DIGITAL FORENSIC ANALYSIS OF ARTIFACTS EXTRACTED FROM VARIOUS REMOTE ACCESS APPLICATIONS

Thesis Submitted for the Award of the Degree of

DOCTOR OF PHILOSOPHY

in

Forensic Sciences

By

Nishchal Soni

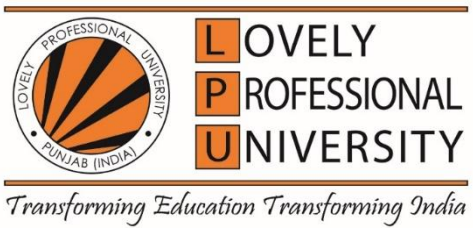
Registration Number: 12207908

Supervised By

Dr. Manpreet Kaur (28193)

Department of Forensic Science (Assistant Professor)

Lovely Professional University, Punjab

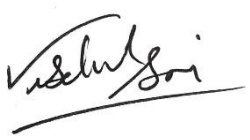


LOVELY PROFESSIONAL UNIVERSITY, PUNJAB

2026

DECLARATION

I, hereby declared that the presented work in the thesis entitled "**Digital Forensic Analysis of Artifacts Extracted from Various Remote Access Applications**" in fulfilment of degree of **Doctor of Philosophy (Ph. D.)** is outcome of research work carried out by me under the supervision of Dr. Manpreet Kaur, working as Assistant Professor, in the Department of Forensic Science, School of Bioengineering and Biosciences of Lovely Professional University, Punjab, India. In keeping with general practice of reporting scientific observations, due acknowledgements have been made whenever work described here has been based on findings of other investigator. This work has not been submitted in part or full to any other University or Institute for the award of any degree.



(Signature of Scholar)

Name of the scholar: Nishchal Soni

Registration No.: 12207908

Department/school: Forensic Science

Lovely Professional University,

Punjab, India

CERTIFICATE

This is to certify that the work reported in the Ph. D. thesis entitled **"Digital Forensic Analysis of Artifacts Extracted from Various Remote Access Applications"** submitted in fulfillment of the requirement for the award of degree of **Doctor of Philosophy (Ph.D.)** in the School of Bioengineering and Biosciences, is a research work carried out by Nishchal Soni, 12207908, is bonafide record of his original work carried out under my supervision and that no part of thesis has been submitted for any other degree, diploma or equivalent course.

24-9-25

(Signature of Supervisor)

Name of supervisor: Dr. Manpreet Kaur

Designation: Assistant Professor

Department/school: Department of Forensic Science, School of Bioengineering and Biosciences

University: Lovely Professional University

ABSTRACT

Remote Access Applications (RAAs), originally designed to facilitate remote administration, collaboration, and support, have emerged as one of the most frequently misused categories of tools by malicious actors. Their dual-use (legitimate and illicit) nature allows them to function as a legitimate tool in enterprise and professional contexts, while at the same time enabling criminals to bypass physical barriers and remotely obtain unauthorized access to devices and networks to commit illicit activities. RAAs have become a popular tool for remote work in personal and professional ecosystems after the COVID-19 pandemic. It also provided grounds for attackers who employ social engineering and identity deception techniques to infiltrate target systems. Despite this sharp rise in cybercrimes involving RAAs, the forensic community lacked a structured, comparative, and systematic body of knowledge that could guide the interpretation of the artifacts generated by these Remote Access Applications. The present study addresses this critical gap by conducting a systematic forensic analysis of five widely used RAAs, such as TeamViewer, AnyDesk, Zoho Assist, RemotePC, and LogMeIn (GoToMyPC), across four operating systems (Windows, macOS, Android, and iOS), focusing on installation and uninstallation states as well as on free and licensed versions of these tools.

The present research examined one hundred samples in total, comprising 80 structured samples across applications, platforms, and license types, along with 20 blind samples to validate the results of the present study. Each application was installed and used under controlled yet realistic scenarios that included but were not limited to remote sessions, chats, file transfers, clipboard interactions, and session recordings. Acquisition of data was performed using advanced forensic tools such as Magnet AXIOM, MD-NEXT, MD-RED, PECmd.exe, and DB Browser for SQLite, with both automated extraction and manual validation to ensure completeness and accuracy. Post-uninstallation acquisitions were also performed to evaluate the persistence of artifacts.

The findings presented variability in artifact generation and persistence across platforms. Windows systems were consistently the most artifact-rich, yielding

logs such as TeamViewer15_Logfile.log and Connections.txt, configuration identifiers like rolloutfile.tv13, prefetch entries, and event logs. These artifacts provided comprehensive data, including session metadata, IP addresses, device specifications, usernames, licensed versions, and screen recordings of sessions. Importantly, many of these traces persisted after uninstallation, enabling reconstruction of activity even when users attempted to conceal usage by removing the software. MacOS produced similarly valuable artifacts, with files such as Info.plist, global.conf, and client.conf retaining cryptographic keys, identifiers, and detailed session logs. Like Windows, macOS also preserves artifacts post-uninstallation, offering investigators opportunities for timeline reconstruction.

Mobile platforms were found to be more restrictive in artifact retention. On Android, valuable data such as device IDs, IP addresses, session logs, and chat metadata were found in files like TVLog.html and client.conf during installation states. Screenshots of remote activity further enhanced the evidentiary potential. However, only limited remnants such as installation traces in library.db persisted, leaving little to reconstruct user activity. iOS presented an even greater challenge, as during installation on iOS, artifacts such as keychain entries, connection records, and application state logs were available. However, after uninstallation, nearly all of this evidence was removed, with only installation logs (mobile_installation.log.0) left behind

A major contribution of this study lies in the comparative analysis of free versus licensed versions of RAAs. Licensed versions consistently generated richer forensic footprints, including advanced session logs, file transfer histories, screenshots, session recordings, and chat records. These additional traces significantly strengthened the ability to reconstruct timelines and corroborate evidence. In contrast, the free versions left behind only minimal identifiers or sparse log records. This asymmetry is particularly relevant because free versions are more commonly used in cybercrime, thus presenting investigators with fewer opportunities for reconstruction. The study demonstrates that licensing

models directly influence the forensic landscape, an insight that has been largely absent from existing literature.

The persistence of artifacts across installation and uninstallation states also adds to the contribution of this research. On desktop systems, investigators can expect a substantial degree of persistence, with logs, identifiers, and system metadata remaining accessible even after removal of the application. On mobile platforms, persistence of artifacts was limited. This dichotomy between desktop and mobile ecosystems illustrates how operating system design fundamentally shapes evidentiary availability.

The present study also highlights how software licensing models and operating system architecture collectively make changes to the forensic footprint of applications. It underscores the dual nature of RAAs as both tools of utility and abuse, demonstrating that even in malicious contexts and application of anti-forensic techniques such as uninstallation of applications, they generate valuable forensic footprints that can be used for attribution, reconstruction, and legal proceedings. By establishing a structured framework validated through blind testing, this study bridges the gap between theoretical research and applied forensic practice, ensuring that its contributions are not limited to controlled settings but are transferable to real investigative scenarios.

This research systematically demonstrates that Remote Access Applications leave behind a diverse range of forensically valuable artifacts. While desktop environments provide abundant and persistent evidence, mobile platforms present greater challenges due to limited artifact retention. Licensed versions produce significantly richer forensic traces compared to free versions, shaping the strategies investigators must adopt. The validation through blind samples confirms the reliability of the identified artifacts. The present study not only fills the critical gap in forensic literature but also equips digital forensic practitioners with actionable knowledge. This is also advancing the discipline and laying the groundwork for future studies into other categories of dual-use software and emerging platforms.

Collectively, the present study makes significant contributions at both academic and practical levels. Academically, it represents the first systematic, cross-platform and cross-license forensic investigation of multiple Remote Access Applications across desktop and mobile ecosystems, thereby expanding the existing body of forensic literature where structured guidance has been limited. Practically, the study establishes a validated forensic framework that assists investigators in identifying the location, nature, and evidentiary value of artifacts under different operating systems, licensing conditions, and post-uninstallation states. The documentation of artifact persistence, licensing-based variations, and platform-specific behaviors provides investigators with realistic expectations and operational clarity, thereby reducing the risk of misinterpretation or evidentiary oversight in real-world cybercrime investigations. The blind sample validation further strengthens the reliability, robustness, and real-world applicability of the findings, clearly establishing the importance and superiority of the present research.

ACKNOWLEDGEMENT

First and foremost, I bow in gratitude to my late grandfather, **Shri Ram Ji Lal Verma**, who has been no less than a divine presence in my life. His blessings and spiritual presence have guided me at every step and given me the courage to pursue this research journey with dedication and resilience.

I extend my deepest and most sincere thanks to my research supervisor, **Dr. Manpreet Kaur**, for her constant support, invaluable guidance, and remarkable patience. Her trust in my decisions, her ability to handle my thoughts and processes with understanding, and her encouragement at every stage have been instrumental in shaping this work. Her problem-solving skills, perseverance, and motivating presence helped me endure the most challenging phases of my Ph.D. Beyond academic mentoring, I have learned many important personal traits like patience, calmness, and positiveness from her, which will remain with me throughout my life.

I am grateful to **Prof. Neeta Raj Sharma**, Head of the School of Bioengineering and Biosciences, and **Dr. Tejasvi Pandey**, Head of the Department of Forensic Science, along with all respected faculty members of the department at **Lovely Professional University, Punjab**, for their support and encouragement during the course of my research. I would also like to extend my sincere thanks to all the volunteers of this study who assisted me during the blind testing process.

I am thankful to the **University Grants Commission (UGC)** for awarding me a fellowship, which not only supported this research financially but also motivated me to work with greater responsibility.

I am deeply grateful to **Dr. Vishwas Bhardwaj**, Assistant Director, SFSL Jaipur, who first recognized the spark of research within me during my professional tenure. He nurtured this early interest, encouraging and guiding my research efforts with patience and insight. His suggestion of the research gap and thoughtful guidance greatly simplified my task and provided clarity in shaping this study. His belief in my potential and his unwavering support have left an indelible mark on my academic journey.

I also wish to acknowledge **Dr. Prabal Gupta**, whose guidance during challenging research problems, combined with his positive and encouraging demeanor, provided me with the strength and confidence to persevere and move forward.

I owe my deepest gratitude to my family for their unconditional love and encouragement. I especially dedicate this work to my father, **Mr. Manoj Kumar Soni**, who has made countless sacrifices to ensure I could stand on my own feet. My success and my life are a reflection of his unwavering support and dedication. His guidance has been my constant source of inspiration, teaching me to live with courage and to cherish every moment of life. My mother, **Mrs. Madhu Soni**, with her calmness and unwavering support, has always been my pillar of strength. I also want to thank my brother **Mr. Nirmal Soni**, my sister **Ms. Poonam Soni**, and my sister-in-law **Mrs. Poornima Soni**, for always standing by my side and making me feel like I had true friends within my family. Their presence gave me solace during the toughest times.

I am also grateful to my classmates and friends, especially **Mr. Khalid Aziz, Ms. Kanika Chauhan, Ms. Akshita Sankhla, Ms. Sushobhita Singh, Mr. Abhijith A. R., and Mr. Shahid Shah**, whose encouragement, cheerful spirit, and companionship kept me motivated and helped me maintain balance during this journey.

Above all, I would like to express my deepest and most heartfelt thanks to my life partner, **Dr. Chitra Barotia (PT)**. She has been my unwavering source of strength, support, and inspiration throughout this journey. More than a companion, she has shared every challenge, celebrated every achievement, and stood by me with patience, understanding, and unconditional care. Her trust, encouragement, and support have made even the most difficult moments bearable, and her belief in me has inspired me to reach heights I never thought possible. I feel truly blessed to have her by my side, not only during this research but in every step of my life.

To all who contributed to this journey in ways big and small, I extend my sincere gratitude. This thesis is as much yours as it is mine.

Nishchal Soni

Date: 26 Sep. 2025

TABLE OF CONTENTS

DECLARATION I
CERTIFICATE II
ABSTRACT III
ACKNOWLEDGEMENT VII
TABLE OF CONTENTS IX
LIST OF TABLES XIV
LIST OF FIGURES XVII
LIST OF ABBREVIATIONS XXII
CHAPTER 1 1
INTRODUCTION 1
CHAPTER 2 5
LITERATURE REVIEW 5
CHAPTER 3 17
METHODOLOGY 17
3.1 Material 17
3.1.1 Remote Access Applications 17
3.1.1.1 TeamViewer 18
3.1.1.2 LogMeIn (GoToMyPC) 19
3.1.1.3 RemotePC 19
3.1.1.4 Zoho Assist 19
3.1.1.5. AnyDesk 20
3.1.2 Operating Systems and Devices 20
3.2 Methodology 21
3.2.1. Setup of Test Environments 21
3.2.2 Configuration of Virtual Machines 21
3.2.3 Configuration of Physical Devices 21
3.2.4 Network 22
3.2.5 Sample Size and Grouping 22
3.2.6 Tools and Software Used 24

3.2.7 Data Collection 24
3.2.7.1 Installation and Simulation of Usage 24
3.2.7.2 Post-Installation Data Collection 25
3.2.7.3 Post-Uninstallation Data Collection 25
3.2.7.4 Blind Sample Preparation and Data Collection 26
3.2.8 Classification of Digital Artifacts 26
3.2.9 Artifact Analysis 28
CHAPTER 4 30
RESULTS AND DISCUSSION 30
4.1 Results 30
4.1.1 TeamViewer 30
4.1.1.1 Android 31
4.1.1.1.1 Installation State 31
4.1.1.1.2 Uninstallation State 34
4.1.1.1.3 Interpretation and Forensic Significance 34
4.1.1.2 Windows 38
4.1.1.2.1 Installation 38
4.1.1.2.2 Uninstallation State 40
4.1.1.2.3 Interpretation and Forensic Significance 40
4.1.1.3 iOS 43
4.1.1.3.1 Installation 43
4.1.1.3.2 Uninstallation State 46
4.1.1.3.3 Interpretation and Forensic Significance 47
4.1.1.4 macOS 49
4.1.1.4.1 Installation 49
4.1.1.4.2 Uninstallation State 50
4.1.1.4.3 Interpretation and Forensic Significance 51
4.1.2 LogMeIn (GoToMyPC) 53
4.1.2.1 Android 53
4.1.2.1.1 Installation State 53
4.1.2.1.2 Uninstallation State 54

4.1.2.1.3 Interpretation and Forensic Significance 54
4.1.2.2 Window 56
4.1.2.2.1 Installation State 56
4.1.2.2.2 Uninstallation State 57
4.1.2.2.2 Interpretation and Forensic Significance 57
4.1.2.3 iOS 60
4.1.2.3.1 Installation State 60
4.1.2.3.2 Uninstallation State 61
4.1.2.3.3 Interpretation and Forensic Significance 61
4.1.2.4 macOS 63
4.1.2.4.1 Installation State 63
4.1.2.4.2 Uninstallation State 65
4.1.2.4.3 Interpretation and Forensic Significance 65
4.1.3 RemotePC 67
4.1.3.1 Android 67
4.1.3.1.1 Installation State 67
4.1.3.1.2 Uninstallation State 71
4.1.3.1.3 Interpretation and Forensic Significance 72
4.1.3.2 Window 75
4.1.3.2.1 Installation State 75
4.1.3.2.2 Uninstallation State 77
4.1.3.2.3 Interpretation and Forensic Significance 77
4.1.3.3 iOS 81
4.1.3.3.1 Installation State 81
4.1.3.3.2 Uninstallation State 82
4.1.3.3.3 Interpretation and Forensic Significance 83
4.1.3.4 macOS 85
4.1.3.4.1 Installation State 85
4.1.3.4.2 Uninstallation State 86
4.1.3.4.3 Interpretation and Forensic Significance 86
4.1.4 Zoho Assist 89

4.1.4.1 Android 89
4.1.4.1.1 Installation State 89
4.1.4.1.2 Uninstallation State 92
4.1.4.1.3 Interpretation and Forensic Significance 92
4.1.4.2 Windows 95
4.1.4.2.1 Installation State 95
4.1.4.2.2 Uninstallation State 97
4.1.4.2.3 Interpretation and Forensic Significance 97
4.1.4.3 iOS 100
4.1.4.3.1 Installation State 100
4.1.4.3.2 Uninstallation State 102
4.1.4.3.3 Interpretation and Forensic Significance 102
4.1.4.4 macOS 104
4.1.4.4.1 Installation State 104
4.1.4.4.2 Uninstallation State 106
4.1.4.4.3 Interpretation and Forensic Significance 106
4.1.5 Anydesk 109
4.1.5.1 Android 109
4.1.5.1.1 Installation State 109
4.1.5.1.2 Uninstallation State 111
4.1.5.1.3 Interpretation and Forensic Significance 111
4.1.5.2 Windows 114
4.1.5.2.1 Installation 114
4.1.5.2.2 Uninstallation State 116
4.1.5.2.3 Interpretation and Forensic Significance 116
4.1.5.3 iOS 119
4.1.5.3.1 Installation 119
4.1.5.3.2 Uninstallation State 120
4.1.5.3.3 Interpretation and Forensic Significance 121
4.1.5.4 macOS 123
4.1.5.4.1 Installation 123

4.1.5.4.2 Uninstallation State 124
4.1.5.4.3 Interpretation and Forensic Significance 125
4.2 Discussion 127
CHAPTER 5 135
SUMMARY AND CONCLUSION 135
5.1 Summary 135
5.2 Conclusion 138
5.3 Limitations and Future Recommendations 140
BIBLIOGRAPHY 141
List of Publications 147
List of Conferences 147

LIST OF TABLES

Table 3a: TeamViewer Variants Employed for Forensic Analysis on Different Platforms…… 17

Table 4a: Represents the digital artifacts recovered from the installation state of TeamViewer on an Android device, including corresponding file names, storage locations, Content, and license-specific availability …… 28

Table 4b: Represents the digital artifacts recovered from the installation state of TeamViewer on Windows, including corresponding file names, storage locations, Content, and license-specific availability ……... 35

Table 4c: Represents the digital artifacts recovered from the installation state of TeamViewer on iOS, including corresponding file names, storage locations, Content, and license-specific availability ……... 40

Table 4d: Represents the digital artifacts recovered from the installation state of TeamViewer on macOS, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 46

Table 4e: Represents the digital artifacts recovered from the installation state of GoToMyPC on Android, including corresponding file names, storage locations, Content, and license-specific availability ……..….. 50

Table 4f: Represents the digital artifacts recovered from the installation state of GoToMyPC on Windows, including corresponding file names, storage locations, Content, and license-specific availability …….…….. 53

Table 4g: Represents the digital artifacts recovered from the installation state of GoToMyPC on iOS, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 57

Table 4h: Represents the digital artifacts recovered from the installation state of GoToMyPC on macOS, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 60

Table 4i: Represents the digital artifacts recovered from the installation state of RemotePC on Android, including corresponding file names, storage locations, Content, and license-specific availability …….…….. 64

Table 4j: Represents the digital artifacts recovered from the installation state of RemotePC on Windows, including corresponding file names, storage locations, Content, and license-specific availability …….. 72

Table 4k: Represents the digital artifacts recovered from the installation state of RemotePC on iOS, including corresponding file names, storage locations, Content, and license-specific availability ……..…… 78

Table 4l: Represents the digital artifacts recovered from the installation state of RemotePC on macOS, including corresponding file names, storage locations, Content, and license-specific availability …….. 82

Table 4m: Represents the digital artifacts recovered from the installation state of Zoho Assist on Android, including corresponding file names, storage locations, Content, and license-specific availability …….……. 86

Table 4n: Represents the digital artifacts recovered from the installation state of Zoho Assist on Windows, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 92

Table 4o: Represents the digital artifacts recovered from the installation state of Zoho Assist on iOS, including corresponding file names, storage locations, Content, and license-specific availability …….…….. 97

Table 4p: Represents the digital artifacts recovered from the installation state of Zoho Assist on macOS, including corresponding file names, storage locations, Content, and license-specific availability …….……. 101

Table 4q: Represents the digital artifacts recovered from the installation state of AnyDesk on Android, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 106

Table 4r: Represents the digital artifacts recovered from the installation state of AnyDesk on Windows, including corresponding file names, storage locations, Content, and license-specific availability …….…….. 111

Table 4s: Represents the digital artifacts recovered from the installation state of AnyDesk on iOS, including corresponding file names, storage locations, Content, and license-specific availability ……. 116

Table 4t: Represents the digital artifacts recovered from the installation state of AnyDesk on macOS, including corresponding file names, storage locations, Content, and license-specific availability ……..……. 120

Table 4u: Forensic Artifacts Identified in Licensed Versions of Remote Access Applications Across Operating Systems…….. 128

Table 4v: Comparative Persistence of Forensic Artifacts After Uninstallation Across Remote Access Applications and Operating Systems……. 131

Table 4w: Comparative Extraction of Forensic Artifacts Across Remote Access Applications and Operating Systems…….. 133

LIST OF FIGURES

Figure 1: Flowchart of the methodology outlining the process from the selection of Remote Access Applications (RAAs) to forensic analysis and result interpretation.…… …… 22

Figure 2: Illustrates the *TVLog.html* file related to the TeamViewer Quick Support application, highlighting artifacts such as file transfer activities, including the size of the file, clipboard operations, and application uninstall commands with corresponding bundle names …… 33

