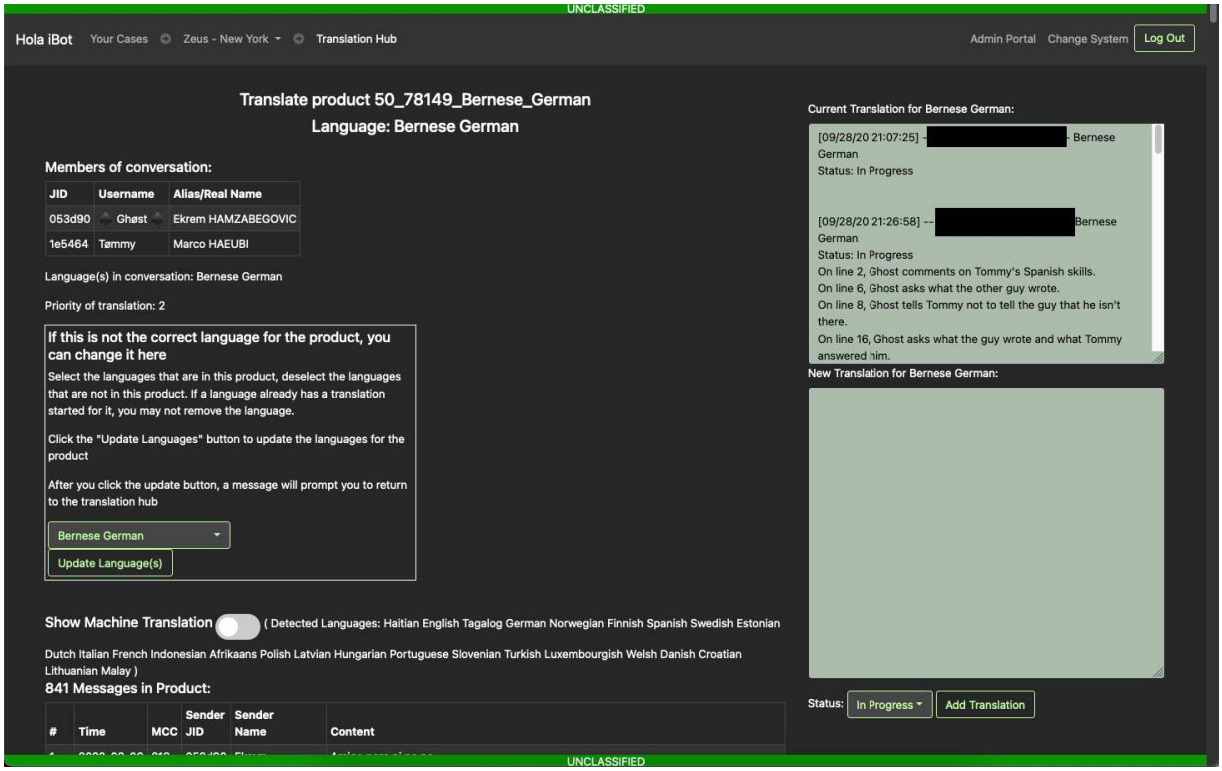
The Translation Hub contains all user-submitted translations for products. Screenshot 10.H contains an example view of the Translation Hub.



Screenshot 10.H

The two lists indicate the status from the 'has_language' table for the corresponding product ID (PID). If a translation request in the 'has_language' table has a status of "Completed", it will no longer be displayed in the two lists.

After clicking a linked product from the Translation Hub, a page similar to the product page will be displayed, as shown in Screenshot 10.I.



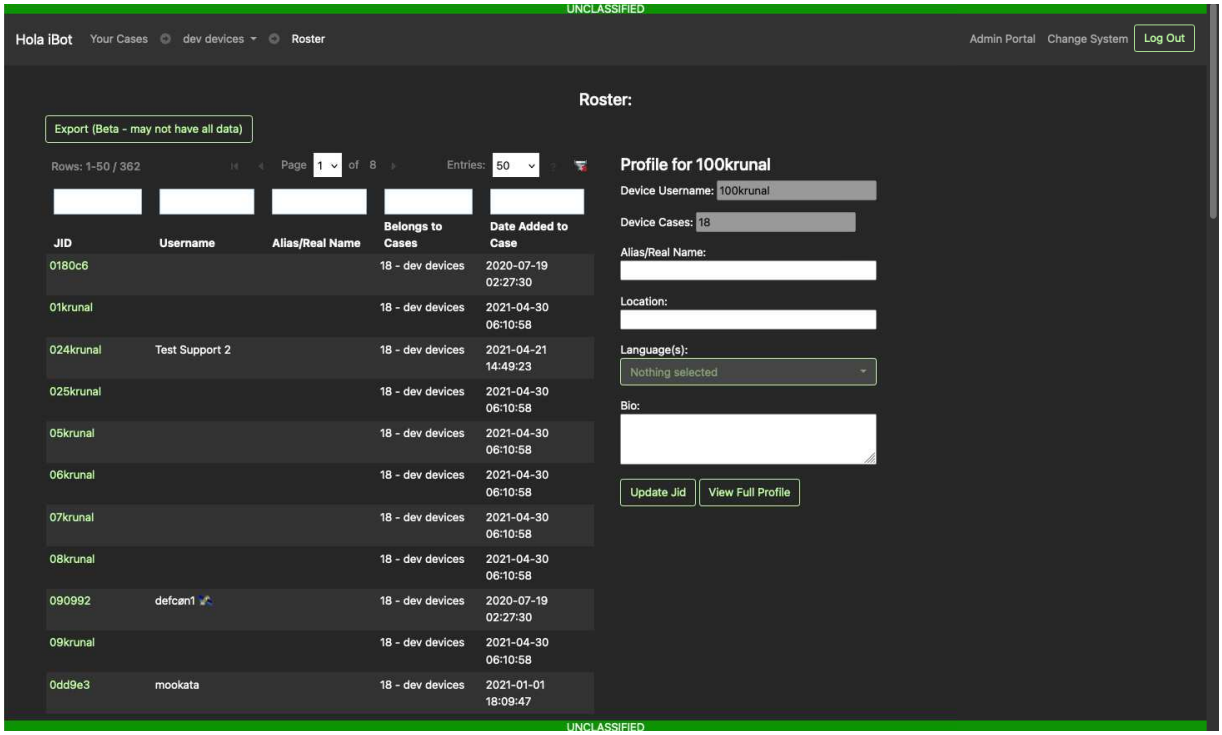
Screenshot 10.I

This page is specifically used by linguists for adding translations of products. The pane on the right-hand side of Screenshot 10.I shows the space where users can add translations. Once a translation is added, it is similar to when review platform users add notes to products. They move to the “Current Translation for [name of language]” text area. Translations cannot be updated that have already been submitted and moved to the other pane. They are also logged with a timestamp and username of the review platform user that added the translation.

All notes that have been added to the corresponding product through the product page are visible at the bottom of the translation page.

10.6 Roster Page

The page that displays the roster shows a list of device users (JIDs) that have a relation to a case in the ‘belongs_to’ table. Screenshot 10.J displays an example roster page for the “dev devices” case.



Screenshot 10.J

The page is separated vertically into two panes.

The left pane displays the list of device users (JIDs) with their Username (device user display name), real name (entered by review platform users if discovered through investigations), corresponding cases (reference the 'belongs_to' table), and the date they were added to the case (reference the 'belongs_to' table).

