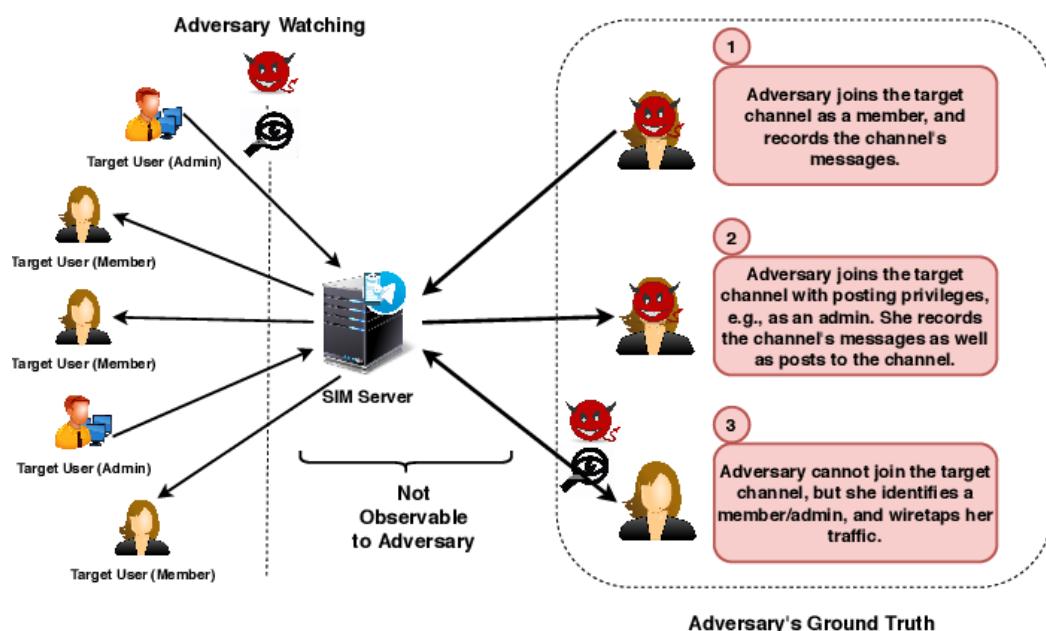


# Practical Traffic Analysis

## Attacks on Secure Messaging Applications



by: Liraz balas 211801220  
Vivian Umansky 322880857

## **Table of contents**

Dry part -----	3-4
Beginning of the research -----	5-8
Event-Based Detector-----	9-20
CDFF explanations -----	21
Summary & Conclusions-----	22-23
bibliography -----	24

## Dry Part

The attacker obtains the ground truth about the traffic of the target IM channel in one of three ways: Joining an open channel: If the target IM channel is an open or public channel, the attacker can join the channel as On can be actively active in the channel, record the messages sent along with their metadata, such as the time and size of the messages. This information provides real information about the traffic patterns of the channel.

Sending messages to the channel: A rule that in cases where the attacker has already joined the target's instant messaging channel sent the message to post messages. This can happen if the channel is a closed group that allows any member to post messages or if an admin role has been accepted for the channel. By recording the incoming and outgoing messages, the attacker can capture the traffic patterns of the channel including its messages with different traffic patterns.

Member/admin eavesdropping: If the attacker is unable to join the target channel as a member or administrator, he can still obtain ground truth information by detecting the IP address of one of the channel's members or administrators. The attacker then eavesdrops on the identified member/manager's network traffic, intercepting their encrypted communications. By recording the traffic patterns of the detected member/manager, the attacker can use this information as ground truth to match the traffic patterns of other intercepted flows.

In summary, the attacker obtains the ground truth about the traffic of the instant messaging channel by actively participating in the channel, intercepting the traffic of administrators/members, or by listening to the network traffic of specific people associated with the channel.

Furthermore the attacker eavesdrops on the network traffic of IM users to identify the IP addresses of the members/administrators of the targeted IM channel. The specific methods mentioned include:

Eavesdropping on Internet Service Providers (ISPs) or IXPs: The attacker can control or gain access to network infrastructure, such as Internet Service Providers (ISPs) or Internet Exchange Points (IXPs). By eavesdropping on the network traffic passing through these control points, the attacker can intercept and monitor the encrypted network traffic of instant messaging users.

Eavesdropping on certain people: In some cases, the attacker may target certain people, such as suspected activists. After obtaining a wiretapping warrant or gaining unauthorized access, the attacker intercepts the network traffic of these individuals to record their traffic patterns.

It is worth noting that eavesdropping on network traffic usually involves capturing and analyzing the data packets transmitted over the network. This can be done by various means, such as deploying network monitoring devices, using special software or hardware tools, or compromising the network infrastructure or endpoints to gain access to traffic. The exact details of how the attacker communicates with network traffic can depend on the technological capabilities and resources available to the attacker. It can be seen in the study that Table II provides an overview of the distribution of the different types of messages in the collected instant message traffic. The table shows statistics such as the count, volume (in MB), size range, and average size for each message type, including text, image, video, file, and audio messages. From the table, we can observe the relative proportions of different message types in the instant messaging traffic database. For example, text messages account for about 29.4% of the total, followed by images (48%), videos (15.4%), files (2.1%) and audio messages (5.1%).

In addition, the table provides insights into the size characteristics of different message types, such as their size range and average values.

This data is essential for understanding the composition and characteristics of instant message communication, which informs us about building models and analyzing instant message traffic, as well as developing traffic analysis attacks.

In communication systems, each SIM event, such as sending an image, produces a bubble of MTU-sized packets in the cryptographic traffic. These packets have tiny delays between the packets and their size corresponds to the size of the MTU.

The bubbles represent SIM events, while scattered packets of small size represent the SIM protocol messages such as alert messages, manuals, updates, etc. To detect these bubbles and reveal the SIM events, the researchers use Inter-Packet Time Threshold (IPD), which is denoted as "te".

Two packets with a time distance smaller than te will be considered to be part of the same bubble.

The te value is a hyperparameter term in the model, and its choice is discussed in the paper.

For each bubble detected using te, the enemy can reveal a SIM event.

The time of the arrival of the last packet in the bubble indicates the arrival time of the event, and the sum of all the sizes of the packets in the bubble gives the size of the event. In addition, two SIM messages sent with an inter-message distance (IMD) less than te will be considered part of the same event.

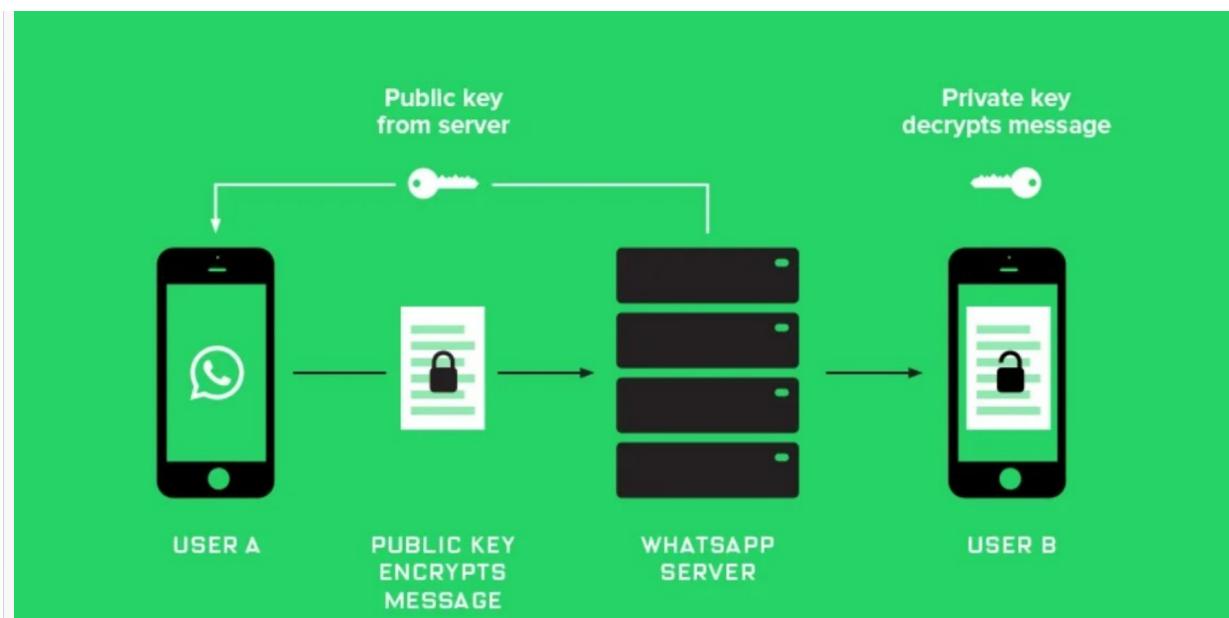
The adversary combines events that are closer than te when running a grow on the target channel, this approach allows the adversary to detect SIM events by looking for bubbles of MTU-sized packets, even though the contents of the packets remain encrypted and inaccessible.