Figure 3: Represents the *TVLog.html* file related to the TeamViewer Host application, highlighting artifacts such as the client device make and model, client TeamViewer ID, host Username, and TeamViewer ID ……. 33

Figure 4: Illustrates the *client.conf* file, highlighting artifacts such as BuddyDisplayName (display name), BuddyLoginName (email), BuddyLoginTokenID (unique session token), encrypted login tokens, SUID, and UserAccountID …….. 34

Figure 5: Represents the *Connection.txt* file, highlighting artifacts such as outgoing connection details, including TeamViewer ID of connected device (Host), start and end timestamps, Client Display name and make/model, and connection type/action …….. 34

Figure 6: Represents the *TeamViewer15_Logfile.log* file, highlighting artifacts such as timestamps, software version, operating system version, private IP of the Windows system, and the default AppPath of the TeamViewer full client software……. 38

Figure 7: Represents the Prefetch file, highlighting artifacts such as creation, modification, and last access timestamps, along with run count and other active timestamps…… 39

Figure 8: Represents the *System.evtx* file, highlighting artifacts such as event creation timestamps, along with Event ID, Service name, and associated File path …….. 39

Figure 9: Represents the *kSecClassGenericPassword* file, highlighting artifacts such as creation/modification timestamp and Email address of the TeamViewer client account …….. 44

Figure 10: Represents the *iTunesMetadata.plist* file, highlighting artifacts such as App Version, Name, Apple ID, and Install timestamp (UTC)……. 45

Figure 11: Represents the *mobile_installation.log.0*, highlighting artifacts such as uninstall request timestamp and logs of uninstallation……. 45

Figure 12: Represents the *com.teamviewer.teamviewer.preferences.plist* file, highlighting host device artifacts such as meeting username, TeamViewer ID, display name, and login email address…….. 49

Figure 13: Represents the *G2PPreferences.xml* file, highlighting user-specific data such as email address, plaintext password, biometric authentication flag (IS_BIOMETRICS_ENABLED), and password save prompt status…….. 52

Figure 14: Represents the *GoToMyPCUnattended.log* file, highlighting Windows metadata, including but not limited to platform, OS details, hostname, device name, private IP address, MAC address, system model, manufacturer, processor, BIOS version, memory, and time zone. ……. 56

Figure 15: Represents the *com.logmein.ios.gotomypc.plist* file, highlighting the last active email used for the session…… 59

Figure 16: Represents the *com.logmein.GoToMyPC.plist* file, highlighting artifacts such as email address, machine name, PlanType, and primary username…….. 63

Figure 17: Represents the *TCC.db* file, highlighting the access permissions granted to the application by the user…… 63

Figure 18: Represents the *RPC Viewer Log.txt* file, highlighting artifacts such as the host's public and private IP address, client private IP address, host device username, and host first name…….. 70

Figure 19: Represents the *com.remotepc.host.xml* file, highlighting artifacts such as host device make and model, personal key, trial period details, plain-text password, client and host public IP addresses, client first name, email address, license information, and plan type…… 70

Figure 20: Represents the *AndroidHost.txt* file, highlighting artifacts such as log timestamps, session recording entries, host device make and model, client device username, host public IP address, host first name, and client email address…….. 71

Figure 21: Represents the *library.db* file, highlighting the installation of the RemotePC Host application via the Play Store along with the associated email ID…….. 71

Figure 22: Represents the *RemotePCService.txt* file, highlighting artifacts such as the client email address, client private IP address, timestamp, and host device username…… 76

Figure 23: Represents the *RPCSettings.ini* file, highlighting artifacts such as host email address, license type, and free trial details…… 76

Figure 24a: Represents the *RPCSuite_digit.log* file, highlighting artifacts such as host email address, timestamp, last name host, license type, license name, and allowed permissions……. 77

Figure 24b: Represents the *RPCSuite_digit.log* file, highlighting whiteboard activity details such as tool usage, pen color, width, state, and timestamp. The numerical values indicate the type of tool used, while state '1' denotes true…… 77

Figure 25: Represents the *settings.json* file, highlighting email address, authentication token, user type, First name, plan description, and session recording file path……. 77

Figure 26: Represents the *metadata.clsrecord* file, highlighting artifacts such as host device model, machine identifier, OS build, display version, platform, locale, and application details, including bundle ID, build version, and display version…… 81

Figure 27: Represents the *trace.log* file, highlighting artifacts such as client name, client public and private IP addresses, host name, model identifier, email address, host public IP address, and application version details…… 81

Figure 28: Represents the *.dat.nosync0480.nWpx91* file, highlighting artifacts such as host descriptions, application version, and client public IP address…….. 85

Figure 29: Represents the *.HostPref.plist* file, highlighting artifacts such as last name, email address, Mac model identifier, details of the plan, and session recording path…… 85

Figure 30: Represents the *iamoauthlib.db-wal* file, highlighting artifacts such as full name, email address of the Zoho account, and client Zoho Assist ID…… 91

Figure 31: Represents the *session_history_database-wal* file, highlighting artifacts such as Zoho Assist account email, chat messages with timestamps, Zoho Assist ID, and sent file names with timestamps (host-to-client only). ……. 91

Figure 32: Represents the *ACServer.log* file, highlighting artifacts such as timestamp, pcname, displayname, and name associated with the Zoho Assist account (Email)…….. 95

Figure 33: Represents the *LogFileTemp.log* file, highlighting artifacts such as timestamp, Zoho Assist ID of Windows (Client), and the name associated with the Zoho Assist account (Email)…… 96

Figure 34: Represents the *CustomerPluginApp.log* file, highlighting artifacts such as timestamp, Zoho Assist ID of the Window (Host), and customer name… 96

Figure 35: Represents the *000003.log* file, highlighting artifacts such as email address, Date, Zoho Assist ID of host, and sent & received messages…… 96

Figure 36: Represents the *com.zoho.assist.zohoassist 2025-01-23_08-34-10.523.log*file, highlighting artifacts such as timestamps and file exchange logs, including file name, size, and path……. 100

Figure 37: Represents the *audit.log*, highlighting artifacts such as timestamps, host Zoho Assist ID, email address, file transfer details (file name and size), ALT+Tab command logs, lock screen commands, and command terminal entry..... 105

Figure 38: Represents the thumbnails of connected devices…… 109

Figure 39: Represents the chat logs, highlighting artifacts such as timestamps, the name of the connected device, and sent and received messages……..……. 109

Figure 40: Represents the *file_transfer_trace.txt* file, containing artifacts such as timestamps, file transfer events, transfer status (start/finish), process type (upload/download), file name, and file size…….. 110

Figure 41: Represents the *connection_trace.txt* file, highlighting artifacts such as incoming session logs, including date, time, and client AnyDesk ID…….. 110

Figure 42: Represents the *user.conf* file, highlighting artifacts such as the AnyDesk ID of Client and email address of the host device……. 115

Figure 43: Represents the *ad_svc.trace* file, highlighting artifacts such as timestamps, the host public IP address, and the AnyDesk ID…… 115

Figure 44a: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, the public IP address, and AnyDesk ID of the client…….. 119

Figure 44b: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, the public IP address, and AnyDesk ID of client. ……. 119

Figure 45: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, screenshot logs with saved file paths, and session recording logs…. 123

Figure 46: Represents the session recording file, whose name consists of the connection type, host device name (make/model), host AnyDesk ID, client device name, and client AnyDesk ID, formatted as: *outgoing samsung SM-J810G (1146523170)-digitalsoni (1564402395) 0.anydesk*…….. 123

LIST OF ABBREVIATIONS

APK	Android Package Kit
ARP	Address Resolution Protocol
CID	Client Identification
CLI	Command Line Interface
CSV	Comma Separated Values
DB	Database
DCIM	Digital Camera Images
DLL	Dynamic Link Library
DNS	Domain Name System
DPI	Dots Per Inch
FBI	Federal Bureau of Investigation
FTP	File Transfer Protocol
GB	Gigabyte
GPS	Global Positioning System
GUI	Graphical User Interface
HDD	Hard Disk Drive
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
ID	Identification
IP	Internet Protocol
iOS	iPhone Operating System
ISP	Internet Service Provider
JSON	JavaScript Object Notation

KB	Kilobyte
LAN	Local Area Network
Log	Record of system or application activity
MAC	Media Access Control
MB	Megabyte
OS	Operating System
PDF	Portable Document Format
PII	Personally Identifiable Information
PKI	Public Key Infrastructure
RAM	Random Access Memory
RAA	Remote Access Application
RDP	Remote Desktop Protocol
ROC	Regional Operation Centre
ROM	Read Only Memory
RPC	Remote Procedure Call
SMS	Short Message Service
SQL	Structured Query Language
SSD	Solid State Drive
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User Interface
USB	Universal Serial Bus
VM	Virtual Machine
VPN	Virtual Private Network
XML	Extensible Markup Language

CHAPTER 1

INTRODUCTION

The rise of electronic devices and continuous advancements of digital technologies have fundamentally changed every aspect of human life (Hoehe & Thibaut, 2020). Digital systems are now serving as the backbone of important infrastructure across social interaction, commerce, governance, healthcare, and national security. Recent reports estimate that over 5.56 billion individuals worldwide use the internet on daily basis, accounting for 67.9 percent of the global population (Statista, 2025). This technological advancement and revolution have brought benefits, including increased efficiency, top-notch connectivity, and faster innovation. However, it has simultaneously expanded the digital attack surface, enabling criminals to perform their malicious activity from physical domains to cyberspace with alarming sophistication (Mallick & Nath, 2024). As a result, forensic science has undergone a revolutionary shift, evolving to include digital forensics as a specialized discipline everywhere to address and mitigate the complexities of modern digital crimes (Klasen et al., 2024).

Digital forensics consists of the process that includes identification, preservation, recovery, and analysis of digital evidence from a wide range of electronic devices and networks. The main objective of digital forensics is to provide scientifically sound and legally admissible evidence and reports that can aid investigations and judicial proceedings (Raghavan, 2012).

The modern digital crime landscape is expanding rapidly. This growth is evident both in the sophistication of tools employed by offenders and in the scale of potential impacts. Cybercrime has escalated into a global threat, according to the 2025 Global Cybersecurity Report, costing an estimated economic loss exceeding $10.5 trillion per year (Cybersecurity Ventures Magazine, 2025). The root of many of these crimes is the exploitation and misuse of legitimate technologies for malicious purposes by attackers. One particularly concerning and growing trend is the misuse of Remote Access Applications (RAAs), which were originally developed to provide remote access convenience and productivity but are increasingly misused by cybercriminals to facilitate unauthorized access and control (Manson, 2022).

Remote access applications (RAA) allow users to remotely access and control a device through a network connection, providing the ability to control a system without any physical access. It works by executing inputs on the remote device and rendering results in real-time (Ismail et al., 2008).

A Remote Access Application functions through two primary components: the host and the client.

- **Host**: The host refers to the device that is made available for remote access. It functions as the recipient of external inputs and permits its operating system, files, and applications to be controlled remotely by a client device. The host runs the Remote Access Application service and remains in a listening state, waiting for incoming client connections and inputs.
- **Client:** The client is the device that initiates and controls the remote session. It transmits commands such as keystrokes, mouse actions, and configuration changes to the host, which executes these instructions and subsequently returns real-time data or visual feedback to the client.

In summary, the client acts as the controlling device, whereas the host operates as the controlled device within the remote access framework.

These Remote Access Applications (RAAs) are legitimately used for IT support, systems administration, collaborative work, and the facilitation of remote working. Commercially available licensed RAAs are often equipped with more advanced features, including but not limited to session recordings, chats, clipboards, and session logging, making them invaluable to enterprise environments (Manson, 2022). However, these advanced features are equally attractive to attackers, who want to use them to bypass physical barriers and operate under the guise of legitimate user activity.

The COVID-19 pandemic has resulted in a significant shift towards remote work, making people rely on RAA software (Sahut & Lissillour, 2022). Internet usage increased drastically during this period, and Remote access applications became central to maintaining business continuity and communication as they allowed employees and administrators to access their systems and files directly from home, ensuring

productivity & reliability despite the absence of physical access to workplace infrastructure (Bal & Bulgur, 2023).

However, the growth of RAAs also attracted the attention of malicious actors who began exploiting these applications for their own personal gains by performing social engineering techniques to trick users into installing these applications and having unauthorized access to their personal data and credentials (Jiang et al., 2019).

In many reported cases (ET Bureau, 2019; Free Press Journal, 2023; NDTV, 2023), attackers convinced victims to install remote access tools, presenting them as legitimate software for support or collaboration. Once installed, these applications provided unauthorized access to the victim's device, enabling the attacker to view and access the sensitive data, capture images and videos, login credentials, one-time passwords (OTPs), and, in some cases, users' private information for blackmail and extortion (India T. O., 2024). This dual-use nature of RAAs is legitimate for remote work, but also dangerous when abused, making them a critical area of concern in digital forensics.

Forensic experts face multiple challenges when investigating crimes involving RAAs. In many cases, investigators receive only the host device as evidence. Since maintaining the integrity of digital evidence requires shutting down the device before preservation and examination, volatile memory forensics used to capture live traces of activities cannot be performed (Hamid & Rahman, 2024). This limitation usually forces investigators to rely on persistent artifacts left on the device's storage by these RAAs, such as download and installation records, configuration files, and residual logs of activities. However, identifying and interpreting these artifacts is a complex task, as information storage varies widely in every RAA, and many attackers try to erase their digital footprints by uninstalling the application. In some cases, even victims uninstall these applications as soon as they suspect unauthorized monitoring or access, in an effort to protect themselves from further harm.

Existing research on the forensic analysis of Remote Access Applications (RAAs) is very limited, leaving substantial gaps in knowledge. Most available studies are limited and fail to provide a detailed examination of RAA-specific artifacts. Critical issues remain unresolved, such as how artifacts vary across different operating systems, the

range of artifacts generated during installation, and their persistence after uninstallation. The absence of such structured research significantly hampers the ability of forensic investigators and law enforcement to accurately reconstruct the user activities.

Another overlooked dimension is the distinction between free and paid versions of RAAs. Commercially licensed RAAs typically provide advanced features, which may generate additional artifacts that could support forensic investigations. In contrast, free versions of the same software often have fewer functions, resulting in fewer traces of access being left behind. However, they are more commonly used by malicious actors as they are easier to install and configure while minimizing the digital footprint.

So, this study involves a systematic forensic examination of selected Remote Access Applications such as AnyDesk, TeamViewer, RemotePC, Zoho Assist, and LogMeIn (GoToMyPC), both in their free and licensed versions, to identify and compare a range of artifacts left on devices. An effort has been made to analyze these applications across multiple operating systems (Windows, macOS, iOS, and Android) in order to determine the variations in forensic footprints and the persistence of digital traces after installation and uninstallation. The study also attempts to simulate real-world scenarios by conducting blind extractions of artifacts to verify the reliability of this study.

Thus, the present study has been conducted with the following aims and objectives:

1. To determine and compare a range of artifacts extracted from selected licensed Remote Access Applications and their free version.
2. To compare the extracted artifacts of Remote Access Applications on devices with different operating systems.
3. To conduct a forensic analysis of the artifacts generated by remote access applications after installation and uninstallation from a device.
4. To conduct blind study on extraction of artifacts from selected Remote Access Applications to determine real case scenario with minimum one-fourth of the total samples.

CHAPTER 2

LITERATURE REVIEW

Cybercriminals are increasingly exploiting legitimate Remote Access Applications (RAAs) to gain unauthorized access to victims' devices for committing crimes like data theft, financial fraud, and manipulation of system files. The misuse of RAAs is particularly concerning because these tools are designed to appear legitimate, making detection difficult and forensic attribution complex. Despite the continued and growing prevalence of such cybercrimes, comprehensive forensic research focused on the systematic extraction and analysis of artifacts generated by RAAs remains limited. Much of the existing body of work addresses individual applications, isolated operating systems, or narrow forensic questions, leaving considerable methodological and evidentiary gaps.

The following review critically examines the available literature, drawing from peer-reviewed journals, scholarly books, conference proceedings, and credible online repositories, in order to map the trajectory of the field, identify its limitations, and establish the specific gaps that the present research addresses.

Kerai (2010) analysed the use of virtual network computing and remote desktop protocol on Windows mobile phones. The use of remote access applications on these devices could leave significant amounts of information that would be analysable by examining the device's registry structure. A mobile registry editor tool was utilized to conduct a comprehensive analysis of the registry of a device. During the analysis, it was discovered that poorly developed applications would often store remote session passwords in plain text, thereby exposing the device to the risk of exploitation by malicious actors in the event of theft or loss. Applications like Mocha VNC and Smart VNC on Windows Mobile devices would leave behind registry values and artifacts, including IP addresses and port numbers used during remote connections. Forensic investigators or law enforcement agencies could use these values as potential evidence in locating the corresponding computer system. While this study provided an early and important demonstration that RAAs leave forensically recoverable traces, it

was limited to a single mobile operating system (Windows Mobile) and did not examine artifact persistence across installation and uninstallation states. Proper analysis of these artifacts would significantly enhance the effectiveness of investigations into cases involving unauthorized access to sensitive information on Windows mobile devices.

Kerai (2012)) extended the earlier work by examining the presence of artifacts related to remote access protocols on a Windows PC. This study aimed to develop a forensic protocol for extracting artifacts from VNC and RDP clients and servers. The registry of Windows mobile PCs and phones exhibited valuable information regarding configurations and usage of devices. This hierarchical structure was similar to that of other Microsoft operating systems. Various types of tools like VMware, Windows XP operating systems, Forensic Tool Kit, Abel and Cain, FTK Imager, Mount Image Pro, Password Recovery Tools, Sysinternals Suite, and MiTeC Windows Registry Analyser were used for data collection, acquisition, and analysis of images. During the session initiation, it was noted that a Data Encryption Standard (DES) key was utilized for authentication in the process between the viewer and the server. Remote access protocols were observed to leave valuable information like identification data, IP addresses, existing files, deleted files, and traces of connections that could be retrieved forensically. Taken together, the studies by Kerai (2010, 2012) established foundational evidence that remote access protocols generate recoverable forensic artifacts, but their scope and methodological constraints underscore the need for more comprehensive and cross-platform investigations.

Altschaffel et al. (2013) proposed a statistical pattern recognition method for accurately differentiating various activities present in encrypted network traffic produced from TeamViewer sessions. The study consisted of four different phases, such as pre-processing, feature extraction, feature selection, and classification. The experiment involved the creation of 52 network traffic recordings on 32-bit Intel Ubuntu Linux using TeamViewer version 7. A custom feature extraction method was developed to obtain distinguishing features and assign labels for four categories: transfer, audio, video, and chat. Multiple experiments were conducted across various training and testing datasets. The performance was benchmarked against a reference

system to assess classification time. For validation, a dataset comprising TeamViewer sessions was utilized. The results demonstrated that this method effectively differentiated between file transfers, voice and video conferences, text chats, and standard remote sessions within TeamViewer. While methodologically innovative in its application of machine learning to network-level forensics, this study was confined to a single application version on a Linux environment, which significantly constrains the generalisability of its findings. The absence of any examination of disk-level or registry-level artifacts means the study captures only one dimension of the forensic footprint.

Lahaie & Leberfinger (2013) conducted a forensic analysis of TeamViewer. This study was aimed at issues related to privacy, security, data transfer, and artifacts left behind after uninstalling the application. Two virtual machines were created to simulate the victim's and the suspect's computers. Both machines were equipped with TeamViewer, which was used to carry out various activities like file transfers, meetings, and remote control. Digital forensic tools like FTK and manual examination techniques were used to analyse the data gathered during the study. The analysis revealed that the users were given the option to either remove or retain settings after uninstalling TeamViewer. However, only a few artifacts could be left behind in case of removing all settings. Log files created by the program exhibited extensive information and a detailed record of events during a session, such as all transferred files, as well as information about the participants in that session. This finding is significant in that it reveals the direct forensic impact of user behaviour during the uninstallation process. However, the study was limited to the Windows platform and a single RAA (TeamViewer), and the test environment of only two virtual machines constrains the representativeness of its findings. The absence of a cross-platform comparison and the lack of any examination of licensed versus free versions are notable limitations. This study nonetheless provided one of the earliest empirical demonstrations of post-uninstallation artifact persistence in RAAs, a theme that remains underexplored in the broader forensic literature and which the present study systematically investigates across five RAAs and four operating systems.

Saidi et al. (2013) carried out a forensic study of cyber-attacks by using Virtual Network Computing (VNC) and keylogger applications on a Windows 7 Home Premium (32-bit) Operating System. This study was aimed at tracking down the registry artifacts left by attackers who accessed the system using the VNC protocol. It also aimed at detecting any unauthorized access to the machine or installation of unwanted applications through the VNC connection with respect to the user activity. It was observed that the log files generated by the VNC program contained comprehensive information, including every file transfer and details about the members in the session. Forensic Toolkit (FTK) Imager and Yet Another Registry Utility (YARU) were used to investigate the potential evidence of illegal activities from registry files and image files. Alterations in registry entries were detected and examined by using the Windows Registry File Viewer. The registry provided digital footprints that would verify user activities. However, no evidence of keyloggers being found. Nonetheless, artifacts related to the installation of the VNC application were discovered in other keys of the registry. The analysis was found to be significant in tracing illegal activities to the registry.

