Annals of Telecommunications (2023) 78:183–208
https://doi.org/10.1007/s12243-022-00919-6

Forensic investigation of Cisco WebEx desktop client, web, and Android smartphone applications

Zainab Khalid[1] · Farkhund Iqbal[2] · Faouzi Kamoun[3] · Liaqat Ali Khan[4] · Babar Shah[2]

Received: 13 January 2022 / Accepted: 8 July 2022 / Published online: 12 August 2022

Abstract
Digital forensic analysis of videoconferencing applications has received considerable attention recently, owing to the wider adoption and diffusion of such applications following the recent COVID-19 pandemic. In this contribution, we present a detailed forensic analysis of Cisco WebEx which is among the top three videoconferencing applications available today. More precisely, we present the results of the forensic investigation of Cisco WebEx desktop client, web, and Android smartphone applications. We focus on three digital forensic areas, namely memory, disk space, and network forensics. From the extracted artifacts, it is evident that valuable user data can be retrieved from different data localities. These include user credentials, emails, user IDs, profile photos, chat messages, shared media, meeting information including meeting passwords, contacts, Advanced Encryption Standard (AES) keys, keyword searches, timestamps, and call logs. We develop a memory parsing tool for Cisco WebEx based on the extracted artifacts. Additionally, we identify anti-forensic artifacts such as deleted chat messages. Although network communications are encrypted, we successfully retrieve useful artifacts such as IPs of server domains and host devices along with message/event timestamps.

Keywords Cisco WebEx · Disk-space forensics · Memory forensics · Network forensics · Videoconferencing · VoIP forensics

1 Introduction

COVID-19 has been a prime catalyst in the widespread adoption of videoconferencing applications such as Zoom, Cisco WebEx, Microsoft Teams, Adobe Connect, and BlueJeans for professional and personal use. In fact, during the pandemic, the work landscape changed dramatically as more companies shifted to a *work-from-home* model. As a result, videoconferencing market is expected to grow from US $6.28 billion in 2021 to US $12.99 billion by 2028 according to a Fortune Business Insight report [1].

✉ Zainab Khalid
zkhalid.msis18seecs@seecs.edu.pk

[1] School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan

[2] College of Technological Innovation, Zayed University, Dubai, UAE

[3] ESPRIT School of Engineering, ESPRIT School of Business, Ariana, Tunisia

[4] Air University, Islamabad, Pakistan

The growing popularity of videoconferencing applications has also attracted malicious users, who use them to launch targeted attacks such as hacking online meetings and subjecting attendees to offensive content. The terms *Zoom bombing* or *Zoom raiding* have recently been used to designate the disruptive intrusion into a videoconference call, whereby a hacker leverages weak authentication features (or other vulnerabilities) to either stream improper content or bully/harass meeting participants [2, 3]. The security and privacy of the videoconference session have therefore become a major concern. Forensic artifacts carved in digital investigation of such applications can provide useful insights into the *what-whom-when-where* of incidents, and attribute malicious actions to a *device* or an *individual*, which can serve as *digital evidence* (DE) in criminal investigations.

Considering a regular Cisco WebEx user's scenario, our goal is to perform an exhaustive investigation of the DE (forensic artifacts) left behind by the application on a client's device. This research is an extension of our earlier work [4], which investigated various data localities (memory, disk space, and network) in a Windows 10 operating system (OS) for DE pertaining to the Cisco WebEx desktop application.

The present contribution extends our earlier research [4] in at least four aspects:

- We extend the scope of our forensic analysis in [4] to cover WebEx Meetings smartphone application running under Android OS.
- We introduce a memory parsing tool for Cisco WebEx based on our findings from (manual) memory forensic analysis.
- We extract additional disk space forensic artifacts from Cisco WebEx desktop client application such as prefetch files.
- We present the results of the forensic analysis of Cisco WebEx web application, targeting the Google Chrome data directory for client-side artifacts with an illustrated case study to highlight the relevance of artifacts in a forensic investigation.

The results of the forensic investigation of three versions (desktop, web, smartphone) of Cisco WebEx applications revealed several relevant artifacts including user account information (display names, email addresses, credentials, default WebEx sites, profile pictures), running processes, encryption keys, exchanged and deleted chat messages, media/text files, meeting records, registry keys, prefetch files, network information, history, downloads, cookies, cache, and bookmarks. The extracted artifacts can be used as potential DE in a court of law.

The remaining of this paper is organized as follows. Section 2 discusses the state of the art in videoconferencing applications' forensics research. Section 3 details our research methodology and experimental setup. Sections 4, 5, and 6 present forensic artifacts extracted from Cisco WebEx desktop, web, and smartphone applications, respectively. Section 7 illustrates two case studies, and Section 8 provides a summary of the paper and some suggestions for future research.

2 Related work

Digital forensics of Voice over Internet Protocol (VoIP), and videoconferencing applications has become a popular research topic recently as highlighted by increase in the number of published papers on these topics during the past few years. Both client and server-side forensic analyses are vital to reconstruct a holistic picture of events pertaining to these applications. However, server-side forensics require access to data stored in application servers, which (1) is not easily available, (2) can be time-consuming to gain access to, and (3) is costly and potentially complicated to collect due to privacy policies implemented by Internet service providers (ISPs) [5]. Nonetheless, a network forensic analysis may divulge information about client-server connections and vice versa. Also, the Windows 10 roaming folder can provide valuable forensic insights since it stores a server's copy of the account data that is loaded into any device the user account is logged into [6]. On the other hand, artifacts extracted from the client device, while accessible, may be stored in encrypted form. While encryption enhances user privacy, it also introduces additional complexity for the forensic analyst. The entailing discussion explores methods for forensic analysis of videoconferencing, and/or VoIP applications (with respect to different data localities, and OSs), and various challenges in the process.

Mahr et al. [7] presented a forensic analysis of Zoom, with a special focus on exploring the disk space for extracting artifacts. The authors investigated various databases associated with the Zoom data directory and extracted artifacts related to chats, contacts, cache, video meetings, and user/device configurations. The authors also presented preliminary results associated with network and memory forensic analysis. At the time their research was conducted, the databases investigated were stored in unencrypted format on the client desktop. However, we have observed that with recent updates of Zoom, these databases are now encrypted which introduces an additional layer of complexity for forensic investigators as a passphrase, or key is now required to decrypt the databases.

Yang et al. [8] performed an in-depth forensic analysis of the Windows Store Skype application for memory, disk space, and network artifacts on a Windows 8.1 client machine. The experiments and results revealed valuable terrestrial (client side) forensic artifacts, including installation information, login, conversations, and records of exchanged files. The authors also observed that uninstalling Skype from the client machine automatically removes the *application folders* of Skype from Windows' file system, which makes them inaccessible for forensic investigation. However, Skype installation folders still reside in the file system, and these can reveal several artifacts of forensic value.

Nicoletti and Bernaschi [9] adopted a multiple case study approach to forensically analyze Skype for business. Their research focused on examining the communication architecture, protocols, and the VoIP codec to extract artifacts. In addition, the Windows Registry, Event Viewer, client application folder, and log files have been identified as sources of forensic evidence.

Recently, Nicoletti, and Bernaschi [10] performed forensic analysis of Microsoft Teams exploring different usage scenarios of the application. This included the Teams-public switched telephone network (PSTN) and Teams walkie-talkie communication scenarios. The authors also explored whether the Teams-PSTN integrated environment is more

vulnerable to attacks and whether forensic evidence can be extracted.

Bowling [11] performed client-side forensic analysis of Microsoft Teams with respect to multiple OSs: Windows, iOS, and Android. The focus of the study was disk space forensics, identifying SQLite databases (in Android) and the chromium cache structures (in Windows) as main sources of artifacts. However, an important database/structure called LevelDB was not explored by the author because of difficulties in parsing the said database.

Bilz [12] also performed disk space forensics to extract Microsoft Teams' artifacts, exploring the data directory for artifacts. In his analysis, the author proposed a Python script to analyze the LevelDB structure discussed in [11]. The script parses the database/structure and outputs data in *.json*. The LevelDB structure seems to store a plethora of information including text messages, comments, posts, exchanged files, contacts, and call logs according to the author's findings.

Anglano [13] performed a forensic analysis of WhatsApp Messenger on an Android smartphone. For this purpose, the YouWave virtualization platform was used to emulate multiple Android smartphones. Various forensic artifacts were extracted including contacts and existing/deleted messages.

Other works [14–17] detail forensics of VoIP applications such as WhatsApp, Viber, Skype, and Tango.

Most of earlier research contributions focused on disk space/file-system forensics to extract client artifacts. Memory and network forensics, on the other hand, were addressed only in few studies such as those reported by Mahr et al. [7], Yang et al. [8], and Nicoletti and Bernaschi [9, 10]. This is mainly attributed to the fact that memory acquisition on a digital device that is in a shutdown mode becomes obsolete, while network traffic encryption renders network forensics a daunting task. Our aim, on the other hand, was to perform an exhaustive analysis of Cisco WebEx, by exploring all three data localities.

3 Experimental setup

Forensic analysis of Cisco WebEx was conducted on a client's device in a controlled test environment. For this purpose, a Windows 10 ISO file was used to create a virtual machine (VM). This was done in order to avoid mixing of WebEx artifacts with other applications (or system files). A total of 4 GB of memory/RAM and 60 GB disk space were allocated to the VM. The Cisco WebEx desktop application was downloaded and installed. A new WebEx account was setup in the VM, and the actions of a typical user were performed while interacting with the WebEx desktop application. The user actions were based on the videoconferencing features provided by WebEx. Similarly, a separate VM was setup for the WebEx web application for test usage. Compared to the desktop application, WebEx's web application naturally offers limited features.

WebEx assigns a *personal meeting room ID*, a *default WebEx site*, and a *default video address* to each user, much like an in-person scenario would take place. Users can invite other guests to join them in their personal meeting room by sharing their meeting link. This way, videoconferencing group meetings can be setup instantly or scheduled at given times.

User actions performed as test activity include the following:

1. Setting up a username, password, and profile photo
2. Adding/deleting contacts
3. Exchanging/deleting chat messages
4. Exchanging/deleting text files, media files, and URLs
5. Using the keyword search to find acquaintances and friends
6. Conducting a meeting using personal room
7. Creating groups
8. In-meeting chat messages, media, and text files
9. Record meetings
10. Screen sharing
11. Conducting one-to-one, group, and scheduled meetings

The test activities associated with users deleting certain contacts, messages, and exchanged files from the WebEx application were conducted to be investigated later in the analysis phase of our forensics process to see if such obstructive/anti-forensic behaviors could be detected.

After the user actions were performed, memory and disk space were captured using the AccessData FTK Imager, intermittently. In case of memory, the captures were made after major events such as login, exchange of chat messages, exchange of media files, and one-on-one/group/scheduled meetings. The hashes of these images were computed using FTK Imager to ensure their integrity before image processing.

For network forensic analysis of WebEx, a Wi-Fi hotspot was used to streamline the network traffic. Wireshark network protocol analyzer was used to capture the traffic which was saved as a *.pcap* file. NetworkMiner was also used for analysis. Network traffic for each user activity was captured separately in order to analyze the artifacts of each activity individually. The login events, chat messages/files/URLs exchange events, and meeting events were recorded using Wireshark separately.

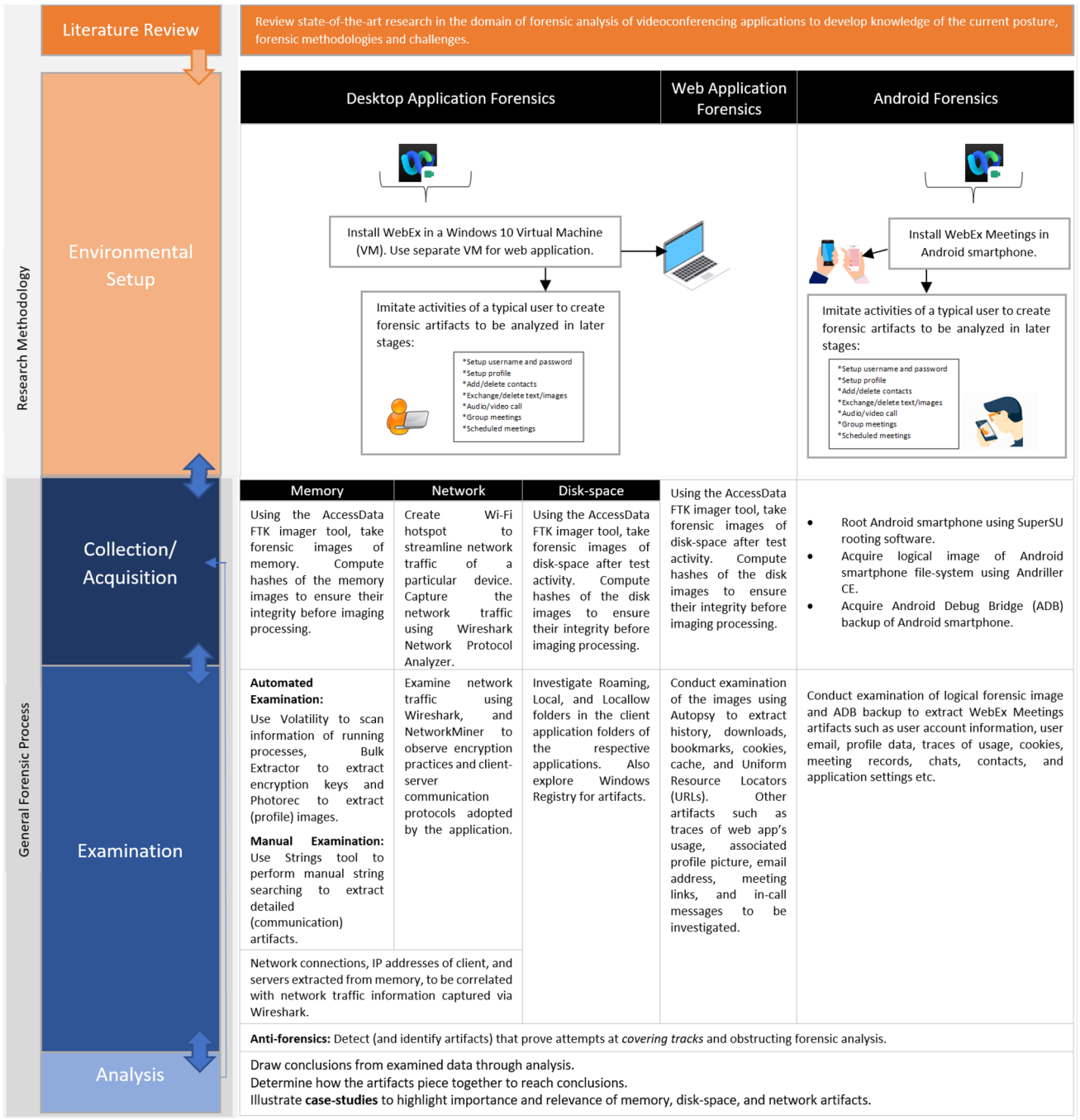

Fig. 1 Forensic methodology

To perform forensic analysis of the Android smartphone, we used Andriller Community Edition (CE) to acquire a logical forensic image of the smartphone. This operation was first performed on an unrooted Android smartphone, in which case Andriller CE was unable to acquire the image. On the other hand, we were able to successfully complete a logical acquisition after rooting the smartphone. Additionally, we were able to successfully acquire and investigate the Android Debug Bridge (ADB) backup for potential DE.