## Beginning of research:

In this part of the project, we tried how to see in the most abstract way the relationship of our computer with the WhatsApp server, including all communications that call the application.

We discovered that it's not that simple, as we know, WhatsApp is an end-to-end encrypted application  
From Wikipedia:

In November 2014, Open Whisper Systems announced a collaboration with WhatsApp to establish an infrastructure for end-to-end encryption with the Signal protocol. After the establishment of the infrastructure in April 2015, the WhatsApp company stated that all forms of communication of the application are strongly encrypted (WhatsApp conversations are encrypted using the SRTP protocol) in such a way that, according to the company's claim, no one, including Facebook itself, which owns WhatsApp, and not even the NSA or any other government organization can get access to any Misron content or conversation. In October 2015, WhatsApp began to transfer all the messages of the



application's users to Google servers using Google Drive by default. This step raised many doubts regarding the company's statement that no third party will be able to read the users' messages, because all the user's messages are uploaded to Google's servers without encryption on the part of WhatsApp, but only if there is encryption by the server.

Starting in April 2016, WhatsApp began to encrypt all users' messages, photos, videos, voice messages, documents and conversations, using end-to-end encryption. And she claimed that: "When you send a message, the only person who can read it is the person or group you are sending the message to. No one will be able to read your message: not cyber criminals, not hackers, not oppressive regimes, not even us."

photo: <https://www.urtech.ca/2020/07/how-to-verify-that-whatsapp-messages-are-encrypted/>

As part of our attempts, we were not able to record with wireshark packets that pass through our network when we enter the application from the mobile, all our attempts within the framework of our knowledge, failed to understand how it is possible to identify any activity from the mobile. Therefore, we focused on investigating the packets that are transferred between our computer (which is connected via the browser to the application) to the WhatsApp server. First, to understand the process, we recorded a conversation between two people on WhatsApp so that we are connected to the application through our computer, and the other party is not in our vicinity. In this conversation, text messages, voice messages, photos, videos and files were transferred.

In addition, we tried to understand which protocols the application uses, and we found that there are several protocols that WhatsApp uses as part of its services:

WhatsApp uses a combination of protocols to provide secure and efficient messaging services. These protocols ensure confidentiality and efficiency.

1.TLS: Encrypts the connection between the user's device and the WhatsApp servers.

This ensures privacy and prevents unauthorized access to data during transmission.

No.	Time	Source	Destination	Protocol	Length	Frame	Server Name	Info
1 0.0000000000	192.168.86.38	157.240.253.60	192.168.86.38	TLSV1.2	343 ✓			Application Data
2 0.000028341	192.168.86.38	157.240.253.60	192.168.86.38	TCP	66 ✓			48446 → 443 [ACK] Seq=1 Ack=278

2.TCP: provides reliable delivery, ensuring that messages are received in the correct order without data loss, especially in text-based communication.

0 2.39000 / 471	192.168.86.38	157.240.253.60	TLSV1.2	141 ✓	Application Data
7 2.424211149	157.240.253.60	192.168.86.38	TCP	66 ✓	443 → 48446 [ACK] Seq=278 Ack=159
8 2.457914510	157.240.253.60	192.168.86.38	TCP	66 ✓	443 → 48446 [ACK] Seq=278 Ack=234

3.UDP: Faster data transfer, used for real-time communication, such as voice calls and maybe some multimedia sharing.

Improves user experience by reducing delays in time-sensitive interactions.

4.QUIC: Optimizes data transfer over unreliable networks, combining features of TCP and UDP. Especially messages and voice calls.

No.	Time	Source	Destination	Protocol	Length	Frame	Server Name	Info
1 0.0000000000	192.168.86.38	157.240.253.60	157.240.253.60	QUIC	1399 ✓		pps.whatapp..	0-RTT, DCID=6ec9a8ed868e50cf, SCI
2 0.120043722	157.240.253.60	192.168.86.38	192.168.86.38	QUIC	1274 ✓			Initial, DCID=2b7bde, SCID=b51d00
3 0.120072451	157.240.253.60	192.168.86.38	192.168.86.38	QUIC	254 ✓			Handshake, DCID=2b7bde, SCID=b51d00

And back to the conversation we recorded-

In Wireshark you can see the packages relevant to the call as follows:

In the above recording you can see many packets that pass through different protocols from our computer (10.100.102.10) to the server 157.240.251.60 which is the server of WHATAPP WEB.

No.	Time	Source	Destination	Protocol	Length	Frame	Server Name	Info
10.000000000	10.100.102.10	157.240.251.60	QUIC	1399✓			web.whatssapp...	0-RTT, DCID=f793efb3a38fd9b5, SCID=8ef07, DCID=8ef07, SCID=401d43
20.066538928	157.240.251.60	10.100.102.10	QUIC	1274✓				Initial, DCID=8ef07, SCID=401d43
30.067126543	10.100.102.10	157.240.251.60	QUIC	82✓				Handshake, DCID=401d43339543eb66,
40.068743158	157.240.251.60	10.100.102.10	QUIC	254✓				Handshake, DCID=8ef07, SCID=401d
50.068770229	157.240.251.60	10.100.102.10	QUIC	90✓				Protected Payload (KPO), DCID=8e8
60.068796877	157.240.251.60	10.100.102.10	QUIC	122✓				Protected Payload (KPO), DCID=8e8
70.069496744	10.100.102.10	157.240.251.60	QUIC	127✓				Handshake, DCID=401d43339543eb66,
80.090465541	10.100.102.10	157.240.251.60	QUIC	77✓				Protected Payload (KPO), DCID=401
90.126852795	157.240.251.60	10.100.102.10	QUIC	84✓				Handshake, DCID=8ef07, SCID=401d
100.127559714	157.240.251.60	10.100.102.10	QUIC	154✓				Protected Payload (KPO), DCID=8e8
110.127640502	157.240.251.60	10.100.102.10	QUIC	314✓				Protected Payload (KPO), DCID=8e8
120.128539741	10.100.102.10	157.240.251.60	QUIC	77✓				Protected Payload (KPO), DCID=401
133.999840426	10.100.102.2	10.100.102.255	UDP	77✓				60420 → 15600 Len=35
144.000669793	10.100.102.2	10.100.102.255	UDP	77✓				51535 → 15600 Len=35
1516.096889705	10.100.102.2	10.100.102.255	UDP	77✓				35742 → 15600 Len=35
1622.137491520	10.100.102.2	10.100.102.255	UDP	77✓				57452 → 15600 Len=35
1728.005624097	10.100.102.2	10.100.102.255	UDP	77✓				41310 → 15600 Len=35
1834.005752453	10.100.102.2	10.100.102.255	UDP	77✓				43096 → 15600 Len=35
1938.037849477	10.100.102.10	62.0.32.32	QUIC	1399✓			media.fhfa2...	0-RTT, DCID=d29fb84c5815270a206e,
2038.039450743	10.100.102.10	62.0.32.32	QUIC	473✓				0-RTT, DCID=d29fb84c5815270a206e,
2138.044303729	62.0.32.32	10.100.102.10	QUIC	1274✓				Initial, DCID=c154c7, SCID=4007088e7233960d,
2238.045215490	10.100.102.10	62.0.32.32	QUIC	82✓				Handshake, DCID=4007088e7233960d,
2338.052091078	62.0.32.32	10.100.102.10	QUIC	255✓				Handshake, DCID=c154c7, SCID=4007
2438.052133599	62.0.32.32	10.100.102.10	QUIC	90✓				Protected Payload (KPO), DCID=c15
2538.052146425	62.0.32.32	10.100.102.10	QUIC	122✓				Protected Payload (KPO), DCID=c15
2638.052153572	62.0.32.32	10.100.102.10	QUIC	90✓				Protected Payload (KPO), DCID=c15
2738.053587686	10.100.102.10	62.0.32.32	QUIC	161✓				Protected Payload (KPO), DCID=400
2838.057809789	62.0.32.32	10.100.102.10	QUIC	84✓				Handshake, DCID=c154c7, SCID=4007
2938.060182444	62.0.32.32	10.100.102.10	QUIC	378✓				Protected Payload (KPO), DCID=c15
3038.081402193	10.100.102.10	62.0.32.32	QUIC	77✓				Protected Payload (KPO), DCID=400
3138.262324140	62.0.32.32	10.100.102.10	QUIC	1274✓				Protected Payload (KPO), DCID=c15