Kerai & Vekariya (2016) conducted a study with the aim of detecting and recovering artifacts by using different remote desktop applications on a Windows 10 computer. The analysis involved five commonly used remote applications, namely RealVNC, TightVNC, LogMeIn, TeamViewer, and Citrix GoToMyPC. The study involved the utilization of various tools like Abel and Cain v4.9.35, VNCPass, VNCSniff, and Microsoft Sysinternals Suite. For VNC connections, the client's TCP port 5900 and TCP port 5800 were opened. However, other remote applications could run on TCP port 443 without requiring any special ports to be opened on the router. It was observed that VNC applications employed the Data Encryption Standard (DES) with a small key size of 56 bits, which was considered insecure, as an attacker could potentially intercept the traffic after cracking the encrypted password. Various artifacts were retrieved from RDA that were crucial for forensic investigations, particularly for solving crimes involving the transfer of illegal and terrorism-related materials over remote connections.

Jiang et al. (2019) conducted a forensic evaluation of six widely used remote desktop applications, including AnyDesk, MicroRDS, RealVNC, ConnectWise, Zoho Assist, and TeamViewer, on Windows 7 and 10 platforms to investigate side-channel information leakage. The study utilized machine learning approaches, including logistic regression, support vector machines, gradient boosting decision trees, and random forests, along with statistical features of flow bursts, to classify encrypted remote desktop traffic. Findings revealed that an attacker could reliably identify five common daily activities by editing documents, reading documents, web browsing, watching videos, and software installation with high accuracy. Additionally, the study showed that four fine-grained activities related to document editing using Microsoft Word and other tools could be precisely distinguished, achieving a high true positive rate and a low false positive rate. These results highlighted that current remote desktop encryption mechanisms are insufficient to prevent side-channel information leakage, emphasizing the urgent need for both users and service providers to address this critical privacy issue.

Khalid et al. (2021) conducted a forensic examination of Cisco WebEx application installed on an Android device. The study was aimed at introducing a memory parsing tool and extracting additional artifacts from the desktop client. Chrome data was examined, and a case study was conducted to showcase the importance of artifacts in such investigations. To ensure that artifacts were not mixed with other applications, a VM with 3 GB of RAM and 60 GB of disk space was created using a Windows 10 ISO file. The Cisco WebEx desktop app was then installed, and a new user account was created to replicate common user interactions, including the use of personal meeting room IDs for video conferencing meetings with guests. Various types of tools like FTK Imager, bulk extractor, volatility, Photorec, and Wireshark were used to extract generic forensic artifacts. It was observed that device memory exhibited rich artifacts like user account data, deleted messages, shared media, and plaintext meeting information. The Windows Registry and network artifacts also revealed useful details, such as credentials and server domains. To gain insights into the user's activities and the chronological trail of events, the artifacts from all sources were correlated and reconstructed.

Mahr et al. (2021) carried out a forensic investigation of the Zoom video conferencing application, with the specific objective of performing disk, memory, and network

analysis. The investigation was carried out to identify potential artifacts that could provide insight into the user's activities during Zoom sessions, including file transfers, chat messages, and screen sharing. A set of SQL queries were used to extract pertinent data from the application database. Various types of tools, such as Wireshark, Magnet Acquire, DumpIt, FTK Imager, Autopsy, Android Debug Bridge (ADB), Android Data Extraction Tool, DB Browser, checkra1n, Android Data Extraction Tool, SuperSur, DB Browser, and Volatility were used in this research. It was observed that various user information like plain-text user information was accessible, despite Zoom's efforts to fix and prevent security risks. Some data was also observed to be stored even after the deletion of contact by the user. Memory analysis revealed that evidence could be found even after the application had been closed.

Azhar et al. (2022) conducted a forensic examination of the two widely used video conferencing platforms, namely Microsoft Teams and Google Meet. The study aimed to identify and evaluate the security features and forensic artifacts associated with the applications. The collection and potential use of digital evidence related to these applications in legal investigations was also discussed. The study involved Windows-based virtual machines and multiple Android devices for the creation of test accounts, sending messages and voice calls, and analysing network traffic to mimic standard conference calls. Various software like FTK Imager, Autopsy, Wireshark, and Volatility3 were used to collect and analyse the data. It was observed that email data was retrievable from both the logs and cache. Additionally, the Windows registry was observed to contain a limited number of artifacts like email addresses. The extraction of user information like login details and transmitted messages was found to be possible by capturing and analysing network traffic.

Iqbal et al. (2022) conducted a forensic analysis of Google Meet’s memory to extract artifacts, and a digital forensic tool was also developed to automate this process. The study aimed to determine the impact of RAM sizes on extracted artifacts from Chrome, Firefox, and Edge. The focus of the analysis was to extract various artifacts from Google Meet, such as Email addresses, bookmarks, cache, communication records, cookies, downloads, meeting records, history, profile pictures, and other related data. Three virtual machines with varying memory sizes and operating systems were created

to test the Google Meet Web client on different browsers. The tests simulated user activities in three categories such as creating and starting meetings, scheduling meetings in Google Calendar, and joining meetings set up by other users. Several forensic tools like AccessData FTK Imager, Volatility, PhotoRec, Autopsy, Strings, DB Browser for SQLite, Chromecacheview, and Chromecookiesview were used to analyse the memory and disk images, browse application databases, and Google Meet cache. The results showed successful extraction of various artifacts like usage traces, meeting & communication records, whiteboard activity downloads, and correspondence emails from memory.

Khalid et al. (2022a) conducted a detailed memory forensic analysis with an in-depth analysis of network traffic related to the Microsoft Teams application. The objective of the study was to extract artifacts that could be used as digital evidence in legal proceedings and serve as a reference for forensic analysts working on cases related to Microsoft Teams. To achieve this, a controlled test environment consisting of ten virtual machines with 60GB of ROM and 4GB of RAM was utilized. To extract user account information, profile photos, exchanged messages and files (including deleted ones), timestamps, encryption keys, and other related data, a combination of memory, disk-space, and network forensics tools were utilized, including FTK Imager, Memory Dump, Volatility, Bulk Extractor, Photorec, and Wireshark. Encrypted network traffic was also analyzed to reconstruct client-server connections during a Microsoft Teams meeting, which involved investigating IP addresses, timestamps, and digital certificates. Despite the implementation of robust security measures, the study revealed that user data can still be extracted from a client's desktop.

Khalid et al. (2022b) conducted a forensic examination of the Cisco WebEx application using various methods, including memory, disk-space, and network forensics, to identify relevant digital artifacts. In addition, a case study was presented that focused on the analysis of encrypted WebEx traffic to extract pertinent forensic information. A controlled test environment of 10 virtual machines (60GB ROM and 4GB RAM) was used to test the Cisco Webex application. Various types of tools like FTK Imager, Photorec, and Wireshark were used to perform disk, memory, and network analysis. The findings revealed various forensic artifacts of WebEx Meetings, revealing

communication details, media files, meeting records, and network artifacts that could be linked with the reconstruction of user activities. The WebEx web application exhibited detailed meeting logs, with indexedDB-levelDB as a prime source of meeting details. Through the forensic analysis of Android OS backups, various forms of digital artifacts were discovered, which included user account information, media, cookies, and records of past meetings.

Manson (2022) carried out a forensic investigation of remote desktop software (RDS), with a particular focus on the popular Windows solution, TeamViewer. A test environment was created using virtual machines to investigate the artifacts left behind by TeamViewer remote connection software. The setup involved two Windows 10 VMs connected via VirtualBox, with the same version of TeamViewer and RegistryChangesView installed on both machines. Snapshots of virtual machines were utilized to monitor alterations made to the Registry Hives and filesystem across different scenarios. The possibility of privacy concerns occurred because of insufficient removal procedures that might result in sensitive information being left behind on a system. To assist investigators in analysing TeamViewer artifacts, a Python package was made available for Autopsy software, which could identify, handle, and display important TeamViewer evidence automatically. The module included functionality for the identification of TeamViewer installations, sessions, file transfers, and other events. It was observed that the module could be easily extended to support other RDS solutions.

Paligu & Varol (2022) conducted a forensic analysis of Microsoft Teams' desktop application, specifically exploring the data stored in IndexedDB storage from a forensic perspective. Two primary questions were raised regarding the potential value of the information stored in IndexedDB for forensic investigations and the feasibility of conducting time-frame analysis using data from Microsoft Teams' IndexedDB storage. A single-case pretest-posttest quasi-experiment in an experimental environment was conducted that included two laptops with Windows 10 and one phone with Android OS. The devices were configured by creating three Microsoft 365 accounts and installing Teams desktop and Android applications. The extracted artifacts were evaluated on the basis of signature patterns and assessed for their significance. Traditional database

queries were utilized to cluster the information by relevancy, and a time-frame analysis was conducted to format the data for investigators. Valuable artifacts were found in Microsoft Teams' IndexedDB storage, including private chat messages, voice mail, and team extensions, which could be highly useful in digital forensic investigations, despite the presence of security flaws in the application.

Bowling et al. (2023) carried out a forensic analysis of Microsoft Teams installed on three different operating systems, such as Android, iOS, and Windows 10, and provided methods for artifact recovery and evaluation of current automated recovery tools. Data population and analysis techniques for communication platforms like Teams were also presented. Four machines, like a virtual admin machine, a physical laptop with Windows 10, and jailbroken/rooted physical mobile devices running iOS and Android, were used in this study. Basic functionalities of Microsoft Teams, like messaging, file sharing, and video conferencing, in an isolated virtual testing environment were performed. The data obtained from the analysis was processed using various forensic tools, including FTK Imager, Cellebrite UFED Physical Analyser, and Magnet AXIOM Examine. Manual investigation of data resulted in the retrieval of a significant number of artifacts, including calls, calendar events, and account details on all three operating systems. The manual investigation yielded a recovery rate of 77.6% for important artifacts, either partially or in full. Despite the use of forensic tools, many artifacts identified through manual investigation could not be recovered, with only 13.8% of the artifacts being partially or fully retrieved across all three devices.

Hur et al. (2023) carried out a forensic study of the multi-platform Cisco Webex tool. This study aimed to extract and analyze synchronized user data, such as chatting, file transfer history, calendars, profiles, and login information from cloud-stored artifacts for digital investigations. To simulate real-world usage, devices including a Ryzen 3700X desktop (Windows 10), MacBook Air M1 (macOS 12.2.1), iPhone X (iOS 16.0.2), and Galaxy S21 5G (Android 11) were used with Webex versions ranging from 42.11.0 to 43.4.0.185. Techniques involved decrypting SQLite Encryption Extension (SEE) databases using RC4/AES with acquired DB keys and migrating credentials for server access. Three key databases (spark_persistent_store.db, spark_roaming_store.db, Webex_pre_login_store.db) were observed to contain

encrypted forensic artifacts. Deleted messages could be recovered from remnants and credentials, enabling post-acquisition cloud data retrieval. In addition, artifacts from all platforms and cloud sources were correlated and reconstructed to gain insights into the user's activities and the chronological trail of events.

Paneda et al. (2024) conducted a forensic examination of file exfiltration using AnyDesk, TeamViewer, and Chrome Remote Desktop on Windows PCs, application logs, registry entries, prefetch files, and network traffic to determine the type of transferred files, the time of exfiltration, and the sender or receiver of that transfer. The aim was to provide digital evidence for legal purposes and identify behavioral patterns in encrypted communications for proactive security alerts via network monitoring. Base scenarios involved controlled file transfers in telecommuting setups by using Windows 10/11 with specific app versions like TeamViewer 15.43.0200, AnyDesk 8.0.0, and Chrome Remote Desktop 114.0.5735.199. Tools like Autopsy, Wireshark, and Windows Event Viewer were used to collect and analyze persistent artifacts. Results showed that logs and registry entries offered partial answers requiring complementary sources for complete evidence. Certain traffic behaviors, like unusual HTTPS tunneling, could trigger early alarms. Insights into exfiltration events were gained by correlating artifacts across logs, system files, and network data. This study is one of the few to include AnyDesk and to examine prefetch files as a forensic source for RAA activity, representing a meaningful contribution to the field.

Tiwari et al. (2025) carried out forensic analysis of browser-based GoToMeeting clients to uncover memory and browser artifacts that was focussed on recovering hidden forensic evidence, such as meeting records (including type, ID, timezone, duration, and participant count), user details (such as username, display name, email, account creation date, and profile pictures), participant details (such as display name, email, join time, and virtual background usage), chat messages, shared notes with metadata (creator ID, email, and timestamps), and scheduled meeting details (name, start time/date, duration, invitation link, and organizer info). The study aimed to address challenges in SaaS forensics for web-based video conferencing applications and provide insights for law enforcement in digital investigations, as well as for developers implementing security measures. To simulate user interactions, a virtual machine was

created using Windows 10 with 3GB RAM and 30GB disk space, Chrome browser installed, and a temporary GoToMeeting account. Activities like instant meetings, scheduling, chatting, note-sharing, and screen sharing were performed in this process. Various types of tools, like FTK Imager, Volatility, Strings, OSForensics, Autopsy, and Bulk Extractor, were used to acquire and analyze memory dumps and disk images. It was observed that memory exhibited rich artifacts like unencrypted profile pictures, message texts (without metadata), and detailed notes/meeting info. In addition, browser data revealed usage traces in jump lists, top sites, favicons, sessions, IndexedDB logs, bookmarks, history, cookies, cache, emails, and thumbnails. The artifacts from all sources were correlated and reconstructed to gain insights into the user's activities and the chronological trail of events.

A detailed review of the above-mentioned has highlighted significant findings on the artifacts generated by Remote Access Applications (RAAs) and screen mirroring applications. Numerous studies have highlighted the presence of valuable artifacts, such as registry entries, log files, network traffic data, memory dumps, and browser cache, which may provide critical evidence of user activities, including file transfers, remote sessions, chat messages, and meeting records. These artifacts have been instrumental in reconstructing user interactions and identifying unauthorized access or data exfiltration. The review has also indicated that artifacts vary depending on the application, operating system, and the respective settings retained or removed during uninstallation, with some applications leaving sensitive data like plaintext passwords, IP addresses, and user credentials, posing privacy and security risks. The convergence of findings across studies using different tools and methodologies from FTK Imager and Autopsy to Magnet AXIOM and Volatility lends credibility to the broader principle that RAAs and communication applications generate recoverable forensic evidence even in adverse conditions such as application closure or contact deletion. Furthermore, the finding of Bowling et al. (2023), that automated forensic tools recovered only 13.8% of artifacts compared to 77.6% through manual investigation, is a methodologically important caution against over-reliance on tool-based acquisition without human validation is a concern that the present study addresses through its combined automated and manual analysis approach, reinforced by blind-sample validation.

Despite the extensive body of research, significant gaps in related research have been found. The review of literature has demonstrated the most persistent limitation across the reviewed studies is the restriction of forensic examination to a single operating system, most frequently Windows, which fundamentally limits the generalisability of findings to the multi-platform environments in which RAA-facilitated cybercrime actually occurs. While Hur et al. (2023) and Bowling et al. (2023) stand out as exceptions in their cross-platform scope, neither study examines dedicated remote access and control applications. Equally notable is the near-absence of forensic research on several of the most widely deployed RAAs: no prior study has systematically examined Zoho Assist, RemotePC, or GoToMyPC from a dedicated forensic perspective, despite their established presence in both professional and criminal-use contexts. TeamViewer and, to a lesser extent, AnyDesk have received the majority of forensic attention, while the remaining applications remain almost entirely uncharted. Furthermore, none of the reviewed studies has systematically compared the forensic footprints generated by free versus licensed versions of the same RAA, a distinction with significant practical consequences, given that free versions are more commonly encountered in cybercrime investigations yet may yield substantially fewer recoverable artifacts. Post-uninstallation artifact persistence, though touched upon by Lahaie & Leberfinger (2013) and Manson (2022), has not been systematically examined across multiple applications, operating systems, and license conditions. Finally, no reviewed study has incorporated blind-sample validation to confirm the reliability and real-world transferability of its findings, a methodological weakness that limits the operational applicability of the existing literature.

Thus, a systematic research study has been designed to determine and compare the range of forensic artifacts generated by licensed and free versions of the selected Remote Access Applications. The current research also aims to analyze variations in artifacts across different operating systems and the persistence of these artifacts in the post-installation and uninstallation of these applications. In addition, this research incorporates a blind study to validate the methodology and results of the investigation conducted.

CHAPTER 3

METHODOLOGY

A user can remotely control devices over a network and also perform tasks such as file sharing, remote collaboration, troubleshooting, and monitoring with the help of remote access applications. These applications are widely beneficial for society, but they are often exploited by cybercriminals for unauthorized access, data theft, and other illegal activities. The licensed version of these RAAs provides more advanced features, which often leave different digital footprints as compared to their free versions. Additionally, the artifacts generated by RAAs vary across operating systems due to differences in storage mechanisms, permissions, and logging behavior. Artifacts generated by these applications often persist after uninstallation, which can further aid forensic investigation related to RAAs. Thus, digital artifacts extracted from free and licensed versions of selected RAAs were analyzed in installed and uninstalled states across multiple operating systems (Windows, macOS, iOS, and Android) to identify patterns, differences, and forensic relevance.

Some artifacts may be misinterpreted or overlooked if not systematically examined, which can lead to incomplete & incorrect reconstructions of cyber incidents. Thus, the controlled simulated experiments were conducted for systematic examination of RAAs across different operating systems, states, and licenses.

A systematic approach was followed to achieve the aims and objectives of this present research study, as discussed below:

3.1 Material

3.1.1 Remote Access Applications

This research study was conducted to determine the range of artifacts generated by various Remote Access Applications (RAAs) and their importance in digital forensic investigations. RAAs with broad cross-platform support (e.g., Android, iOS, Windows, macOS) and features that are commonly abused by cybercriminals, like remote control, file transfer, and screen mirroring, were given priority (ET Bureau, 2019). Both free and licensed versions were included and compared to determine the artifact generation

across different access levels. Five Remote Access Applications (RAAs) were analyzed in this study: TeamViewer, AnyDesk, Zoho Assist, RemotePC, and LogMeIn (GoToMyPC). The five Remote Access Applications (RAAs) analyzed in this study are described as follows:

3.1.1.1 TeamViewer

TeamViewer is a widely recognized and popular remote access tool available across multiple platforms, including Android, iOS, macOS, and Windows. It is trusted globally with more than 320 million users with 2.5 billion installations across 190 countries (Heidelberg, 2025). TeamViewer has many features like high-quality remote access and control, cross-platform compatibility, end-to-end encryption, file transfer, unattended access, session recording, chat, drag-and-drop file sharing, clipboard, remote app uninstalls, remote call, privacy mode, send Alt+Delete, and remote lock screen (TeamViewer, 2025). It has multiple separate applications for different functions, as discussed in Table 3a.

Table 3a: TeamViewer Variants Employed for Forensic Analysis on Different Platforms

Operating System	**Application Name**	**Description**
Android	TeamViewer QuickSupport v15.58.555	Allows the device to receive remote assistance.
	TeamViewer Host v15.58.556	Functions as an add-on for unattended remote access.
	TeamViewer Remote Control v15.58.553	Enables the user to provide remote support to other devices.
Window	TeamViewer full client v15.58.5 (64-bit)	Provides complete functionality for both incoming and outgoing remote connections.
iOS	TeamViewer QuickSupport v15.62.652	Allow the device to receive remote assistance.
	TeamViewer Remote Control v15.59.0	Enables the user to provide remote support to other devices.
macOS	TeamViewer full client v15.62.4	Provides complete functionality for both incoming and outgoing remote connections.

3.1.1.2 LogMeIn (GoToMyPC)

LogMeIn (GoToMyPC) is a prominent remote desktop software available across multiple platforms, including Android, iOS, macOS, and Windows. It serves more than 10 million users across 180+ countries. GoToMyPC is known for its simplicity and advanced security. For the purpose of this study, the versions examined were Android (6.2.3093), Windows (2025.1.4266), iOS (6.2.3099), and macOS (2025.1.4266). The key features of GoToMyPC include remote access and control, seamless installation, multi-monitor support, file transfer, unattended access, remote sound, session timeout for security, remote lock screen, and screen blacking for privacy (GoToMyPC, 2025). GoToMyPC uses a single full-featured client application for all kinds of operations, eliminating the need for separate applications for different functions like incoming or outgoing connections.

3.1.1.3 RemotePC

RemotePC is a well-known Remote Access Application available for multiple operating systems, including Android, iOS, macOS, and Windows. It has been used by millions of users throughout the world, with a strong reputation for feature-rich functionality, reliability, and affordability. For the purpose of this study, the versions examined were Windows (7.6.79), macOS (7.7.35), iOS (7.9.7), and Android, which included two separate applications: RemotePC Host (1.1.5) for receiving support and RemotePC Viewer (5.5.7) for providing support. The key features of the RemotePC application include unattended access, remote file access & transfer, multi-platform, remote reboot, drag and drop file transfer, chat, remote sound, whiteboard, and session recording (RemotePC, 2025). RemotePC generally uses a single full-featured client application for all operations, eliminating the need for separate applications for different functions such as incoming or outgoing connections. However, on Android, it specifically provides the two distinct applications mentioned above.