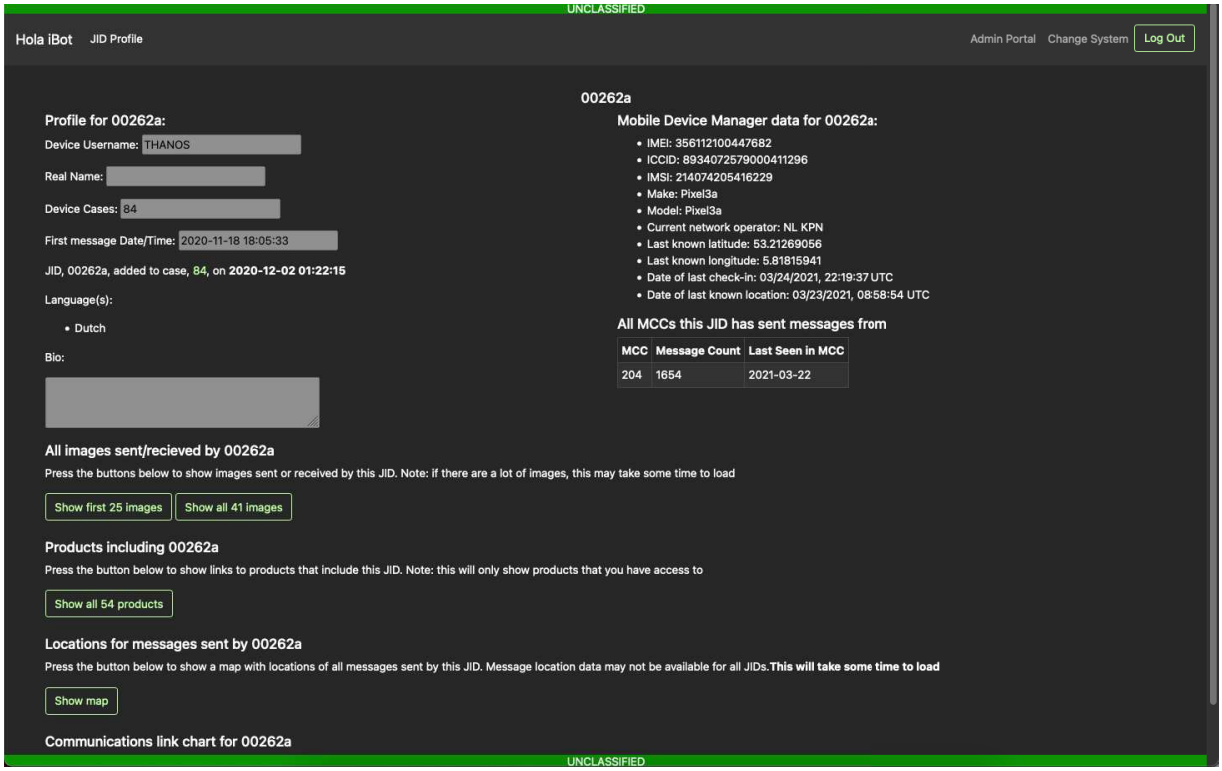
Each row in the list is clickable and will populate the right pane with the following additional information that can be edited:

- Alias/Real Name
- Location
- Language(s)
- Bio

Section 8.3 has additional information regarding the fields in the roster.

10.7 JID Profile

The JID profile can be linked to from a number of different pages. An example JID profile is shown in Screenshot 10.K.



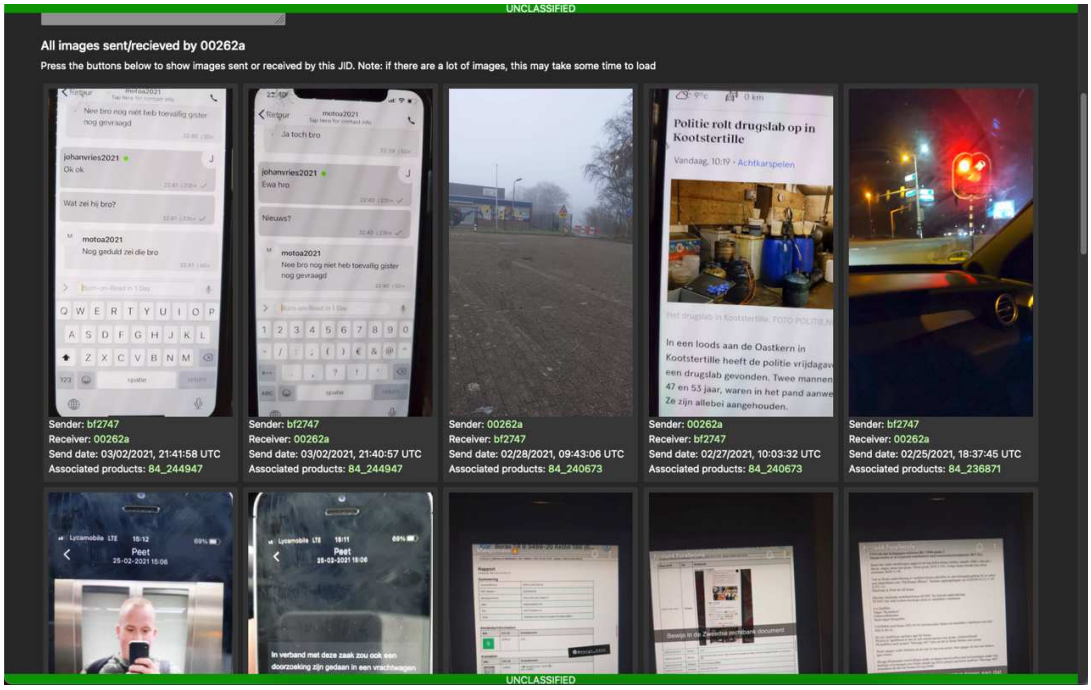
Screenshot 10.K

The remainder of section 10.7 references Screenshot 10.K.

10.7.1 Information About the Device User (JID)

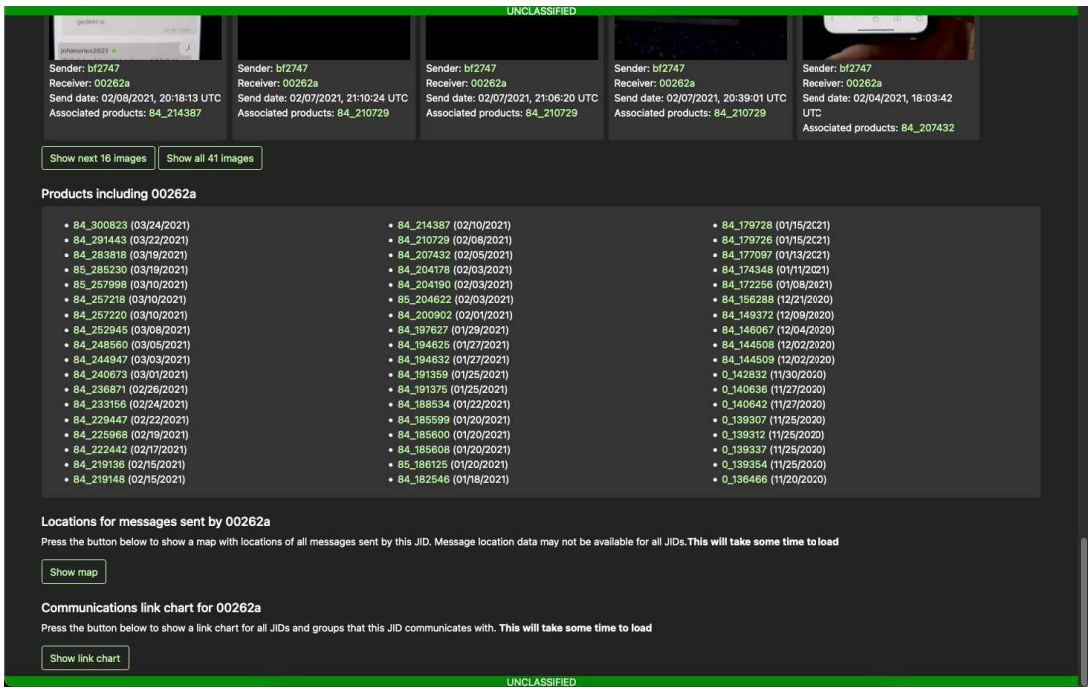
All roster data on the device user (JID) will be displayed on the JID profile page. There is additional data that can be retrieved from the JID profile page as well, such as