Figure 1 details the forensic methodology of this study, while Table 1 lists the tools employed during the digital investigation process.

4 Cisco WebEx desktop client forensics

This section gives a detailed client-side forensic analysis of Cisco WebEx desktop application targeting the memory, disk space, and network data localities.

Table 1 Tools used for forensic analysis

Tool/device	Software version	Usage
Windows 10 virtual machine (VM)	10	Test environment
Samsung Galaxy Grand Prime+	Marshmallow 6.0.1	Test device
Cisco WebEx desktop application	41.3.0.18191	Videoconferencing application to test for forensic artifacts
Cisco WebEx web application	42.1.1.1286	Videoconferencing web application to test for forensic artifacts
WebEx meetings smartphone application	42.2.0.242020258	Videoconferencing application to test for forensic artifacts
AccessData FTK Imager	4.5.0.3	Creation and analysis of forensic images
Volatility	2.6	Analysis of memory dumps
Autopsy	4.19.1	Analysis of forensic images
Andriller CE	3.6.1	Logical acquisition (and report generation) for Android smartphone
Strings	2.53	Manual string searching
Bulk Extractor	1.6.0	Analysis of image dumps
PhotoRec	7.2	Carve .jpeg images
DB Browser for SQLite	3.12.1	Browse application databases in client application folder
Regedit	6.1.7600.16385	View Windows Registry
PECmd, Eric Zimmerman	1.5.0.0	Parse prefetch files
SuperSU	2.82	Android rooting software
Wireshark	3.4.3	Capture/analyze network traffic
NetworkMiner	2.6	Analyze network traffic
ChromeCacheView	2.27	View WebEx cache
ChromeCookiesView	1.66	View WebEx cookies
DCode	5.5.21194.40	Timestamp decoding
Android Debug Bridge (ADB)	1.0.41	Acquisition for ADB backup

4.1 Memory forensics

The volatile memory provides a wealth of information about running processes and applications in a device among other artifacts. It is a subject of great interest for forensic investigators because data that would normally be stored in encrypted form on hard drive can reside in memory as plaintext. Our analysis of captured memory dumps focused on five *profiles* [18], namely communication content (activity/relationship context, details), contacts, communication history, passwords, and encryption keys.

4.1.1 Automated analysis

Several WebEx processes, listed below, were identified from memory dumps (using *pstree* Volatility):

1. atmgr.exe
2. CiscoCollabHost.exe
3. Ciscowebexstart.exe
4. webexAppLaunch.exe
5. washost.exe
6. webexmta.exe

The *atmgr.exe* and *washost.exe* processes in particular exist when WebEx meetings occur (Fig. 2).

Targeted *yarascan* searches (particular to the process IDs obtained with the running processes of WebEx) against the memory dumps revealed interesting information regarding calls, but the output that appeared was a limited block of information and required parsing through the dump based on the physical/virtual offset of the yarascan result for further analysis (Fig. 3); similar artifacts were easily extracted manually as well.

Advanced Encryption Standard (AES) keys (Fig. 4), the email addresses of the test user, and corresponding parties were recovered from memory using Bulk Extractor (Fig. 5).

Photographic images reconstructed from the memory dumps using PhotoRec included the WebEx logo and favicon images related to the application. Despite acquiring several successive memory dumps for carving avatars, we were not able to extract the profile photos of the participants. Most likely, the profile photos of WebEx accounts are stored in encrypted form in memory.

Fig. 2 Pstree output for WebEx extracted via Volatility

```
Name                                                 Pid   PPid   Thds   Hnds Time
---------------------------------------------------- ------ ------ ------ ------ ----
 0xfffffa80087f3b00:CiscoCollabHos                   2152   1980     51   1237 2021-04-26
. 0xfffffa8008dedb00:CiscoCollabHos                  2948   2152     11    229 2021-04-26
. 0xfffffa80090c4310:CiscoCollabHos                  1200   2152     14    322 2021-04-26
. 0xfffffa80071fc060:CiscoCollabHos                  1520   2152     14    205 2021-04-26
. 0xfffffa8003ebfb00:CiscoCollabHos                  1060   2152     35    441 2021-04-26
. 0xfffffa80048bfb00:CiscoCollabHos                  1552   2152     13    212 2021-04-26
. 0xfffffa8003f71060:CiscoCollabHos                  2540   2152      9    145 2021-04-26
 0xfffffa8008f82b00:explorer.exe                     1296   1260     24    817 2021-04-26
. 0xfffffa80091a4b00:igfxtray.exe                    1536   1296      3     83 2021-04-26
. 0xfffffa80092f3b00:ciscowebexstar                  1192   1296     13    360 2021-04-26
.. 0xfffffa8004637060:atmgr.exe                      3252   1192     30    465 2021-04-26
...0xfffffa8003710060:webexAppLaunch                 1676   3252      5    234 2021-04-26
...0xfffffa8003010060:washost.exe                    3364   3252      9    213 2021-04-26
...0xfffffa8002981060:webexmta.exe                   3588   3252     10    123 2021-04-26
```

Fig. 3 Yarascan search on WebEx process (PID 2152) via Volatility

```
Rule: r1
Owner: Process CiscoCollabHos Pid 2152
0x058be3f1  68 74 74 70 73 3a 2f 2f 63 6f 6e 76 2d 6b 2e 77   https://conv-k.w
0x058be401  62 78 32 2e 63 6f 6d 2f 63 6f 6e 76 65 72 73 61   bx2.com/conversa
0x058be411  74 69 6f 6e 2f 61 70 69 2f 76 31 2f 61 63 74 69   tion/api/v1/acti
0x058be421  76 69 74 69 65 73 2f 34 33 39 32 32 39 37 30 2d   vities/43922970-
0x058be431  61 36 35 37 2d 31 31 65 62 2d 61 65 37 38 2d 63   a657-11eb-ae78-c
0x058be441  66 33 35 36 38 39 36 34 66 39 63 22 2c 22 70 75   f3568964f9c","pu
0x058be451  62 6c 69 73 68 65 64 22 3a 22 32 30 32 31 2d 30   blished":"2021-0
0x058be461  34 2d 32 36 54 30 36 3a 31 38 3a 34 39 2e 30 39   4-26T06:18:49.09
0x058be471  35 5a 22 2c 22 76 65 72 62 22 3a 22 70 6f 73 74   5Z","verb":"post
0x058be481  22 2c 22 61 63 74 6f 72 22 3a 7b 22 69 64 22 3a   ","actor":{"id":
0x058be491  22 7a 61 69 6e 61 62 6b 68 61 6c 69 64 32 33 31   "zainab[redacted]
0x058be4a1  35 40 67 6d 61 69 6c 2e 63 6f 6d 22 2c 22 6f 62   5@gmail.com","ob
0x058be4b1  6a 65 63 74 54 79 70 65 22 3a 22 70 65 72 73 6f   jectType":"perso
0x058be4c1  6e 22 2c 22 64 69 73 70 6c 61 79 4e 61 6d 65 22   n","displayName"
0x058be4d1  3a 22 5a 61 69 6e 61 62 20 4b 68 61 6c 69 64 22   :"Zainab.Khalid"
0x058be4e1  2c 22 6f 72 67 49 64 22 3a 22 34 63 32 35 30 38   ,"orgId":"4c2508
```

Fig. 4 AES encryption keys extracted via Bulk Extractor

```
# BULK_EXTRACTOR-Version: 1.6.0 ($Rev: 10844 $)
# Feature-Recorder: aes_keys
# Filename: memdump.mem
# Feature-File-Version: 1.1
35156448        60 37 9b 87 ad 51 5b bc de 04 ca fd 32 82 64 7f f9 51 96 e8 46 5a 70 7b 0b aa 10 38 d2 d8        AES256
69006724        d3 46 cf f5 56 74 25 1d 8f 3f 7a 90 f7 0c         AES128
585827652       25 9a d5 74 a3 b8 e3 6e 41 d5 b6 cc 92 06 a9 8c 06 6f 7e f9 53 8d d8 a9 be db b2 2c 85 e4        AES256
1486159016      ad f9 65 ab e7 55 37 10 9d 4b a1 0e 95 20 35 ae 6d 3d 04 31 63 35 a1 49 4b 6e cc 71 d3 56        AES256
1610195700      25 9a d5 74 a3 b8 e3 6e 41 d5 b6 cc 92 06 a9 8c 06 6f 7e f9 53 8d d8 a9 be db b2 2c 85 e4        AES256
1789058636      25 9a d5 74 a3 b8 e3 6e 41 d5 b6 cc 92 06 a9 8c 06 6f 7e f9 53 8d d8 a9 be db b2 2c 85 e4        AES256
1816222472      ad f9 65 ab e7 55 37 10 9d 4b a1 0e 95 20 35 ae 6d 3d 04 31 63 35 a1 49 4b 6e cc 71 d3 56        AES256
1816223120      89 77 12 1d 40 ce c0 41 eb 17 ad bb db c3 ea 72 aa 58 a6 13 b5 7a b2 30 e3 69 4c ce bf 15        AES256
1921482224      d5 9f 24 dc 70 2e 95 a1 aa f8 cd cf c1 9c         AES128
1936150856      6a a0 85 09 69 7d 29 13 8e 9b 78 4a f9 83 90 09 a1 a1 30 9c c6 58 94 93 52 28 c4 93 2b 18        AES256
1936151504      0c b0 3a 4d ab bf 85 ca 13 88 fc 19 a1 0a b7 ea 4c 6b b3 9c bc 92 28 f0 b9 13 9b 63 29 b6        AES256
2008310240      60 37 9b 87 ad 51 5b bc de 04 ca fd 32 82 64 7f f9 51 96 e8 46 5a 70 7b 0b aa 10 38 d2 d8        AES256
2024600964      d3 46 cf f5 56 74 25 1d 8f 3f 7a 90 f7 0c         AES128
2048572432      60 37 9b 87 ad 51 5b bc de 04 ca fd 32 82 64 7f f9 51 96 e8 46 5a 70 7b 0b aa 10 38 d2 d8        AES256
2227338540      25 9a d5 74 a3 b8 e3 6e 41 d5 b6 cc 92 06 a9 8c 06 6f 7e f9 53 8d d8 a9 be db b2 2c 85 e4        AES256
2352972384      2d bc f8 f8 ee 29 42 10 b9 7a 2f d3 15 f3         AES128
2706685096      b9 f8 29 be ab 65 59 58 21 4f 19 93 09 2e db 0c c0 0c e9 2b 47 6a f5 f3 95 79 3e dc eb 61        AES256
2822651048      b9 f8 29 be ab 65 59 58 21 4f 19 93 09 2e db 0c c0 0c e9 2b 47 6a f5 f3 95 79 3e dc eb 61        AES256
2825587964      25 9a d5 74 a3 b8 e3 6e 41 d5 b6 cc 92 06 a9 8c 06 6f 7e f9 53 8d d8 a9 be db b2 2c 85 e4        AES256
2826188268      1c fa c1 b4 fe 2c 1f ab 9d e0 0c 4b c8 8f         AES128
```

4.1.2 Manual analysis

String searching using relevant phrases and keywords was performed against the captured memory dumps. Such analysis is purely an "unstructured" (/manual) analysis of the memory [19], which proved vital in extracting detailed communication artifacts.

A memory dump contains scattered information consisting of *lines* of data that do not necessarily follow any order. To extract application-specific information from the dump, we first developed a sense of the patterns and behaviors that WebEx artifacts exhibit. Consider the following line of information extracted from the memory using the command "*strings [memory dump filename]*| *grep* '*messagesContainer*'":

{{avatar:raw}} {{sparkMessagesDiv:raw}} "" data-object-id="5e6ec853-9ff3-465a-a9d8-768fd9dd62ea"><img

```
# BULK_EXTRACTOR-Version: 1.6.0 ($Rev: 10844 $)
# Feature-Recorder: email
# Filename: memdump.mem
# Histogram-File-Version: 1.1
n=232  zainab          @gmail.com      (utf16=125)
n=189                  gmail.com
n=176  pr1824381488@meet68.webex.com   (utf16=30)
n=93   info@diginotar.nl
n=66   r1824381488@meet68.webex.com    (utf16=65)
n=64   appro@openssl.org
n=45   zainab          @yahoo.com      (utf16=12)
n=44   premium-server@thawte.com
n=31   cps-requests@verisign.com
n=27   1824381488@meet68.webex.com     (utf16=2)
n=26   pkiadmin@trustcentre.co.za
```

Fig. 5 Email addresses of meeting participants via Bulk Extractor

This snippet of memory dump gives every detail about a chat message that was sent by the test user: the actual message, timestamp, message ID, display name of the user receiving the message, and the fact that this user is also a contact of the test user on WebEx. The presence of the *<messagesContainer>* string tag (a constant tag for chat messages) enabled us to extract all the messages exchanged by the test user along with the associated timestamps and other metadata (Fig. 6). A similar analysis of the memory dump using searches based on phrases/keywords such as *webex*, *message*, *meeting*, and *query* was performed to investigate the patterns/tags that would fetch other useful artifacts pertaining to the application. As a result, a plethora of information related to the test user account was extracted. Basic *user account information* extracted includes the name of the test account, associated email address, default WebEx site, personal room number, and video address (Fig. 7). We also observed that the password of the test account was encrypted in memory since we were unable to retrieve it in plaintext. The keywords entered by the user into the search bar to find acquaintances and friends were extracted under the *<query>* tag (Fig. 8). Exchanged text files, media files, and their metadata along with other details were found under the *<sparkShareInfo>* tag (Fig. 9–10). Exchanged URLs were also extracted (Fig. 11). Information related to scheduled meetings was extracted under *<WebExMeetingData>* (Fig. 12). Among other information, the passwords of the scheduled meetings, and in-meeting chat messages were found in plaintext (Figs. 12, 13).