3.1.1.4 Zoho Assist

Zoho Assist is also known for its versatility, security, and availability across multiple operating systems such as Android, iOS, macOS, and Windows. It has been installed over 850,000 times, completed more than 11 million remote sessions, and served more than 130 million customers in 180+ countries. For the purpose of this study, the

versions examined were Windows (2.7.1), macOS (2.7.2), iOS (3.2.25), and Android, which included two separate applications: Zoho Assist–Customer App (1.37.7) for receiving support and Zoho Assist–Remote Desktop (3.22.2) for providing support. Key features of Zoho Assist include unattended remote access, screen sharing, file transfer, chat, voice and video chat, reboot and reconnect, session recording, swap screen, and annotation (Zoho Assist, 2025). Zoho Assist generally uses a single full-featured client application for all operations, eliminating the need for separate applications for different functions like incoming or outgoing connections. However, on Android, the two distinct applications mentioned above are provided.

3.1.1.5. AnyDesk

AnyDesk is a leading and the most popular Remote Access Application, recognized for its speed, security, versatility, and availability across multiple operating systems, including Android, iOS, macOS, and Windows. It has been downloaded over 1 billion times and served more than 200,000 businesses in 190 countries (AnyDesk, 2025). For the purpose of this study, the versions examined were Android (8.0.4), Windows (9.6.0), iOS (7.1.2), and macOS (9.5.0). Key features of AnyDesk include remote access and control, cross-platform compatibility, file transfer, session recording, chat, clipboard, clipboard file transfer, screenshot, and privacy mode.

AnyDesk uses a single full-featured client application for all kinds of operations, eliminating the need for separate applications for different functions like incoming or outgoing connections.

3.1.2 Operating Systems and Devices

The four operating systems, including Android, iOS, macOS, and Windows, were chosen on the basis of their prevalence in personal, professional, and cybercrime contexts. These platforms dominate global device ecosystems and have different data storage and logging systems compared to each other (StatCounter Global Stats, 2025). The selection included specific versions to balance modern functionality with compatibility for the tested RAAs, while virtualization tools were employed to create controlled yet realistic testing environments.

Operating systems such as Windows 10, macOS Ventura, Android 10, and iOS 15.8.4 were chosen for their extensive adoption worldwide. These versions of the operating

system showed robust compatibility with all the selected Remote Access Applications and facilitated the generation of a comprehensive range of artifacts during preliminary testing. For Windows and macOS, virtual machines (VM) were created using VMware Workstation 17 software (VMware Workstation Player, 2025). This tool allowed isolated environments with snapshot capabilities, enabling resets to clean states between tests to avoid data contamination

3.2 Methodology

3.2.1. Setup of Test Environments

To facilitate a controlled and reproducible analysis of Remote Access Application artifacts, test environments were configured using both virtual and physical devices across the selected operating systems.

3.2.2 Configuration of Virtual Machines

VMware Workstation 17 software was used to create virtual machines for Windows 10 and macOS Ventura. For Windows, each VM was configured with 4GB RAM and 30GB storage to replicate a real-world user system, ensuring sufficient resources for RAA to execute smoothly. For macOS, VMs were allocated 4GB RAM and 80GB storage to manage the resource demands of the APFS file system and to execute RAA smoothly. Each VM was installed with a clean operating system image sourced from official Microsoft and Apple distributions. Network adapters were configured to enable internet access via a bridged connection, simulating real-world connectivity for remote sessions.

3.2.3 Configuration of Physical Devices

Physical devices were used for mobile platforms to ensure realistic artifact generation. A Samsung Galaxy J8 running Android 10 and an iPhone 7 Plus running iOS 15.8.4 were selected for testing. Before each test, both devices were reset to factory settings to eliminate residual data from previous experiments. Each device was rooted or jailbroken for every setup to grant full-file system access, enabling comprehensive extraction of RAA artifacts, such as logs and databases. Rooting and jailbreaking were performed anew for every test session to maintain a clean baseline, ensuring that artifacts were solely attributable to the tested RAA and usage scenario. Devices were

installed with RAAs via official sources (Google Play Store for Android, App Store for iOS) to reflect standard user behavior.

3.2.4 Network

To simulate realistic remote access scenarios, two distinct personal Wi-Fi networks were used for all test sessions, representing separate network environments for client and host devices. The network's public and private IP addresses were recorded at regular intervals from the start to the end of each user session. The use of two Wi-Fi networks ensured variability in connection scenarios, mimicking real-world conditions where devices may operate on different networks.

3.2.5 Sample Size and Grouping

The total number of samples analyzed in the study was 100, broken down as follows:

- 40 samples from free versions of RAAs
- 40 samples from licensed versions of RAAs
- 20 real-life blind samples

Each RAA was tested on all four operating systems in both installed and uninstalled states. This resulted in 80 structured samples (5 RAAs × 4 OS × 2 license types × 2 states: installation/uninstallation) and 20 blind test samples, which were used to validate the findings of this study (Figure 1).

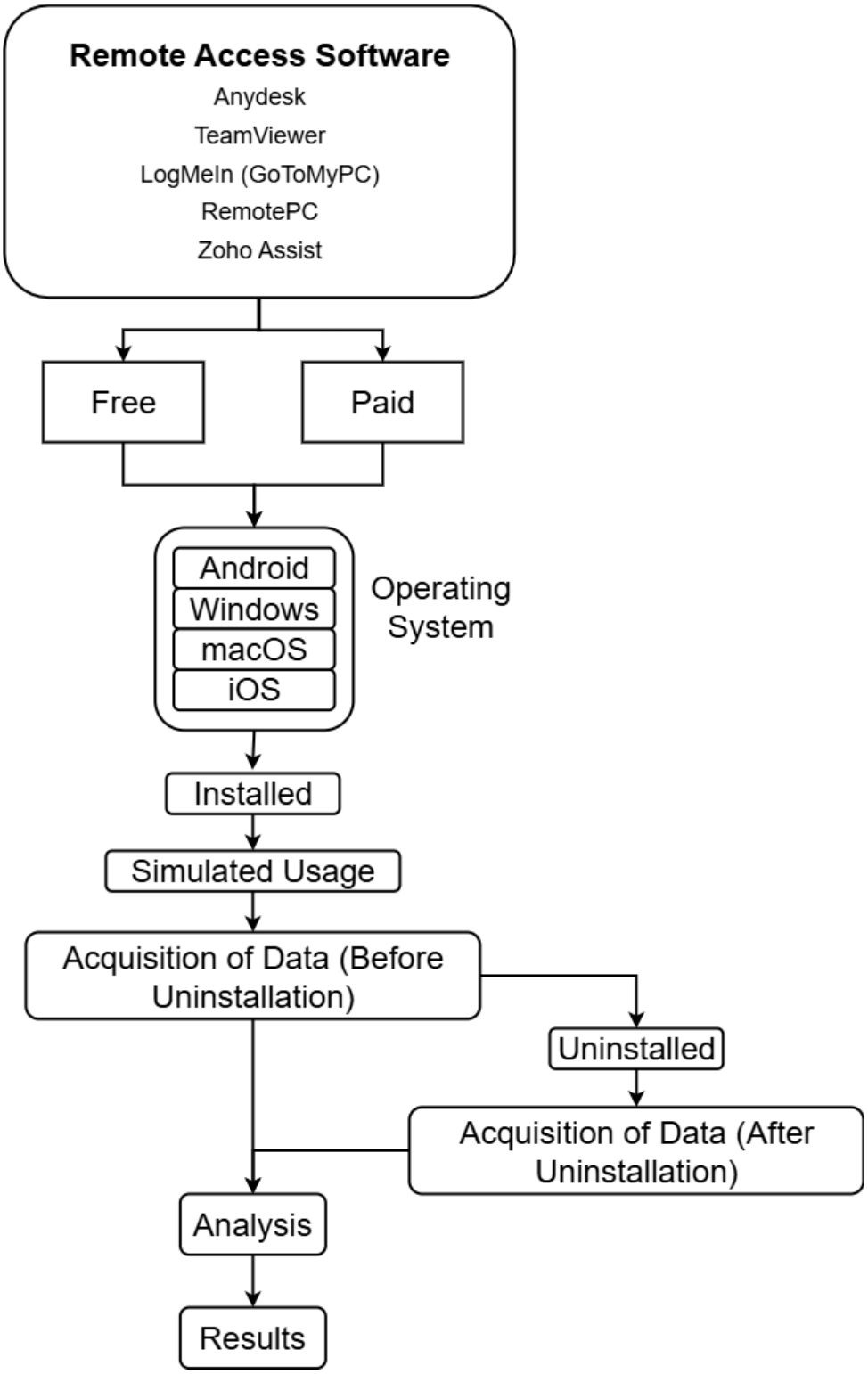


Figure 1: Flowchart of the methodology outlining the process from the selection of Remote Access Applications (RAAs) to forensic analysis and result interpretation.

3.2.6 Tools and Software Used

The following tools were used throughout the study for acquisition and analysis:

- **Magnet AXIOM Process v8.4.0.41469**: Utilized for data extraction from Windows, macOS, and Android devices. Magnet AXIOM Process enabled the extraction of system and application data, including logs, configuration files, and user activity records, critical for forensic analysis of RAAs.
- **Magnet AXIOM Examine v8.4.0.41469**: Employed for in-depth analysis and correlation of extracted data (Magnet Forensics, 2025).
- **MD-NEXT v1.92.10.1959**: Utilized for data acquisition from iOS devices. MD-NEXT enabled the extraction of application data, including databases and logs generated by RAAs, ensuring comprehensive artifact collection from mobile platforms (GMDSOFT, 2025).
- **MD-RED v3.10.13.2287**: Applied for the analysis and correlation of data extracted from iOS devices (GMDSOFT, 2025).
- **DB Browser for SQLite v3.12.2**: Used for manual inspection and analysis of SQLite databases, which are commonly utilized by mobile and desktop RAAs (DB Browser for SQLite, 2025).
- **Sixo Online APK Analyzer:** Utilized to extract and analyze the AndroidManifest.xml file from the base.apk (Sixo Online APK Analyzer, 2025)
- **PECmd.exe v0.6.0.0**: Employed for the analysis of prefetch files on Windows 10 systems. Prefetch files are used in Windows devices to optimize application loading times, contain valuable metadata about application execution, such as timestamps and file paths, making them essential for reconstructing user activity related to RAAs (Hammazahmed, 2024)

3.2.7 Data Collection

3.2.7.1 Installation and Simulation of Usage

For each RAA and OS combination, the following steps were performed, and all the entries were documented:

1. The latest version of the application was downloaded from its official source.

2. The application was installed on a clean virtual machine or device.
3. Standard configuration and login procedures were followed.
4. Realistic user activity was conducted to utilize the full range of functionalities available in each RAA, whether in the free or licensed version. All available functionalities were used to ensure comprehensive artifact generation, including but not limited to:

    - File transfer (documents, media files)
    - Remote control sessions
    - Screen capture or session recording
    - Chat or text communication
    - Clipboard operations

    This ensured that a variety of user interactions were logged and could be traced during forensic examination.

3.2.7.2 Post-Installation Data Collection

After simulating realistic user activity using all available features of each RAA, forensic acquisition was performed using the above-mentioned digital forensic tools. For desktop environments (Windows and macOS), the VMDK files from VMware were extracted and used for analysis. This approach allowed full access to the system storage.

For mobile platforms (Android and iOS), full file system extraction techniques were applied to capture complete device data, including system files, application data, and residual traces. This ensured consistency across platforms and allowed for a thorough forensic examination of artifacts left by RAAs. All the findings during analysis were tagged, documented, and categorized for further comparison.

3.2.7.3 Post-Uninstallation Data Collection

To analyze residual artifacts, a copy of the original virtual machine was created by duplicating the associated folder of the virtual machine (using standard file copy methods). The replica was verified to ensure it was an exact clone of the pre-uninstallation state.

Next, the copied virtual machine was launched, and the corresponding RAA was uninstalled using the default Windows uninstaller. Following uninstallation, the

updated VMDK file was again extracted and subjected to forensic analysis to detect the artifacts. The same process was mirrored for mobile devices. An extraction was performed post-uninstallation, enabling a direct comparison between the artifact sets before and after application removal.

This method provided a reliable baseline for identifying the persistence and forensic relevance of artifacts left by each RAA across different operating systems and platforms.

3.2.7.4 Blind Sample Preparation and Data Collection

In addition to the structured and controlled tests, 20 blind samples of the Remote Access application were taken to validate the results of this study.

For this phase, mobile devices and virtual machines were provided to volunteer subjects. Participants were instructed to install, configure, and interact with one or more RAAs. All interactions were performed without external supervision, and sessions were self-documented by the volunteers for potential reference during result comparison.

Following usage, forensic acquisition was conducted on both mobile phones and virtual machines. Previously identified artifact locations served as reference points for examining residual traces in blind samples.

Artifact analysis was conducted exclusively on the extracted forensic data, without relying on participant notes. This approach aimed to validate the findings of this study in detecting, comparing, and distinguishing artifacts generated by different RAAs under blind conditions.

3.2.8 Classification of Digital Artifacts

In this study, the term *digital artifact* refers to any trace generated or modified by a Remote Access Application during its installation, configuration, active usage, communication events, or removal from the device. These traces include logs, configuration files, identifiers, metadata entries, temporary files, and post-uninstallation residues that collectively represent the application's operational footprint. The examination of such artifacts is essential for reconstructing user activity, validating the presence of an application, identifying session details, determining

network interactions, and assessing the persistence of evidence across different operating systems.

To maintain uniformity during analysis and to enable clear comparison across platforms, all collected artifacts were classified into specific categories based on their function and forensic relevance. These categories include installation artifacts, configuration and identity artifacts, session and usage artifacts, network and communication artifacts, cache and temporary artifacts, and residual artifacts left after uninstallation. Because each operating system stores and structures data differently, artifacts were further organised according to the OS in which they were recovered. This OS-wise classification helped highlight differences in forensic visibility, artifact richness, and evidence persistence across Windows, macOS, Android, and iOS.

The following table summarises the artifact types identified in this study and presents the corresponding operating system locations and structures where these artifacts were observed.

Table 3.x: Artifact Types Identified Across Operating Systems (Based on Actual Study Findings)

Artifact Type	Android	Windows	iOS	macOS
Installation Artifacts	base.apk metadata, XML preference files, app directories	installation logs, application folders, registry keys (RAA entries), Prefetch files	mobile_installation.log entries	install.log, plist installation records, Application Support directories
Configuration & Identity Artifacts	client.conf, global.conf, XML configs (settings, SUID, tokens)	configuration folders, client/device IDs stored in program data, registry-based IDs	settings plist, Keychain identifiers (where present)	plist preference files containing clientID, machineID

Session & Usage Artifacts	TVLog.html, TVLogOld.html, internal app logs, screenshots	application log files, Connections.txt, session metadata, .tvs recordings (TeamViewer), Event Logs	minimal session logs inside app sandbox, plist entries	plist session logs, system logs, .tvs screen recordings (TeamViewer)
Network & Communication Artifacts	heartbeat logs (Firebase), connection logs stored in XML/app logs	IP logs, connection history inside application log files	restricted network metadata (limited to internal logs)	network metadata inside system/application logs
Cache & Temporary Artifacts	cached XML, internal app cache files, temporary logs	session cache files, temporary log fragments	very limited cache traces	plist cache, temporary system log traces
Uninstallation Residues	small XML/cache remnants (in some cases)	rolloutfile.tv13 (TeamViewer ID), small leftover logs, registry remnants	almost no recoverable remnants	leftover plist fragments, installation log traces

3.2.9 Artifact Analysis

Following the collection of forensic data from virtual machines and physical mobile devices, a systematic analysis was conducted to examine digital artifacts generated by each Remote Access Application. The objective of this study was to identify, characterize, and compare artifacts associated with the installation and uninstallation of RAAs in both their free and licensed versions across the above-mentioned operating systems, thereby evaluating their forensic significance in investigations.

A semi-automatic analysis approach, combining forensic tools with manual verification, was employed to ensure accuracy and minimize the risk of overlooking artifacts. Tools such as Magnet AXIOM, MD-RED, MD-NEXT, PECmd.exe, and DB Browser for SQLite were used to extract and review the extracted files, including but not limited to logs, configuration files, databases, and system traces.

All associated files and logs of RAAs were examined for evidence of user activities, including file transfers, chat logs, screen recordings, connection histories, and configuration details. Specific focus was given to identifying IP addresses, timestamps, usernames, session IDs, and communication records to aid in the reconstruction of activities.

Artifact locations identified during controlled experiments served as reference points for searching similar traces in blind samples. Each recovered artifact was documented and evaluated for persistence post-uninstallation and forensic relevance.

This approach facilitated a comprehensive understanding of RAA behavior across platforms, highlighting variations in artifact generation and detectability, and provided insights into their utility for reconstructing events and attributing actions in forensic investigations.

CHAPTER 4

RESULTS AND DISCUSSION

Artifacts were extracted, analyzed, and compared across different scenarios using a combination of forensic tools and manual approaches to achieve the above-mentioned objectives of the study. The results obtained in the present research study are presented and discussed as follows:

4.1 Results

This section presents the observations made during the extraction and analysis of artifacts generated by the selected Remote Access Applications. Each application's results have been divided into subsections based on operating system and further subdivided by state (installation or uninstallation).

The artifact tables have been structured into multiple columns such as Application Name, File Name, File Path, File Content, and lastly License Type, where the Content column explains the nature of the artifact, its evidentiary details, such as logs, identifiers, or user activity traces and the License Type column indicates whether the artifact was present in both free and paid versions, or exclusively in one variant.

4.1.1 TeamViewer

TeamViewer application was subjected to detailed forensic analysis across four major operating systems, such as Android, Windows, iOS, and macOS. Both installation and post-uninstallation states were examined to identify the persistent artifacts. The application was tested in its free and licensed variants to evaluate any differences in artifact generation and retention. The results are organized platform-wise, presenting key evidence, including logs, configuration files, system traces, residual data, etc.

The results related to the analysis of the TeamViewer application with regard to different operating systems and states are compiled as follows:

4.1.1.1 Android

4.1.1.1.1 Installation State

The artifacts were extracted after installation and active use of the TeamViewer application. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4a.

Table 4a: Represents the digital artifacts recovered from the installation state of TeamViewer on an Android device, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
TeamViewer Quick Support	base.apk	app\com.teamviewer.quicksupport.marketwJnPZ5wPZss0SSq4CVvw==\base.apk	Contains application version details and desired permissions.	Free & Paid
	TVLog.html	data\com.teamviewer.quicksupport.market\files\TVLog.html & media\0\Android\data\com.teamviewer.quicksupport.market\files\export\files.zip\TVLog.html	Records ClientID, ClientIC (Client Identification Code), IP addresses of both devices, timestamps, network type, TeamViewer IDs with roles, chat/message metadata, device-specific data (model, release, brand, display, fingerprint, etc.), application version details of host/client, screenshot requests, uninstallation activities (with names of uninstalled apps), host Username, and details of shared files (name and path). (Figure 2)	Free & Paid
	com.teamviewer.quicksupport.market_p	data\com.teamviewer.quicksupport.market\shared_prefs\com.teamviewer.quicksupport.	Stores user preferences, including app version, EULA acceptance, onboarding completion, and UI	Free & Paid

	references.xml	market_preferences.xml	settings, reflecting active app usage.	
	Screenshot	media\0\DCIM\Screenshots\	Stores Screenshots captured via TeamViewer.	Free & Paid
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with package name and associated email ID.	Free & Paid
TeamViewer Host	Base.apk	app\com.teamviewer.host.market-NEfaOi31vU5DXZOtxGcLEg==\base.apk	Contains application version details and desired permissions.	Free & Paid
	TVLog.html	data\com.teamviewer.host.market\files\TVLog.html & media\0\Android\data\com.teamviewer.host.market\files\export\files.zip\TVLog.html	Records timestamps, public IP address of Windows system, ClientID, Application Versions, network type, TeamViewer IDs of both systems with their respective roles, device-specific Android details (model, release, brand, fingerprint, etc.), version details of both devices, and Username of Client/Host. (Figure 3)	Free & Paid
	global.conf	data\com.teamviewer.host.market\files\global.conf	Stores cryptographic keys, client IDs, and session settings, indicating installation, version, and configuration of TeamViewer Host.	Free & Paid
	client.conf	data\com.teamviewer.host.market\files\client.conf	Stores the Username of the connected Windows device.	Free & Paid
	rolloutfile.tv13	data\com.teamviewer.host.market\files\rolloutfile.tv13	Stores the TeamViewer ID of the Android device.	Free & Paid
	com.teamviewer.host.market_preferences.xml	data\com.teamviewer.host.market\shared_prefs\com.teamviewer.host.market_preferences.xml	Stores user preferences such as app version, onboarding completion, and UI settings, showing active usage of TeamViewer Host.	Free & Paid
	FirebaseHeartBeatW0RFRkFVTFR	data\com.teamviewer.host.market\shared_prefs\FirebaseHeartBeatW0RFRkFV	Records the last usage date of the application.	Free & Paid

	d+MTox MDA3Nj k2ODc5 MDU5O mFuZHJ vaWQ6Z DhhYzE 3Y2ZjZ mM3ZjI1 Ng.xml	TFRd+MToxMDA 3Njk2ODc5MDU5 OmFuZHJvaWQ6Z DhhYzE3Y2ZjZm M3ZjI1Ng.xml		
	library.db	data\com.android.v ending\databases\li brary.db	SQLite database confirming installation via the Play Store with package name and associated email ID.	Free & Paid
TeamVie wer full client	base.apk	data\app\com.teamv iewer.teamviewer.m arket.mobile-bSBC4or-jsQbz7I2gKVMcg= =\base.apk	Contains application version details and desired permissions.	Free & Paid
	TVLog.ht ml	data\com.teamview er.teamviewer.mark et.mobile\files\TVL og.html	Records all TeamViewer session activities, including chat metadata such as identifiers, timestamps, and chat room IDs, but does not store the actual content of chat messages. It also records device information (model, serial number, OS, TeamViewer version), public IP of receiver device, Black Screen and Lock Screen commands, TeamViewer IDs of both devices, timestamps, and file transfer histories (source and destination paths). Excludes dynamic TeamViewer passwords, associated email addresses, and IP addresses of the Android device.	Free & Paid
	client.con f	data\com.teamview er.teamviewer.mark et.mobile\files\clien t.conf	Contains configuration details, such as BuddyDisplayName (display name), BuddyLoginName (email), BuddyLoginTokenID (unique session token), encrypted login tokens, SUID (system user ID), and UserAccountID. (Figure 4)	Free & Paid
	global.co nf	data\com.teamview er.teamviewer.mark	Stores cryptographic keys, client IDs, and session settings,	Free & Paid

		et.mobile\files\global.conf	indicating installation, version, and configuration.	
	Connections.txt	data\com.teamviewer.teamviewer.market.mobile\files\connection.txt	Logs outgoing connection data, including TeamViewer ID of connected device, start and end timestamps, Client Display name and make/model, and connection type/action (RemoteControl/FileTransfer). (Figure 5)	Free & Paid
	TVLogOld.html	data\media\0\Android\data\com.teamviewer.teamviewer.market.mobile\files\export\files.zip\TVLogOld.html	Records similar artifacts as TVLog.html but additionally includes audio call metadata (timestamps). Does not include video call artifacts.	Paid Only
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with the package name and associated email ID.	Free & Paid

4.1.1.1.2 Uninstallation State

After uninstalling the three TeamViewer applications (TeamViewer QuickSupport, TeamViewer Host, and TeamViewer Full Client) in both free and licensed versions on Android, only minimal traces were found stored. The primary remnants were the library.db file from the Google Play Store, which records installation details of uninstalled applications, and screenshots saved in the device's media folders. No logs, configuration files, or user data were recovered, indicating that TeamViewer uninstalls relatively cleanly on Android devices.