- Images sent by the device user.
    - “All images sent/received by 00262a” in the referenced screenshot



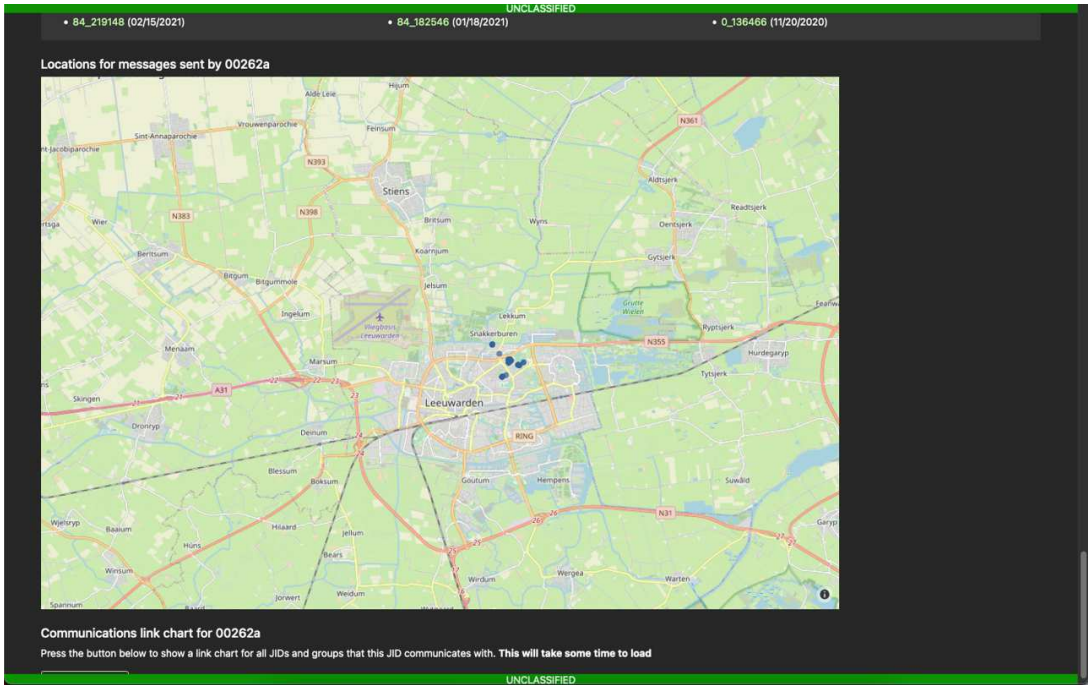
Screenshot 10.K.1

- Products that the user had sent or received messages in
  - "Products including 00262a" in the referenced screenshot



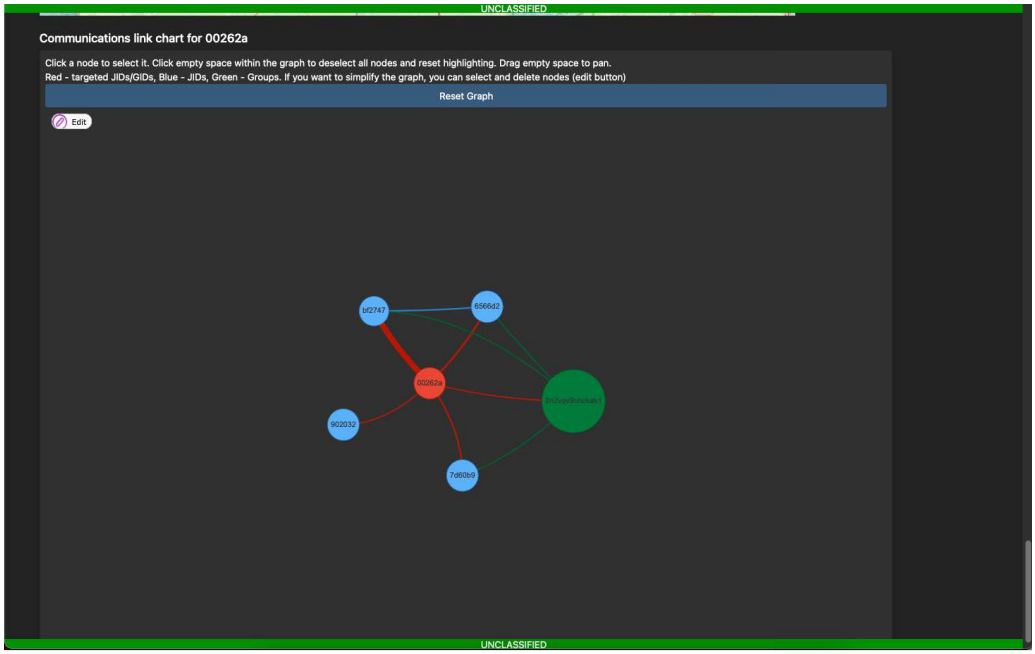
Screenshot 10.K.2

- If GPS data for messages was available it was displayed. Additional information can be found in Appendix A, 8.2.
    - “Locations for messages sent by 00262a” in the referenced screenshot



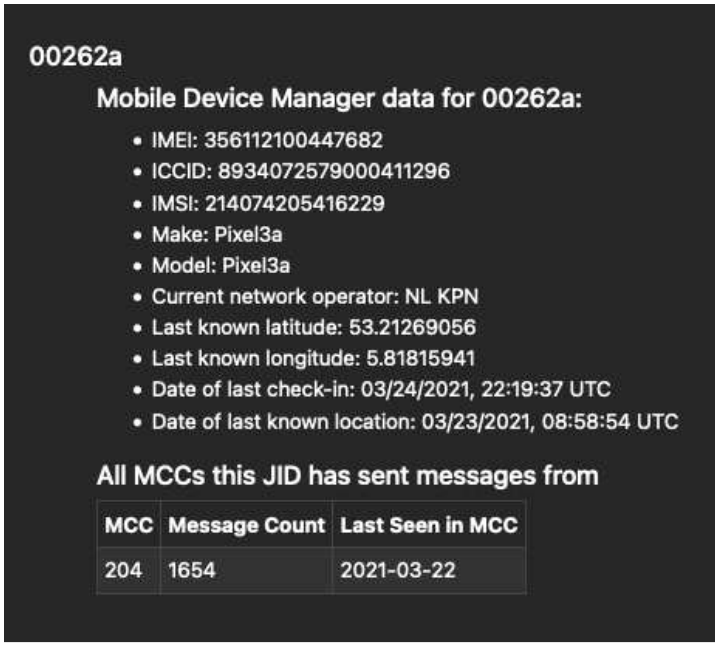
Screenshot 10.K.3

- A communications interactive link chart
    - “Communications link chart for 00262a” in the referenced screenshot
    - When hovering over a node, it displays the sending JID, case, and last MCC.
    - If double-clicked, the review platform user is navigated to the JID profile of the selected node.



Screenshot 10.K.4

- Mobile Device Manager (MDM) Information
  - As referenced in Appendix A.5 – 24.a, the ANØM platform was managed by a mobile device manager, and some of the information from the from MDM portals was ingested into the iBot_dec database for the purposes of providing additional information about device users.
  - Screenshot 10.K.5 shows a magnified screenshot of the MDM data for the JID in Screenshot 10.K.