We have developed a *Cisco WebEx memory parsing tool* based on our findings from the manual (unstructured) forensic analysis of the memory dumps. Our memory parsing tool ([20]) can be used to retrieve memory artifacts (including user account information, search keywords, exchanged text/media files, exchanged URLs, deleted URLs, exchanged chat messages, deleted chat messages, scheduled meeting information, and contacts) from any

Fig. 6 Exchanged chat message information via manual string search

```
<div class="sparkAvatar me "></div><div class="messagesContainer "><div tabIndex="-1"
messageId="ca7428d0-a0e4-11eb-89c1-1bf3ae40567c" role="row"  class="messageContainer  "
aria-describedby="infoca7428d0-a0e4-11eb-89c1-1bf3ae40567c"><p class="messageHeader">
<span class='actor' data-object-id='45a83b43-ca6a-4244-832f-8e937c8087aa'>You
<span class='externalDomain'></span></span> <span class="messageTime">08:17 AM</span>
</p><div class="sparkMessage  "
messageId="ca7428d0-a0e4-11eb-89c1-1bf3ae40567c">
<div id="infoca7428d0-a0e4-11eb-89c1-1bf3ae40567c"
aria-hidden="false" class="visualhidden"></div><div class="msgContainer">
<div class="sparkShares ">
</div><div>The meeting is rescheduled for 3 PM, today.</div></div><div class="flagContainer ">
</div><div class="actionContainer"
aria-label=" List item. "><div class="reactionActions ">
<span class="sparkIcon sparkReaction animated partypopper "
reactionSelected="" reactionId="partypopper" messageId="ca7428d0-a0e4-11eb-89c1-1bf3ae40567c"
tabIndex="-1" onclick="sparkBase.toggleReaction(event);" data-sparktt="Celebrate"></span>
```

class="sparkAvatar" draggable="false" participantid="PID-5e6ec853-9ff3-465a-a9d8-768fd9dd62ea" data-object-type="person" data-object-id="5e6ec853-9ff3-465a-a9d8-768fd9dd62ea" onclick="return sparkBase.clickEventHandler(event);" onmouseenter="return sparkBase.mouseEventHandler(event);" onmouseleave="return sparkBase.mouseEventHandler(event);"> <div tabIndex="-1" **messageId="cfa73e80-9a1f-11eb-bd43-b734da68796a"** role="row" class=**"messageContainer "** aria-describedby="infocfa73e80-9a1f-11eb-bd43-b734da68796a"> <spark-contact data-object-type='USER' data-object-id='5e6ec853-9ff3-465a-a9d8-768fd9dd62ea'>**Usmani</spark-contact>** <span class=**"messageTime">4/10/2021, 10:11 PM** <div class="sparkMessage " **messageId="cfa73e80-9a1f-11eb-bd43-b734da68796a"**> **What is going on?**

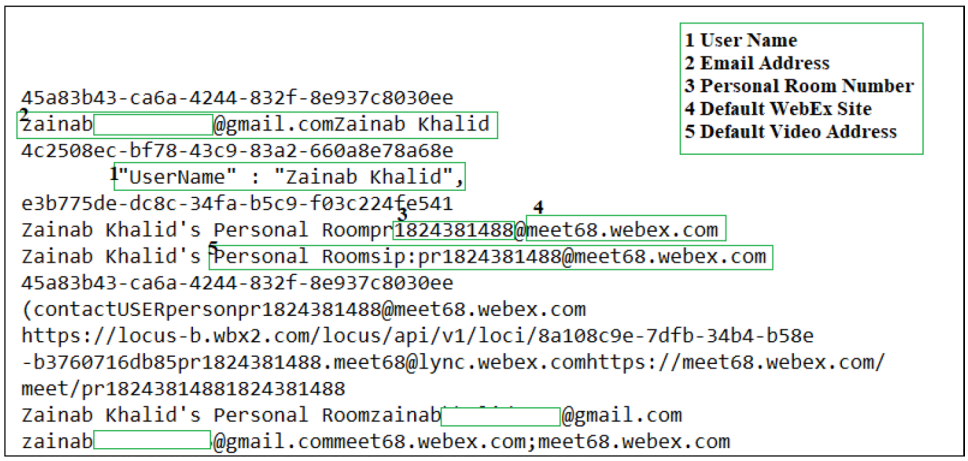

Fig. 7 User account information via manual string search

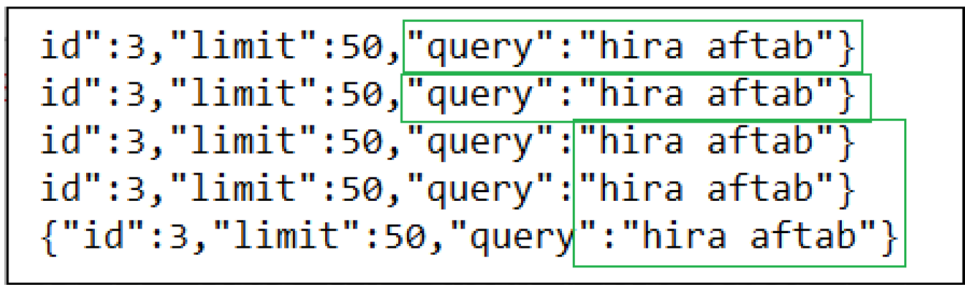

Fig. 8 Keyword search via manual string search

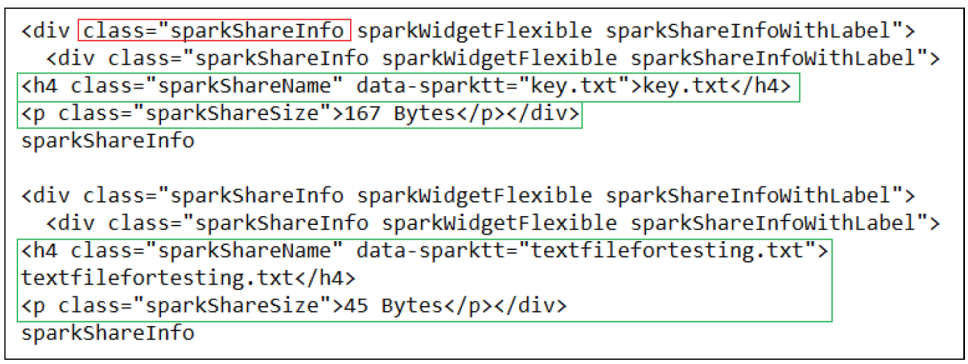

Fig. 9 Exchanged text file information via manual string search

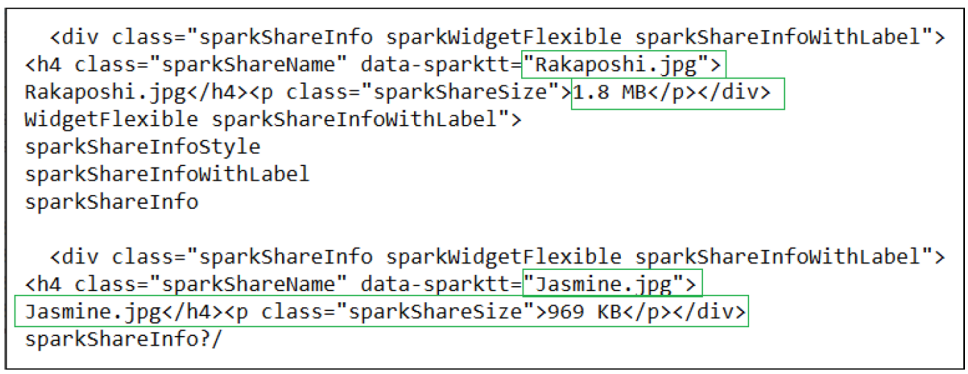

Fig. 10 Exchanged media file information via manual string search

Fig. 11 Exchanged URLs via manual string search

```
<a href=https://www.forensicfocus.com/
class="unfurUrl hideTooltip"
onclick="return sparkBase.clickEventHandler(event);">
```

Fig. 12 Scheduled meeting information via manual string search

```
<?xml version="1.0" encoding="UTF-16"?><CopyData>
<WebExMeetingData><Version>1.0.0.0</Version><UserType>0</UserType>
<MeetingKey>1932698210</MeetingKey><Password>XEhSqTxE736</Password>
<ShareStatus>0</ShareStatus><PDShareStatus>0</PDShareStatus>
<MeetingName>Test Meeting for Sched</MeetingName><SiteName>meet68.webex.com</SiteName>
<SiteID>13918757</SiteID><UserName>Zainab Khalid</UserName>
<UserID>757681932</UserID><MeetingType>1</MeetingType>
<TelephoneInfo>Audio connection:,
Call in phone number(US/Canada): +1-415-655-0001</TelephoneInfo>
<InviteEmailTitle><!CDATA[Join Webex meeting in progress: Test Meeting for Sched]]>
</InviteEmailTitle>
<InviteEmailText><!CDATA[My Webex meeting in progress.
Meeting password: XEhSqTxE736 (93477793 from video systems)
```

Fig. 13 In-meeting chat message extracted via manual string search

```
<span class="messageTime">7:16 AM</span>
 </p><div class="sparkMessage  "
 messageId="e8854310-aec5-11eb-a156-6f4fc1567g44">
<div id="infoe8854310-aec5-11eb-a156-6f4fc1567g44"
aria-hidden="false"
 class="visualhidden"></div><div class="msgContainer">
<div class="sparkShares ">
</div><div>Let us go over the agenda first.</div></div>
<div class="flagContainer ">
```

Fig. 14 Deleted chat message information via manual string search

```
?U53e193b0-a657-11eb-8f76-71d27fc1330deac61ec0-9781-11eb-837a-d761b877e81e
Did you contact ABC?45a83b43-ca6a-4244-832f-8e937c8030ee

?U53e193b0-a657-11eb-8f76-71d27fc1330deac61ec0-9781-11eb-837a-d761b877e81e
Did you contact ABC?45a83b43-ca6a-4244-832f-8e937c8030ee

<p>Did you contact ABC?</p>
<p>Did you contact ABC?</p>
```

memory dump taken from a Windows machine with Cisco WebEx as one of the running applications.

4.1.3 Anti-forensics

When a WebEx user deletes information such as chat messages and exchanged files, the information is still recoverable from memory using manual string searches, thus enabling the detection of attempts to obstruct evidence. In our case, we recovered deleted URLs from the memory. While deleted messages were also recovered from the memory dumps, it was observed that they were not found in the usual *message containers*. Although discarded from the *message containers* upon deletion, these messages were still recoverable, preceding the *user identity*, and under the paragraph href tags, i.e., (Fig. 14). Chat messages exchanged during the videoconference meeting were also recovered from the memory dump under the paragraph href tags (Fig. 15).

4.2 Disk-space forensics

As opposed to volatile memory, a device's hard disk retains application data for a longer time. The application folder,

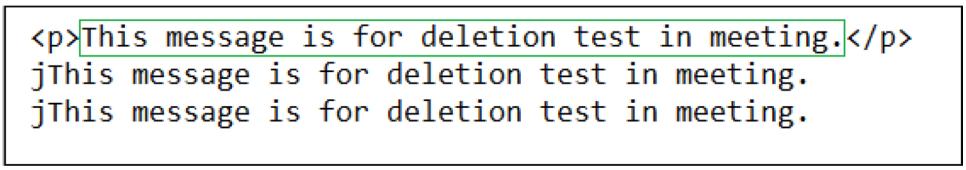

Fig. 15 In-meeting deleted chat message extracted via manual string search

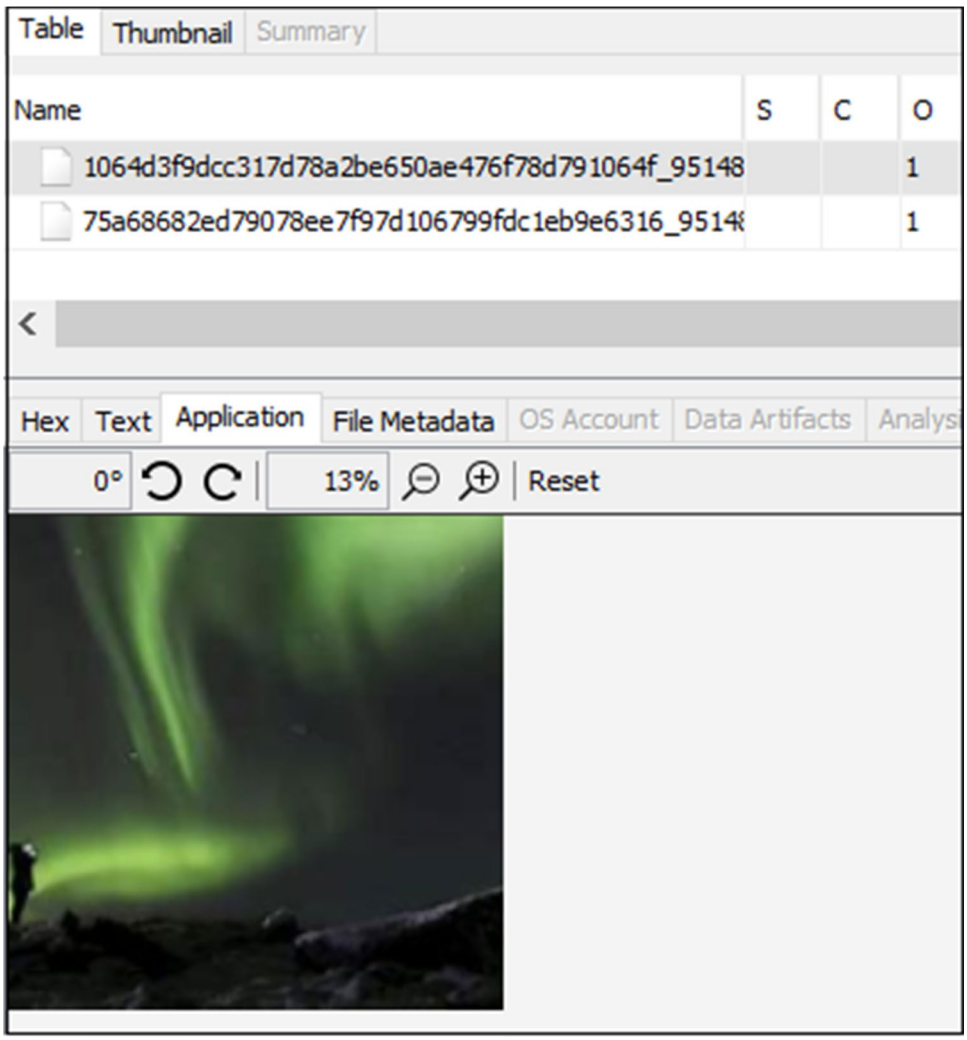

Fig. 16 Avatars of logged-in accounts via Autopsy

Windows Registry, and prefetch files on disk are potential sources of forensic artifacts.