All the packets that are transmitted by the QUIC protocol, probably in the picture, are the packets that transmit the information of the WhatsApp conversation between two people. You can see that all the information is encrypted and cannot be accessed.

```

4 Frame 46: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface wlp2s0, id 0
4> Ethernet II, Src: ADBITAL_29:45:15 (10:5a:f7:29:45:15), Dst: Epigram_0d:f4:3e (00:90:4c:0d:f4:3e)
4> Internet Protocol Version 4, Src: 157.240.251.60, Dst: 10.100.102.10
4> User Datagram Protocol, Src Port: 443, Dst Port: 45731
4> QUIC IETF
5   > QUIC Connection information
5   [Packet Length: 48]
5   > QUIC Short Header DCID=8ef07
5   > [Expert Info (Warning/Decryption): Failed to create decryption context: Secrets are not available]
5   [Failed to create decryption context: Secrets are not available]
5   [Severity level: Warning]
5   [Group: Decryption]
5 Remaining Payload: ab83eaf... 0000 00 90 4c 0d f4 3e 10 5a f7 29 45 15 08 00 45 00 ...L->Z )E..E.
5 0010 00 4c 00 00 40 00 55 11 1c 06 9d f0 fb 3c 0a 64 :L-@U...<.
5 0020 66 0a 01 bb b2 a3 00 38 9d 78 4b 8e 8f 07 ab 83 f....8 xK...
5 0030 ea fd 48 cc c6 5e 08 d8 f4 26 31 b8 b9 56 61 d2 ..H.^...&Va.
5 0040 7a e9 54 3b ca bc 2d 0f cd 72 97 14 d1 59 d0 51 z-T;...r...Y.Q
5 0050 18 77 c2 77 1a 99 0d 97 09 a6 .w-w.....

```

Additionally, we created the **messages\_over\_time.py** program which simply shows us the size of the packets sent on the network throughout the call, You can see the sharp jumps when it comes to sending photos/files or videos and voice messages. The significantly smaller packets can be part of the whole big message or packets that are sent as part of the protocol (handshakes, etc.).

In our project we focused on the Event-Based Detector attack algorithm according to the article.

Therefore

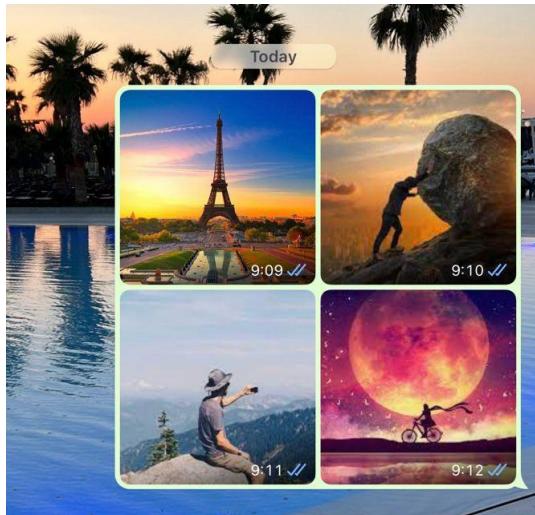
In order to understand the subject we have to investigate, we continued to document relatively simple WhatsApp groups (a group of a few people with a distinct character such as sending pictures only), in order to understand how to draw conclusions and reach good results, from there we continued to more complex groups in our project, we start by studying the simple things and drawing the conclusions from them in order to continue to the more complex parts.

## Event-Based Detector

### Group number 1:

This group is characterized by sending pictures, a group in which the manager sends pictures, that is, the transfer of the content is through pictures only through the manager to the participants.

We carried out an event where every minute a photo was sent to the group from the manager (who is connected to whatsapp web).



Similarly, as we saw earlier, if we filter all the packets that pass through our network and leave those that use the QUIC protocol, the wireshark recording will look like this

No.	Time	Source	Destination	Protocol	Length	Frame	Server Name	Info
1 0.000000000	192.168.86.38		213.57.24.97	QUIC	1399✓		media.fhfaf1...	0-RTT, DCID=8db1ed47c1406cf310a7e:0-RTT, DCID=8db1ed47c1406cf310a7e
2 0.002112761	192.168.86.38		213.57.24.97	QUIC	476✓			Initial, DCID=d44ae0, SCID=400105
3 0.018320302	213.57.24.97		192.168.86.38	QUIC	1274✓			Handshake, DCID=d44ae0, SCID=4001
4 0.018373967	213.57.24.97		192.168.86.38	QUIC	256✓			Protected Payload (KP0), DCID=d44
5 0.019213872	213.57.24.97		192.168.86.38	QUIC	90✓			Handshake, DCID=400105127193df6c, DCID=d44
6 0.019601540	192.168.86.38		213.57.24.97	QUIC	127✓			Protected Payload (KP0), DCID=d44
7 0.020348564	213.57.24.97		192.168.86.38	QUIC	122✓			Protected Payload (KP0), DCID=d44
8 0.021241132	213.57.24.97		192.168.86.38	QUIC	90✓			Protected Payload (KP0), DCID=d44
9 0.021380312	192.168.86.38		213.57.24.97	QUIC	76✓			Protected Payload (KP0), DCID=d44
10 0.031720834	192.168.86.38		157.240.253.60	QUIC	1399✓		pps.whatsapp...	Initial, DCID=323192de96e91a0a66e
11 0.034848246	213.57.24.97		192.168.86.38	QUIC	84✓			Handshake, DCID=d44ae0, SCID=4001
12 0.035991238	213.57.24.97		192.168.86.38	QUIC	378✓			Protected Payload (KP0), DCID=d44
13 0.056820967	192.168.86.38		213.57.24.97	QUIC	77✓			Protected Payload (KP0), DCID=400
14 0.099363939	213.57.24.97		192.168.86.38	QUIC	282✓			Protected Payload (KP0), DCID=d44
15 0.0996442186	192.168.86.38		213.57.24.97	QUIC	73✓			Protected Payload (KP0), DCID=d400
16 0.100116917	192.168.86.38		213.57.24.97	QUIC	390✓			Protected Payload (KP0), DCID=d400
17 0.104752081	157.240.253.60		192.168.86.38	QUIC	1274✓			Initial, DCID=25b693, SCID=bb1d00
18 0.104871017	157.240.253.60		192.168.86.38	QUIC	1274✓			Handshake, DCID=25b693, SCID=bb1d
19 0.105467601	192.168.86.38		157.240.253.60	QUIC	87✓			Handshake, DCID=bb1d008f41700b34,
20 0.105837234	157.240.253.60		192.168.86.38	QUIC	1274✓			Handshake, DCID=25b693, SCID=bb1d
21 0.105969898	192.168.86.38		157.240.253.60	QUIC	87✓			Handshake, DCID=bb1d008f41700b34,
22 0.120450175	192.168.86.38		213.57.24.97	QUIC	77✓			Protected Payload (KP0), DCID=d400
23 0.148213008	157.240.253.60		192.168.86.38	QUIC	809✓			Handshake, DCID=25b693, SCID=bb1d
24 0.148240398	157.240.253.60		192.168.86.38	QUIC	122✓			Protected Payload (KP0), DCID=25b
25 0.148550387	213.57.24.97		192.168.86.38	QUIC	90✓			Protected Payload (KP0), DCID=d44

Frame 10: 1399 bytes on wire (11192 bits), 1399 bytes captured (11192 bits) on interface wlp2s0, id 0  
 ▷ Ethernet II, Src: Epigram\_0d:f4:3e (00:90:4c:0d:f4:3e), Dst: Google\_29:2d:87 (24:05:88:29:2d:87)  
 ▷ Internet Protocol Version 4, Src: 192.168.86.38, Dst: 157.240.253.60  
 ▷ User Datagram Protocol, Src Port: 48776, Dst Port: 443  
 ▷ QUIC IETF  
 ▷ QUIC IETF

In the recording we record an intriguing thing:

It seems that in addition to the connection between our computer (192.168.86.38) and the WhatsApp server (157.240.253.60)

There is another connection between our computer and the address (213.57.24.97), after checking we discovered that it is a HOT server in Israel.