Screenshot 10.K.5

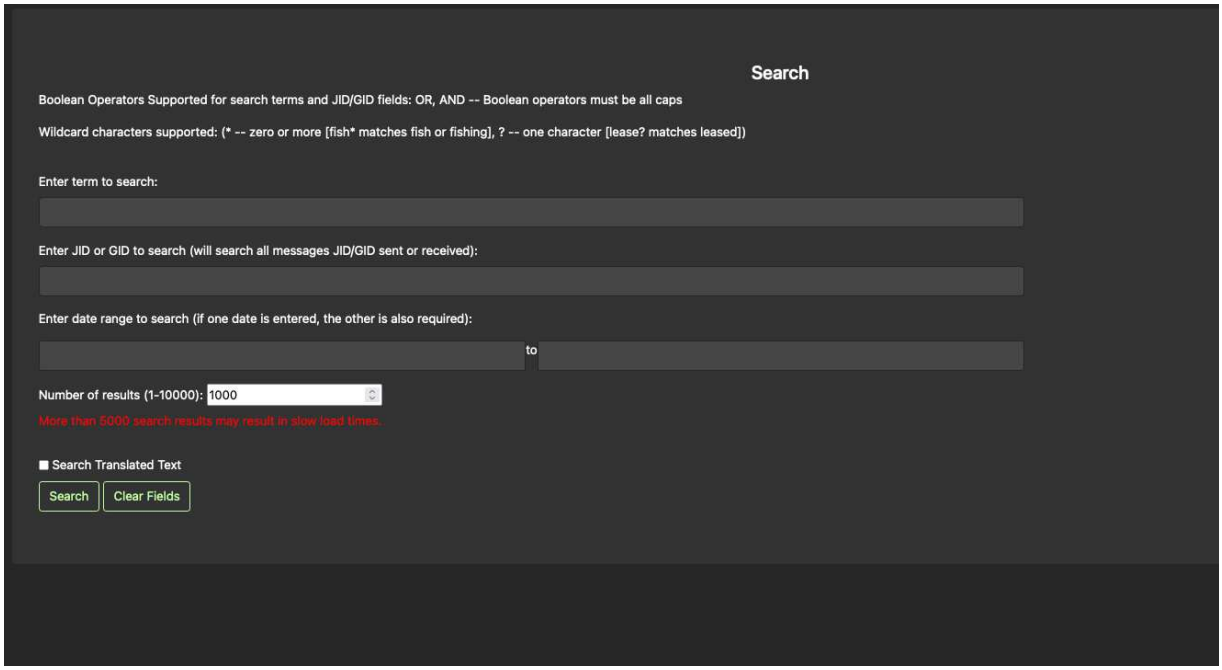
10.8 Other Pages

10.8.1 Search

The search page allows for searching within

- the ‘content’ field of messages.
- device user IDs (JIDs).
- group IDs (GIDs).

Date ranges can also be applied to filter results provided.



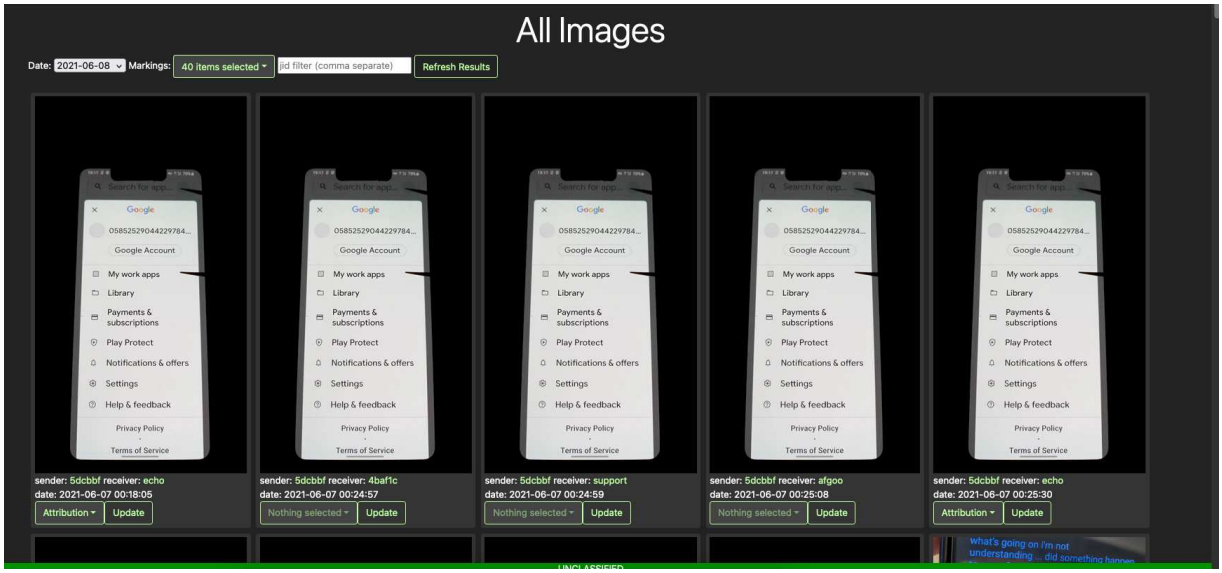
Screenshot 10.L

10.8.2 Search Notes

When review platform users add notes, they also become searchable through the Search Notes page. It has a similar interface to the Search page.

10.8.3 All Images

The All Images page provides a display page of all images within the scope of current access controls can be scrolled through, marked with attachment markings, and viewed.



Screenshot 10.M

10.8.4 Blocked

A list of messages that were blocked as described in 9.3.1.1. Additional information can be found in a future addendum.

All other pages that are navigable based on user role (reference section 8.2) are used for review platform administrators and other investigative purposes.

11. MLAT EXPORTING

MLAT requests are fulfilled by the San Diego FBI Field Office. They are submitted by different countries for a list of device users (JIDs). In order to prepare for these MLAT requests, several steps were taken to create an automated process of exporting data in a secure fashion.

11.1 All Attachment Files Pulled Down

For preparing for these requests, all attachment files were pulled down from the I1 server within AWS GovCloud. These attachment files are stored on a workstation at the San Diego FBI Field Office that can automatically create the exports.

11.2 Final MySQL Database Dump Pulled Down

Upon completion of new data packages on 6/8/2021, a full MySQL Database Dump was created and pulled down in preparation of creating exports. Timestamps of all messages exported are in Pacific Time.

11.3 MLAT Export Creation

Given a list of device users (JIDs), the database will be queried for all messages sent or received, pertinent to the device user. These messages are temporarily stored in a second database that is dumped to a '.sql' file to be added to the request fulfillment. This also includes any conversations containing the requested JID regardless of syndicate (case) assignment.

An external drive is prepared with LUKS encryption, protected with a password pre-shared with the MLAT requesters. The '.sql' file is copied to the encrypted drive, along with a list of device users (JIDs) that were requested.

The list of attachments is saved to a file while processing the data, and this file is enumerated with a second process copying each attachment file to the encrypted drive.

Following the steps described in this section will result in an MLAT export ready to be sent to the requesting country.

APPENDIX A

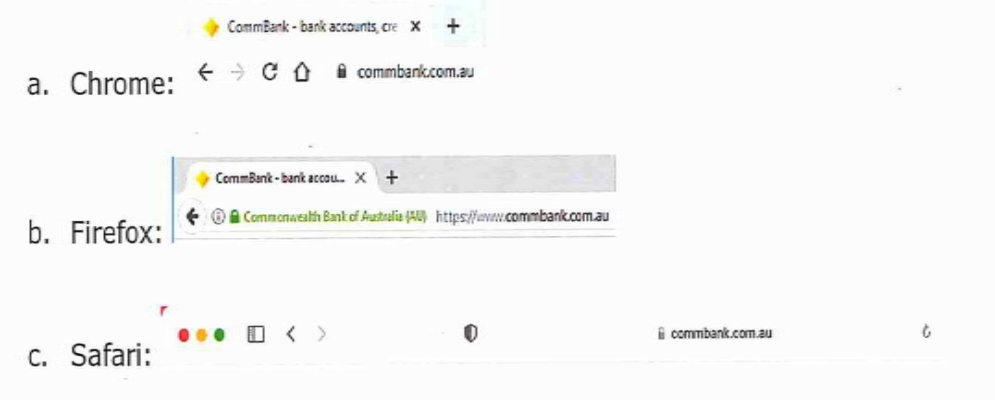
A.1 Public Key cryptography

1. Encryption involves encoding information so that it cannot be read by anyone who does not possess the means to decrypt it. Public Key (PK) cryptography is a recognized encryption approach whereby two different keys, referred to as the 'public' key and the 'private' key, are used to encrypt and decrypt information. These two keys are mathematically related, such that information encrypted using one can only be decrypted using the other. A single key cannot be used to encrypt and then decrypt the same information. This means that information encrypted using the public key is secure so long as the private key is protected. It will be illegible to anyone who lacks the private key.