4.2.1 Cisco WebEx data directory structure

WebEx leaves remnants in the *Local*, *Locallow*, and *Roaming* application folders. In the Roaming subfolder, *\AppData\Roaming\webex\Avatar*, WebEx saves avatar of the currently logged-in account and also accounts that were previously logged into the application, if any (Fig. 16). In addition, a *.ini* file with information regarding the last joined location is stored in *\AppData\Roaming\webex\Avatar\latestjoinedlocation.ini* (Fig. 17). The *Roaming* folder also stores a temporary *.dat* file (*\AppData\Roaming\webex\Avatar\QSXMLFile.dat*) that has information regarding previous meetings of the account including meeting numbers, timestamps, and passwords of the meetings (Fig. 18). It is pertinent to note that the QSXMLFile.dat file resides in the file system for a temporary time following the event of a scheduled meeting that has recently been scheduled/conducted by the user.

In the *Local* folder, the application creates (1) WebEx and (2) CiscoSpark folders separately. The presence of these two separate folders is attributed to the fact that WebEx was a separate application before Cisco acquired it. The WebEx folder contains (1) *.json* scripts for different browsers to

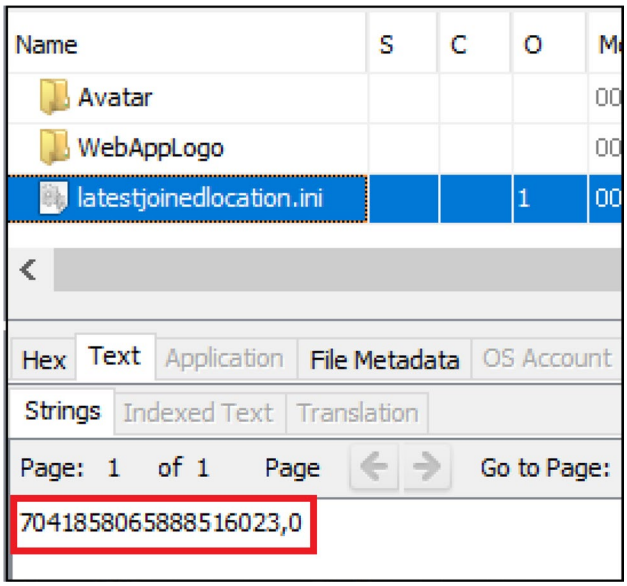

Fig. 17 Last joined location via Autopsy

enable the application startup, (2) site information that consists of the default WebEx site of the user (or multiple default sites if more than one account was logged in on the device), (3) WebEx cache, and (4) the WebEx application itself along with application extensions.

From the *CiscoMeetings* database in the same folder, *AppData\Local\WebEx\CiscoMeetings.db*, we were able to retrieve records of meetings, and their timestamps, along with default WebEx sites of previous and currently logged-in accounts (Fig. 19). Client info of the logged-in accounts was also found in the same database including blackList, siteURL, URLroot of the account (Fig. 20), and the timestamp of when the last meeting was held along with other application-related information (Fig. 21).

The CiscoSpark folder contains several artifacts of interest for digital forensic investigation as well. In particular, the subfolder *\AppData\Local\CiscoSpark\media\calls* stores call logs of the logged-in user. Timestamps from these logs can be corroborated with evidence extracted from the *CiscoMeetings* database to infer useful insights regarding meetings and calls from the logs. Although log analysis requires intensive manual searching, they provide useful information such as call IDs, media session IDs, IP addresses, media statistics such as screenshare/video resolution, and number of audio/video packets received.

The *\AppData\Local\CiscoSpark\[User ID]* folder contains *spark_persistent_store.db* and *spark_roaming_store.db* databases that are encrypted.

Another database, *spark_shared_store.db* (*\AppData\Local\CiscoSpark\spark_shared_store.db*), stores installation ID, and user ID of the previous account that was logged in. In the CiscoSpark folder, *lifecycle.dat* stores login state of the user account (logged in vs logged out).

4.2.2 Windows Registry keys for Cisco WebEx

Following an in-depth analysis of Windows Registry for the presence of WebEx artifacts, we identified several registry keys that revealed valuable information about previous user activity (Fig. 22). These include credentials, profile avatars, configuration settings, application settings, and product information. In particular, the *HKCU\SOFTWARE\WebEx\ProdTools\Logon\[Default WebEx Site]* key stores the login credential and profile avatar of the user. Credentials of previously logged-in users are also stored in this key. The *HKCU\SOFTWARE\WebEx\FeaturePayloads* key stores user configuration settings in reference to the default WebEx site, e.g., *"EnableAES256GCM":true.* A total of 141 configuration settings are stored in this key. The *HKCU\SOFTWARE\WebEx\Config* key stores application settings such as the theme type setup by user and whether panelist and attendees' names are shown.

Fig. 18 Temporary QSXML-File.dat file

```
Test Meeting for Sched
Friday, June 20, 2021
8:56 AM | Pacific Daylight Time (San Francisco, GMT-07:00) | 50 mins
Meeting number: 193 269 8210
Meeting password: XEhSqTxE736 (93477793 from video systems)

Join Meeting
https://meet68.webex.com/meet68/e.php?MTID=m9b386686257137ceb0d979f27b0d5715

JOIN BY PHONE
Use VOIP only

JOIN FROM A VIDEO SYSTEM OR APPLICATION
Dial sip:1932698210@meet68.webex.com
You can also dial 173.243.68 and enter your meeting number.
```

Table MeetingHistory 11 entries Page 1

Id	Site	Date
17	meet31.webex.com	2021-10-26 13:08:08
18	meet31.webex.com	2021-10-26 13:13:48
19	meet31.webex.com	2021-10-26 13:39:10
20	meet31.webex.com	2021-10-26 13:39:52
21	meet31.webex.com	2021-10-26 14:10:57
22	meet31.webex.com	2021-10-26 14:11:05
23	meet31.webex.com	2021-10-26 14:11:19
24	meet68.webex.com	2021-10-26 14:11:39
25	meet31.webex.com	2021-12-15 13:56:35
26	meet68.webex.com	2021-12-15 14:07:47
27	meet68.webex.com	2021-12-15 14:25:40

Fig. 19 CiscoMeetings.db via Autopsy

4.2.3 Cisco WebEx prefetch files

Cisco WebEx has two associated prefetch files stored in the Windows file system, namely CISCOCOLLABHOST.EXE-49749B78.pf and WEBEXHOST.EXE-7D2F62CC.pf. The eight characters trailing the name of the executable is a hash of the application's location and may differ from one device to another. The sizes of the two prefetch files in our test device were 116 KB and 26 KB, respectively. This suggested that the application was used frequently. The prefetch files were parsed (using PECmd) for information about the run times of the executables and related timestamps. As illustrated in Fig. 23, the CISCOCOLLABHOST.EXE-49749B78.pf and WEBEXHOST.EXE-7D2F62CC.pf files had run counts of 25 and 15, respectively.

In addition to run counts, parsing results related to creation time, modified time, last accessed time, and volume information along with directories, and files referenced by the executables were extracted as shown in Fig. 23.

Fig. 20 CiscoMeetings.db client info via Autopsy

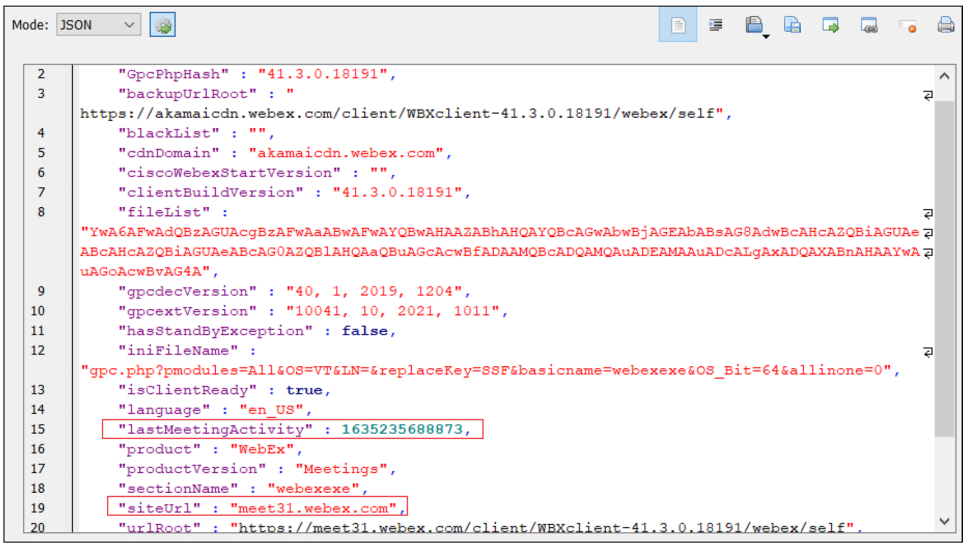

Fig. 21 CiscoMeetings.db client info via DB Browser for SQLite

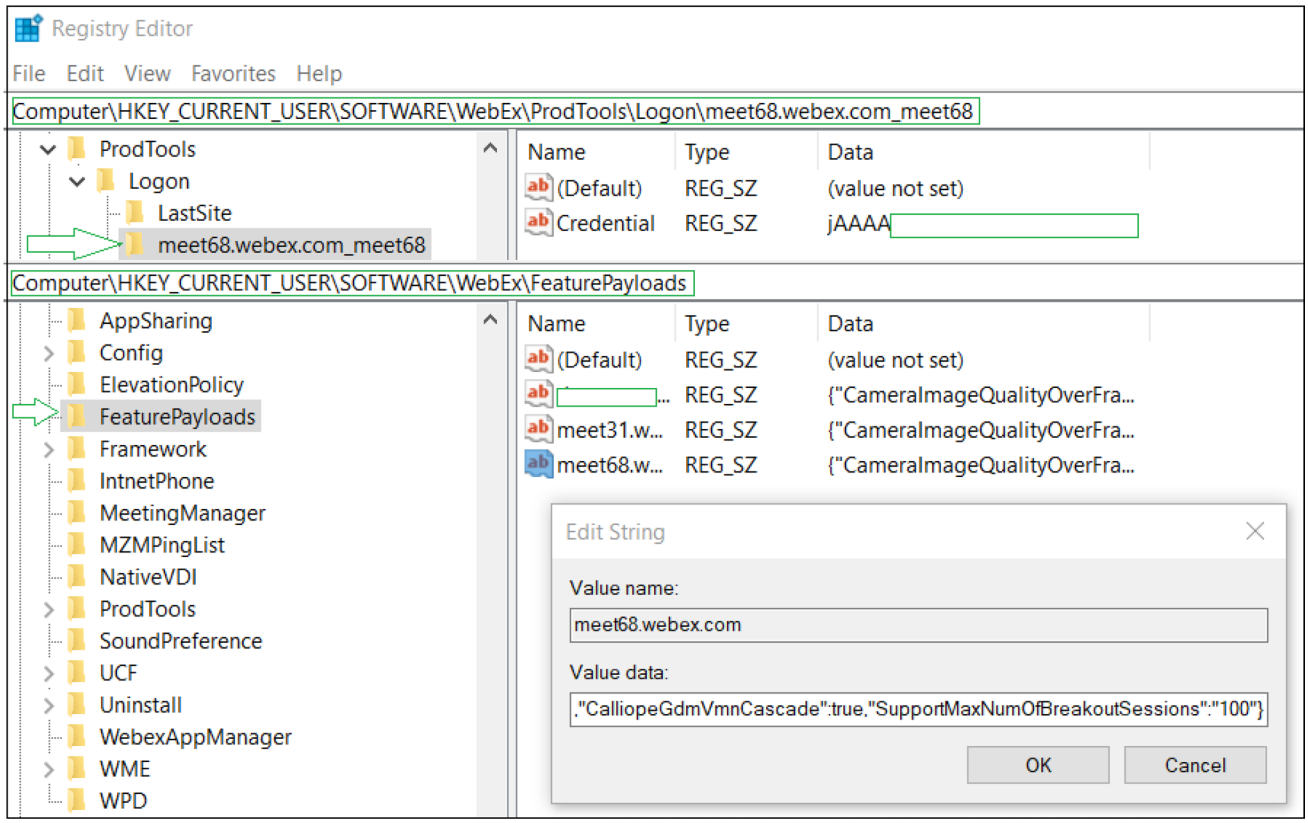

Fig. 22 Cisco WebEx Registry keys

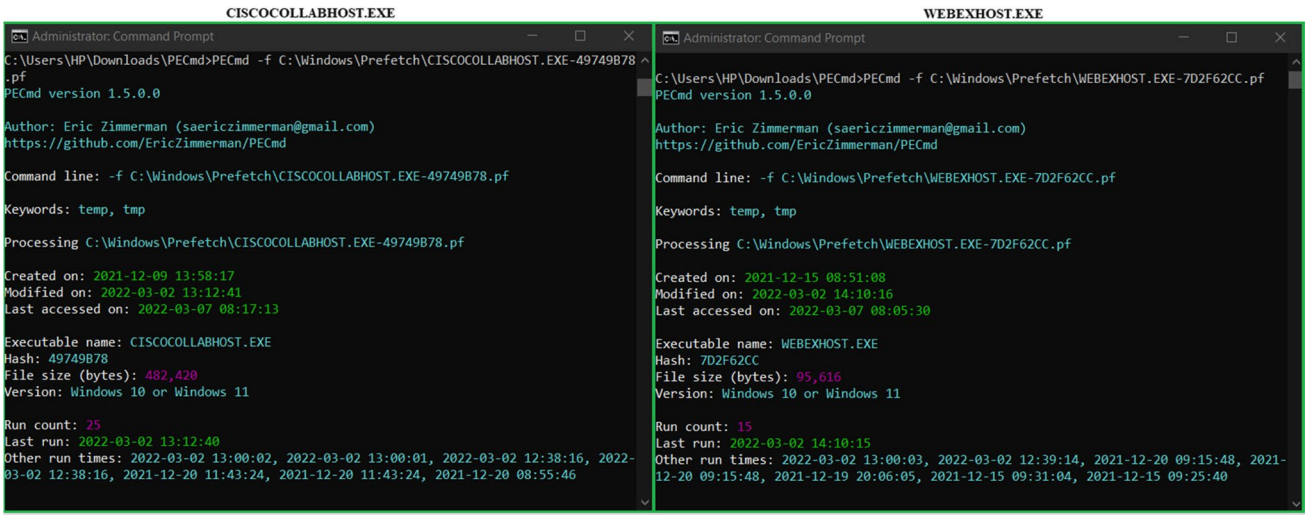

Fig. 23 Parsing results for prefetch files CISCOCOLLABHOST.EXE-49749B78.pf and WEBEXHOST.EXE-7D2F62CC.pf

4.3 Network forensics

WebEx has three related processes, among others: *CiscoCollabHost.exe*, *atmgr.exe*, and *washost.exe*, which may emerge while parsing memory for artifacts. Figure 24 shows the *netscan* output for WebEx. The additional system processes, such as *svchost.exe*, have been filtered for the sake of brevity. In addition to the *CiscoCollabHost.exe* process, the *atmgr.exe* process also appeared. This process belongs to the AtMgr module of Cisco WebEx. After the connection was established, and the meeting began, this process appeared during the transfer of meeting media (audio/video/text). As