We assume that sending the photos from our computer to the WhatsApp server is done as follows:

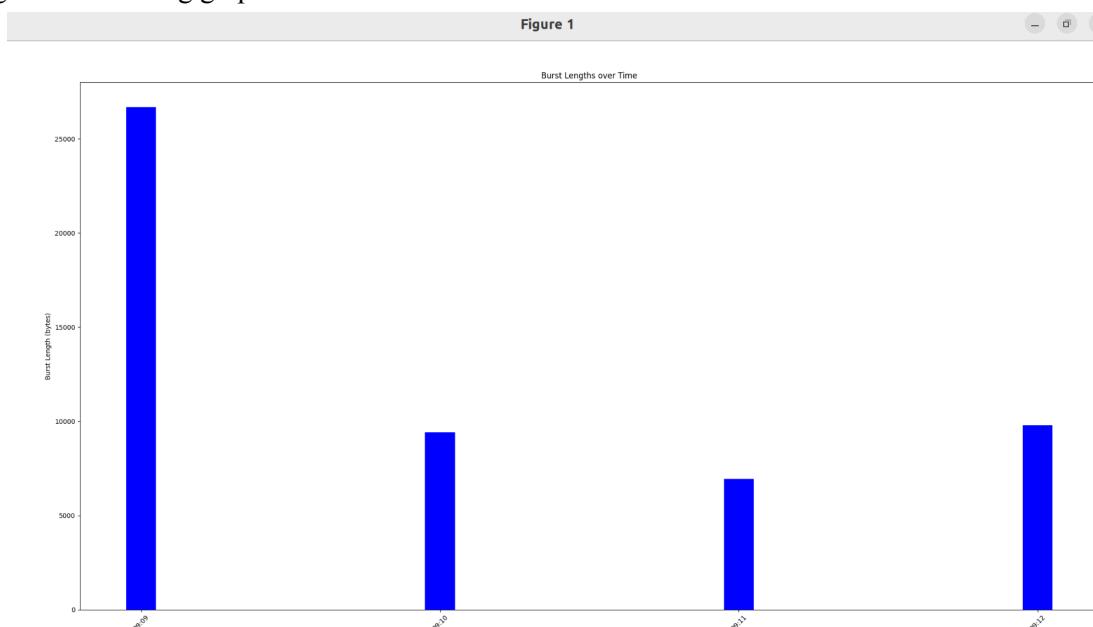
Our computer (the client), starts the process by accessing WhatsApp Web. This involves opening a web browser and visiting the WhatsApp web interface.

WhatsApp may use content delivery networks (CDNs) to optimize the delivery of media, such as images. These CDNs consist of servers distributed in different locations around the world. It is possible that the server in question is the server we found in the recording. And when we send an image, it is routed through this server.

#### event detector graph:

In order to carry out the attack algorithm we will use the program event\_extraction.py which creates the event separation for us (a detailed explanation can be found in detail on github).

We will get the following graph: th 0.00001



Which appropriately shows us the sending of the 4 photos every minute in the group as a separate event.

It can be concluded that the nature of the group is such that relatively large files are sent (more than 25000 bytes per minute) at consecutive and odd times, so it may also be possible to conclude that it is one person who is sending the information.

But we will note that the size of the images here does not accurately describe their original size, we tried to understand why this is so and came to the following conclusions:

When we send pictures through WhatsApp, the application compresses them before sending them over the network. The compression is done to reduce the size of the image, since smaller images require less bandwidth and storage.

The disparity between file sizes can be attributed to several factors, including image format conversion, metadata abstraction, and additional compression applied by WhatsApp.

1. Image format conversion: WhatsApp may convert the images to a more efficient format (eg, WEBP) which will result in smaller file sizes.
2. Remove metadata: WhatsApp may remove some metadata or non-essential information from the images, further reducing the file size. Therefore we see that the size of the packages that are sent is relatively small.

In summary, the difference in file sizes is due to WhatsApp's image compression and optimization techniques, which reduce file sizes for efficient network transfer and user experience.

## Group number 2:

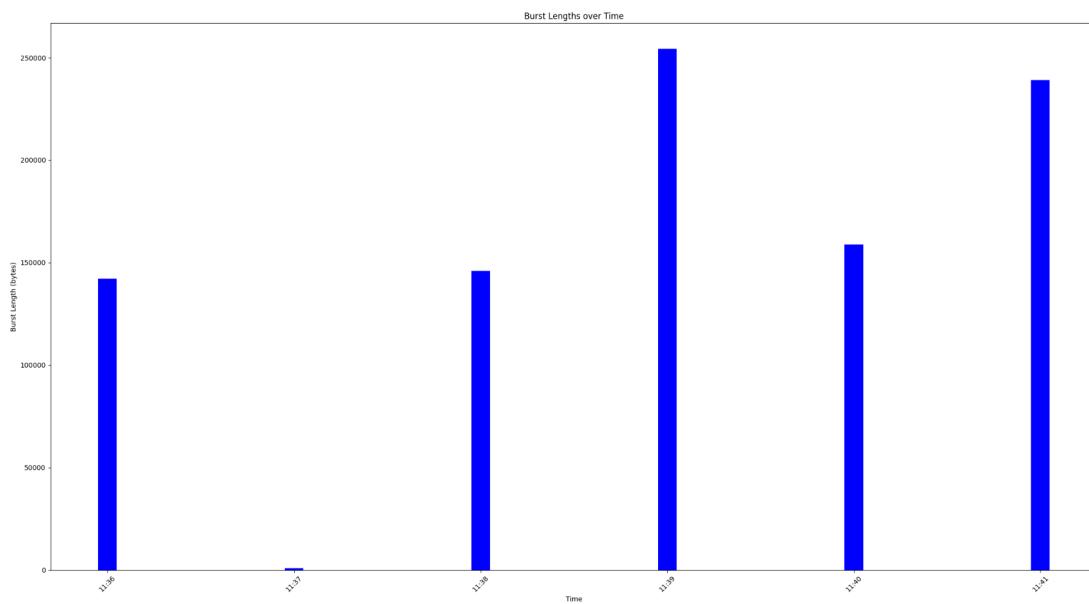
A group where 5 videos were sent to participants every minute.



## wireshark:

No.	Time	Source	Destination	Protocol	Length	Frame	Server Name	Info
10.000000000	172.20.10.6	34.107.221.82	TCP	66 ✓				35736 → 80 [ACK] Seq=1 Ack=1 Win=
20.0000020767	172.20.10.6	34.107.221.82	TCP	66 ✓				43114 → 80 [ACK] Seq=1 Ack=1 Win=
30.216525935	34.107.221.82	172.20.10.6	TCP	66 ✓				[TCP ACKed unseen segment] 80 → 4
40.220537165	34.107.221.82	172.20.10.6	TCP	66 ✓				[TCP ACKed unseen segment] 80 → 3
52.739466935	172.20.10.6	157.240.195.56	TLSv1.2	195 ✓				Application Data
62.880021419	157.240.195.56	172.20.10.6	TCP	66 ✓				443 → 59972 [ACK] Seq=1 Ack=130 W
73.009454072	157.240.195.56	172.20.10.6	TLSv1.2	213 ✓				Application Data
83.009491655	172.20.10.6	157.240.195.56	TCP	66 ✓				59972 → 443 [ACK] Seq=130 Ack=148
98.8.114367551	172.20.10.6	34.117.65.55	TLSv1.2	105 ✓				Application Data
108.8.336031295	34.117.65.55	172.20.10.6	TLSv1.2	105 ✓				Application Data
118.336108458	172.20.10.6	34.117.65.55	TCP	66 ✓				34948 → 443 [ACK] Seq=40 Ack=40 W
1210.240966676	172.20.10.6	34.107.221.82	TCP	66 ✓				[TCP Dup ACK 1#1] 35736 → 80 [ACK]
1310.244975117	172.20.10.6	34.107.221.82	TCP	66 ✓				[TCP Dup ACK 2#1] 43114 → 80 [ACK]
1410.278830747	34.107.221.82	172.20.10.6	TCP	66 ✓				[TCP Dup ACK 4#1] [TCP ACKed unse
1510.278877106	34.107.221.82	172.20.10.6	TCP	66 ✓				[TCP Dup ACK 3#1] [TCP ACKed unse
1610.945864705	172.20.10.6	157.240.195.56	TLSv1.2	186 ✓				Application Data
1711.058334835	157.240.195.56	172.20.10.6	TLSv1.2	101 ✓				Application Data
1811.058374638	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=36
1911.070705891	157.240.195.56	172.20.10.6	TCP	1446 ✓				443 → 59956 [ACK] Seq=36 Ack=121
2011.070748907	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=141
2111.070758783	157.240.195.56	172.20.10.6	TLSv1.2	200 ✓				Application Data
2211.070767048	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=155
2311.070876667	157.240.195.56	172.20.10.6	TCP	1446 ✓				443 → 59956 [ACK] Seq=1550 Ack=12
2411.070888369	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=293
2511.072097006	157.240.195.56	172.20.10.6	TLSv1.2	1446 ✓				Application Data [TCP segment of
2611.072145476	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=431
2711.072164507	157.240.195.56	172.20.10.6	TCP	1446 ✓				443 → 59956 [ACK] Seq=4310 Ack=12
2811.072179499	172.20.10.6	157.240.195.56	TCP	66 ✓				59956 → 443 [ACK] Seq=121 Ack=569