2. PK cryptography has many applications and is commonplace on the Internet, Transport Layer Security (TLS) is an example of the implementation of PK cryptography.

3. TLS is a commonly used security mechanism that protects data exchanged with many internet websites and can be identified by the presence of a padlock icon in the address bar of a web browser. For example, the padlock icon is often present on websites such as banks, webmail, and Google, assuring the user the information they exchange with the site is secure. This transport security further prevents data that is being exchanged between a server and a client from being observed in transit by a third party. Any website that you visit where the padlock icon is displayed in the address bar of the web browser is using PK cryptography to protect your data.

4. Examples of the padlock icon displayed in the address bar when visiting the Commonwealth Bank of Australia website, for the Chrome, Firefox and Safari web browser applications are shown for reference:



5. Because PK cryptography requires significant processing power, it is not generally used to encrypt large amounts of data. A recognized alternative to this is to use a symmetric key, known as the session key in TLS, to encrypt the data and then use PK cryptography to encrypt this key. The encrypted key is then bundled with the encrypted data. The holder of the private key can then decrypt the symmetric key, and subsequently use it to decrypt the data.

A.2 Hashes

6. A ' hash ' function is a mathematical relationship, whereby input data of arbitrary size (referred to as the 'message') is processed into output data of fixed size (referred to as the 'message digest' or ' hash value'). Some hash functions have cryptographic applications. These include the

Secure Hash Algorithm 2 ('SHA-2') family of algorithms developed by the US National Security Agency and published by the US Department of Commerce as part of the Federal Information Processing Standards (FIPS), which are publicly available on the Internet.

7. The SHA-2 algorithms can be used to verify the integrity of data (including computer files) because it is highly improbable that two different messages will correspond to the same hash value. This includes where one message has been altered, resulting in a different message. This is explained in FIPS Publication 180-4:

> *The hash algorithms specified in this Standard are called secure because, for an algorithm, it is computationally infeasible 1) to find a message corresponds to a given message digest, or 2) to find two different messages that produce the same message digest. Any change to a message will, with a very high probability, result in a different message digest. This will result in a verification failure...*

8. The Message Digest 5 (MD5) algorithm, although no longer considered suitable for cryptographic purposes, is suitable and commonly used for purposes such as creating checksum values to verify against unintentional data corruption. It is often represented as 32 hexadecimal characters, which consist of the numbers 0 to 9, and the letters A to F.

9. A hash is designed to be an irreversible process, as it produces a fixed-length message digest for any given length message input. A simple example analogy is to take the sum of two numbers, which produces a result (for example: 23 + 78 = 101). It is extremely difficult to determine exactly which two numbers were used to produce the output result (101), but one could work through every single combination to identify combinations that do. This process would be known as a 'brute force', and each combination found would be known as a 'hash collision'. If you further complicated the equation that produces the result - for example, including multiplication, subtraction and division, in combination with the original addition - this would in turn make identifying number combinations much more difficult and time consuming. This further complication would also result in a lower subset of potential original input values.

10. Given the complexity of the hashing functions, mathematicians and crypto-analysts spend a lot of time trying to reverse the function, to prove that it is indeed a one-way function. This results in the only remaining way to identify an original input message from a hash digest is to undertake a brute force attempt, which is a prohibitively lengthy process.

A.3 THE ANØM PLATFORM

11. The ANØM communications network (ANØM) provides end-to-end encrypted, closed- network messaging capability between users of ANØM. By closed- network, I mean that the end-to-end encrypted communications can only occur between users of ANØM. It is a subscription-based service, requiring the purchase of smartphones (handsets) which are specifically configured to communicate on ANØM, and only handsets that have been accordingly set up can participate.

12. The Federal Bureau of Investigation enlisted third parties to setup and operate ANØM, including primary responsibility for development of the application, management and maintenance of the infrastructure to facilitate the ANØM functionality (ANØM administrators).

A.3.1 End-to-end encryption

13. End-to-end encryption refers to an encryption scheme where message content can only be read by the parties involved in the communication, typically involving the implementation of Public Key cryptography.

14. End-to-end encryption ensures that the original message content is not able to be read by the provider of the messaging service, or any third-party system or network which the message may be transmitted over. End-to-end encrypted messaging is implemented and available on recognized platforms such as Apple's iMessage; Facebook's Messenger and WhatsApp platforms; and the Telegram messaging platform.

15. Without end-to-end encryption, parties that are not the sender and intended recipients can access and read the original message. Examples where end-to-end encryption is not commonly implemented include Email services, where the service provider can read emails stored within an individual 'mailbox'; SMS communications, and facsimiles, where a network provider who operates infrastructure between the parties can also view the original content of the message as it transits their infrastructure. A postcard, sent via traditional public postal mail services, is a non-technical example of how without end-to-end encryption, anyone can read the message during its journey from the sender to the intended recipient.

16. ANØM uses Extensible Messaging and Presence Protocol (XMPP) to enable communications between participants of ANØM. XMPP is an industry recognized and publicly available framework that can be modified, adapted and built upon as required.

17. ANØM enables participants to send end-to-end encrypted messages - consisting of text and attachments such as images, videos, notes and short voice messages - to other users on ANØM.

18. Each user on ANØM is identified by a unique user identifier (user ID), known technically as a Jabber Identifier (JID). Initially, this user ID took the form of 6 alpha-numeric characters (for example 'aab2c3') and was automatically generated based on the handset International Mobile Equipment Identity (IMEI) number.

19. The handset IMEI number was combined with a fixed set of characters (known as a 'salt'), to create an input value. The input value was hashed using the MD5 algorithm. The last six characters of the resulting hash was the handset user ID, and the user password was the entire hash value. This process was performed automatically by the ANØM app, which used this user ID and password to automatically authenticate with ANØM servers.

20. It is possible to calculate a user ID for a given IMEI, as the MD5 hash process, including the salt value, is known to AFP.

A.4 The Provisioning Portal

21. The provisioning portal (the portal) was a website set up in January 2021 by the ANØM operator, and enabled the management of end-user accounts, handsets and subscriptions.

22. Access to the portal was limited generally to handset resellers and ANØM administrators, providing functionality to set-up new handsets for operation on ANØM; retire a handset; or initiate a remote wipe of a handset. Access to the portal was protected by a username and password, and also required the authorized user to connect via a Virtual Private Network (VPN) connection, which was only available to those authorized by the ANØM operator.

23. In December 2020, due to changes in the Android Operating System, the IMEI number was not able to be used to automatically generate the user ID. As a result, the user ID format was changed to be automatically generated by the provisioning portal. In this new process, the user ID consisted of two random English words combined together, for example: "LITTLE" and " FI SH", would create the user ID " LITTLEFISH ".

A.5 Handsets

24. Handsets used on ANØM consisted of mobile smartphones running the Android mobile operating system that:

    a. Have been enrolled into a private Mobile Device Management (MDM) application, set up and controlled by the ANØM administrators. The MDM essentially enabled the administrators to facilitate or limit certain features on the handsets; and

    b. Have the ANØM end- to-end custom encrypted communications application installed (the app). The app is only available to handsets that had been enrolled into the MDM.