Offset(P)	Proto	Local Address	Foreign Address	State	Pid	Owner	Created
0x139e744f0	UDPv4	0.0.0.0:58652	*:*		4212	CiscoCollabHos	2021-11-16 06:14:05 UTC+0000
0x13a020670	UDPv4	0.0.0.0:0	*:*		4212	CiscoCollabHos	2021-11-16 06:13:59 UTC+0000
0x13a2dec00	UDPv4	0.0.0.0:3702	*:*		4212	CiscoCollabHos	2021-11-16 06:14:47 UTC+0000
0x13a53f8c0	UDPv4	0.0.0.0:64757	*:*		4212	CiscoCollabHos	2021-11-16 06:14:46 UTC+0000
0x13a017ee0	TCPv4	0.0.0.0:49154	*:*		4212	CiscoCollabHos	
0x13ab471c0	UDPv6	:::64758	*:*		116	svchost.exe	2021-11-16 06:14:46 UTC+0000
0x13a760950	TCPv4	0.0.0.0:49152	0.0.0.0:0	LISTENING	548	wininit.exe	
0x13a8b3010	TCPv4	192.168.216.1:139	0.0.0.0:0	LISTENING	4	System	
0x13ab6e480	TCPv4	0.0.0.0:49155	0.0.0.0:0	LISTENING	632	lsass.exe	
0x13a920010	TCPv4	127.0.0.1:49291	127.0.0.1:49290	ESTABLISHED	-1		
0x13c0be300	TCPv4	0.0.0.0:445	0.0.0.0:0	LISTENING	4	System	
0x13c0be300	TCPv6	:::445	:::0	LISTENING	4	System	
0x13c3adcd0	TCPv4	127.0.0.1:49233	127.0.0.1:49234	ESTABLISHED	-1		
0x13c5f2d00	UDPv6	fe80::98fc:1030:989e:9b3a:1900	*:*		3832	svchost.exe	2021-11-16 06:15:26 UTC+0000
0x13e8b7660	TCPv4	0.0.0.0:49163	0.0.0.0:0	LISTENING	608	services.exe	
0x13e6a36a0	TCPv4	127.0.0.1:49510	127.0.0.1:49511	ESTABLISHED	-1		
0x13f275b10	UDPv4	127.0.0.1:49512	0.0.0.0:0	LISTENING	3252	atmgr.exe	
0x13f275b50	UDPv4	127.0.0.1:49512	0.0.0.0:0	LISTENING	3252	atmgr.exe	
0x13f275b90	UDPv4	127.0.0.1:49512	0.0.0.0:0	LISTENING	3252	atmgr.exe	

Fig. 24 Netscan output extracted via Volatility

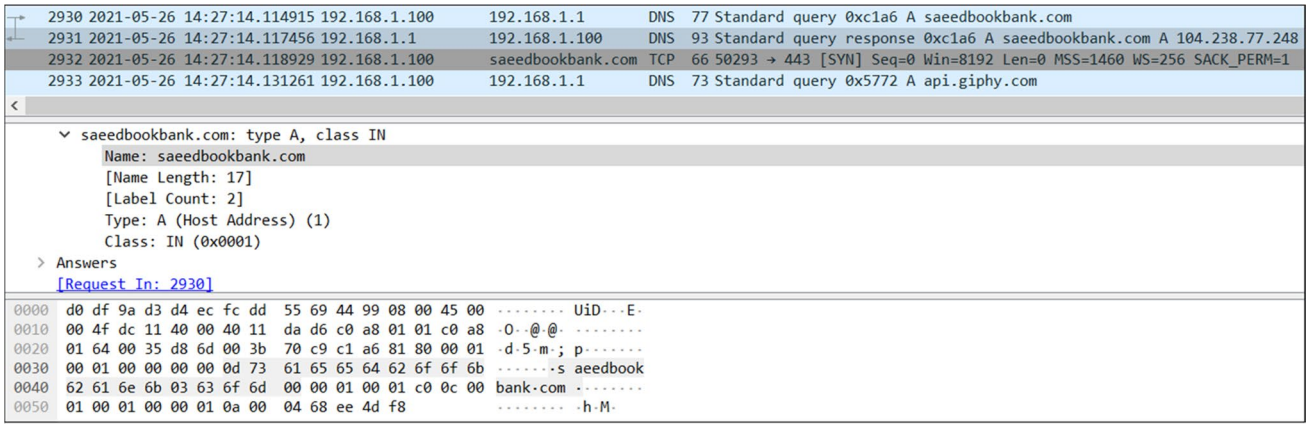

Fig. 25 Plaintext URLs sent in WebEx chat extracted via Wireshark

can be seen in Fig. 24, the atmgr.exe process is associated with User Datagram Protocol (UDP), which is the protocol WebEx uses for transferring meeting media. Timestamps, and PIDs from *netscan*, can be corroborated with the *pslist/pstree* output or with other artifacts. The *netscan* output also lists Internet Protocol (IP) foreign addresses that can be further analyzed for origin traces. The physical or virtual offsets can be used to locate the processes in the dump files via a hex editor.

While memory and disk are valuable sources for forensic artifacts, memory is not always available because of its volatile nature, and disk can be manipulated (to an extent). Accordingly, network forensics can be a practical alternative as *one advantage to watching the network is that the network can't lie* [21].

WebEx does encrypt all of its sessions using Transport Layer Security (TLS) v1.2. Multimedia traffic is sent using the UDP protocol in encrypted form. From our observations, signaling data and media are encrypted since no credentials, images, or files were found in plaintext. However, the URLs exchanged via chat sessions did appear in plaintext over the network. Images pertaining to exchanged URLs were also observed as shown in Fig. 25.

Logging into WebEx, we found that a session with Amazon.com, Inc. was created on port 443. This makes sense since Cisco uses both Amazon Web Services and Microsoft Azure to provide cloud services. Then, a session with Cisco WebEx LLC servers (global-idbroker-eu.webex.com, idbroker-eu.webex.com, and identity-eu.webex.com) was established on port 443 to authenticate the host to the cloud. Another session with ocsp.quovadisglobal.com was subsequently established to request the certificate revocation lists (CRLs) on Hypertext Transfer Protocol (HTTP) port 80. Account information such as profile ID, contacts, and meeting information was retrieved from the servers (jabber-integration-k.wbx2.com, conv-k.wbx2.com, contacts-service-k.wbx2.com, avatar-k.wbx2.com, calendar-k.wbx2.com) on port 443. As discussed before, since all network traffic

Table 2 Network information—Cisco WebEx

Cisco WebEx LLC	
• global-idbroker-eu.webex.com	62.109.208.10
• idbroker-eu.webex.com, identity-eu.webex.com	206.197.206.43
• global-nebulab.webex.com	
• meet68.webex.com [user's default WebEx site]	
Amazon.com. Inc.	
• afra-sni.wbx2.com, ha-k-3az-sni.wbx2.com	18.156.21.71
• u2c.wbx2.com, metrics-k.wbx2.com	18.156.21.15
• ocsp.quovadisglobal.com, ds.ciscospark.com	18.156.9.248
• wdm.k1.ciscospark.com, conv-k.wbx2.com	34.249.165.103
• jabber-integration-k.wbx2.com	18.157.51.3
• meet-wbx2-3az-[id].elb.eu-central-1.amazonaws.	35.158.159.116
com, calliope-k.wbx2.com, contacts-service-k.	3.126.199.220
wbx2.com	3.125.194.2
• avatar-k.wbx2.com, files-api-k.wbx2.com	35.158.159.125
• people.webex.com, hydra-k.wbx2.com	52.12.48.103
• janus-k.wbx2.com, calendar-k.wbx2.com	
Akamai Technologies, Inc.	
• e11070.b.akamaiedge.net	23.12.98.160
• query.prod.cms.rt.microsoft.com.edgekey.net	88.221.196.113
• crl.certum.pl, crl.certum.pl.edgekey.net	
Highwinds Network Group, Inc.	
crl.usertrust.com, crl.comodoca.com	151.139.128.14

is encrypted, captured frames associated with users' credentials, files, profile IDs, and contacts were not helpful for the forensic investigation. However, NetworkMiner does retrieve digital certificates exchanged during the videoconference which validate whether communicating nodes were authenticated or not.

The IP addresses, and timestamps from the traffic, can be useful in reconstructing events, and attributing whom the host communicated with and when. The communication artifacts can also be utilized as signatures/flags for Cisco WebEx network traffic.

Table 2 illustrates captured network details associated with the IP addresses and the servers that the host communicated with.

5 Cisco WebEx web application forensics

Cisco WebEx web version is meant to alleviate the need to install the full desktop version (which requires storage space) while allowing the users to still benefit from the usage of Web Services (WS) *on-the-go*. One might conjecture that since the application is not installed on the actual OS, it must not leave any artifacts behind that may be of sensitive value. While this may be true in the sense that no application/installation folders are created for these web applications, nonetheless, a substantial amount of information can still be extracted from the Google Chrome data directory, as elaborated in this section. Our target artifacts in browser forensics included (1) traces/indicators of Cisco WebEx web application's usage, (2) meeting records, (3) history, (4) downloads, (5) bookmarks, (6) cache, (7) cookies, and (8) associated profile picture, and email address. These are further discussed below:

5.1 Traces of usage

The first question to be addressed during our forensic analysis of the WebEx web application was whether the application was used in the first place, and if so, to whom the application's usage can be attributed. We found that there are several artifacts that may help in tracing Cisco WebEx web application's usage. In particular, *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\JumpListIconsMostVisited* and *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\JumpListIconsRecentClosed* store the icons of most visited and recently closed web applications. *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Top Sites* is a similar indicator. *Topsites* is an SQLite database that stores the thumbnails of top sites visited by the user. The favicons database at *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Favicons* also stores favicons of the web pages/applications. In our case, the Cisco WebEx web application was present in all the four folders/databases, which might not be the case in instances of less frequent usage.

In addition to the above, the presence of a Cisco WebEx URL in the Network Action Predictor SQLite database (*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Network Action Predictor*) is an indicator of its past usage.

The QuotaManager SQLite database (*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\QuotaManager*) listed the use count of the web application along with the last accessed timestamp. The last accessed timestamp may also be extracted from the shortcuts SQLite database (*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Shortcuts*).

Session data stored in the folder *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Sessions* also indicated that the WebEx web application was used.

Once it is established that Cisco WebEx web application was used, an artifact of attribution as to whom the user was can be found at *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Accounts\Avatar Images* and at *C:\Users\[usermane]\AppData\Local\Google\Chrome\User Data\Default\Google Profile Picture*. We extracted the email address associated with the profile from the *Sessions* folder mentioned earlier. The logs stored

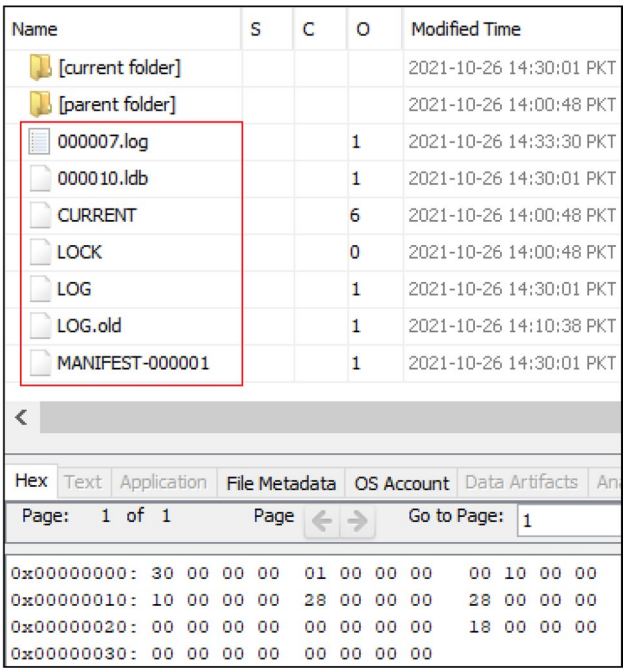

Fig. 26 IndexedDB-levelDB structure via Autopsy

in the folder disclosed the email address of the user. These session logs may also divulge links of the meetings that were conducted. However, any additional details, such as in-call messages, were not found in the sessions folder.

5.2 IndexedDB-levelDB

Whenever a web application is invoked from Google Chrome, an indexedDB-levelDB database corresponding to that web application is created in the Google Chrome data directory on the client's desktop. This artifact is a strong indicator of *trace of usage* for the Cisco WebEx web application. IndexedDB is a novel browser Application Programming Interface (API) based on the levelDB key-value pair database structure. It essentially stores and retrieves session data for various web applications that are activated from a browser [12, 22]. In the Cisco WebEx case, a levelDB named "https_[default WebEx site]_0.indexeddb.leveldb" is created in the *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\IndexedDB* folder.

Fig. 26 shows structure of a levelDB as seen via Autopsy.

The *.log* and *.ldb* files were of particular interest in our forensic investigation. Accessing the raw data in the database through Autopsy was tricky since it showed the database contents in unordered/scattered manner. It should be noted that although Autopsy presented the contents in an unordered manner, these were however presented in a readable (strings) format (which is not the case when this database is viewed directly via a text editor such as notepad).

To demystify the data stored in Cisco WebEx's levelDB, we dumped the contents from the database into a *.json* file using a Python script developed in [12] which essentially converts electron-based levelDB into *.json*; although Cisco WebEx is not based on electron, the subject Python script worked fine for Cisco WebEx levelDB. Accordingly, we investigated the contents of the database to see how it is organized and what information can be retrieved from it.