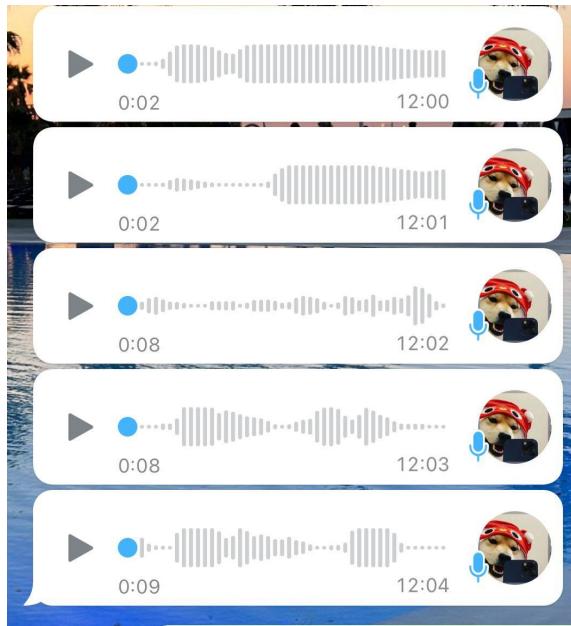
event detector graph:



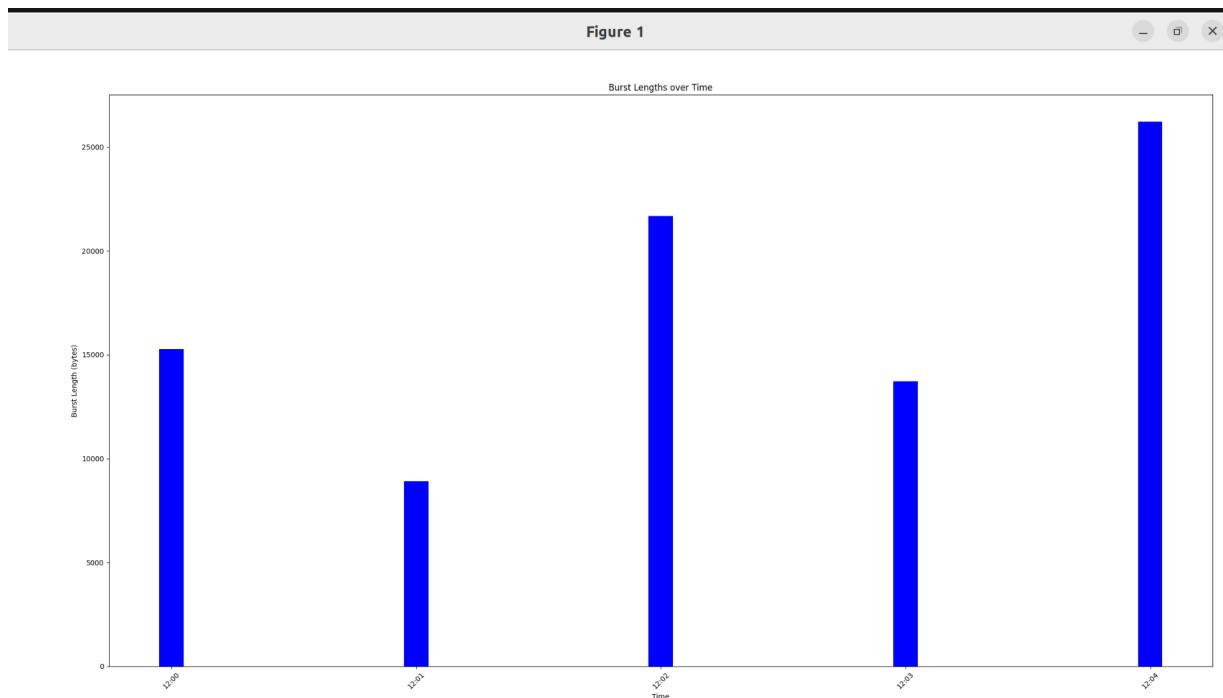
With  $\text{th}=0.1$  we were able to identify the sizes and export the corresponding events of sending a video every minute, in addition each event corresponds to the size of the videos and at 11:39 we sent a longer video and this is also seen from the graph.

In addition, we had a small break between sending the messages at the beginning and you can see in the 37th minute that there is minimal sending of packages.

Group number 3: In this group we received voice messages every minute from a participant in the group



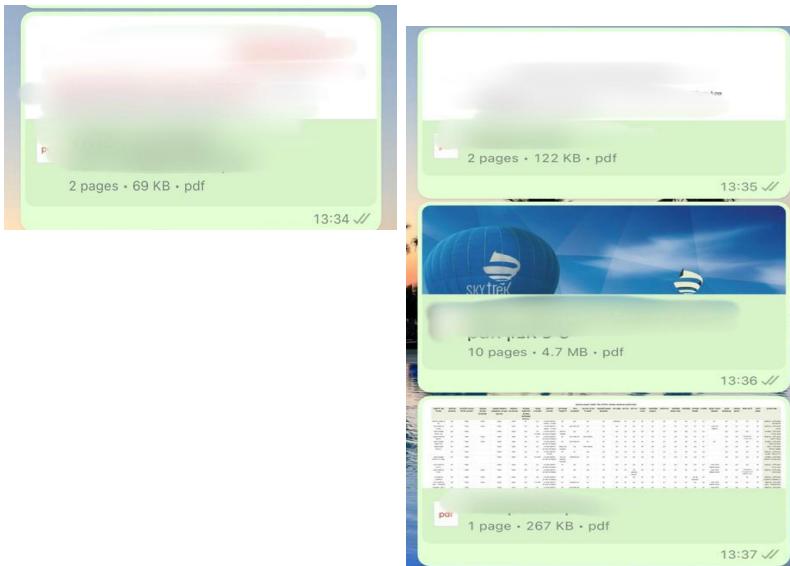
Event detection graph:



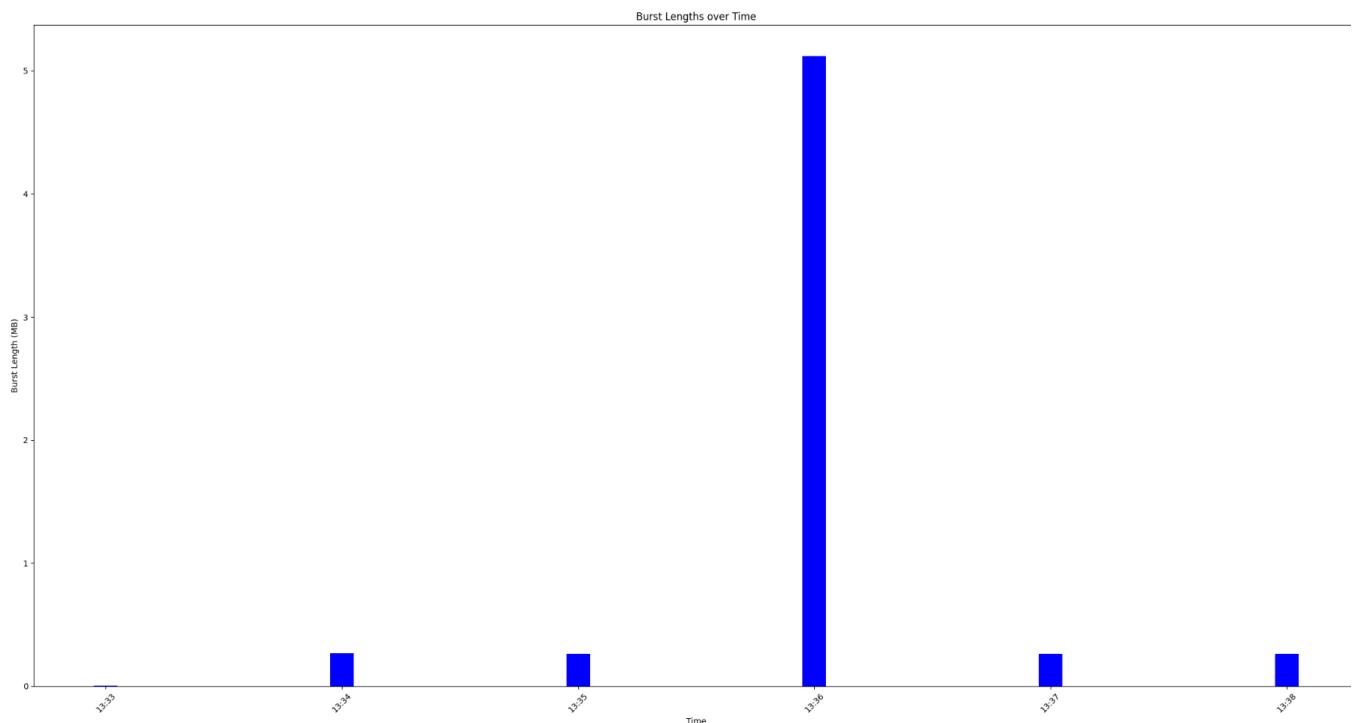
According to  $\text{th}=0.1$  we see that even though there are messages that the recording time is the same but there is still a difference in the sizes of the events, therefore we can assume that the size not only depends on the time but also on the "voice sequence" which affects the size of the packet.

#### Group number 4:

In this group we sent different PDF files of different sizes every minute.