25. Handsets were provisioned prior to being delivered to the intended user. By provisioned, I mean installation and configuration of the required software and settings to enable the handset to access and communicate using ANØM. Included as Annexure B and C are two documents that were provided by the ANØM administrators as guides for the provisioning process. Provisioning includes:

    a. Installation of the required version of mobile operating system, if required;

    b. If required, provision of an active Subscriber Identity Module (SIM) card, typically sourced from one of the following international mobile network telecommunications providers: Jersey Telecom (JT), Telefonica or POD Group. These SIM cards were not required to be registered in the name of the handset user;

    c. Install the MDM application and enroll the handset into the MDM;

    d. Upon successful enrolment into the MDM, the ANØM app was automatically installed onto the handset;

    e. If the installed ANØM app was a version released later than January 2021, use of the provisioning portal was required to obtain an automatically generated user ID and password to uniquely identify the handset on ANØM. This was not required for earlier versions of the ANØM app, as explained at paragraph 41; and

    f. For handsets that could not automatically generate a user ID and password, manually configure the ANØM app with the user ID and password.

26. The end-user pays the reseller a subscription fee for the period of time for which they desire to remain an active user of the platform.

27. For the majority of users of ANØM, there were no other means of communicating via these handsets. Traditional features associated with mobile handsets, such as voice/video calling, Short Message Service (SMS) messages, social media applications, and access to public internet websites and email services, cannot be used on a provisioned handset, as the ability to do so is prevented by the MDM.

28. As of 8 June 2021, there were only 73 of approximately 12,500 total handsets that did have the ability to make phone calls in the traditional sense, or browse the internet via an installed 'ToR' (The Onion Router) browser application. The ability to do so was allowed by the MDM for a specific group of users referred to as 'Corporate Owned, Personally Enabled' (COPE).

29. Due to the closed-network nature of ANØM, the private app, and specific authentication credentials required, it was not possible to inadvertently send a communication across the platform from a normal mobile phone handset.

30. Each mobile handset can be uniquely identified by its International Mobile Equipment Identity (IMEI) number. The IMEI is a 15-digit number, comprising 14 digits plus a check digit, which provides information about the handset make, model and serial number. The IMEI can often be found physically printed or located on the handset.

31. The· first 8 digits of an IMEI are known as the Type Allocation Code (TAC), which identify the make (manufacturer) and the model of a handset. The next 6 digits are the device serial number, with the final digit being a check digit calculated using the Luhn algorithm.

32. As an example, the IMEI 358503082383242 would consist of:

    a. A TAC of 35850308, which identifies this IMEI as belonging to a Samsung Galaxy Note 8; and

    b. A serial number of 238324; and

    c. A check digit of 2.

33. During the course of ANØM's operation, there have been a variety of different handset makes and models provisioned, the most common being either Google Pixel, Samsung Galaxy or Xiaomi. All handsets were running a version of the Android operating system.

A.5 Network connectivity to the platform

34. As stated at paragraph 42.b.ii, handsets may be provided to the end-user with an international SIM card, which would provide international roaming data using a mobile network located within the host country where the handset is operating. Alternatively, an end-user can utilize a SIM card from a mobile telecommunications provider they have sourced.

35. The handset user may also elect to connect to any available wireless data (WiFi) n.et work, instead of using a mobile network data connection.

A.6 The Mobile Device Management Component

36. During the operation of ANØM, there were two MDM applications and associated infrastructure utilized, MobileIron and FieldX. Both MDM applications provide similar core functionality as described below.

37. The role of the MDM component was to provide and enforce certain handset features via the application of an 'MDM policy', set by the ANØM administrators. Examples of such features include, but are not limited to:

    a. Initiate a remote 'factory reset' of a handset, which would erase all data, including applications, from the handset;

    b. Enforcement of a device password policy, ensuring that the password meets complexity requirements such as minimum length and character sets;

    c. Prevent enabling ' developer options', which would facilitate a user to enable 'USB debugging' amongst other available options, that could be used to circumvent handset restrictions, or otherwise tamper with the handset's integrity;

d. Maintaining a secured handset boot loader, which ensures only the platform approved mobile handset operating system is allowed to run, ensuring handset integrity and security;

e. Prevent installation of untrusted or unapproved applications, which ensures only applications available through the ANØM MDM can be installed and run on the handset;

f. In the case of the FieldX MDM, ensuring the ANØM app is updated automatically;

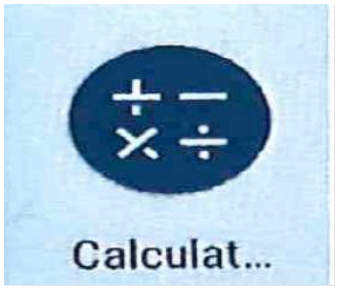
g. Reporting handset information periodically, such as but not limited to, unique handset IMEI, compliance with MDM policy, handset boot loader and operating system versions, and handset status such as battery and internal storage levels.

38. The MDM component utilizes functionality that is part of the core Android mobile operating system on the handset, communicating with MDM servers to report handset status, and retrieve policies to enforce on the handset.

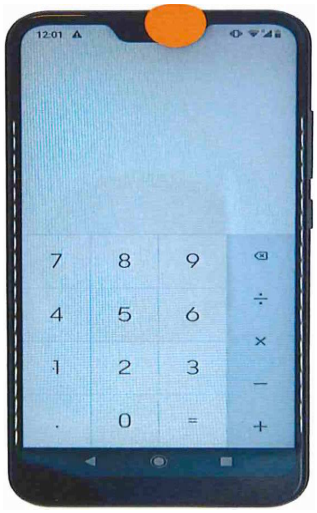
A.7 The ANØM Application

39. The ANØM application (the app) appears on the handset, under the disguise of a working 'calculator' application. Both the app icon, and calculator label, do not reveal the application's true purpose. Examination by an unknowing user would not indicate the presence of an encrypted messaging application.

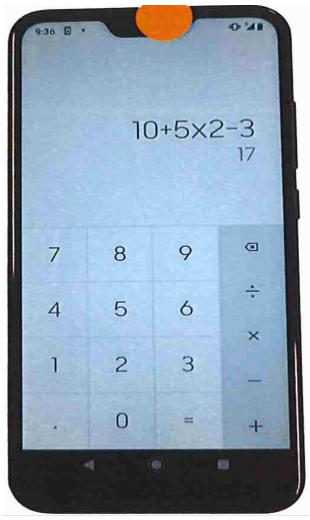
40. An example of the app icon, as would have appeared on some handsets is shown (the style of icon was dependent on the operating system version):



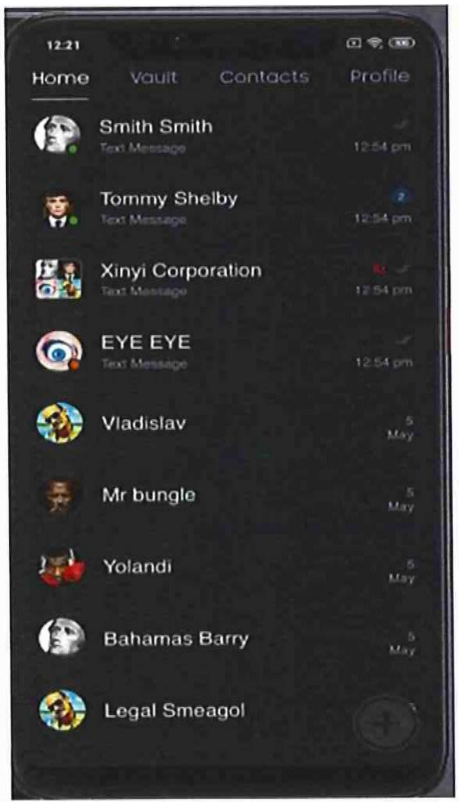
41. Once the app is launched, the user is presented with an interface that continues to reflect that of a calculator:



42. This calculator functions as expected, in that a user can enter mathematical functions, and the correct result is returned.



43. If the user enters a secret personal identification number (PIN) (access PIN), and then presses and holds the equals ('=') symbol, the true platform application appears:



44. Along with the app access PIN, there also exists a 'duress' PIN, which if used instead of the access PIN, along with a long press on the equals symbol, initiated a deletion of all information from within the app.