Cisco WebEx levelDB's *object stores* are arranged *meeting wise*, meaning the logs of each meeting that successively takes place are stored in successive object stores, namely logStore1, logStore2, logStore3, and logStore4 (Figs. 27, 28, 29, 30, 31, 32, 33). These stores log detailed information about every meeting that takes place, including event logs. The logs are numbered/serialized chronologically. The *value* in key-value pairs that encompass the object stores is an important element of particular interest. It stores the timestamps of every event, serial numbers, records of meetings, and associated events. These events can also be categorized according to one of the service managers that manage events emanating from a WebEx web application session, namely serviceMgr, confMgr, conf, videoMgr, mediaCSIMgr, activeMgr, chatMgr, webServiceMgr, CMSC, among others. For instance, records of exchanged chat messages can

Fig. 27 Chat message recovered from levelDB

```
{
   "key": "b'\\x00\\x01\\x01\\x01\\x01\\x04\\x00g\\x003\\x003\\x001\\x01\\xd0\\x0f\\x00\\x00\\x00\\x00\\x00'",
   "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000010.ldb",
   "seq": 4048,
   "state": "KeyState.Live",
   "store": "logStore1",
   "value": {
       "id": "g331",
       "p": "[\"[334] [02:01:04.210]\",\"handleMeetingParameters4Client is {\\\"Client\\\":{\\\"AllowAttendeeToUnmuteSelf\\\":1,
            \\\"AvatarServiceUrl\\\":\\\"https://avatar-a.wbx2.com/avatar/api/v1\\\",\\\"ClosedCaptionEnableByDefault\\\":0,\\\
            "DelayEstEx\\\":0,\\\"DynamicFecModeWithRTX\\\":\\\"TABLE_LOOKUP_ONLY\\\",\\\"EnableAES256GCM\\\":1,\\\"EnableAES256GCMForNonE2E\\\":1,
            \\\"EnableAppHub\\\":1,\\\"EnableAppHubForLargeEvent\\\":0,\\\"EnableAppHubForLoginUser\\\":1,\\\"EnableAttendeeSessionRoster\\\":1,\\\
            "EnableBreakoutSessionForTeams\\\":1,\\\"EnableCaptionsPanel\\\":1,\\\"EnableCbPreCreateCacheEx\\\":0,\\\"EnableEvent\\\":false,\\\
            "EnableFECForRetransmission\\\":\\\"1\\\",\\\"EnableHardMute\\\":1,\\\"EnableHardMuteModeratedModeInBO\\\":0,\\\"EnableImmersiveSharing\\\":1,
            \\\"EnableMQEAudioProcessDataUpload\\\":0,\\\"EnableMeetingReactionForTeams\\\":1,\\\"EnableMuteNotification\\\":false,\\\"EnablePostMeetingHook\\\":1,
            \\\"EnablePreMeetingLobby\\\":1,\\\"EnableQA\\\":1,\\\"EnableQAForLargeEvent\\\":1,\\\"EnableRaiseHandInBO\\\":1,\\\"EnableReaction\\\":1,
            \\\"EnableShareOnMCSInBO\\\":1,\\\"EnableShortAttendeeID\\\":0,\\\"EnableShowHideReactorName\\\":1,\\\"EnableSimplifyCMR\\\":1,
            \\\"EnableSubCBRaiseHand\\\":false,\\\"EnableUnifyRaiseHand\\\":1,\\\"EnableWebinar\\\":0,\\\"HostId\\\":757681932,\\\"MMPChooseHybridASN\\\":
            \\\"BO_SUPPORT\\\",\\\"MediaDropTimeOut\\\":45,\\\"MeetingInstanceId\\\":\\\"6c27111b81e1407db79e4b0540a03fc5_I_209222404475484472\\\",
            \\\"MuteUponEntry\\\":0,\\\"NoPostMeetingHook\\\":0,\\\"RealtimeTranscriptEnableByDefault\\\":0,\\\"SupportBreakoutSessions\\\":1,
            \\\"SupportInMeetingCrossOrgPolicy\\\":1,\\\"SupportNonEnglishASR\\\":0,\\\"SupportRealtimeTranscript\\\":0,\\\"SupportRealtimeTranslation\\\":0,
            \\\"SupportSeparatedClosedCaption\\\":0,\\\"TurnOnBreakoutSessions\\\":0,\\\"Video1080PMaxbitrateEnabled\\\":0,\\\"Video1080PResolutionEnabled\\\":0,
            \\\"VideoMute\\\":1,\\\"confLockWithLobby\\\":1,\\\"embeddedAppForTeams\\\":0,\\\"lobbyTimeoutMinutes\\\":30},\\\"compressed\\\":false}\"]",
       "t": 1635238864211.0
   }
```


Fig. 28 Meeting settings recovered from levelDB

```
{
    "key": "b'\\x00\\x01\\x04\\x01\\x01\\x03\\x00l\\x008\\x007'",
    "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000007.log",
    "seq": 92515,
    "state": "KeyState.Live",
    "store": "logStore4",
    "value": {
        "id": "l87",
        "p": "[\"[86] [02:29:59.619]\",\"[(jsMeetingPing)]parameter is \\\"action=pre_join_meeting_ping&gdm=1&
             site_id=13918757&user_id=0&meeting_id=209224234739726878&meeting_name=Test%202.209224234739726878&
             meeting_key=255247111128est_num=10&correlationId=d7d855c6-72a3-424e-b9b0-bd4c44acf06d&trackingId=D7
             &nodeType=7936&startLocalTime=2021y10m26d9h29m59s619ms&startTicket=1635240599&\\\"\"]",
        "t": 1635240599619.0
    }
```


Fig. 29 Meeting name, password, and timestamp of scheduled meeting recovered from levelDB

```
{
    "key": "b'\\x00\\x01\\x01\\x01\\x01\\x05\\x00g\\x001\\x002\\x002\\x008\\x01H:\\x00\\x00\\x00\\x00\\x00'",
    "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000010.ldb",
    "seq": 14920,
    "state": "KeyState.Live",
    "store": "logStore1",
    "value": {
        "id": "g1228",
        "p": "[\"[1240] [02:01:06.701]\",\"[(shareMgr)]onDocCreateConnectionSuccess, join docshare session success\"]"
        "t": 1635238866701.0
    }
```


Fig. 30 Screensharing event recovered with timestamp from levelDB

be found in the object stores under the *chatMgr* identifier (Fig. 27). As may be seen, information regarding the sender, and receiver(s) of the message, the actual message, and relative timestamps can be retrieved.

Information regarding meeting settings (Fig. 28), meeting name, and meeting keys/passwords in case of scheduled meetings (Fig. 29), number of participants in each meeting, added, and removed participant counts can also be retrieved. Instances/events (and timestamps) of screensharing (Fig. 30), international call numbers of users (Fig. 31), user IDs, session IDs, node IDs of client device(s), and whether audio/video was on (and timestamps of when they were) were successfully retrieved. The CMSC service manager logged networking details about the meetings including the client's IP address, the server IP

```
{
    "key": "b'\\x00\\x01\\x01\\x01\\x01\\x05\\x00g\\x001\\x002\\x008\\x002\\x01\\xcc<\\x00\\x00\\x00\\x00\\x00'",
    "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000010.ldb",
    "seq": 15564,
    "state": "KeyState.Live",
    "store": "logStore1",
    "value": {
        "id": "g1282",
        "p": "[\"[1294] [02:01:06.944]\",\"[(hybrid)]callInNumbers = \\\"+1-415-655-0001\\\",
            m_GTNIndex = 0, callInNumInternational = \\\"\\\", hostLabel = \\\"\\\", hostCode = \\\"\\\",
            attendeeLabel = \\\"\\\", attendeeCode = \\\"\\\", TSNo = 0, mergeCode = undefined\"]",
        "t": 1635238866944.0
    }
```


Fig. 31 User call number recovered from levelDB


```
{
    "key": "b'\\x00\\x01\\x04\\x01\\x01\\x04\\x00l\\x001\\x004\\x003'",
    "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000007.log",
    "seq": 93171,
    "state": "KeyState.Live",
    "store": "logStore4",
    "value": {
        "id": "l143",
        "p": "[\"[142] [02:30:00.811]\",\"[(jsMeetingPing)]on_ping_return, response 5:
            \\\"client_ext_ip>119.160.65.147</client_ext_ip><enhance_GDM_ping>0</enhance_GDM_ping><zone_type>2</zone_type></HTML>\\\"\"]",
        "t": 1635240600811.0
    }
```


Fig. 32 Client IP address recovered from levelDB

Fig. 33 Network details recovered from levelDB


```
{
    "key": "b'\\x00\\x01\\x01\\x01\\x01\\x05\\x00l\\x002\\x003\\x001\\x007\\x01\\xccm\\x00\\x00\\x00\\x00\\x00'",
    "origin_file": "C:\\Users\\HP\\Desktop\\https_meet68.webex.com_0.indexeddb.leveldb\\000010.ldb",
    "seq": 28108,
    "state": "KeyState.Live",
    "store": "logStore1",
    "value": {
        "id": "l2317",
        "p": "[\"[2339] [02:02:45.12]\",\"[CMSC]
            [simpleapi.js?v=fdeb827e7e4ec2938fca576167dca44b:14:5771]\",
            \"\\\" showData = \\\"\",\"\\\"{\\\\\\\"mediaType\\\\\\\":\\\\\\\"video
            \\\\\\\",\\\\\\\"common\\\\\\\":{\\\\\\\"bytesReceived\\\\\\\":524779,
            \\\\\\\"bytesSent\\\\\\\":0,\\\\\\\"bitrateReceived\\\\\\\":419823,\\\\\\\"
            "bitrateSent\\\\\\\":0,\\\\\\\"pdrReceived\\\\\\\":2.59,\\\\\\\"
            "remotePdr\\\\\\\":0,\\\\\\\"packetsReceived\\\\\\\":693,\\\\\\\"
            "packetsSent\\\\\\\":0,\\\\\\\"rtt\\\\\\\":0.134,\\\\\\\"inBW\\\\\\\":329739,
            \\\\\\\"outBW\\\\\\\":300000,\\\\\\\"server\\\\\\\":\\\\\\\"mcs\\\\\\\",\\\\\\\"
            "localIp\\\\\\\":\\\\\\\"127.0.0.1\\\\\\\",\\\\\\\"
            "reflexIp\\\\\\\":\\\\\\\"127.0.0.1\\\\\\\",\\\\\\\"
            "remoteIp\\\\\\\":\\\\\\\"23.89.129.71\\\\\\\",\\\\\\\"
            "peerReflexiveIP\\\\\\\":\\\\\\\"119.160.65.147\\\\\\\",\\\\\\\"
            "transportType\\\\\\\":\\\\\\\"UDP\\\\\\\"},\\\\\\\"peers\\\\\\\":
            {\\\\\\\"0\\\\\\\":{\\\\\\\"video\\\\\\\":{\\\\\\\"codec\\\\\\\":\\\\\\\"h264
            \\\\\\\",\\\\\\\"sscRx\\\\\\\":2416348620,\\\\\\\"bytesReceived\\\\\\\":524779,
            \\\\\\\"packetsReceived\\\\\\\":693,\\\\\\\"packetsLostReceived\\\\\\\":18,\\\\\\\"
            "jitterReceived\\\\\\\":4.463,\\\\\\\"avgJitterDelay\\\\\\\":0.09163,\\\\\\\"
            "pdrReceived\\\\\\\":2.59,\\\\\\\"bitrateReceived\\\\\\\":419823,\\\\\\\"
            "packetRateReceived\\\\\\\":69.3,\\\\\\\"fHeightReceived\\\\\\\":360,
            \\\\\\\"fWidthReceived\\\\\\\":640,\\\\\\\"frameRateRendered\\\\\\\":9.1,
            \\\\\\\"frameRateReceived\\\\\\\":9.2,\\\\\\\"framesDecoded\\\\\\\":91,\\\\\\\"
            "keyFramesDecoded\\\\\\\":3,\\\\\\\"pliCount\\\\\\\":0,\\\\\\\"avgDecodeTime\\\\\\\"
            ":0.0023,\\\\\\\"localPort\\\\\\\":56771}},\\\\\\\"1\\\\\\\":{\\\\\\\"video\\\\\\\":{}},
            \\\\\\\"2\\\\\\\":{\\\\\\\"video\\\\\\\":{}},\\\\\\\"3\\\\\\\":
            {\\\\\\\"video\\\\\\\":{}},\\\\\\\"4\\\\\\\":{\\\\\\\"video\\\\\\\":{}},
            \\\\\\\"5\\\\\\\":{\\\\\\\"video\\\\\\\":{}},\\\\\\\"6\\\\\\\":
            {\\\\\\\"video\\\\\\\":{}}}\\\"\"]",
        "t": 1635238965012.0
    }
```


addresses that the client communicated with during the meeting (i.e., 62.109.233.128, 62.109.233.150, 62.109.233.129, 62.109.233.177, 62.109.233.168, and 170.72.63.182 etc.), and other details such as *mediaType* (audio/video), *bytesReceived*, *bytesSent*, *bitrateReceived*, *bitrateSent*, *packetsReceived*, *packetsSent*, *transportType* (protocol used for transmission), *packetsLostReceived*, and *localPort* were also disclosed from the database (Fig. 32–33). It is pertinent to note that the IP addresses of parties

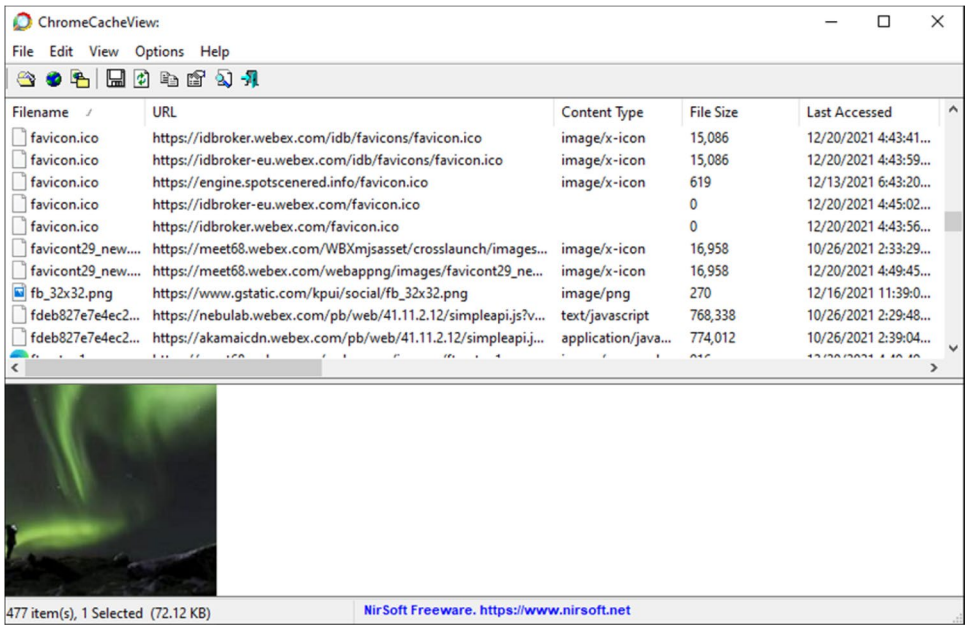

Fig. 34 WebEx cache via ChromeCacheView

the client had communicated with were not logged in the levelDB log stores.

5.3 Bookmarks, history, cookies, and cache

The bookmark of Cisco WebEx web application saved by the user on Google Chrome was extracted from *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Bookmarks*. This gave us the GUID of the bookmark along with the timestamp of when it was added into the Google Chrome browser.

The history SQLite database (*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\History*) provided detailed information regarding (1) keyword search terms entered into Google Chrome (e.g., "cisco webex web app"), (2) history of the visited URLs (along with visit counts, timestamp of last visit, and durations of visits), and (3) downloads from the web application, if any.

The cookies related to the Cisco WebEx web application were also found in the SQLite databases *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Cookies*, and *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Extension Cookies*.