4.1.1.1.3 Interpretation and Forensic Significance

The forensic analysis of TeamViewer on Android in both installation and uninstallation states revealed a clear evidentiary pattern, which is highly relevant to investigations of unauthorized access through Remote Access Applications.

During the installation state, rich forensic artifacts were recovered across all three variants (TeamViewer QuickSupport, TeamViewer Host, and TeamViewer Full Client). The base.apk confirmed the application version and desired permissions, establishing proof of installation, version, package name, and potential functional capabilities. The *TVLog.html* files emerged as the most significant source of evidence, as this file logs

comprehensive session details, including ClientID, ClientIC, IP addresses of both devices, timestamps, network types, TeamViewer IDs with roles, chat metadata, device specifics (e.g., model, OS, fingerprint), screenshot requests, uninstallation commands, host usernames, and file transfer details (e.g., names, paths, sizes). These logs enable investigators to reconstruct the sequence of remote connections, identify the devices involved, and detect suspicious activities such as unauthorized file sharing or device remote access.

Configuration files like *global.conf*, *client.conf*, and *rolloutfile.tv13* provided persistent identifiers, cryptographic keys, and usernames that are critical for attributing activities to specific accounts and linking them to the suspect or attacker. Screenshots captured during sessions added a layer of direct evidence, showing what was accessed or viewed on the victim's device, while *market_preferences.xml* and *FirebaseHeartBeat.xml* corroborated app usage timelines. Together, these artifacts formed a strong evidentiary base for proving malicious exploitation of TeamViewer in unauthorized access cases.

In the uninstallation state, only a limited set of remnants persisted, most notably the *library.db* file from the Google Play Store and screenshots stored in the media directories. The *library.db* file is forensically significant because it recorded the package names and installation timestamps of uninstalled applications, thereby proving the historical presence of TeamViewer on the device even after its removal. This is particularly useful in countering denial claims by suspects who argue that the application was never installed. Screenshots left behind further strengthened the chain of evidence by visually linking the device to prior remote access sessions, even when logs and configuration files are no longer present.

From a forensic perspective, this dual-state analysis underscores two critical points: first, the installation state of TeamViewer generates abundant, high-value artifacts that allow detailed reconstruction of unauthorized access incidents; second, even after uninstallation, residual traces such as *library.db* entries and screenshots can serve as crucial proof of prior application usage. Therefore, the combination of active and residual artifacts provides investigators with both direct and circumstantial evidence to establish unauthorized access, link sessions to specific devices and accounts, and support the admissibility of findings in legal proceedings (Soni et al., 2024a).



Figure 2: TVLog.html file associated with the TeamViewer QuickSupport application.

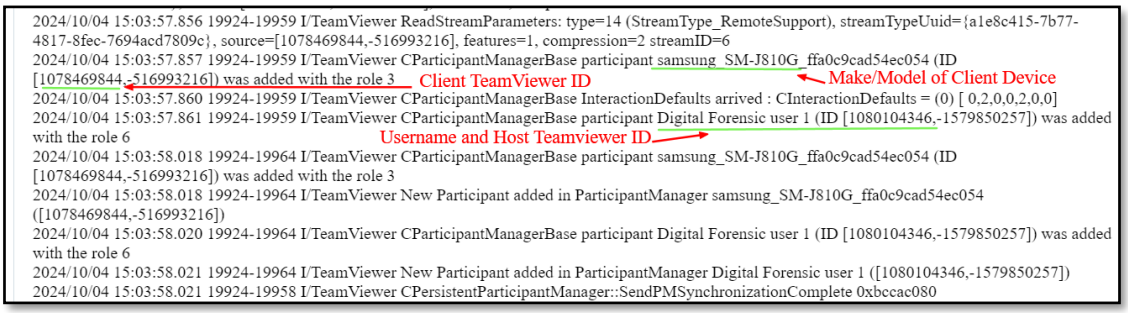


Figure 3: TVLog.html file associated with the TeamViewer Host application.



Figure 4: client.conf file containing configuration and identity-related artifacts.

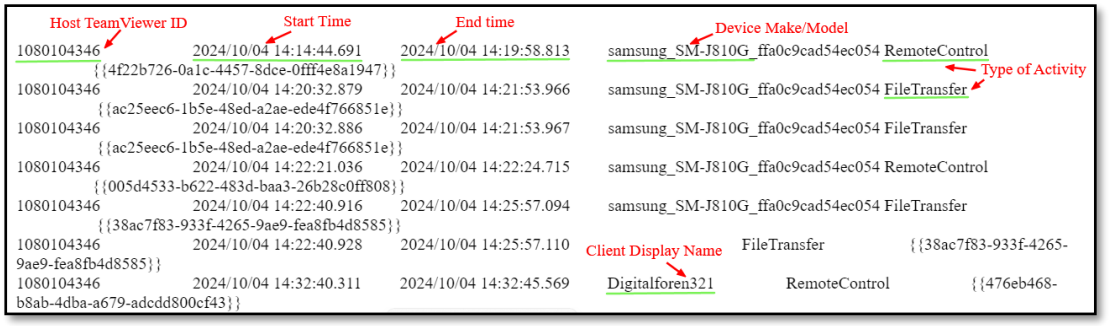


Figure 5: Connection.txt file containing connection and user details.

4.1.1.2 Windows

4.1.1.2.1 Installation

The artifacts were extracted after installation and active use of the TeamViewer application. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4b.

Table 4b: Represents the digital artifacts recovered from the installation state of TeamViewer on Windows, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
TeamViewer full client	rolloutfile.tv13	Program Files\TeamViewer\rolloutfile.tv13	Stores the TeamViewer ID associated with the corresponding Windows user.	Free & Paid
	Connections.txt	Program Files\TeamViewer\Connections.txt	Logs outgoing connection details, including Receiver ID, start and end timestamps, current Windows Username, and connection type or action performed (RemoteControl/FileTransfer).	Free & Paid
	TeamViewer15_Hooks.log	Program Files\TeamViewer\TeamViewer15_Hooks.log	Contains timestamped entries related to driver installation, including initiation, completion, parameters, and file paths, useful for installation timeline reconstruction.	Free & Paid
	TeamViewer15_Logfile.log	Program Files\TeamViewer\TeamViewer15_Logfile.log	Records TeamViewer IDs of both devices, email addresses, application versions, IP addresses, and activity timestamps. Includes device-specific details (Android model, serial number, screen resolution, CPU info, OS version, language settings). Logs file transfer details (name, size, source,	Free & Paid

			destination), host username, network configuration, notifications, and session commands. However, certain actions (e.g., screenshots, chat) are not captured, indicating partial activity logging. (Figure 6)	
	Connections.txt	Users\<Username>\AppData\Roaming\TeamViewer\Connections.txt	Logs Receiver ID (Host TeamViewer ID), connection timestamps, Windows Username, and Connection details.	Free & Paid
	<targetID>.tvc	Users\<Username>\AppData\Roaming\TeamViewer\MRU\RemoteSupport\<targetID>.tvc	Contains details of recent remote sessions, including Receiver ID (Host TeamViewer ID), and connection timestamps.	Free & Paid
	Prefetch Files (Folder)	Windows\Prefetch	Prefetch entries log execution metadata such as run counts, last run time, other run timestamps, and MAC (Modification, Access, Creation) details. (Figure 7)	Free & Paid
	Window event logs	Windows\System32\winevt\Logs	Windows Event Log entry (System.evtx (Figure 8), Security.evtx) records details of user accounts (Security ID, Account Name, Domain, Logon ID) and related process execution, including Process ID and full process path.	Free & Paid
	samsung_SM-J810G_ffa0c9cad54ec054 (1078 117 778)_2024-10-09 15.09.08.tvs	User-defined location (variable)	Video screen recording of a remote session, providing direct visual evidence of on-screen activities. The name of the file encodes device make and model, UUID, Host (Receiver) TeamViewer ID, and timestamp. Save location is user-defined, making it valuable but potentially difficult to locate in forensic acquisition.	Paid Only

4.1.1.2.2 Uninstallation State

All the previously identified artifacts were found to persist even after the uninstallation of TeamViewer from the Windows system. The files were successfully recovered from their respective directories, indicating that the uninstallation process did not completely remove user-specific or system-level data. While the artifact content remained consistent, the hash values of a few files were found to be different, likely due to minor changes in metadata or timestamp updates during system operations.

4.1.1.2.3 Interpretation and Forensic Significance

The forensic analysis of TeamViewer on Windows in its installation state revealed multiple artifacts critical for digital investigations. The rolloutfile.tv13 file serves as a persistent identifier, storing the TeamViewer ID linked to the Windows user, enabling attribution of specific sessions to a system. Connections.txt files, found in both the Program Files and AppData\Roaming directories, log Receiver IDs, session timestamps, usernames, and connection actions, providing cross-verification points.

The TeamViewer15_Hooks.log aids timeline reconstruction by recording driver installation events, while TeamViewer15_Logfile.log captures extensive activity, including device IDs, email addresses, IPs, file transfers, system configurations, and session commands. User-specific <targetID>.tvc files record remote support sessions, linking Receiver IDs with timestamps and usernames. Prefetch files and Windows Event Logs (System.evtx and Security.evtx) offer additional temporal and contextual evidence.

A video session recording (tvs) provides direct visual evidence of remote activities, embedding metadata such as device model, UUID, and TeamViewer IDs in its file name. This artifact is limited to licensed versions and may be stored in user-defined locations, making it less accessible.

Importantly, all artifacts persisted after uninstallation. Although minor hash changes occurred, core content remained intact, demonstrating that uninstalling TeamViewer does not remove forensic traces.

Overall, TeamViewer generates evidence of the user, application, and system levels. The recovered artifacts enable investigators to establish session timelines, identify

participants, reconstruct activities, and correlate application usage with broader system events. Their persistence further strengthens their forensic value, ensuring recoverability even when attempts are made to conceal usage (Soni et al., 2024a).

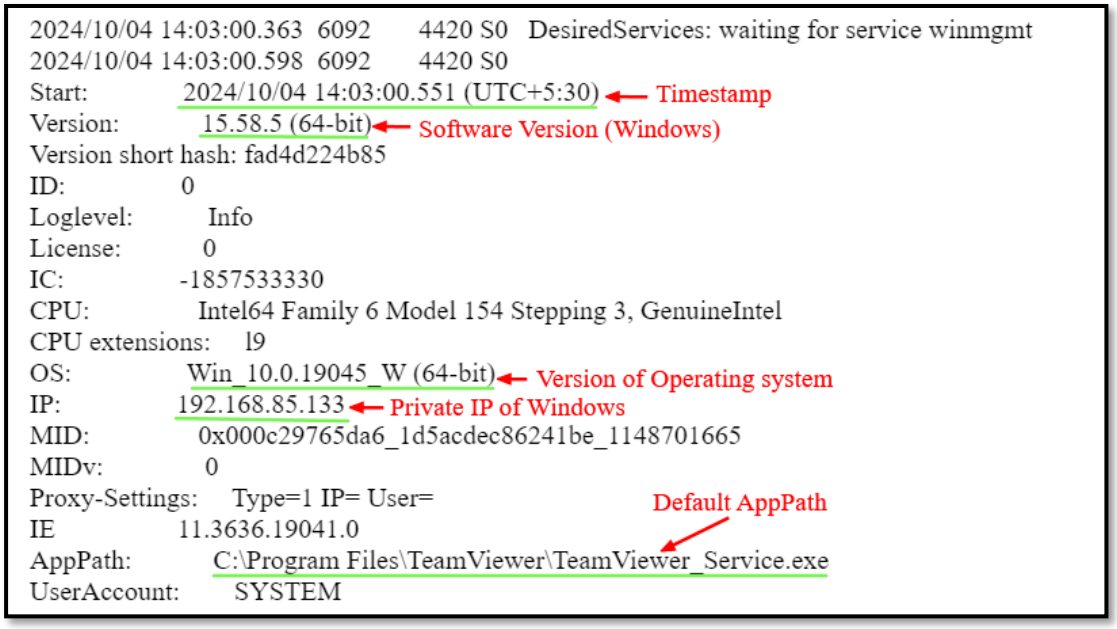


Figure 6: Represents the *TeamViewer15_Logfile.log* file, highlighting artifacts such as timestamps, software version, operating system version, private IP of the Windows system, and the default AppPath of the TeamViewer full client software.

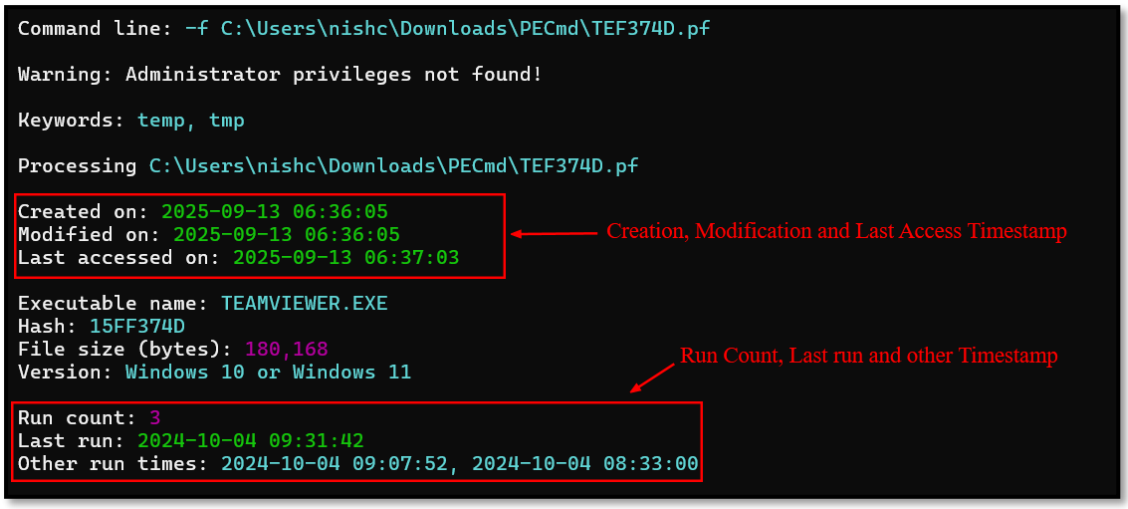


Figure 7: Represents the Prefetch file, highlighting artifacts such as creation, modification, and last access timestamps, along with run count and other active timestamps.

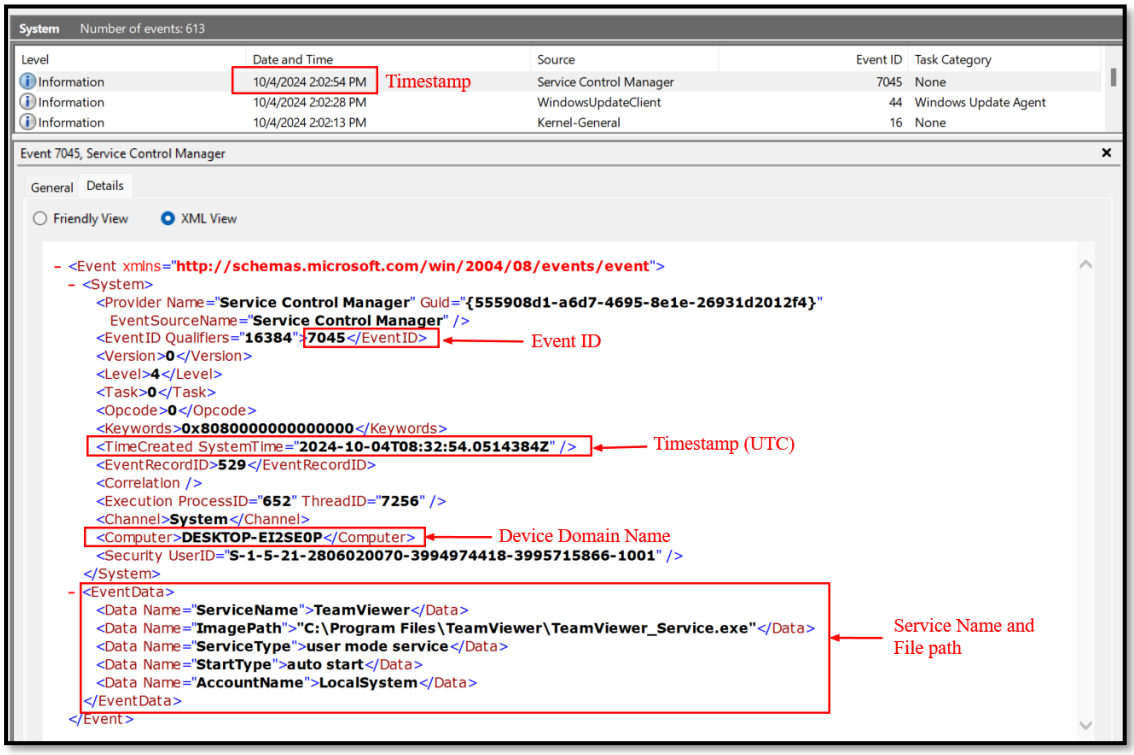


Figure 8: Represents the *System.evtx* file, highlighting artifacts such as event creation timestamps, along with Event ID, Service name, and associated File path.

4.1.1.3 iOS

4.1.1.3.1 Installation

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4c.