45. The app duress PIN could, for example, be provided to an unauthorized user of the handset if the owner is coerced or otherwise agreed to voluntarily unlock the app. While access to the app would still occur, all data would be deleted from within the app. An example of where this might occur is if law enforcement were to seize the phone and require the access PIN, the authorized user could instead provide the duress PIN, resulting in the appearance of cooperation but in fact would trigger the removal of any potential evidentiary material.

A.7.1 ANØM Application Attributes

A.7.1.1 User Identifier

46. Each user of the platform is identified by a unique user ID. Only one handset, at any given point, could access ANØM as a particular user ID. Once the user ID and password were entered into the app, the app automatically updates the password associated with that ANØM user ID. This updated password is stored by the app for subsequent login, however it is never presented or known to the user who entered the values. Attempts to login on another handset with the previously known password and user ID would fail, and need the password to be reset to a known value to succeed

A.7.1.2 User Display Name

47. The user of the handset can set a 'display name' (also known as a nickname, alias, or handle), which can be changed whenever the user desires. This does not change their user ID on the plat

form. This is similar to the "From" name displayed in emails that typically differs from the email address, which is often the username;

A.7.1.3 Vault

48. The vault enables the user to store notes and images on the device securely. This data is encrypted such that it is only accessible from the handset, and no data in the vault is transmitted unless the user includes it as an attachment to a message.

49. Notes stored in the vault can contain a title, along with body text or an itemized list. Images stored in the vault, or taken using the handset camera, can also be included in the body of notes.

A.7.1.4 Messages

50. A user can send an end-to-end encrypted message to another user on ANØM, providing they know the intended recipient's user ID;

51. A user can also send an end-to-end encrypted message to a group of ANØM users, and each user in that group can see the other members of the group. This group can have a label, naming the group, which can be set by any group administrator, who is typically the creator of the group, and any other members that have been granted administrator privileges for that group. This is similar to how a group chat in other messaging applications such as WhatsApp, Signal, Telegram, and Facebook Messenger function

52. A message, either to a single user or a group of users, can consist of a text message, an image (for example, a photo taken by the handset camera), a video obtained using the device's camera, a short voice (also known as Push- To-Talk, or PTT) message, or a 'notes' document from the device vault;

53. Messages can contain different languages. In order to support various language character sets, computer applications can use Unicode as a method for representing symbols and text from most of the world's writing systems. Unicode provides a unique number for every character, no matter what platform, device, application or language. It does so by expressing characters as a code number, not a glyph. For instance, the Arabic characters "ا ب و "would be represented as "\u0627\u0628\u0648" in Unicode. Symbols commonly known as 'emojis' can also be expressed as a Unicode value. For instance, "\u1F603" is the Unicode value for the emoji "😃". These Unicode values need to be converted back into the correct characters in order to be legible. Applications may interpret the Unicode characters and display the character, rather than its code;

54. A message can be quoted by selecting it in a conversation and pressing the 'reply' button, or forwarded to other recipients by selecting it and pressing the 'share' button. When this occurs, the parties to the message can visually see that the message has been quoted or forwarded, as it will appear as a quoted block within the sent message;

55. Unlike other messaging applications such as WhatsApp, or Telegram, the ANØM app did not provide indicators if a message had been received or read by the parties;

56. Audio pitch-shifting ability

57. The sender of an audio message has the option to adjust the pitch of the recording prior to it being made, either up (option known as 'Helium' in the app) or down (option known as 'Jellyfish'

in the app), which disguises the true sound of the sender's audio to the recipient. The sender can also opt for no pitch-shift to occur;

A.7.1.5 Self-Destructing Messages

58. The sender of any message type described above can also nominate to set a 'self-destruct' timer for that message, known as a 'burn time'. Setting the burn-time would remove the message from any handset which has received the message once the timer period has elapsed.

A.8 Data Recorded from the Platform

59. The ANØM app included a feature whereby when a message was sent, the app automatically sends a 'blind carbon copy' of the message to a ghost user ID, 'bot'. This process was not visible to the sending user, nor were they aware that this process occurred.

60. The automatic blind carbon copy process resulted in a copy of the message encrypted for, and sent to, the ghost user ID 'bot'. Computers running an application compatible with the platform received these encrypted messages addressed to the 'bot' user ID.

61. This process was identical to the normal platform app behavior when a message is sent, with the exception that the messages sent to the ghost user ID were not visible to the other parties of the communication.

62. A diagram depicting the message flow, and the encryption process for a hypothetical message sent from Alice to Bob, including the blind carbon copy and encryption for the ghost user ID 'bot', is included in Section 1.5.

63. A.8.1 Message contents
64. Messages that are sent using the platform usually have the following, or a combination of the following, attributes that are visible to both sender and recipients:

    a. Burn Time - This is set by the sender at the time of sending a message and defines how long a message will exist before being automatically deleted from both devices once that time has 'elapsed. It is visible on both the sender and recipient handsets, at the bottom of each message. It can be set to 30 seconds; 5 or 30 minutes; 1 or 12 hours; or 1, 3 or 7 days;

    b. Sender - The sending handset user ID and display name;

    c. Recipients - The recipients to a message can be identified by viewing the parties in the relevant message thread, which like other common chat applications such as WhatsApp, Telegram, and Facebook Messenger, appear as separate items in a message list;

    d. Content - The message content, which can consist of text or an attachment such as audio, an image, video or note.

    e. Time - The time and date in Greenwich Mean Time (GMT) according to the sending handset, when a message was sent. This would be displayed on the handset in the configured time zone. For example, a message sent on 10 June 2020, at 11: 00 am Perth time, would have the time and date as 3: 00 am on 10 June 2020 GMT. A handset in Sydney would present this time as being 1: 00 pm on 10 June 20 20 . Handsets which had an active SIM card would synchronize their time via Network Time Protocol (NTP) to the telecommunications carrier network to which it is connected.

i. See section 8.6.1 for time zone discrepancies.

A.8.2 Message meta-data

65. Messages sent over ANØM included additional meta-data that was not visible to either the sender or recipients of a message via the app . Some of these fields were included for the benefit of Law Enforcement Agencies (LEA). Due to changes and enhancements not all the following meta-data fields, or combinations of, were sent with each message:

    a. A Unique Message ID - A Universally Unique Identifier (UUID), automatically generated by the sending handset application at the time a message is sent, and is not displayed to either the sender, or recipients of a message. This is a component of the XMPP framework;

    b. IMEI - On handsets running the Android operating system version 9 or less, the IMEI number was included as meta-data with each message for LEA use;

    c. MCC / MNC - On handsets that contained an active SIM card, and were currently registered to a mobile telecommunications network, the Mobile Country Code (MCC), and Mobile Network Code (MNC), to which the handset is currently connected. These numbers were included as meta-data with the message for LEA use;

    d. Location Information - Following an app update in April 2020, the app would attempt to obtain a location via the Android Location Manager Application Programming Interface (API). If a location could be ascertained, the latitude, longitude, accuracy, and location acquisition time values, were included as meta-data with the message for LEA use. Occasionally, the app was unsuccessful in obtaining a complete and accurate GPS location. When this occurred, the GPS point collected often began with 39.0 and resolved to an area off the coast of Fiji in the South Pacific Ocean;

    e. Audio pitch adjustment - If the message contained an audio attachment, then meta-data which indicates the value of pitch-shift performed was included as meta-data with the message for LEA use;

    f. Quoted or forwarded messages - If the message contains a quote, or is a forwarded message, then the new message content will contain the quoted or forwarded content, which may also include information such as the original sender user ID, and the original message time. This was a feature of the ANØM application, and messaging applications such as WhatsApp, Signal, Telegram and Facebook Messenger have a similar feature to quote and forward messages.

A.9 Authenticity and Integrity

66. The ANØM app behavior of updating a user ID password upon initial successful login, effectively prevents another handset from authenticating and being able to send messages purporting to be another ANØM user ID. Additional information will be included in a future addendum.