WebEx cache on the other hand was collected from the *C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Cache* folder. Figure 34 shows WebEx cache extracted from the cache folder.

6 Android smartphone forensics

6.1 Logical forensic image

Investigation of the logical forensic image of Android smartphone taken using Andriller CE, for evidence pertaining to the WebEx Meetings smartphone application revealed several primary artifacts. The *Shared Storage* folder from logical image contained (1) *whiteboard images* downloaded by the test user during WebEx meetings and (2) pictures shared/annotated by the user. A *Shared Storage report* logged the same, along with their directories/paths, filenames, file sizes, and modified timestamps as shown in Fig. 35.

The email address of the user account logged into the smartphone was revealed in the device specifications *REPORT* (Fig. 36). This test email address was linked to the WebEx Meetings account for attribution using an artifact extracted from the *data*\ directory as further explained below.

The SQLite database *frosting.db* in the *data\apps\com.android.vending\db* folder saved the *apk* path of WebEx Meetings along with last updated timestamp as shown in Fig. 37.

Similarly, several other SQLite databases in the same *data\apps\com.android.vending\db* folder, including *install_queue.db*, *install_source.db*, *localappstate.db*, *suggestions.db*, *verify_apps.db*, and *xternal_referrer_status.db*, provided some digital evidence of application's usage. Attribution of the WebEx Meetings account with an email address can be

Shared Storage

Total: 16

Index	Directory	Filename	Size	Modified
1	shared/0/.applogs	.log.ai.a2a3987523d6dad791eea43ee49065e6	32	2022-03-02 12:04:44 UTC
2	shared/0/.applogs	.log.ai.a2a3987523d6dad791eea43ee49065e6.h	32	2022-03-02 12:04:44 UTC
3	shared/0/.applogs	.log.si	32	2022-03-02 12:04:44 UTC
4	shared/0/.applogs	.log.si.h	32	2022-03-02 12:04:44 UTC
5	shared/0/DCIM/.thumbnails	1646225884175.jpg	2.8KB	2022-03-02 12:58:04 UTC
6	shared/0/DCIM/.thumbnails	1646225884114.jpg	8.0KB	2022-03-02 12:58:04 UTC
7	shared/0/DCIM/.thumbnails	1646225877345.jpg	2.8KB	2022-03-02 12:57:57 UTC
8	shared/0/DCIM/.thumbnails	1646225877267.jpg	8.0KB	2022-03-02 12:57:57 UTC
9	shared/0/DCIM/.thumbnails	1646225306723.jpg	4.2KB	2022-03-02 12:48:26 UTC
10	shared/0/DCIM/.thumbnails	1646225306645.jpg	18.6KB	2022-03-02 12:48:26 UTC
11	shared/0/DCIM/.thumbnails	.thumbdata3--1967290299	412.5KB	2022-03-02 12:58:04 UTC
12	shared/0/DCIM/Camera	20220227_195811.jpg	2.0MB	2022-02-27 14:58:11 UTC
13	shared/0/Pictures	1646225883927.jpg	39.4KB	2022-03-02 12:58:04 UTC
14	shared/0/Pictures	1646225877052.jpg	39.4KB	2022-03-02 12:57:57 UTC
15	shared/0/Pictures	1646225306298.jpg	91.3KB	2022-03-02 12:48:26 UTC
16	shared/0/Samsung/Music	Over the Horizon.mp3	3.2MB	2022-02-27 11:36:54 UTC

Fig. 35 Shared storage report listing downloaded whiteboard documents and shared pictures via Andriller CE

Type	Data
Serial	
Status	device
Ro.Product.Manufacturer	samsung
Ro.Product.Model	SM-G532F
Ro.Build.Version.Release	6.0.1
Local_Time	2022-03-03 05:57:35 EST
Device_Time	2022-03-03 15:57:38 PKT
Accounts	• com.google: zainab@gmail.com
Application	Shared Storage (16)

Fig. 36 Email address and device specifications of Android

done using *library.db* in the same *\data\apps\com.android.vending\db* folder, which lists the email address against WebEx Meetings, and the certificate hash of the application as shown in Fig. 38.

The *\data\apps\com.google.android.googlequicksearchbox\r\app_webview* folder contains *Cookies.db* that stores web cookies pertaining to WebEx meetings. Other SQLite databases indicating traces of installation/usage are *auto_update.db*, and *data_usage.db* in the *\data\apps\com.android.vending\db* folder.

6.2 ADB backup

Further forensic investigation of the ADB backup of Android smartphone revealed the presence of additional interesting

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: frosting — Filter in any column

	pk	frosting_id	last_updated	apk_path
	Filter	Filter	Filter	Filter
219	com.cisco.webex.meetings	2	1646216597671	/data/app/com.cisco.webex.meetings-1/base.apk
220	com.android.shell	0	1524182089000	/system/priv-app/Shell/Shell.apk

Fig. 37 Frosting database acquired via Andriller CE

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: ownership — Filter in any column

	account	ɔrary_	acken	doc_id	doc_type	er_ty	ocument_has	lid_ur	app_certificate_hash
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
133	zainab@gmail.com	3	3	com.cisco.webex.meetings	1	1	550900568...	NULL	30oIrBf

Fig. 38 Library.db acquired via Andriller CE

Fig. 39 calendar.db acquired via ADB backup

Re...	_sync_id	dirty	mutat	lastSy	calen	title	eventLocation	description
	_60q6cch	0		0	2	Zainab Khalid's meeti	https://meet68.webex.com/meet68,	JOIN WEBEX MEI
	_6tj68p9r	0		0	2	TEST2	https://meet68.webex.com/meet68,	JOIN WEBEX MEI
	_c8pj2e9r	0		0	2	TEST	https://meet68.webex.com/meet68,	JOIN WEBEX MEI

first_download_r	referr	account	title	flags	contir	last_notified_	last_update_time	accou	auto_	exterr	persis	permi
1646216453572		zainab [redacted] @gmail.c	Webex Meetings	0		242020285	0			0	1	1

Fig. 40 localappstate.db acquired via ADB backup

forensic artifacts. These included information, and error event logs related to WebEx Meetings. Cookies related to the application (with values and expiration dates) were also extracted from the ADB backup.

An SQLite database, *calendar.db*, containing Google Calendar events that listed a record of past WebEx meetings was extracted, which provided details related to meeting titles, meeting locations, meeting descriptions, timestamps of start, and attend times of meetings, event time zones, and the organizer's email address (Fig. 39).

We were also able to extract another SQLite database, *localappstate.db*, with records of installed applications, listing WebEx Meetings with download timestamp, associated email address, last notification and last update timestamps (Fig. 40).

Other extracted artifacts included profile picture of associated Google account, whiteboard downloads from the application, and shared/annotated images (with thumbnails).

Putting things together, we illustrate in Table 3 the forensic evidence extracted from the three versions (desktop, web, smartphone) of the Cisco WebEx applications.

7 Case studies

To gain better insights on how individual artifacts, when corroborated, can play a vital role in a forensic investigation, we present in the sequel two case studies based on hypothetical scenarios.

7.1 Desktop client

A legal firm is investigating a case of a fraudulent employee who was recently hired as a junior associate and is suspected by the firm's senior partners to have faked a law degree and practicing law without having passed the bar exam. The suspicion started when the firm rang up the listed college on his resume for a routine reference check and found no records of the said student. As a result, the

Fig. 41 Message received by X

```
messageId="4cee96d0-a0ed-11eb-89c1-1bf3ae40235b"
role="row"  class="messageContainer  "
aria-describedby="info4cee96d0-a0ed-11eb-89c1-1bf3ae40235b">
<p class="messageHeader"><span class='actor'
data-object-id='5d3b137a-e240-4cc1-9d25-1e0dd4bb5dac'>
<spark-contact data-object-type='USER'
data-object-id='5d3b137a-e240-4cc1-9d25-1e0dd4bb5dac'>Mike Ross</spark-contact>
<span class='externalDomain'>@yahoo.com</span></span>
<span class="messageTime">1:57 PM</span>
 </p><div class="sparkMessage  "
messageId="4cee96d0-a0ed-11eb-89c1-1bf3ae40235b">
<div id="info4cee96d0-a0ed-11eb-89c1-1bf3ae40235b"
aria-hidden="false" class="visualhidden"> Has attachments.
</div><div class="msgContainer"><div class="sparkShares ">
</div><div>I have the degree ready for you. I'll come by
your apartment at midnight. Keep my payment ready.</div></div>
<div class="sparkShare previewImage loading sparkDownloadStateNotStarted"
messageId="4cee96d0-a0ed-11eb-89c1-1bf3ae40235b"
contentIndex="0"
data-conversation-id="eac61ec0-9781-11eb-837a-d761b877e81e"hbin
```

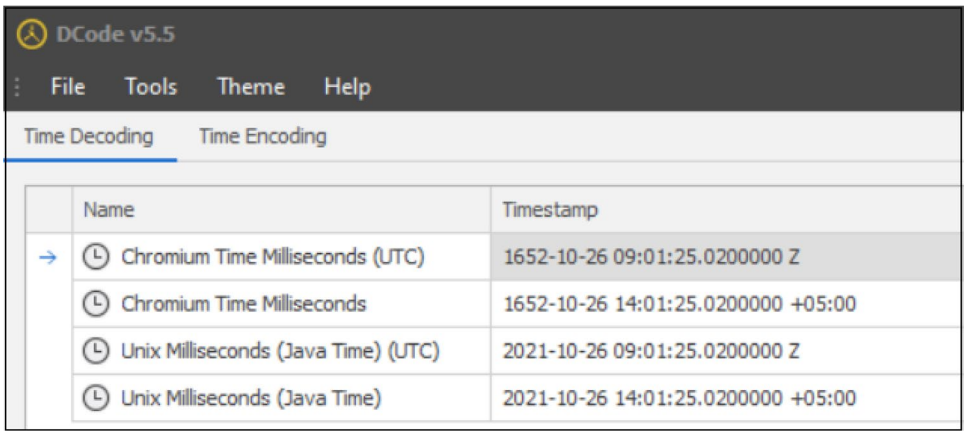

Fig. 42 Decoded timestamp of Y's meeting with XYZ Co.

Fig. 43 Message sent by Y

firm decided to further investigate the case. While the employee, thereafter, referred to as *X*, was working on his laptop PC, he was called in by the HR director for interrogation, and his computer was confiscated for forensic examination. Since the computer was on, the forensic analysts immediately captured a memory dump from *X*'s computer, and analyzed it for clues. A pslist/*pstree* Volatility search revealed the running applications, among which Cisco WebEx was listed. Further investigation (a string search against < *messagesContainer* > tag) revealed interesting artifacts. Figure 41 shows details of a suspicious message that *X* received from an account associated with 'Mike Ross'. In addition to the acquired message contents, it was also evident that the account was one of *X*'s contacts on Cisco WebEx. The time the message was received was also recorded (1:41 PM) along with the email domain of the sender (@yahoo.com).

A string search for emails revealed the complete email of the sender, i.e., mikeross@yahoo.com. An email histogram obtained using Bulk Extractor indicated that this account was one of the most frequently contacted accounts that *X* communicated with recently. Furthermore, the text messages exchanged between *X* and Mike were extracted along with the corresponding timestamps. It revealed that *X* was to receive his fake degree at midnight in exchange for a certain amount of money.

Performing memory dump capture revealed to a be an invaluable approach for extracting forensic evidence. However, because of memory volatility, such an approach becomes impractical if the user shuts down the device. In this case, disk space forensics becomes a viable alternative.

7.2 Web application

A long-term senior employee, *Y*, of a food manufacturing company, *ABC*, is in a confidentiality agreement with the company in regard to a trade secret that the company developed decades ago. *Y* received a multimillion dollar offer by a rival company to disclose the trade secret. A virtual meeting related to this business was in

order. Since *Y* did not want to use his company licensed Cisco WebEx account, which was logged into his WebEx desktop client application, he used his personal WebEx account, which he created using his personal Gmail. *Y*, in order to avoid any trace of this communication on his laptop PC, decided to use the Cisco WebEx web application instead. The subject meeting was conducted, and the deal was finalized.

A few weeks in, company *ABC* asked for a forensic investigation of employees' workstations. *Y*'s laptop hard drive were imaged using FTK Imager and investigated for any relevant clues. Although no trace of contact with the rival company was found in the Cisco WebEx data directory, i.e., the CiscoSpark (*AppData\Local\CiscoSpark*) and WebEx (*AppData\Local\WebEx*) folders which were storing information of his company licensed WebEx account, the forensic analyst checked the Google Chrome data directory as a last resort. As it turned out, an indexedDB-levelDB corresponding to a personal Cisco WebEx account was found which not only proved the application's usage via the Chrome web browser, but also provided details related to the conducted meeting with the rival *XYZ* company in the levelDB in *logstore12* object store. The timestamps of logstore12, decoded using DCode (Fig. 42), suggested that the meeting was held on October 26, 2021, around 2 PM.

The text chain of *Y*, and company *XYZ* was recovered in chronological order (following the *chatMgr* service manager to streamline the search using the *find* utility on the text editor), and also using timestamps of each text message. A text message sent by Y to the XYZ Co. representative in the meeting, as shown in Fig. 43, clearly proves that *Y* was guilty of violating the confidentiality agreement because he communicated the trade secret with the company *XYZ*.

Evidently, even when the suspect in subject case employed an anti-forensic route by not using the Cisco WebEx desktop client application, the forensic analyst was able to counter by performing forensic analysis of the web application. Cisco WebEx web application leaves very detailed artifacts pertaining to meetings conducted, and can be presented as DE in cases involving the application.

8 Conclusion and future work

Forensic analysis of videoconferencing applications has attracted considerable attention among researchers, and forensic investigators alike, owing to the accelerated usage of these applications following the COVID-19 pandemic and the emergence of highly publicized cases of Zoom-bombing incidents. The established artifacts pertaining to each videoconferencing application provide a reference for forensic analysts, which can potentially serve as digital evidence in the court of law.

This paper presented a forensic analysis of Cisco WebEx desktop and web applications with respect to memory, disk space, and network on Windows 10. The objective of the research was to analyze what forensic artifacts can be extracted pertaining to the application. Memory forensics revealed detailed information with respect to communications that took place between the user and other parties, such as chat messages, exchanged text and media files, meeting passwords, and contacts. User credentials were not found in the memory in plaintext, but the same was successfully extracted from disk space Windows Registry. Other interesting artifacts extracted from the disk space included profile pictures, call logs, meeting records, and prefetch files. The extracted network artifacts provided useful insights into the client-server communications such as server domains and IP addresses. Artifacts from all these sources can be linked to reconstruct user activities and to develop a chronological trail for activities that took place.

The Cisco WebEx web application was also observed to be keeping detailed logs of the meetings that occurred. Besides traces of the web application's usage in different files/folders in the Google Chrome data directory, the indexedDB-levelDB proved to be a prime source of all the interesting details regarding meetings.