Event detection graph:



Since there is a 5MB file, we created another program that will show us the information in MB. You can clearly see the sending of the largest file in relation to all other messages. In addition at 13:38 we see that another message is received that is not related to the group (from another group) and we see how this hinders us from analyzing the information from the group we are investigating.

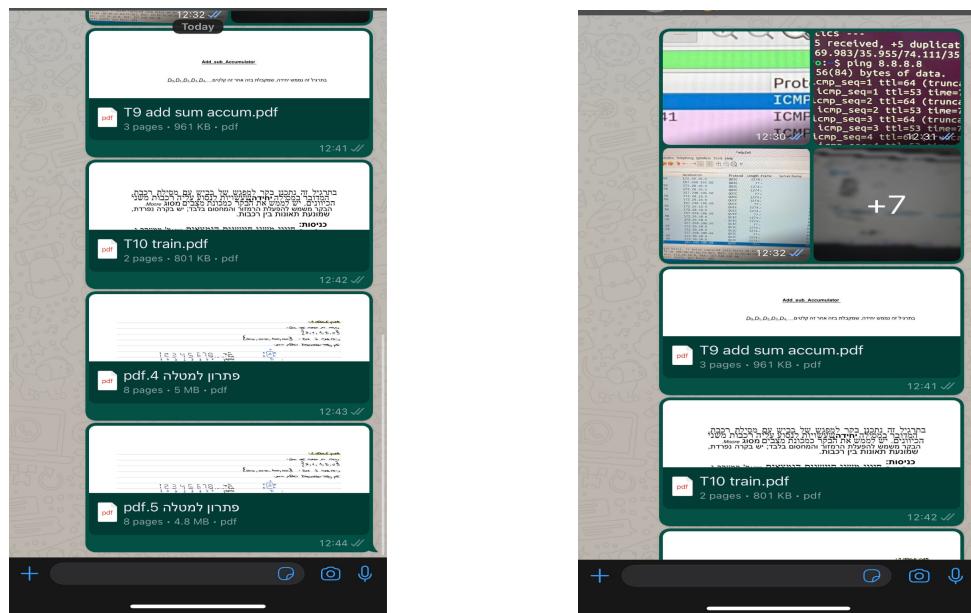
## Group number 5:

A group where the manager (we) transfer photos and files.

While in the open window my Gmail receives emails. In the group the conversation looks like this:

(2)

(1)



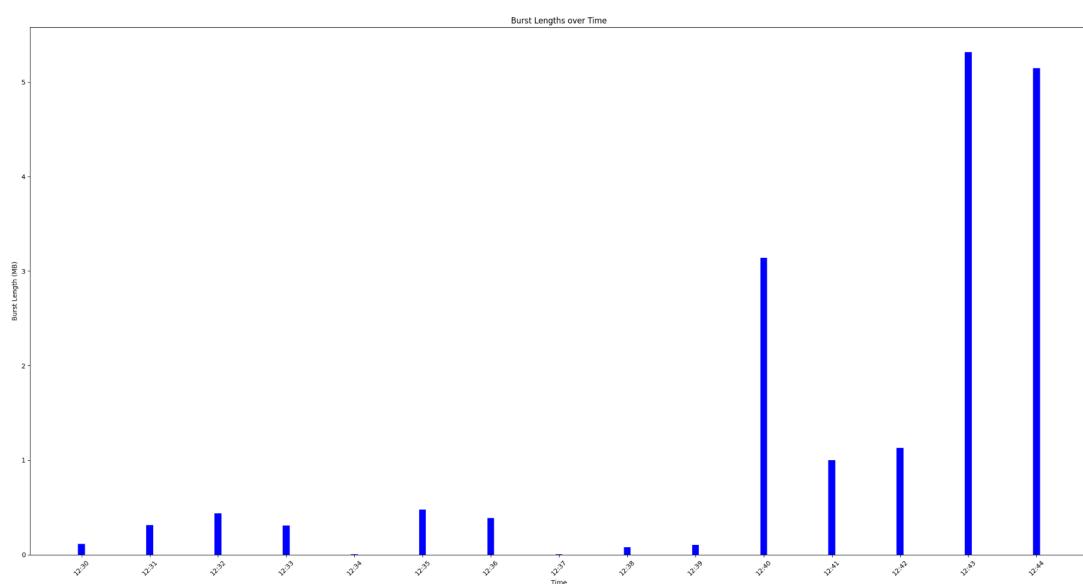
In this group, photos and files were sent as follows:

Images were sent one minute apart, then files were sent one after the other with one minute apart.

While sending these messages, WhatsApp was open to "background noise", email (gmail) was open while receiving the files, in addition, text messages from another group were also received.

So here we simulate a situation where the attacker listens to a certain group but the attacked has noise not only from the application but also from the network.

event detector graph:



The graph shows us the sending of images every minute and in fact you can see that on average the attacker can conclude that the messages sent every minute on average are more or less the same in terms of weight, and then there is a big jump in the graph at 12:40 so the attacker can understand that this is a suspicious jump, that is A message with greater weight and understand that it is a different file type and not an image

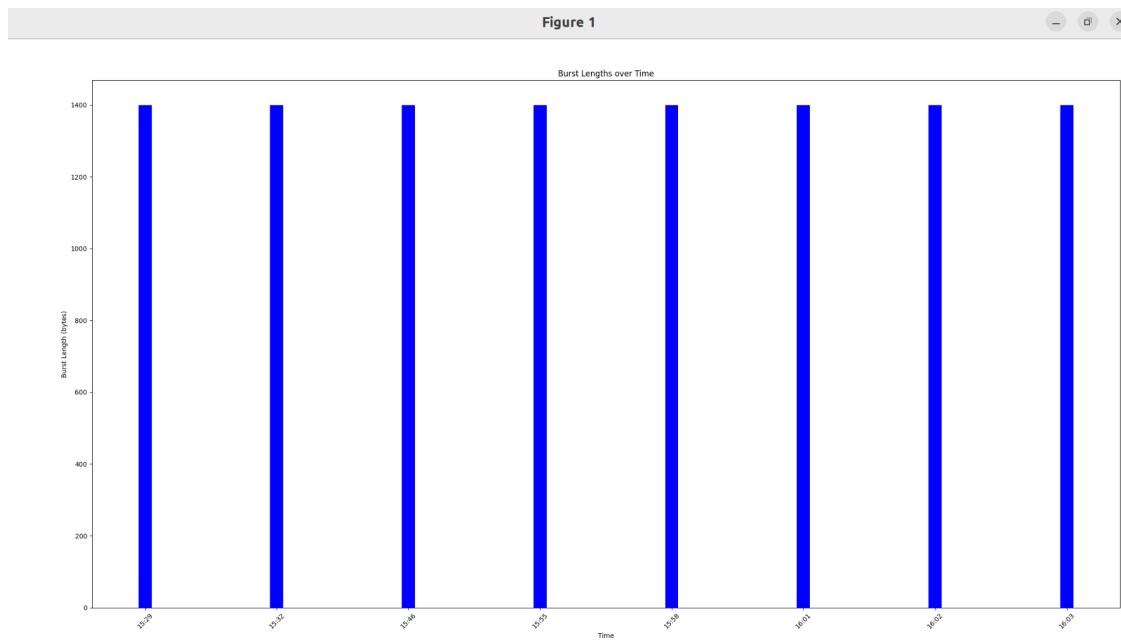
In this way, the attacker actually gets an illustration of what type of messages are in the group and thus understands what its "character" is, is it the desired group and can also conclude that due to the unusual size it is a disturbance.