67. Until the user ID format change in December 2020, the user ID was directly related to the IMEI number of the handset, further limiting the ability for ANØM messages to be sent

68. If a handset was 'factory reset', initiated either from the MDM, or on the handset itself, all data and applications - including the MDM and platform app - was removed from the handset.

69. If the app duress PIN was entered, all data, including the saved login information, was removed from the app. This would result in the user being unable to access or communicate on ANØM.

70. If either of these processes occurred, then the user ID and password would need to be reset before the handset could be used to access ANØM again. In the case of a factory reset, the handset would need to be reprovisioned.

GLOSSARY

- **AES Encryption**: Symmetric (single-key) encryption that was used for the other part of the re-encryption of data collected.
- **ANØM**: An end-to-end encrypted communications platform, utilizing dedicated mobile phone handsets with a custom messaging application installed to communicate within a closed network of participants.
- **Application Programming Interface (API)**: A software interface, commonly utilized with computer-to-computer interaction, whereby a client computer initiates a request, which is responded to by the server computer running the API, typically returning data to the requesting client computer.
- **AWS GovCloud**: Amazon Web Services cloud infrastructure where the Hola iBot web application and data related to Operation Trojan Shield was stored.
- **Boot Loader**: The first item of software that runs on a mobile phone handset upon device power up or reboot, which initializes the hardware and loads the mobile Operating System, for example, Android.
- **Cases**: Also known as Syndicates. This is the central component of relationships within the iBot_dec database.
- **Difference-Dump**: The new or changed data between two MySQL database dumps. This is how the third-party country parceled out new content from the entire database when sending new data packages.
- **Elastic Compute Cloud (EC2)**: AWS-owned computing resources that can be rented for the purpose of running computer applications.
- **End-to-end encryption**: An encryption scheme whereby data is encrypted between two end points, and only those end points have the ability to decrypt the data. The servers and networks that relay the data between the two end points are not capable of decrypting the data.
- **Extensible Messaging and Presence Protocol (XMPP)**: XMPP is an open standard protocol for instant messaging, which can facilitate multi- party chats, supporting multimedia such as voice, images and video. Anyone is free to run their own XMPP server and network and build upon the framework that is provided.
- **Ghost User ID ('bot')**: The user identifier to which the AN0MANØM application installed upon mobile handsets would send a blind carbon copy of each message which was sent from that handset.
- **GnuPG (GPG)**: allows for data encryption and signing for communications.
- **Google Compute Engine**: A service that Google Cloud provides to create and run virtual machines within Google Cloud.
- **Hash (MD5, SHA-2)**: A hash function is a mathematical process whereby the data input, or 'message', is processed into a 'message digest' or hash value of fixed length. It is commonly represented as hexadecimal characters which consist of the numbers Oto 9, and letters A-F. Hash functions are often used for data verification to identify if data has been altered, due to

the very high improbability that two different input 'messages' will result in the same hash value.

- **Hola iBot**: Custom web application built and operated by San Diego FBI to serve as the review platform of all data received from the ANØM platform.
- **iBot_dec**: The persistent database that serves data to the Hola iBot review platform. Contains data collected on the AN0MANØM platform, and presents it for review
- **iBot_enc**: The temporary database which would hold new data packages when new data was received from the third-party country's server.'
- **Ingestion-1 Server (I1)**: The AWS GovCloud EC2 that ran the Ingestion process (section 9), retained the attachment files of messages, and served the iBot_dec database
- **International Model Equipment Identifier (IMEI)**: A 15-digit number assigned to a mobile handset which can be used as an identifier. It consists of an 8-digit Type Allocation Code (TAC) that reports the manufacturer and model of the handset, a 6-digit serial number, and the last number being a 'check digit' using the Luhn algorithm.
- **Location Information**: Information that identifies a location, which may be a region or area, and typically consists of latitude and longitude values as well as a timestamp in Universal Co-ordinated Time (UTC) of the location fix. The location may be obtained from Global Navigation Satellite Systems (GNSS), such as GPS or GLONASS, or determined based on information about nearby mobile cell towers and WiFi access points.
- **Mobile Country Code (MCC)**: The Mobile Country Code uniquely identifies the country of origin for the mobile subscriber, and is the first three digits that make up the International Mobile Subscriber Identity (IMSI) number, securely stored within the Subscriber Identity Module (SIM) card. For example, Australia has the MCC value of 505.
- **Mobile Device Management (MDM)**: A software application, consisting of a mobile device component, and a server component, which is commonly used by entities that wish to enforce certain device features, known as a 'policy', to assist in device integrity, security and useability. MDM's also commonly possess the ability to remotely wipe a device to prevent information
- **Mobile Network Code (MNC)**: The Mobile Network Code can consist of two or three digits which identifies the home mobile network for the mobile subscriber. It is the two or three digits following the MCC that make up an International Mobile Subscriber Identity (IMSI) number, securely stored within the Subscriber Identity Module (SIM) card. The following are Australian mobile netw9rk operator example values (not a comprehensive example); Telstra is 01, Optus is 02, and Vodafone is 03.
- **MySQL Dump**: a logical backup of all tables, including content, within a MySQL Database.
- **New Data Package**: A package of new data collected from the ANØM platform sent by the third-party country to the I1 server.
- **Portal**: A website accessible over the internet, with the use of a private Virtual Private Network connection, requiring a valid username and password provided by the ANØM administrators, to assist with the ANØM end-user device, account and user ID management.
- **Proxy**: A server that acts as a relay of all network traffic between two computers or servers

- **Public Key Cryptography/ Public Key Encryption**: An encryption and decryption scheme whereby a pair of keys that are mathematically related are used to encrypt and decrypt content. The 'public' key can be used to encrypt content, which only the 'private' key can decrypt, and vice-versa. The same key cannot be used to decrypt content that was encrypted by it.
- **RAM**: Temporary computer storage. For example, this is where all temporary data structures would be stored for ingestion.
- **RSA Encryption:** A type of PKI encryption scheme that was used for part of the re-encryption of data collected.
- **Subscriber Identity Module (SIM)**: The Subscriber Identity Module is a physical integrated circuit card that securely stores the International Mobile Subscriber Identity (IMSI) number and its related authentication key, used to uniquely identify and authenticate a mobile subscriber (end-user) to a mobile network. The SIM card also contains a unique serial number, known as the Integrated Circuit Card Identifier (ICCID) which may be printed on the SIM card itself.
- **Transport Layer Security (TLS)**: An implementation of Public Key Cryptography, commonly used to secure information exchanged with internet websites such as banking, email and internet search engines. Its use can be identified by the presence of a closed padlock icon present in the address bar of a web browser. TLS can also be used to secure communications for other purposes, such as instant messaging applications and Voice over Internet Protocol (VoIP).
- **Unicode**: Unicode is a method for representing symbols and text from most of the world's writing systems. This is achieved by expressing characters as a number, not the glyph. 'Emojis' are a common glyph that would be expressed using the Unicode value for example in messaging applications but presented to a human as the glyph.
- **USB Debugging**: The ability to connect a computer to a mobile handset via a Universal Serial Bus (USB) interface cable, and interact with the mobile operating system. It can be used to add or remove data from a handset, read logs, or install and remove applications. It can also be used to replace the operating system on a mobile handset, or install processes that run with elevated privileges - also known as 'root' a device.
- **User ID (JID)**: A username that uniquely identified a user on the AN0MANØM platform. Initially comprising of six characters consisting of the numbers Oto 9, and letters A to F; later, this became two random English words combined together.
- **Virtual Private Network (VPN)**: A networking application used to encrypt data exchanged over an untrusted network connection, commonly the Internet. VPN's were traditionally used to ‘extend' a private network, that is one not accessible if not directly connected such as a corporate network of computers, over an Internet connection. This would enable a remote device to access websites, files, data and services that are not accessible outside the private network. VPN's can also be used to masquerade the identity or destination of a user's internet activity from their Internet Service Provider (ISP).
- **XMPP**: *See Extensible Messaging and Presence Protocol (XMPP)*