Forensics of logical images, and ADB backups of Android OS for WebEx Meetings smartphone application artifacts, revealed user account information, shared media, cookies, event logs, and meeting records etc.

Results presented herein are valid at the time the research was conducted, and they might become outdated with the emergence of potential future WebEx software updates.

This work can be further explored in many directions. Since this research focuses on Windows 10 OS, it would be interesting to perform a similar in-depth analysis of Cisco WebEx on the novel Windows 11 OS. A forensic analysis of a physical Android image may also be performed since our study focused on logical images and ADB backups. It would also be needful to conduct a forensic investigation of Cisco WebEx on other OSs such as iOS, macOS, and Linux etc., and additionally, on other smart devices such as iPad. Future work can also focus on extending our forensic analysis to other videoconferencing applications that have not been explored as of yet such as Google Meet, Adobe Connect, and BlueJeans

Appendix

Table 3 Summary of extracted artifacts

Memory artifacts

Artifact	String tag	Details	Command for extraction
User account information	<"UserName">, <Personal Room>	user account display name, email address, personal room number, default WebEx site, video address	strings [memory dump name] | grep 'Personal Room'
Search keywords	<"query">	Search keywords entered by user	strings [memory dump name] | grep '"query"'
Password	n/a	n/a	n/a
Text files exchanged	<sparkShareInfo>	file name, file size	strings [memory dump name] | grep 'sparkShareInfo'
Media files exchanged	<sparkShareInfo>	image name, image size	strings [memory dump name] | grep 'sparkShareInfo'
URLs exchanged	<a href> <[]> for deleted	URLs	strings [memory dump name] | grep 'a href'
Chat messages	<sparkMessageGroup> for deleted	Chat message body, time of message, message ID	strings [memory dump name] | grep 'sparkMessageGroup'
Scheduled meetings	<WebExMeetingData>	Meeting password, meeting key, meeting name, scheduler's username and default WebEx site, in-meeting messages	strings [memory dump name] | grep 'WebExMeetingData'
Contacts	</spark-contact>	contacts of WebEx user	strings [memory dump name] | grep '/spark-contact'

Disk-space artifacts

Artifact	Directory path
Profile pictures of logged in accounts	*\AppData\Roaming\webex\Avatar*
Last joined location	*\AppData\Roaming\webex\Avatar\latestjoinedlocation.ini*
Meeting logs	*\AppData\Roaming\webex\Avatar\QSXMLFile.dat*
Site startup	*\AppData\Local\WebEx\chrome.json*
Site information	*\AppData\Local\WebEx\siteinfo*
Cache	*\AppData\Local\WebEx\wbxcache*
CiscoMeetings database	*\AppData\Local\WebEx\CiscoMeetings.db*
Call logs	*\AppData\Local\CiscoSpark\media\calls*
Encrypted database	*\AppData\Local\CiscoSpark\[User ID]*\spark_persistent_store.db
Encrypted database	*\AppData\Local\CiscoSpark\[User ID]*\ spark_roaming_store.db
Spark shared store database	*\AppData\Local\CiscoSpark\spark_shared_store.db*
Login state of account	*\AppData\Local\CiscoSpark\lifecycle.dat*
Prefetch files	*C:\Windows\Prefetch\CISCOCOLLABHOST.EXE-49749B78.pf* *C:\Windows\Prefetch\WEBEXHOST.EXE-7D2F62CC.pf*

Registry Key–Value explanation
HKCU\SOFTWARE\Microsoft\Installer\Products\[Product ID]\SourceList–Package name of the application (WebEx.msi), and path to installer application.
HKCU\SOFTWARE\Microsoft\Installer\Products\[Product ID]–Information regarding the WebEx application i.e., Version, Product name, Icon, Package code etc.
HKCU\SOFTWARE\RegisteredApplications–List of registered application in the client desktop (WebEx inclusive).
HKCU\SOFTWARE\Cisco\Spark–Paths to client application folders.
HKCU\SOFTWARE\Cisco Spark Native–Application settings (e.g., block WebEx desktop app after meeting ends).
HKCU\SOFTWARE\CiscoSystems,Inc.\WebexTeams\Capabilities\URLAssociations–URL associations of the application.
HKCU\SOFTWARE\WebEx\Config–Configuration information for WebEx (show panelist name, show attendee name, show timestamp, current theme type etc.).
HKCU\SOFTWARE\WebEx\FeaturePayloads–Default WebEx sites and their configuration information. For example, the *"EnableAES256GCM": true* setting enables AES encryption. 141 such configuration settings are stored in the key.
HKCU\SOFTWARE\WebEx\ProdTools\Logon\[Default WebEx Site]–Credential and path to folder with saved profile avatar.
HKCU\SOFTWARE\WebEx\UCF\Components\meetings–Support for Audio, Video, Recording, and WebEx Web etc. and other related settings such as audio/video file type extensions that are supported by the application.
HKCU\SOFTWARE\WebEx\Framework\RecordPlayback–Path to folder with saved meeting recordings.
HKCU\SOFTWARE\WebEx\Framework\RecentlyCall–Meeting settings from the last meeting.

Cisco WebEx web application artifacts

Artifact	Directory path
IndexedDB-LevelDB	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\IndexedDB\ https_[default WebEx site]_0.indexeddb.leveldb*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\JumpListIconsMostVisited*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\JumpListIconsRecentClosed*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Top Sites*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Favicons*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Network Action Predictor*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\QuotaManager*
Trace of usage	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Shortcuts*
Trace of usage/email	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Sessions*
Profile picture	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Accounts\Avatar Images*
Profile picture	*C:\Users\[usermane]\AppData\Local\Google\Chrome\User Data\Default\Google Profile Picture*
Bookmark	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Bookmarks*
History	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\History*
Cookies	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Cookies*
Cache	*C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\Cache*

Table 3 (continued)

		Artifact	Directory path
WebEx Meetings (Android) application artifacts	Logical acquisition artifacts	Whiteboard docs/shared images	*\Shared Storage*
		Email address	*\REPORT*
		Trace of installation	*\data\apps\com.android.vending\db\frosting.db*
		Trace of usage	*\data\apps\com.android.vending\db\install_queue.db*
		Trace of usage	*\data\apps\com.android.vending\db\install_source.db*
		Trace of usage	*\data\apps\com.android.vending\db\localappstate.db*
		Trace of usage	*\data\apps\com.android.vending\db\suggestions.db*
		Trace of usage	*\data\apps\com.android.vending\db\verify_apps.db*
		Trace of usage	*\data\apps\com.android.vending\db\xternal_referrer_status.db*
		Trace of usage	*\data\apps\com.android.vending\db\library.db*
		Trace of usage	*\data\apps\com.android.vending\db\auto_update.db*
		Trace of usage	*\data\apps\com.android.vending\db\data_usage.db*
		Cookies	*\data\apps\com.google.android.googlequicksearchbox\r\app_webview\Cookies.db*
	ADB backup artifacts	Event logs	*\vol_0\shared\0\logs\com.cisco.webex.meetings_info_0.log*
		Cookies	*\vol_0\apps\com.android.chrome\r\app_chrome\Default\History*
		Google Calendar events database	*\vol_0\apps\com.android.providers.calendar\db\calendar.db*
		Installed application information	*\vol_0\apps\com.android.vending\db\localappstate.db*
		Profile picture	*\vol_0\apps\com.android.chrome\r\app_chrome\Default\Google Profile Picture.png*
		Downloads and shared images	*\vol_0\shared\0\Pictures*

Acknowledgements This research is supported with Research Incentive Funds (R21111) and Provost Research Fellowship Award (R20093), Zayed University, United Arab Emirates. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect those of Zayed University or the publisher. This research is an extension of our earlier work available at https://ieeexplore.ieee.org/abstract/document/9614647. The authors contacted Cisco regarding this work.

Declarations

Ethics approval and consent to participate Forensic research needs to consider the potential ethical and legal considerations associated with privacy, intellectual property, responsible disclosure, secondary uses of personal information, terms of use, informed consent, and use of human participants among others [23]. We have taken the necessary steps for addressing or mitigating these common concerns as explained below. The inherent nature of this forensic study is not to expose flaws, hackable security vulnerabilities, or zero-day exploits whose exploitation might lead to incidents but rather to guide forensic investigators unearth readily accessible digital data that some users may want to hide as evidence of legal wrongdoing. Recall that every user interaction with the WebEx application leaves behind some residual forensic evidence which originates from personal data that the app requires to properly operate as per the WebEx privacy terms and conditions [24]. Hence, in our case, ethical and legal issues pertaining to responsible disclosure of vulnerabilities did not apply, as we did not divulge any vulnerability that might jeopardize the availability, integrity, or confidentiality of Cisco WebEx data and system files.

All our forensic experiments were conducted in a controlled laboratory setting by one of the co-authors who, as the owner of the testing device, provided explicit permission and consent for the residual forensic data (including personal identifying information) to be included in this study. Accordingly, privacy concerns associated with the practice of involving human participants in the research were addressed.
The usage of the three WebEx videoconferencing applications, as reported in this study, was conducted in full compliance with Cisco WebEx license terms and conditions. In particular, we did not perform any form of code inspection, decryption, alteration, or reverse engineering on the WebEx software. The forensic artifacts collected from our study belong to the categories of personal data that is required to use the WebEx Service as per Cisco WebEx privacy data sheet [24].

Conflict of interest The authors declare no competing interests.

References

1. Fortune Business Insights. (2021, June). Video conferencing market size, share, trends and growth. www.fortunebusinessinsights.com. Retrieved from https://www.fortunebusinessinsights.com/industry-reports/video-conferencing-market-100293
2. Lorenz, T. (2020, March 20). 'Zoombombing': when video conferences go wrong. The New York Times. Retrieved from https://www.nytimes.com/2020/03/20/style/zoombombing-zoom-trolling.html
3. Hussain, R. (2020, April 12). Experts warn against vulnerabilities of apps, videoconferencing platforms. Arab News. Retrieved from https://www.arabnews.com/node/1657286/saudi-arabia
4. Khalid, Z., Iqbal, F., Kamoun, F., Hussain, M., & Khan, L. A. (2021). Forensic analysis of the Cisco WebEx application. *2021 5th Cyber Security in Networking Conference (CSNet)*, pp. 90-97. https://doi.org/10.1109/CSNet52717.2021.9614647
5. ENISA. (2012, April 2). Procure Secure: a guide to monitoring of security service levels in cloud contracts—ENISA. Europe: European Union Agency for Network and Information Security (ENISA). Retrieved from https://www.enisa.europa.eu/activities/Resilience-and-CIIP/cloud-computing/procure-secure-a-guide-tomonitoring-of-security-service-levels-in-cloud-contracts
6. Brockschmidt, K. (2014, July 15). Programming Windows Store Apps with HTML, CSS, and JavaScript. Microsoft Press Store by Pearson. Retrieved from https://www.microsoftpressstore.com/store/programming-windows-store-apps-with-html-css-and-javascript-9780735695665

7. Yang TY, Dehghantanha A, Choo KR, Muda Z (2016) Windows instant messaging app forensics: Facebook and Skype as case studies. *PLOS ONE* 11(3):e0150300. https://doi.org/10.1371/journal.pone.0150300
8. Nicoletti M, Bernaschi M (2019) Forensic analysis of Microsoft Skype for business. *Digital Investigation* 29:159–179. https://doi.org/10.1016/j.diin.2019.03.012
9. Mahr A, Cichon M, Mateo S, Grajeda C, Baggili I (2021) Zooming into the pandemic! A forensic analysis of the Zoom application. *Forensic Science International. Digital Investigation* 36. https://doi.org/10.1016/j.fsidi.2021.301107
10. Nicoletti, M., & Bernaschi, M. (2021, September 08). Forensics for Microsoft Teams. Retrieved December 03, 2021, from https://arxiv.org/pdf/2109.06097
11. Herschel, R. B. (2021). A forensic analysis of Microsoft Teams. Purdue University Graduate School. Thesis. https://doi.org/10.25394/PGS.15091329.v1
12. Bilz, A. (2021, September 9). A forensic gold mine III: forensic analysis of the Microsoft Teams desktop client. https://www.alexbilz.com/post/2021-09-09-forensic-artifacts-microsoft-teams/
13. Anglano C (2014) Forensic analysis of WhatsApp Messenger on android smartphones. *Digital Investigation* 11(3):201–213. https://doi.org/10.1016/j.diin.2014.04.003
14. Sgaras C, Kechadi M-T, Le-Khac N-A (2015) Forensics acquisition and analysis of instant messaging and VoIP applications. *Computational Forensics*:188–199. https://doi.org/10.1007/978-3-319-20125-2_16
15. Tandel H, Rughani PH (2018) Forensic analysis of Asterisk-FreePBX based VoIP server. *International Journal of Emerging Research in Management & Technology* 6:166–171. https://doi.org/10.23956/ijermt.v6i8.133
16. Dargahi, T., Dehghantanha, A., & Conti, M. (2017). Forensics analysis of android mobile VoIP apps. *Contemporary Digital Forensic Investigations of Cloud and Mobile Applications*,pp. 7–20. https://doi.org/10.1016/b978-0-12-805303-4.00002-2
17. Sha MM, T M, Abd El-atty SM (2016) VoIP forensic analyzer. *International Journal of Advanced Computer Science and Applications* 7. https://doi.org/10.14569/ijacsa.2016.070116
18. Simon, M., & Slay, J. (2010). Recovery of Skype application activity data from physical memory. *2010 International Conference on Availability, Reliability and Security*, pp. 283-288. https://doi.org/10.1109/ARES.2010.73
19. Case A, Richard GG (2017) Memory forensics: the path forward. *Digital Investigation* 20:23–33. https://doi.org/10.1016/j.diin.2016.12.004
20. Khalid, Z. (2021, December 27). ciscowebexmemory. GitHub. https://github.com/znbkhld/ciscowebexmemory
21. Messier R (2017) *Network Forensics*. Wiley, Indianapolis
22. Caithness, A. (2020, September 23). Hang on! That's not SQLite! Chrome, Electron and LevelDB. cclsolutionsgroup.com. Retrieved December 19, 2021, from https://www.cclsolutionsgroup.com//post/hang-on-thats-not-sqlite-chrome-electron-and-leveldb
23. Glisson, W. B., Storer, T., Blyth, A., Grispos, G & Campbell, M. (2016). In-the-wild residual data research and privacy. *Journal of Digital Forensics, Security and Law*, 11 (1), pp. 77-97. https://doi.org/10.15394/jdfsl.2016.1371
24. Cisco Webex Meetings: Privacy Data Sheet. https://trustportal.cisco.com/c/dam/r/ctp/docs/privacydatasheet/collaboration/cisco-webex-meetings-privacy-data-sheet.pdf