Table 4c: Represents the digital artifacts recovered from the installation state of TeamViewer on iOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
TeamViewer QuickSupport	kSecClassGenericPassword	/Keychain/kSecClassGenericPassword	Stores TeamViewer's encryption key (TVRCCoreEncryptionKey), user identifier (tv_uuid_account), app identifier, and key creation time. Confirms application usage, account linkage, and secure credential storage. **Note:** The Licensed version of this file contains the email address of the iOS Device, too. (Figure 9)	Free & Paid
	app.state	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/io.sentry/1f73e3a3e3f26fb44089b4115ec526f7e5110752/app.state	Records application state metadata, including SDK status, termination flags, vendor ID (UUID), system boot timestamp, OS and release version, debugging flags, and ANR status. Useful for verifying app activity and usage patterns.	Free & Paid
	com.apple.mobile_container_manager.metadata.plist	/private/var/mobile/Containers/Data/Application/<UUID>/.com.apple.mobile_container_manager.metadata.plist	This file helps to confirm the presence and installation of the TeamViewer QuickSupport application on the device.	Free & Paid

	1738316317.498902-00003-94E0EAD7-4B19-42BE-B345-FF10BDE43D8D.json	/private/var/mobile/Containers/Data/Application/<UUID>/tmp/NSIRD_QuickSupport_NDGrew/1738316317.498902-00003-94E0EAD7-4B19-42BE-B345-FF10BDE43D8D.json	Records session data: session start/end time, duration, device identifier (DID), session ID (SID), release version, and environment.	Free & Paid
	breadcrumbs.2.state	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/io.sentry/1f73e3a3e3f26fb44089b4115ec526f7e5110752/breadcrumbs.2.state	Logs UI navigation events, user interactions (like button taps), device orientation changes, battery state, and app lifecycle transitions (foreground/background).	Free & Paid
	com.TeamViewer.Settings.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.TeamViewer.Settings.plist	Contains clientIC, clientID, and installed app version.	Free & Paid
	Connections_incoming.txt	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Logs/TeamViewer/Connections_incoming.txt	Logs incoming sessions: remote username, Controller TeamViewer ID, timestamps, session type (e.g., RemoteControl), and session identifier.	Free & Paid
	previous.app.state	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/io.sentry/1f73e3a3e3f26fb44089b4115ec526f7e5110752/previous.app.state	Records application state metadata, including SDK status, termination flags, vendor ID (UUID), system boot timestamp, OS and release version, debugging flags, and ANR status. Useful for verifying app activity and usage patterns.	Free & Paid
	TeamViewer15_Logfile.log	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Logs/TeamViewer/Team	Logs TeamViewer IDs of both devices, app versions, public IPs, private IP of connected device (Host), VoIP call data, timestamps, and device details (model,	Free & Paid

		Viewer15_Logfile.log	serial, resolution, CPU, OS, language, host username, notification of clipboard and network configuration). Does not include screenshots, chat, email, private IP of Client, or file transfer data.	
	iTunesMetadata.plist	/private/var/containers/Bundle/Application/<UUID>/iTunesMetadata.plist	Contains App Store metadata: Apple ID, item name, version, source app, purchase date, and dsPersonalID. (Figure 10)	Free & Paid
	Screenshot	/private/var/mobile/Media/DCIM/100APPLE	Screenshots of the iOS screen, captured through TeamViewer.	Free & Paid
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records installation/uninstallation events with timestamps, file paths, and package names.	Free & Paid
TeamViewer Remote Control	app.state	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/io.sentry/25fcc51e81e693c7aafc6c7bdd8fd0e333abe504/app.state	Contains application state metadata: SDK working status, termination flag, vendor ID, boot timestamp, OS version, release version, debugging flag, and ANR status.	Free & Paid
	breadcrumbs.2.state	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/io.sentry/25fcc51e81e693c7aafc6c7bdd8fd0e333abe504/breadcrumbs.2.state	Logs UI navigation, user actions, orientation changes, battery state, and lifecycle transitions.	Free & Paid
	com.TeamViewer.Settings.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.TeamViewer.Settings.plist	Contains clientIC, ClientID assigned to the iOS device, BuddyDisplayName, BuddyLoginName (email), app version, and TeamViewer ID of connected device.	Free & Paid
	Connections.txt	/private/var/mobile/Containers/Data/Applic	The file logs details such as the Receiver ID, Start	Free & Paid

		ation/ <UUID>/Library/Logs/TeamViewer/Connections.txt	Date, Start Time, End Date, End Time, and Connection Type or action.	
	session.current	/private/var/mobile/Containers/Data/Application/ <UUID>/Library/Caches/io.sentry/25fcc51e81e693c7aafc6c7bdd8fd0e333abe504/session.current	Records active session details: session ID, device ID, release version, environment, and session sequence.	Free & Paid
	TeamViewer15_Logfile.log	/private/var/mobile/Containers/Data/Application/ <UUID>/Library/Logs/TeamViewer/TeamViewer15_Logfile.log	Logs TeamViewer IDs of both devices, app versions, IPs, file transfers (path and size), timestamps, and device data (model, serial, resolution, CPU, OS, language, host username, clipboard events, network config). Includes connected device info (name, resolution, duration). Excludes screenshots, chat, and email.	Free & Paid
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records installation/uninstallation events with timestamps, file paths, and package names. (Figure 11)	Free & Paid

4.1.1.3.2 Uninstallation State

After uninstalling the TeamViewer apps (Free and Paid) on iOS, only a single residual file named mobile_installation.log.0 was identified. This file captures detailed information related to the installation and uninstallation processes of applications on iOS. Specifically, it records the exact timestamp of each event, the file path of the application package, and its corresponding package name. Although no additional user data or configuration files were found after uninstallation, the presence of mobile_installation.log.0 stands as a crucial forensic artifact for confirming the historical existence of the TeamViewer app on the device.

4.1.1.3.3 Interpretation and Forensic Significance

The forensic analysis of TeamViewer on iOS in its installation state revealed multiple artifacts of substantial evidentiary value. Keychain entries (kSecClassGenericPassword) are critical, storing the TVRCCoreEncryptionKey, user identifiers, app metadata, and the user's email address in licensed versions, linking activity to a specific account. Application state files (app.state, previous.app.state, session.current, breadcrumbs.2.state) capture contextual details such as vendor IDs, device orientation, system status, session identifiers, and user interactions, enabling reconstruction of application usage over time.

System-level files like com.apple.mobile_container_manager.metadata.plist and iTunesMetadata.plist confirm installation history and app source. Settings and connection logs (com.TeamViewer.Settings.plist, Connections.txt, Connections_incoming.txt) provide persistent evidence of client IDs, session timestamps, and user details, supporting correlation of remote activities with specific devices and accounts.

The TeamViewer15_Logfile.log is a comprehensive artifact, recording TeamViewer IDs, IPs, device data, session details, file transfers, and VoIP call activity, though screenshots, chat logs, and email identifiers are not captured. Screenshots recovered from the DCIM directory offer direct visual evidence of remote sessions.

After uninstallation, nearly all artifacts were removed, leaving only mobile_installation.log.0, which records installation and uninstallation timestamps, file paths, and package names, providing historical verification. The absence of chat, email, or dynamic password data indicates that iOS enforces stronger data protection, requiring investigators to rely on configuration files, state logs, and system metadata for forensic analysis.



Figure 9: Represents the *kSecClassGenericPassword* file, highlighting artifacts such as creation/modification timestamp and Email address of the TeamViewer client account.

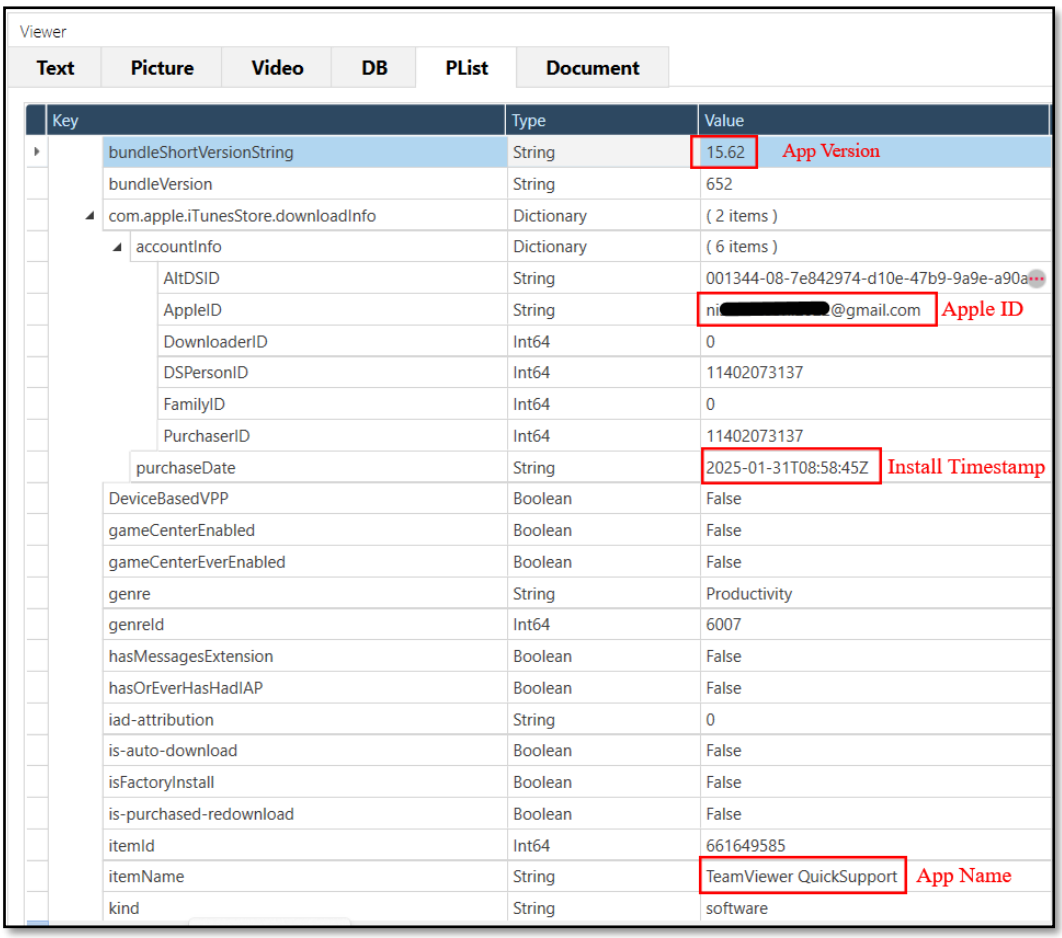


Figure 10: Represents the *iTunesMetadata.plist* file, highlighting artifacts such as App Version, Name, Apple ID, and Install timestamp (UTC).



Figure 11: Represents the *mobile_installation.log.0*, highlighting artifacts such as uninstall request timestamp and logs of uninstallation.

4.1.1.4 macOS

4.1.1.4.1 Installation

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4d.

Table 4d: Represents the digital artifacts recovered from the installation state of TeamViewer on macOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
TeamViewer full client	Info.plist	Applications\TeamViewer.app\Contents\Info.plist & Library\Application Support\TeamViewer\AuthorizationResources.bundle\Contents\Info.plist	Contains application metadata: CFBundleName, CFBundleVersion, and CFBundleIdentifier. Confirms the exact application installed and version on macOS device.	Free & Paid
	com.teamviewer.teamviewer.preferences.plist	Users\<username>\Library\Preferences\com.teamviewer.teamviewer.preferences.plist	Stores user-specific data: Meeting_UserName (macOS username), BuddyLoginTokenID (session/user token), recently connected TeamViewer ID, buddydisplayname, and buddyloginname (name and email used for login). Confirms user identity, login history, and recent connections. (Figure 12)	Free & Paid
	com.teamviewer.teamviewer.preferences.Machine.plist	Users\<username>\Library\Preferences\com.teamviewer.teamviewer.preferences.Machine.plist	Stores machine-specific identifiers: clientID, clientIC, and application version. Used to correlate sessions and link the device to TeamViewer usage.	Free & Paid

	install.log	private\var\log\install.log	Records timestamped installation details, including package file paths. Confirms the timeline of TeamViewer installation on the system.	Free & Paid
	com.apple.finder.plist	Users\<username>\Library\Preferences\com.apple.finder.plist	Shows FXRecentFolders entries referencing TeamViewer, confirming recent access via Finder. Also contains FXDesktopVolumePosition (desktop organization details).	Free & Paid
	TCC.db	Users\<username>\Library\Application Support\com.apple.TCC\TCC.db	Records permissions granted to TeamViewer: screen capture, full disk access, microphone, and camera. Confirms elevated system access.	Free & Paid
	LaunchDaemons (Folder)	Library\LaunchDaemons	Contain multiple .plist files with file paths, program identifiers, and program labels related to TeamViewer.	Free & Paid
	LaunchAgents(Folder)	Library\ LaunchAgents	Contains multiple .plist files with working directory, program arguments, watchpaths, and program labels related to TeamViewer.	Free & Paid
	Session name_2025-02-06 11.22.23.tvs	NA (user-defined save location)	Video session recording of remote connection (screen capture). Filename includes session timestamp. Present only in licensed versions.	Paid Only

4.1.1.4.2 Uninstallation State

After uninstalling TeamViewer on macOS, three key residual files and a screen recording were identified. The file com.teamviewer.teamviewer.preferences.plist contained user-specific forensic data such as the macOS username, session tokens, recently connected TeamViewer IDs (MRU list), and login credentials, including display name and email ID. The com.teamviewer.teamviewer.preferences.Machine.plist file contains machine-specific identifiers such as clientID, clientIC, and application version, which help link the device to its past TeamViewer usage. The install.log file provided timestamped entries related to the application's installation, confirming its presence and helping establish a timeline. Additionally, a screen recording generated through TeamViewer's premium feature was

discovered, which may serve as crucial evidence of remote activity, even after uninstallation.

4.1.1.4.3 Interpretation and Forensic Significance

The forensic examination of TeamViewer on macOS in the installation state revealed artifacts with strong investigative value at both the user and system levels. The Info.plist file confirms the installed version and build, supporting authenticity verification and correlation with version-specific features. User-specific preference files (com.teamviewer.teamviewer.preferences.plist) store macOS usernames, session tokens, recently connected TeamViewer IDs, and login details such as display name and email, establishing direct links to user accounts. Machine-level preference files (com.teamviewer.teamviewer.preferences.Machine.plist) record device identifiers like clientID and application version, enabling session attribution to specific devices. The install.log provides a timestamped record of installation, supporting timeline reconstruction.

System-level artifacts, including com.apple.finder.plist and TCC.db, document recent application access and granted permissions (screen capture, disk, microphone, camera), highlighting potential exposure during remote sessions. LaunchDaemons and LaunchAgents files confirm persistence mechanisms, clarifying how TeamViewer integrates with system startup. Session recording file (.tvs), capturing on-screen activity with embedded timestamps, offering direct visual evidence of remote interactions.

After uninstallation, key artifacts persist, including user and machine preference files, the install log, and session recordings. This persistence allows investigators to reconstruct prior activity, establish user-device linkage, and maintain session timelines even when the application is removed. While chat and file transfer logs are not retained, these artifacts collectively provide a robust basis for forensic investigations, emphasizing the need for cross-platform correlation when analyzing macOS systems.

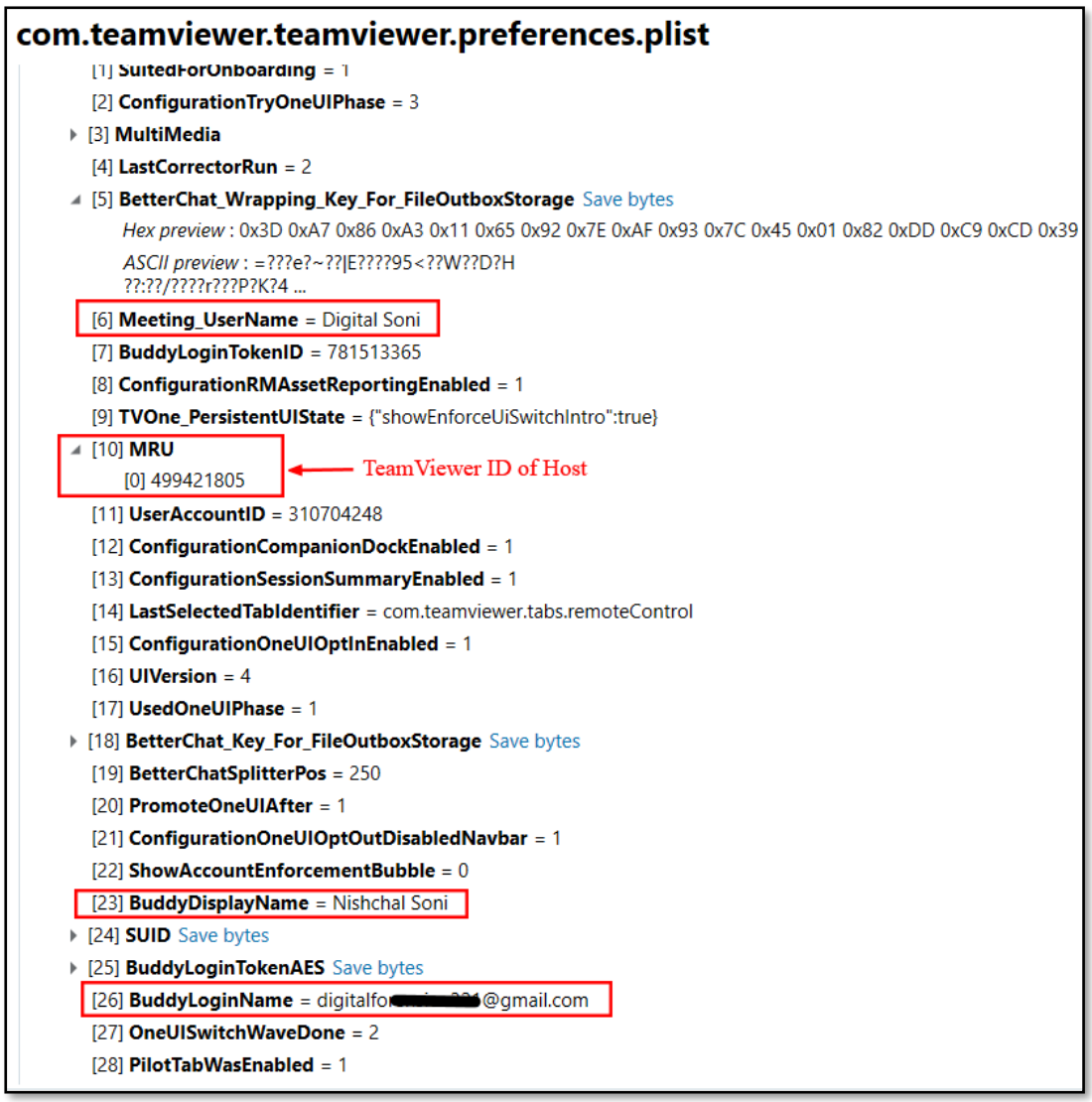


Figure 12: Represents the *com.teamviewer.teamviewer.preferences.plist* file, highlighting host device artifacts such as meeting username, TeamViewer ID, display name, and login email address.

4.1.2 LogMeIn (GoToMyPC)

GoToMyPC was subjected to detailed forensic analysis across four major operating systems: Android, Windows, iOS, and macOS. Both installation and post-uninstallation states were examined to identify persistent artifacts. The application was tested in its free and licensed variants to evaluate any differences in artifact generation and retention. The results are organized platform-wise, presenting key evidence, including but not limited to logs, configuration files, system traces, and residual data.

4.1.2.1 Android

4.1.2.1.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4e.

Table 4e: Represents the digital artifacts recovered from the installation state of GoToMyPC on Android, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
GoToMyPC	base.apk	app\com.gotomypc-izVWzLicAv3cy7Sz7Si59w==\base.apk	Contains application version details and desired permissions.	Free & Paid
	G2PPreferences.xml	data\com.gotomypc\shared_prefs\G2PPreferences.xml	Stores user-specific data: email address, plaintext password, biometric authentication flag (IS_BIOMETRICS_ENABLED), and password save prompt status. Confirms that login credentials and security preferences were saved on the device. (Figure 13)	Free & Paid

	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with the package name and associated email ID.	Free & Paid
	FirebaseHeartBeatW0RFRkFVTFRd+MToxMjIxMjUwNDgyMjU6YW5kcm9pZDplYjRkN2JiZTdiZWVhMjkzZThkM2Uw.xml	data\com.gotomypc\shared_prefs\FirebaseHeartBeatW0RFRkFVTFRd+MToxMjIxMjUwNDgyMjU6YW5kcm9pZDplYjRkN2JiZTdiZWVhMjkzZThkM2Uw.xml	Stores the application's last-used date.	Free & Paid

4.1.2.1.2 Uninstallation State

Upon uninstallation of the GoToMyPC application from the Android device, only the library.db file was found to persist. This file, associated with the Google Play Store, retains metadata about previously installed applications, including package names. Although other application-specific files were removed during uninstallation, the presence of library.db provides a valuable forensic artifact that confirms the prior existence of the GoToMyPC app on the device and helps establish a timeline of its usage.

4.1.2.1.3 Interpretation and Forensic Significance

The forensic analysis of GoToMyPC on Android in the installation state revealed artifacts with high investigative relevance. The base.apk confirmed the application version and desired permissions, establishing proof of installation and potential functional capabilities. The G2PPreferences.xml file contained sensitive user-specific data, including registered email, plaintext password, biometric settings, and password save status, directly linking the device to the account and providing crucial evidence of user identity and login behavior.

The library.db file, maintained by the Google Play Store, recorded installation details, package names, and timestamps, supporting timeline reconstruction and verification of

application presence. FirebaseHeartBeat.xml preserved the last usage date, helping situate activity within a specific timeframe. Together, these artifacts provide insights into application configuration and user identity, making them valuable for investigations of unauthorized remote access.

After uninstallation, sensitive user data was removed, but library.db persisted, allowing investigators to confirm prior installation even without user-level information. While session logs, IP addresses, or remote activity records were absent, the remaining artifacts still offer strong corroborative evidence for forensic investigations.