### Group number 6:

We recorded a group of the degree course after an exam in the course



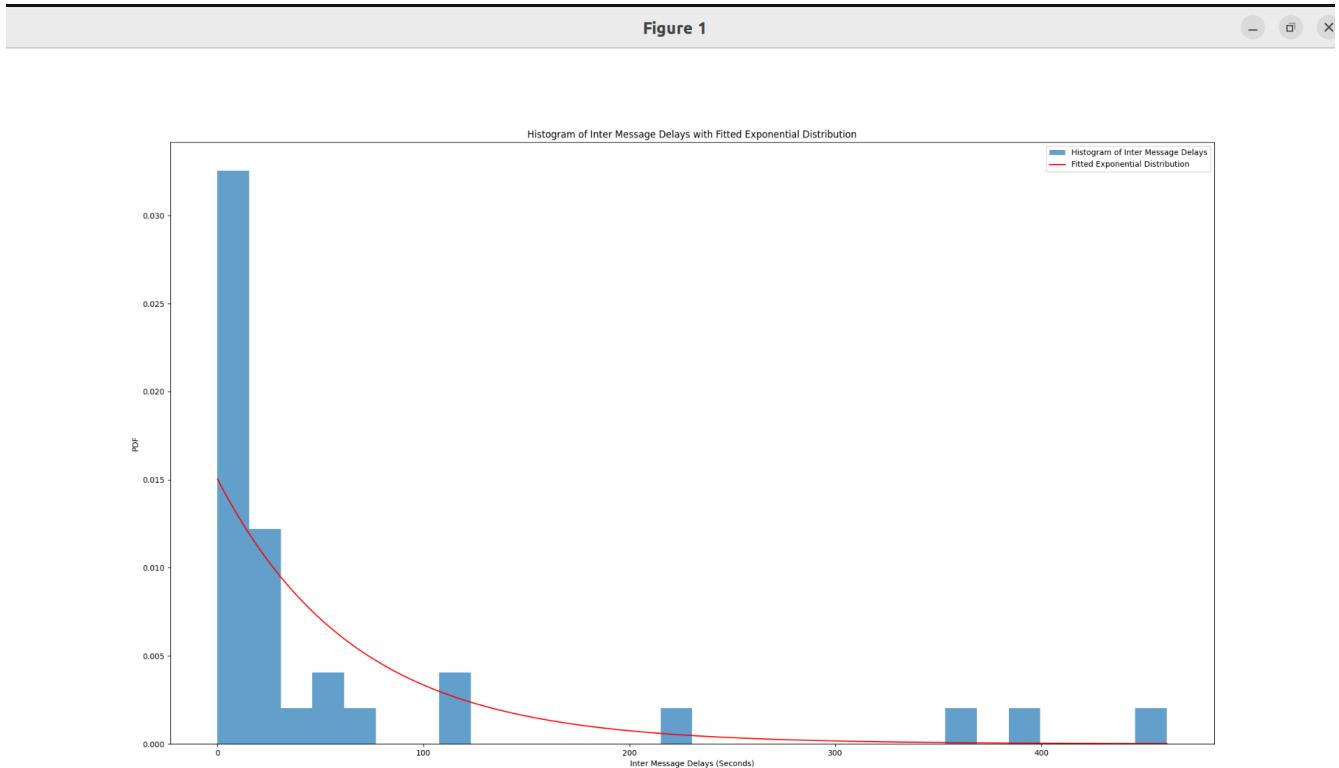
### event detector graph:



Here you see again that there is a kind of "pace" in the messages and every minute several people send a message and the events are still very small, so it can be understood that these are text messages in a group of messages with many participants in a crowded conversation. (thr=0.000001)

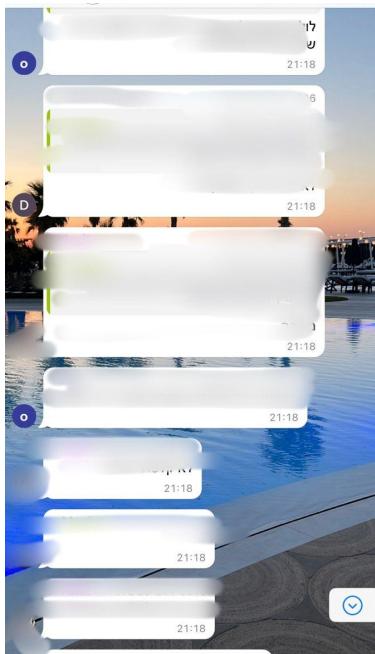
### Histogram graph:

Here we see how the dwell time decreases as the conversation time continues (more people join the conversation and become part of it after leaving the exam.)

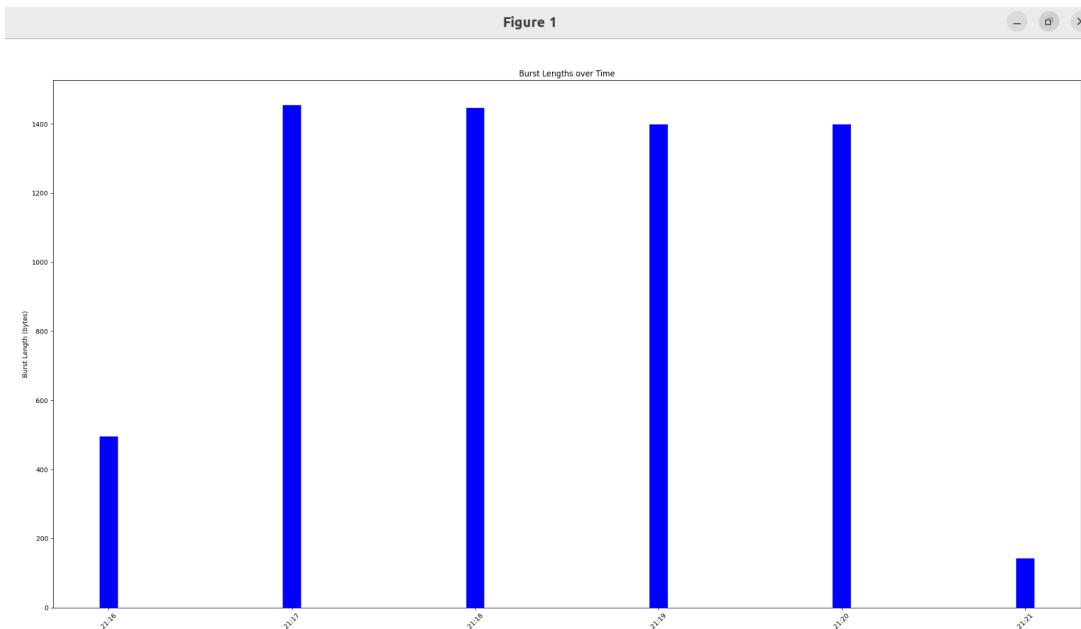


### Group number 7:

Another (another) group that we recorded after a test and you can notice the relatively noisy nature of messages that are sent together per minute!

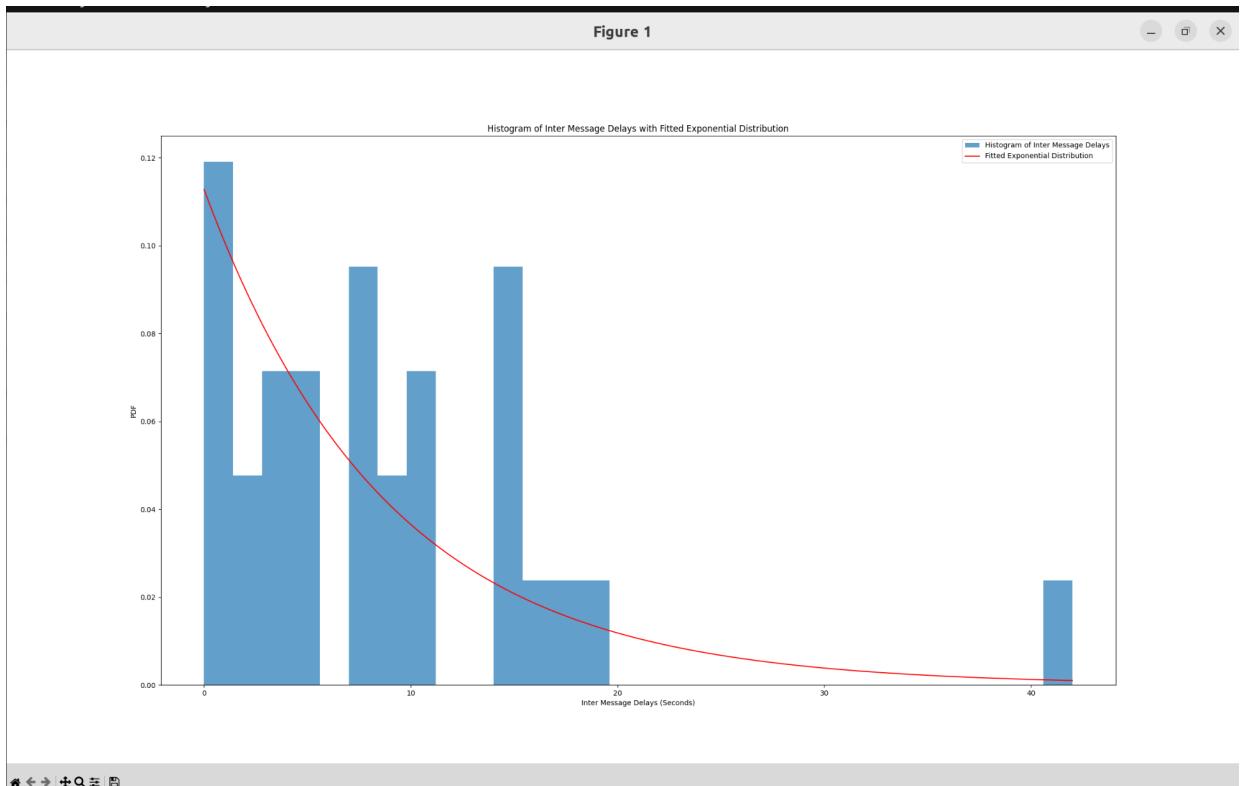


### event detector graph:



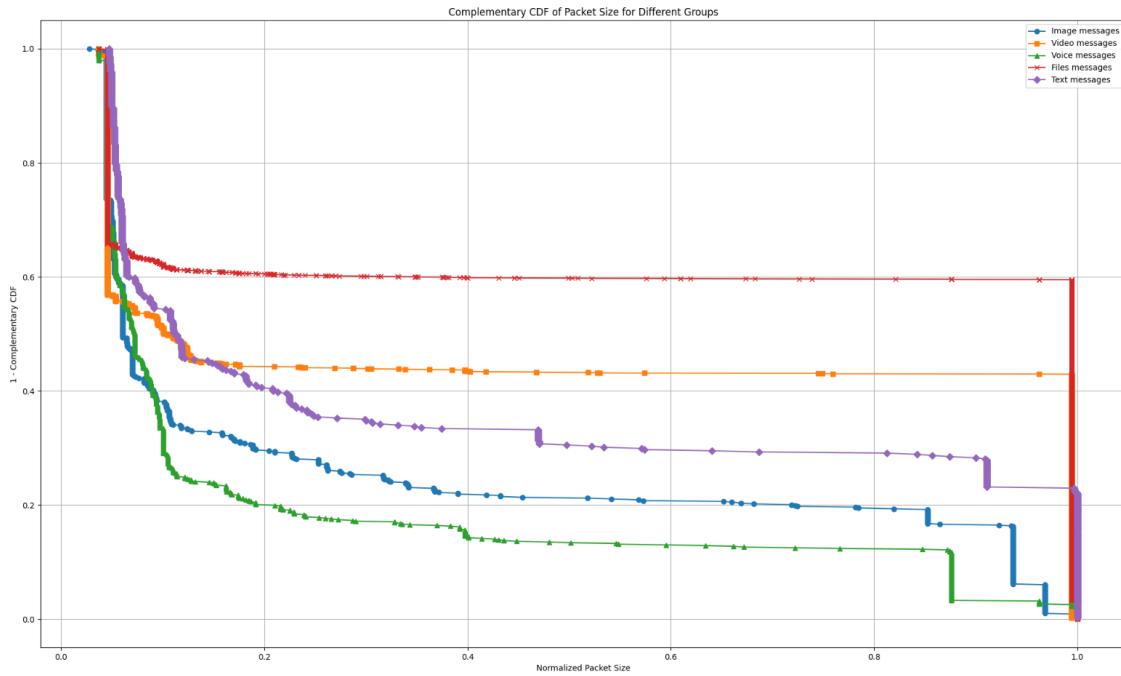
You can see that every minute there is an event but it is relatively small, considering that a text message is between 200 and 100 bytes in size, you can see that a relatively large sequence of messages is sent here every minute (text messages).

### Histogram graph:



It can be seen that the shorter dwell time comes more frequently and indeed the group is very "noisy" as you can notice.

## CDFF:



By groups number 1,2,3,4,6 we created the CCDF graph. The graph shows the complementary cumulative distribution function (CCDF) of the packet sizes normalized to four different groups. Each group represents a different type of message transfer (text, video, images, etc.). The X-axis represents the normalized packet size, ranging from 0 to 1, where 1 corresponds to the maximum size recorded in the group. The Y-axis represents the inverse function of the inverse density distribution function (1 - CCDF), which essentially shows the probability that the message size will be greater than a certain threshold. Or in simple words, for each group each package is normalized by size and we can see from the graph what is the probability of seeing it in the group.

## Summary and conclusions

In our analysis, we investigated the network traffic from WhatsApp messages in different groups. We made interesting observations and faced several challenges in deciphering the data.

### Group number 1: images only

We noticed that sending pictures on WhatsApp led to significant size discrepancies between the original packet capture data and the file sizes displayed in the application.

WhatsApp's image compression and format conversion significantly reduced image sizes during transmission, making it difficult to accurately infer the original image size from the displayed information. This can make it difficult for the attacker to understand what the group in question is and identify their affiliation.

### Group number 2: different types of messages

In this group, we had a clear and reliable pattern of messaging behavior. And the attacker could clearly identify that it is indeed a group if he is able to from the group.

### Group number 3: diverse messages

In groups with various types of messages (photos, files), it was relatively clear and allows discovery.

### Group number 4,5: text messages:

In these groups we conclude that if the attacker is in the groups according to the completion of the two graphs the attacker can understand that we are in the group.

It can be concluded that the algorithms in the article do demonstrate the potential to distinguish group communication patterns. Although challenges exist, this analysis provides important insights into the nature of different groups and can be a valuable tool for understanding the dynamics of WhatsApp groups.

As for the question of whether it is possible to understand whether it is possible to identify the groups in which the attacked participates when he communicates with several groups at the same time, it can be seen that this is relatively very complex. Even if the attacker participates in groups we participate in. Another thing we investigated is why when we listened to the recordings of WhatsApp conversations we can see that sometimes the packets are seen as TCP QUIQ HTTP etc. The question is why it can appear differently each time and what can be learned from this?

The variation in the types of packets observed when intercepting and analyzing traffic on WhatsApp Web stems from the nature of the communication protocols used by the application and the structure of the transmitted data.

WhatsApp Web uses protocols in different data formats to facilitate communication between the client and the server.

When we checked the groups, we tried to do it during "quiet" times in order to get as clean a recording as possible and to be sure that these are indeed the packages related to the only WhatsApp groups in which we are active for a limited time. (whether we send messages or receive messages). When messages are sent simultaneously from several calls, the matter is more complex and it will not be possible to accurately identify which packages are related to which call.

More than that, we noticed that the attacker must also calculate and create a graph as real as possible with an appropriate threshold in order to get the nature of the events in the group. In addition, he must clean the packages that are bothering him.

That's why we think that in the end it takes time and investment and the attacker should be in the group in order not to "get lost" in all the information that passes through the attacked network.

When the attacker tries to coordinate a person's presence in multiple groups or channels, they face several challenges due to the nature of the WhatsApp application we studied. why?

1. End-to-end encryption: when the messages are encrypted, they can only be deciphered on the device of the recipient of the message. This means that even if the adversary can intercept the encrypted messages, he cannot read the content without the encryption keys. This encryption makes it difficult for the attacker to directly access the content of the messages, and understand which messages are related to which group.
2. Mixing messages: Messages from different groups or conversations are often mixed in the network traffic. This mixing challenges the attacker to distinguish between messages belonging to different groups. They can observe the traffic flow and the size of the messages, but without access to the encryption keys, it is not possible to determine the exact content or which messages belong to which group.
3. Packet size and timing: While the content of messages is encrypted, certain metadata such as message timing, size and frequency of communication are still visible to an attacker. However, as we have seen in the work, metadata alone may not be sufficient to reliably correlate an individual's presence in multiple groups. For example, multiple groups may have similar movement patterns, which becomes a challenge.
4. Multiple identities: frequent users, use several platforms when they enter the application, such as from the computer/phone.

This means that even if the attacker watches multiple interactions from different devices, he cannot definitively link them to the same person.

Furthermore, users may switch between devices or change their identity, further complicating the process.

## bibliography

Book:

Kurose, Jim. Computer Networking: A Top Down Approach, 7th Editor.  
Originally published in 2000.

URL:

[https://www.ucg.ac.me/skladiste/blog\\_44233/objava\\_64433/fajlovi/Computer%20Networking%20 %20A%20Top%20Down%20Approach,%207th,%20converted.pdf](https://www.ucg.ac.me/skladiste/blog_44233/objava_64433/fajlovi/Computer%20Networking%20 %20A%20Top%20Down%20Approach,%207th,%20converted.pdf)

Webpage:

Title: "How to Verify That WhatsApp Messages Are Encrypted"

Author: (Author information not provided on the webpage)

Website: urtech.ca

Publication Date: July 2020

URL: <https://www.urtech.ca/2020/07/how-to-verify-that-whatsapp-messages-are-encrypted/>

PDF Paper:

Title: "Practical Traffic Analysis Attacks on Secure Messaging Applications"

Authors: Alireza Bahramali, Amir Houmansadr, Ramin Soltani, Dennis Goeckel, Don Towsley

Affiliations: University of Massachusetts Amherst

Emails: abahramali@cs.umass.edu, amir@cs.umass.edu, ramin.soltani@gmail.com,  
dgoeckel@engin.umass.edu, towsley@cs.umass.edu

Publication Source: Proceedings of the Network and Distributed System Security Symposium (NDSS)

Publication Date: February 2020

URL: <https://www.ndss-symposium.org/wp-content/uploads/2020/02/24347-paper.pdf>

AI Chatbot:

Assistant: ChatGPT

Description: AI language model developed by OpenAI, based on GPT-3 architecture, providing text-based assistance and information.

Knowledge Cutoff: September 2021