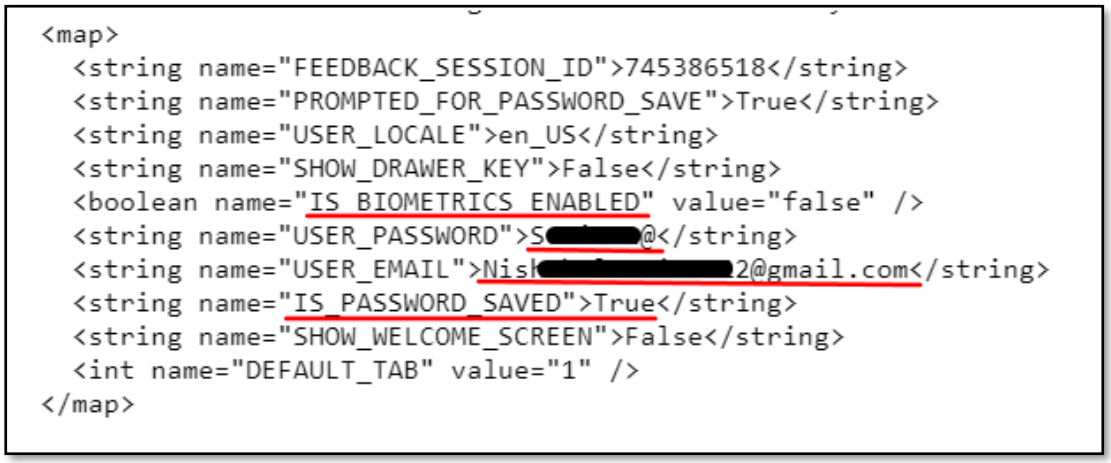


Figure 13: Represents the *G2PPreferences.xml* file, highlighting user-specific data such as email address, plaintext password, biometric authentication flag (IS_BIOMETRICS_ENABLED), and password save prompt status.

4.1.2.2 Window

4.1.2.2.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4f.

Table 4f: Represents the digital artifacts recovered from the installation state of GoToMyPC on Windows, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
GoToMyPC	unattended.json	Program Files (x86)\GoToMyPC WebAccess\75726 67978943005684\ unattended.json	Stores configuration details: service URLs (website, base, monitoring, session backend, data provider), company ID, product type, plan (Personal), application type, session type (Unattended), environment (Production), hostname, session ID, and service name assigned at installation.	Free & Paid
	GoToMyPCUnattended.log	Program Files (x86)\GoToMyPC WebAccess\75726 67978943005684\appdata\GoToMyPCUnattended.log	Contains extensive system (Windows) metadata: platform, OS details, hostname, email address, device name, private IP address, MAC address, system model, manufacturer, processor, BIOS version, memory, time zone, and last logged-in username. Also logs locale (India), unique connectionId, and companyId. Provides a detailed profile of the host machine. (Figure 14)	Free & Paid
	g2host_00.log	ProgramData\LogMeInLogs\GoToM	Records timestamped activity records (date, time recorded in	Free & Paid

		yPC\4266\2025-02-20_13.45.01.226\g2host_00.log	PST). Reveals the Windows username and desktop name of the device.	
	Prefetch Files (Folder)	Windows\Prefetch	Records execution details of GoToMyPC: run counts, last run, other run times, and MAC details. Useful to establish the frequency and timeline of application use.	Free & Paid
	Login Data	Users\<Username>\AppData\Local\Microsoft\Edge\User Data\Default\Login Data	Stores the package name related to GoToMyPC activity in the Edge browser profile.	Free & Paid
	Window event logs	Windows\System32\winevt\Logs	Windows Event Log entry (System.evtx) contains details of the user account (Security ID, Account Name, Login Time, Event ID, Computer Name) and the associated process execution, including Process ID and full Process Name path.	Free & Paid

4.1.2.2.2 Uninstallation State

Upon uninstallation of the GoToMyPC application from the Windows device, only the Prefetch Files, Login Data, and Windows event log files were found to persist. These files can help forensic experts to know the configuration and installation timestamps. Although other application-specific files were removed during uninstallation, the presence of these files provides a valuable forensic artifact that confirms the prior existence of the GoToMyPC app on the device and helps establish a timeline of its usage.

4.1.2.2.2 Interpretation and Forensic Significance

The forensic examination of GoToMyPC on Windows in the installation state revealed artifacts offering strong investigative value. The unattended.json file contained detailed configuration data, including service URLs, company ID, product type, session type,

and environment information, confirming operational modes such as unattended access and linking usage to organizational or account identifiers.

The GoToMyPCUnattended.log captured extensive system metadata, including hostname, email, device details, IP/MAC addresses, BIOS version, memory, processor, time zone, and the last logged-in username. This information enables complete system profiling and direct attribution of activity to a specific user. The g2host_00.log provided timestamped records of activity, while Prefetch files documented execution timestamps and frequency, supporting timeline reconstruction. At the browser level, the Login Data artifact connected GoToMyPC usage to Edge browser activity. Windows Event Logs (System.evtx) offered corroborative evidence of account activity and process execution.

After uninstallation, critical files like unattended.json, GoToMyPCUnattended.log, and g2host_00.log were removed, but residual artifacts in Prefetch, Login Data, and Event Logs persisted, allowing confirmation of prior installation and usage with timestamps and execution metadata.

No chat logs, session recordings, or detailed remote interaction data were recovered, limiting full reconstruction of remote activity. Overall, GoToMyPC on Windows leaves a robust forensic footprint through configuration files, system logs, and prefetch data, with residual artifacts after uninstallation enhancing investigative potential.



Figure 14: Represents the *GoToMyPCUnattended.log* file, highlighting Windows metadata, including but not limited to platform, OS details, hostname, device name, private IP address, MAC address, system model, manufacturer, processor, BIOS version, memory, and time zone.

4.1.2.3 iOS

4.1.2.3.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4g.

Table 4g: Represents the digital artifacts recovered from the installation state of GoToMyPC on iOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
GoToMyPC	kSecClassGenericPassword	/Keychain/kSecClassGenericPassword	Stores encryption key, user identifier, app identifier, key creation time, and LastLoginPassword. Confirms app usage, account linkage, and credential storage.	Free & Paid
	metadata.clsrecord	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/com.crashlytics.data/com.logmein.ios.gotomypc/v5/reports/active/56b09f0ab6f44a7f8fabe990664bb804/metadata.clsrecord	Structured JSON records capturing session_id, app version (display and build), device model, codename, OS build, platform, locale, architecture (arm64), UUID, memory base address, binary size, and encryption status. Links app usage to a specific device and environment. (Figure 15)	Free & Paid
	com.logmein.ios.gotomypc.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.logmein.ios.gotomypc.plist	Contains user-related artifacts, including the LastActiveEmail address.	Free & Paid
	iTunesMetadata.plist	/private/var/containers/Bundle/Application/<UUID>/iTunesMetadata.plist	Confirms official App Store installation. Includes Apple ID, item name, artist name, purchase ID, item ID, bundle version, software version,	Free & Paid

			purchase date, and dsPersonalID.	
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records application installation and uninstallation logs, with timestamps, package path, and package name.	Free & Paid

4.1.2.3.2 Uninstallation State

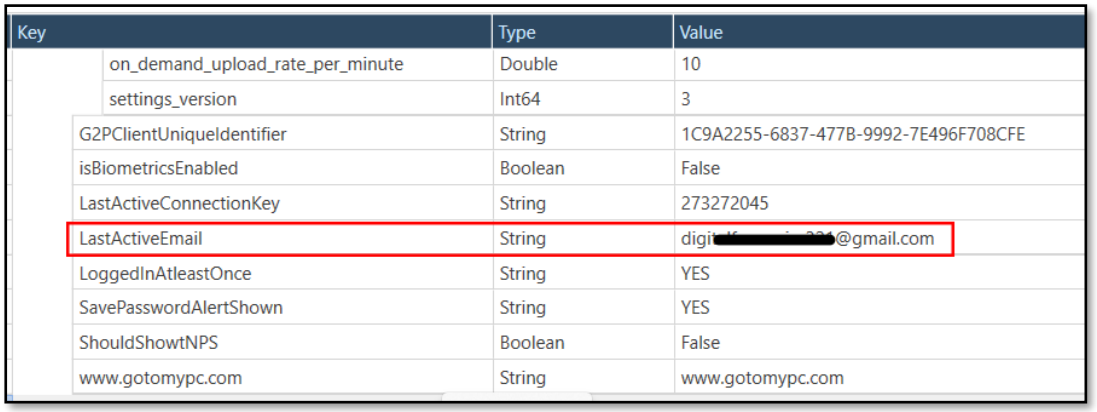
After uninstalling the GoToMyPC apps (Free and Paid) on iOS, only two residual file, named kSecClassGenericPassword and mobile_installation.log.0 were identified. mobile_installation.log.0 file captures detailed information related to the installation and uninstallation processes of applications on iOS. Specifically, it records the exact timestamp of each event, the file path of the application package, and its corresponding package name. Although no additional user data or configuration files were found after uninstallation, the presence of mobile_installation.log.0 stands as a crucial forensic artifact for confirming the historical existence of the TeamViewer app on the device.

4.1.2.3.3 Interpretation and Forensic Significance

The forensic analysis of GoToMyPC on iOS revealed artifacts that provide strong evidence of user activity and system-level traces. The kSecClassGenericPassword entry in the iOS Keychain stores encryption keys, unique user and app identifiers, key creation times, and the last login password, linking account credentials directly to GoToMyPC and supporting user attribution.

The metadata.clsrecord file contained structured JSON data capturing session IDs, app build, device model, OS build, locale, architecture, and memory addresses, providing contextual metadata for verifying the application's operational environment and supporting timeline integrity. com.logmein.ios.gotomypc.plist recorded user information, including the last active email, while iTunesMetadata.plist confirmed official App Store installation and captured Apple ID, purchase ID, and dsPersonalID, strengthening provenance. Installation and uninstallation events were logged in mobile_installation.log.0, documenting package paths, timestamps, and package names for reconstructing application timelines.

After uninstallation, only kSecClassGenericPassword and mobile_installation.log.0 persisted. The Keychain entry continued to store sensitive account data, while the installation log confirmed historical presence. Other user and session data were removed, limiting reconstruction of fine-grained remote interactions. In summary, GoToMyPC on iOS produces high-value forensic artifacts at the system and account level, with residual evidence persisting post-uninstallation, though session-level reconstruction remains limited compared to Windows.



Key	Type	Value
on_demand_upload_rate_per_minute	Double	10
settings_version	Int64	3
G2PClientUniqueIdentifier	String	1C9A2255-6837-477B-9992-7E496F708CFE
isBiometricsEnabled	Boolean	False
LastActiveConnectionKey	String	273272045
LastActiveEmail	String	digi[redacted]@gmail.com
LoggedInAtleastOnce	String	YES
SavePasswordAlertShown	String	YES
ShouldShowtNPS	Boolean	False
www.gotomypc.com	String	www.gotomypc.com

Figure 15: Represents the *com.logmein.ios.gotomypc.plist* file, highlighting the last active email used for the session.

4.1.2.4 macOS

4.1.2.4.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4h.

Table 4h: Represents the digital artifacts recovered from the installation state of GoToMyPC on macOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
GoToMyPC	Info.plist	Applications\GoToMyPC Starter.app\Contents\Info.plist **&** Library\Application Support\GoToMyPC\GoToMyPC.app\Contents\Info.plist **&** Users\<Username>\Library\Application Support\GoToOpener\GoToOpener.app\Contents\Info.plist	Stores application metadata including CFBundleName, CFBundleVersion, and CFBundleIdentifier. Confirms exact version installed and uniquely identifies the application, supporting correlation with system logs and activity records.	Free & Paid
	GoToOpener.log	Users\<Username>\Library\Logs\com.logmein.GoToOpener\GoToOpener.log	Contains installation logs with timestamps and process details, useful for confirming when the application was installed.	Free & Paid
	com.logmein.GoToMyPC.plist	Library\Preferences\com.logmein.GoToMyPC.plist	Stores configuration details, including EventID, Trial status, primary username, access code, email address, and Machine Name. Confirms account and device association. (Figure 16)	Free & Paid

	com.logmein.GoToMyPC.SystemStatusUIHost.plist	Users\<Username>\Library\Preferences\com.logmein.GoToMyPC.SystemStatusUIHost.plist	Records application launch count, supporting timeline reconstruction of usage.	Free & Paid
	com.logmein.mac.GoToOpener.plist	Users\<Username>\Library\Preferences\com.logmein.mac.GoToOpener.plist	The file contains the UUID of the GoToMyPC application.	Free & Paid
	install.log	private\var\log\install.log	Contains timestamped installation records with file paths of installation packages.	Free & Paid
	com.apple.finder.plist	Users\<Username>\Library\Preferences\com.apple.finder.plist	References to FXRecentFolders confirm recent access of GoToMyPC via Finder. FXDesktopVolumePosition indicates how the app was organized on the desktop, supporting user interaction evidence.	Free & Paid
	TCC.db	Users\<Username> Library\Application Support\com.apple.TCC\TCC.db	Records application permissions and access granted to system functionalities. Useful for assessing security and privacy implications of the app. (Figure 17)	Free & Paid
	LaunchDaemons (Folder)	Library\LaunchDaemons	Contains .plist files related to GoToMyPC, including file paths, program identifiers, labels, and KeepAlive status. Confirms persistence and auto-launch configuration.	Free & Paid
	LaunchAgents(Folder)	Library\ LaunchAgents	Contains .plist files related to GoToMyPC, including file paths, program labels, descriptions, and KeepAlive status. Supports analysis of auto-start and background activity.	Free & Paid

4.1.2.4.2 Uninstallation State

After uninstalling GoToMyPC on macOS, all the previously identified artifacts remained present on the system, indicating that the uninstallation process does not completely remove related configuration, log, and preference files. The only exception was the Info.plist file located at Applications\GoToMyPC, which was removed as part of the uninstallation. This persistence of residual artifacts, including user account details, configuration settings, and installation logs, underscores the potential for post-uninstallation forensic recovery and analysis.

4.1.2.4.3 Interpretation and Forensic Significance

The forensic value of GoToMyPC artifacts on macOS is substantial. Info.plist files confirm application presence, version, and build, which are essential for verifying authenticity and correlating activity with system events. GoToOpener.log and install.log provide timestamped installation details, supporting timeline reconstruction and application provenance verification.

User-related artifacts such as com.logmein.GoToMyPC.plist, com.logmein.GoToMyPC.SystemStatusUIHost.plist, and com.logmein.mac.GoToOpener.plist allows investigators to link accounts to devices through usernames, email addresses, access codes, UUIDs, and launch counts, supporting direct user attribution and assessment of usage frequency. System-level artifacts, including com.apple.finder.plist and TCC.db, reveal interactions and granted permissions (screen, microphone). Persistence files in LaunchDaemons and LaunchAgents confirm background execution and automatic startup, providing insight into ongoing application activity.

Most of these artifacts remain after uninstallation, ensuring investigators can reconstruct historical usage, confirm prior installation, and link users and devices, even after software removal. The broad range of surviving artifacts on macOS makes it particularly valuable for forensic reconstruction compared to other platforms.

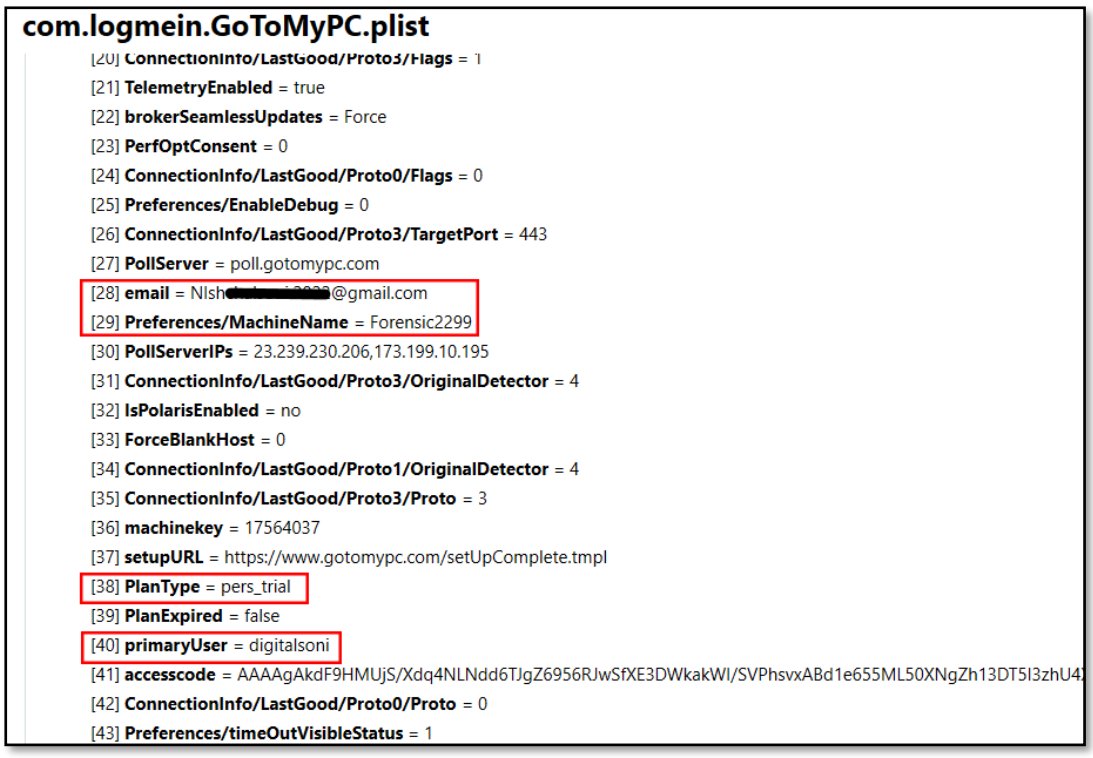


Figure 16: Represents the *com.logmein.GoToMyPC.plist* file, highlighting artifacts such as email address, machine name, PlanType, and primary username.

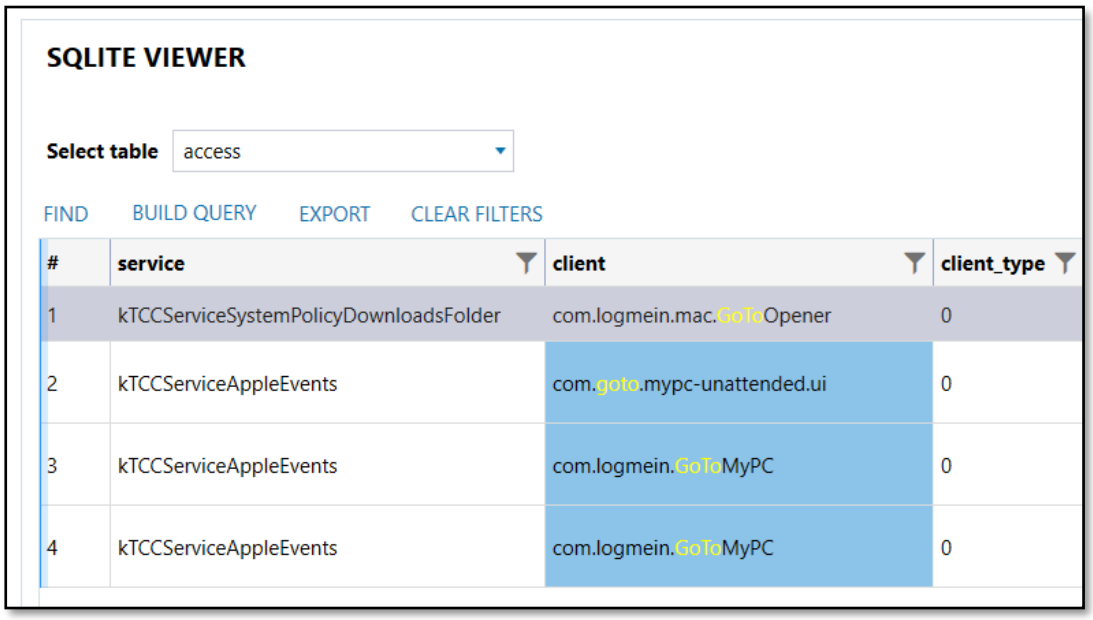


Figure 17: Represents the *TCC.db* file, highlighting the access permissions granted to the application by the user.

4.1.3 RemotePC

RemotePC was subjected to detailed forensic analysis across four major operating systems: Android, Windows, iOS, and macOS. Both installation and post-uninstallation states were examined to identify persistent artifacts. The application was tested in its free and licensed variants to evaluate any differences in artifact generation and retention. The results are organized platform-wise, presenting key evidence, including but not limited to logs, configuration files, system traces, and residual data.

4.1.3.1 Android

4.1.3.1.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4i.

Table 4i: Represents the digital artifacts recovered from the installation state of RemotePC on Android, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Remote PC Viewer	base.apk	app\com.prosoftnet.rpcnew-xW3SF-gfcdroIz5DB9Jklg==\base.apk	Contains application version details and desired permissions.	Free & Paid
	RPC Viewer Log.txt	data\com.prosoftnet.rpcnew\files\AppLogger\RPC Viewer Log.txt	Logs extensive session metadata, including host/client IP addresses, timestamps, login attempts, device details, chat timings (without chat content), remote lock events, session recordings, file transfers, sticky note activations, whiteboard activity, first	Free & Paid

			name of host, and sound events. Does not include user credentials, chat/sticky note content, or clipboard data. (Figure 18)	
	EmailList.txt	data\com.prosoftnet.rpcnew\files\RemotePC_Logger\EmailList.txt	Stores host and client email addresses.	Free & Paid
	report	data\com.prosoftnet.rpcnew\files\.com.google.firebase.crashlytics.files.v2:com.prosoftnet.rpcnew\open-sessions\660CE421032A00012D288FF0ACCF971C\report	Records SDK version, app ID, UUID, build details, session ID, OS information, and device specifications. Confirms app presence, links usage to specific device, and provides session timeline.	Free & Paid
	device.json	data\com.prosoftnet.rpcnew\files\.com.google.firebase.crashlytics.files.v2:com.prosoftnet.rpcnew\open-sessions\660CE421032A00012D288FF0ACCF971C\native\device.json	Records hardware details, including processor count, manufacturer, model, architecture, RAM, disk space, and emulator flag. Confirms device identity and environment.	Free & Paid
	com.prosoftnet.rpcnew.xml	data\com.prosoftnet.rpcnew\shared_prefs\com.prosoftnet.rpcnew.xml	Stores sensitive information, including the user's email address (username), plaintext password, access token, first name, and account status, enabling direct user attribution. It also captures the host machine's name, public IP address, session ID, and proxy topic, linking the session to specific devices and network activity. Additional artifacts include trial subscription details, plan description, and supported monitor	Free & Paid

			resolutions, confirming the application's configuration and usage context.	
	FirebaseHeartBeatW0RFRkFVTFRd+MToxMzc2OTQyNjMzMjQ6YW5kcm9pZDpkZmIyMWY0NmQxNzBkNzliYWUwN2Q5.xml	data\com.prosoftnet.rpcnew\shared_prefs\FirebaseHeartBeatW0RFRkFVTFRd+MToxMzc2OTQyNjMzMjQ6YW5kcm9pZDpkZmIyMWY0NmQxNzBkNzliYWUwN2Q5.xml	Records the last-used date of the app, device brand, model, and Android platform details. Supports timeline analysis.	Free & Paid
	RPCViewer-wal	data\com.prosoftnet.rpcnew\databases\RPCViewer-wal	Contains the host Windows device name, public IP, client device name, received file names, and session date. Useful for correlating devices, IPs, file transfers, and timeline reconstruction.	Free & Paid
	Session recording (RemotePC_Recording_20240402_152708_DESKTOP-EI2SE0P.mp4)	media\0\DCIM\RemotePC\RemotePC_Recording_20240402_152708_DESKTOP-EI2SE0P.mp4	This file is a video screen recording of the remote session, which can provide visual evidence of on-screen activities during the connection. The name of this file reveals the timestamp and receiver device name.	Free & Paid
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with the package name and associated email ID.	Free & Paid
Remote PC Host	base.apk	app\com.remotepc.host-xc7FI8ue1WPEpC09JG-iiw==\base.apk	Contains application version details and desired permissions.	Free & Paid

	com.remotepc.host.xml	data\com.remotepc.host\shared_prefs\com.remotepc.host.xml	Stores user email, plaintext password, access token, account status, host machine name, public IP, and session details. Supports direct user identification and network correlation. (Figure 19)	Free & Paid
	com.prosoftnet.rpcnew.xml	data\com.prosoftnet.rpcnew\shared_prefs\com.prosoftnet.rpcnew.xml	Same as com.remotepc.host.xml, it stores user email, plaintext password, access token, account status, host machine name, public IP, and session details. Supports direct user identification and network correlation.	Free & Paid
	HelpDeskViewerEmailList.txt	data\com.remotepc.host\files\HostLogger\HelpDeskViewerEmailList.txt	This file contains the email address of the host and client.	Free & Paid
	AndroidHost.txt	data\com.remotepc.host\files\AppLogger\AndroidHost.txt	Records login timestamp, plan details, machine ID, host username, account status, user type, subscription type, first/last names, and email address. Also, records host and client device names, make/model of Connected device, and private IP addresses, and client public IP address. Captures file paths of transferred files (excluding file sizes and received files) and session recording timestamps. (Figure 20)	Free & Paid
	report	data\com.remotepc.host\files\.com.google.firebase.crashlytics.files.v2:com.remotepc.host\open-sessions\661629780	Records the application's SDK version, Google Mobile Platform app ID, platform type, unique installation UUID, build and display versions, and	Free & Paid

		3B500011E1A18699C96DD21\report	session details such as a unique session identifier. Additionally, it captures operating system details (platform, version, build, and jailbreak status) and device-specific information, including architecture, device model, core count, RAM, disk space, manufacturer, and model class.	
	device.json	data\com.prosoftnet.rpcnew\files\.com.google.firebase.crashlytics.files.v2:com.prosoftnet.rpcnew\open-sessions\660CE421032A00012D288FF0ACCF971C\native\device.json	This JSON file contains critical device-specific metadata, including the number of available processors, device manufacturer, product name, device model, architecture, device state, available disk space, total RAM, and a flag indicating whether the device is an emulator.	Free & Paid
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via Play Store with package name and associated email ID. (Figure 21)	Free & Paid

4.1.3.1.2 Uninstallation State

Upon uninstallation of the RemotePC application from the Android device, only the library.db and session recording files were found to persist. library.db file, associated with the Google Play Store, retains metadata about previously installed applications, including package names and the associated email address of the Google Play Store. The session recording file may contain traces of remote sessions, such as timestamps or activity details, which could provide insight into possible user actions and remote access activity. Although other application-specific files were removed during uninstallation, the presence of library.db provides a valuable forensic artifact that confirms the prior existence of the GoToMyPC app on the device and helps establish a timeline of its usage.

4.1.3.1.3 Interpretation and Forensic Significance

RemotePC on Android leaves a robust forensic footprint, providing evidence of user activity, device interactions, and session characteristics. base.apk files confirm the presence, version, and desired permissions of both RemotePC Viewer and Host applications, establishing proof of installation. User-specific files such as com.prosoftnet.rpcnew.xml and com.remotepc.host.xml store email addresses, plaintext passwords, access tokens, account status, and host device details, directly linking accounts to devices and supporting attribution.

Session-related artifacts like RPC Viewer Log.txt, RPCViewer-wal, and AndroidHost.txt capture host/client IPs, login attempts, timestamps, device identifiers, file transfer traces, and activity metadata, enabling timeline reconstruction and cross-correlation between users, devices, and networks.

EmailList.txt and HelpDeskViewerEmailList.txt confirm account linkages and communication endpoints, while report and device.json provide system-level technical details, aiding verification of real versus virtual environment usage. Timeline validation is further supported by FirebaseHeartBeat.xml and library.db, recording last usage and installation metadata. session recordings may offer direct visual evidence of remote activity, supporting detailed reconstruction. Post-uninstallation, most sensitive and log files are removed; however, library.db persists, confirming prior installation, and session recordings remain accessible, offering conclusive visual evidence.

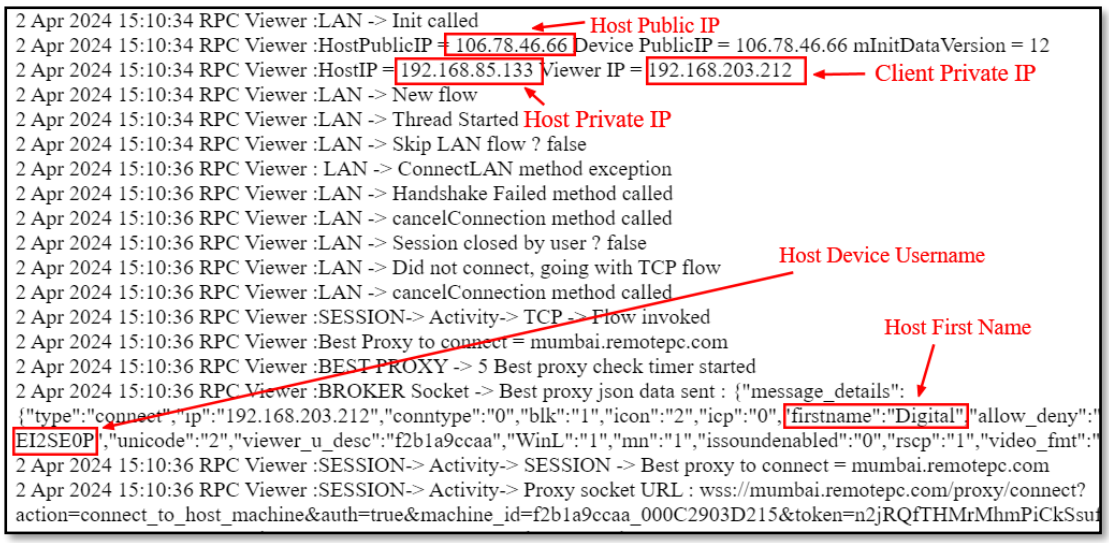


Figure 18: Represents the *RPC Viewer Log.txt* file, highlighting artifacts such as the host's public and private IP address, client private IP address, host device username, and host first name.



Figure 19: Represents the *com.remotepc.host.xml* file, highlighting artifacts such as host device make and model, personal key, trial period details, plain-text password, client and host public IP addresses, client first name, email address, license information, and plan type.

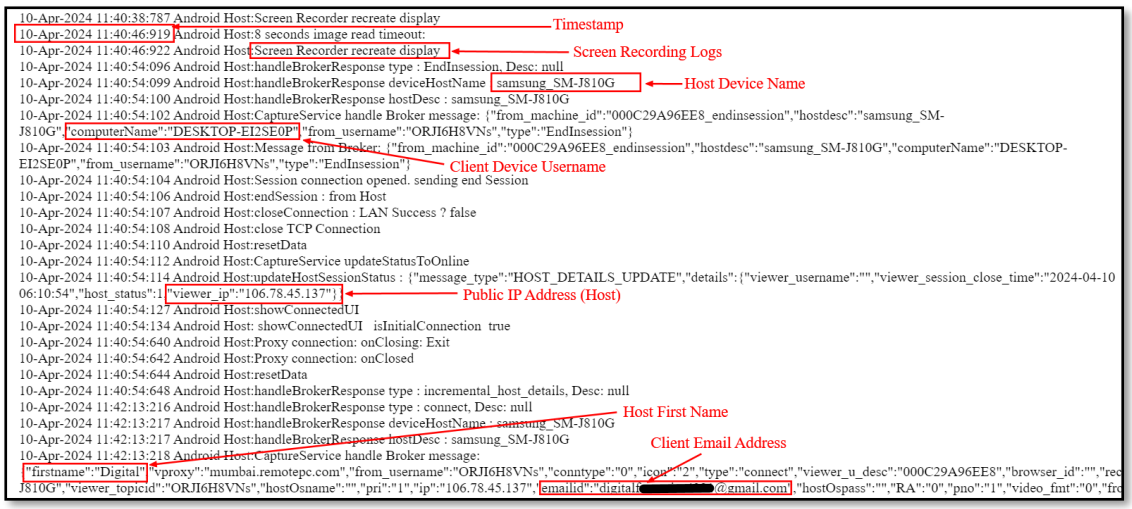


Figure 20: Represents the *AndroidHost.txt* file, highlighting artifacts such as log timestamps, session recording entries, host device make and model, client device username, host public IP address, host first name, and client email address.

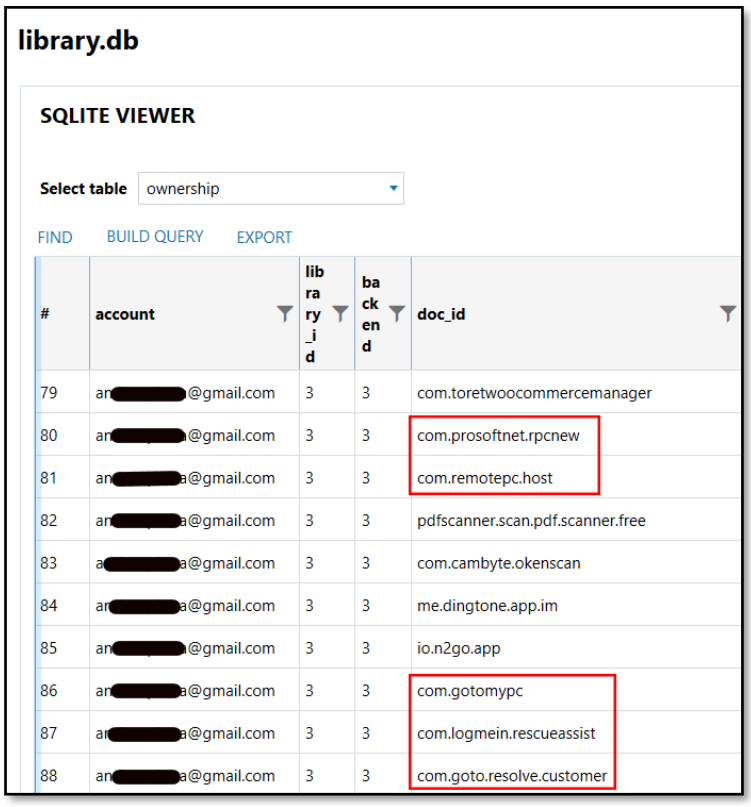


Figure 21: Represents the *library.db* file, highlighting the installation of the RemotePC Host application via the Play Store, along with the associated email ID.

4.1.3.2 Window

4.1.3.2.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4j.

Table 4j: Represents the digital artifacts recovered from the installation state of RemotePC on Windows, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
RemotePC	HostDesc.ini	ProgramData\RemotePC\HostDesc.ini	Contains the email address and host device name.	Free & Paid
	RemotePCService.txt	ProgramData\RemotePC\RemotePCService.txt	Logs session timestamps, private IP addresses of host and client, public IP of host, lock device feature timestamps, device names, Client device make/model, host email ID, first and last names, username, machine ID, and chat message timings (without content). Missing clipboard, whiteboard, sticky note, file transfer, and session recording logs. (Figure 22)	Free & Paid
	RPCSettings.ini	ProgramData\RemotePC\RPCSettings.ini	Contains application version, host username, public and private IP addresses (host and client), user's first name, email address, host device name, session start and end times, and client device make and model. (Figure 23)	Free & Paid
	RPCSuite_digit.log	ProgramData\RemotePC\RPCSuite_digit.log	Contains email address, host private and public IP addresses, user type, allowed hosts, last name,	Free & Paid

			subscription type, plan description, first name, client device make and model, and whiteboard activity details (tool usage, pen color, width, state). (Figure 24a)	
	RPCSuite_digit_Inc.log	ProgramData\RemotePC\RPCSuite_digit_Inc.log	Contains host device name, application version, session start and end times, and public and private IP addresses (host and client). (Figure 24b)	Free & Paid
	StoredValues.xml	ProgramData\RemotePC\StoredValues.xml	Contains the RemotePC account email address.	Free & Paid
	ZGlnaXRhbGZvcmVuc2ljczMyMUBnbWFpbC5jb20=.rpc	ProgramData\RemotePC\ZGlnaXRhbGZvcmVuc2ljczMyMUBnbWFpbC5jb20=.rpc	It contains the host public and private IP address, client private IP address, host registration timestamp, host username, host device name, application version, and device make and model.	Free & Paid
	ZGlnaXRhbGZvcmVuc2ljczMyMUBnbWFpbC5jb20=.rpc_inc	ProgramData\RemotePC\ZGlnaXRhbGZvcmVuc2ljczMyMUBnbWFpbC5jb20=.rpc_inc	Contains the public and private IP addresses of the host & Client, host device name, and app version.	Free & Paid
	settings.json	ProgramData\RemotePC Performance Host\settings.json	Contains the email address, host username, machine ID, and detailed features and permission settings, specifying the application's access rights and enabled functionalities.	Free & Paid
	RPCPerformanceService.log	ProgramData\RemotePC Performance Host\Logs\RPCPerformanceService.log	Contains the public IP address of the connected device.	Free & Paid
	viewerSettings.json	ProgramData\RemotePC Performance Viewer\viewerSettings.json	Contains the public IP address of the Windows device.	Free & Paid

	settings.json	ProgramData\RemotePC Performance Viewer\settings.json	Contains email address, host username, machine ID, authentication token, user type, first and last name, plan description, and session recording file path. (Figure 25)	Free & Paid
	Prefetch Files (Folder)	Windows\Prefetch	Contains run count, last run time, other execution times, and MAC details of the application.	Free & Paid
	Window event logs	Windows\System32\winevt\Logs	Windows Event Log entry (System.evtx, Application.evtx,) contains details of the user account (Security ID, Account Name, Login Time, Event ID, Computer Name) and the associated process execution, including Process ID and full Process Name path.	Free & Paid

4.1.3.2.2 Uninstallation State

Upon uninstallation of the RemotePC application from the Windows device, it was observed that the uninstallation process did not remove or delete any associated files. All the artifacts previously identified and listed in the analysis table remained intact at their original file paths, indicating that the application leaves behind significant residual data even after being removed. These remnants include configuration files, logs, preference files, and other critical components, which retain sensitive information such as user details, session data, and application metadata. This persistence of data poses both forensic advantages, as it aids in post-uninstallation investigations, and potential security concerns, as sensitive information remains accessible on the system

4.1.3.2.3 Interpretation and Forensic Significance

RemotePC on Windows produces a comprehensive set of artifacts that are highly valuable for forensic investigations. Configuration and preference files such as HostDesc.ini, StoredValues.xml, and settings.json store user emails, hostnames,

usernames, account details, machine IDs, and permission settings, directly linking activity to specific users and devices and supporting attribution and usage analysis.

Network and session logs, including RemotePCService.txt, RPCSettings.ini, RPCSuite_digit.log, and RPCSuite_digit_Inc.log, capture public and private IPs, device names, session start/end times, subscription details, and whiteboard activity metadata, enabling detailed reconstruction of remote sessions and verification of interactions between host and client. Encoded files like ZGlnaXRhbGZvcmVuc2ljczMyMUBnbWFpbC5jb20=.rpc and .rpc_inc provide unique identifiers for users, IP addresses, device names, model identifiers, timestamps, and application versions, acting as digital fingerprints that link activity across devices and networks.

Performance and contextual artifacts, including RPCPerformanceService.log and viewerSettings.json, offer additional insights into session management, authentication, and session recording locations. Timeline reconstruction is further supported by Prefetch files and Windows Event Logs, confirming execution frequency, last-run times, and linking processes to user accounts.

Unlike many other platforms, all artifacts persist post-uninstallation, including logs, configurations, and preference data. This persistence allows investigators to recover session histories, account activity, and system information even if the application was deliberately removed. While advantageous for forensic analysis, it also poses security risks due to retained plaintext credentials and session data.

Overall, RemotePC on Windows provides one of the richest forensic datasets among examined applications, supporting precise reconstruction of remote access events, user attribution, device correlation, and timeline verification both before and after uninstallation.

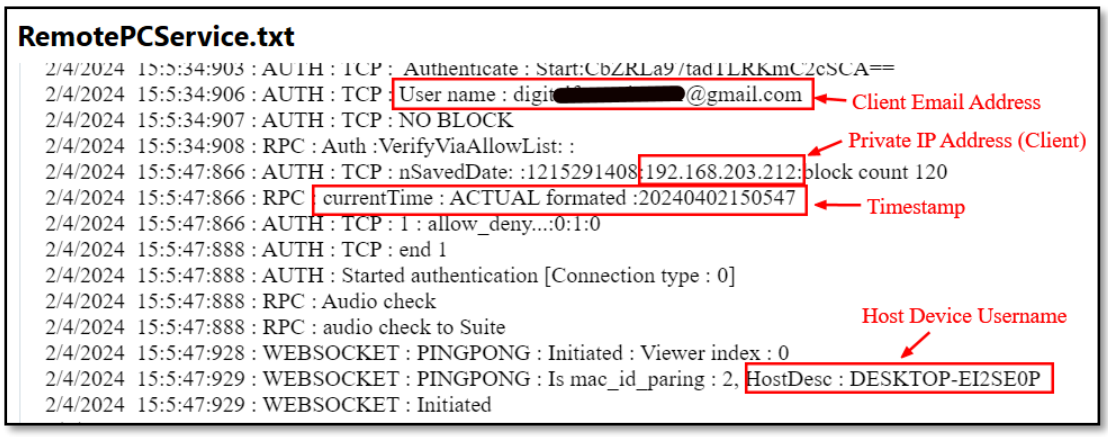


Figure 22: Represents the *RemotePCService.txt* file, highlighting artifacts such as the client email address, client private IP address, timestamp, and host device username.



Figure 23: Represents the *RPCSettings.ini* file, highlighting artifacts such as host email address, license type, and free trial details.

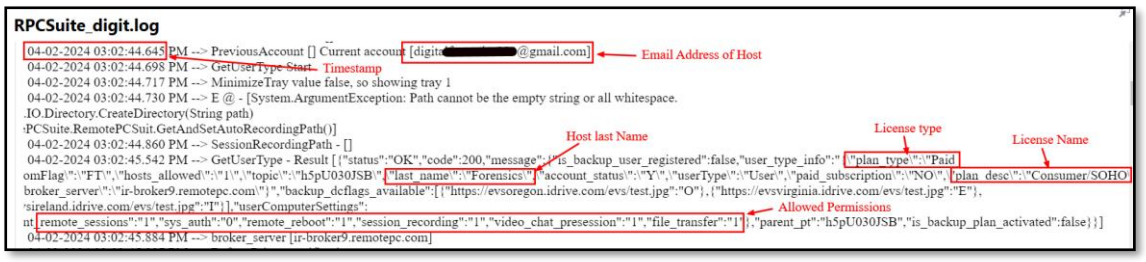


Figure 24a: Represents the *RPCSuite_digit.log* file, highlighting artifacts such as host email address, timestamp, last name host, license type, license name, and allowed permissions.

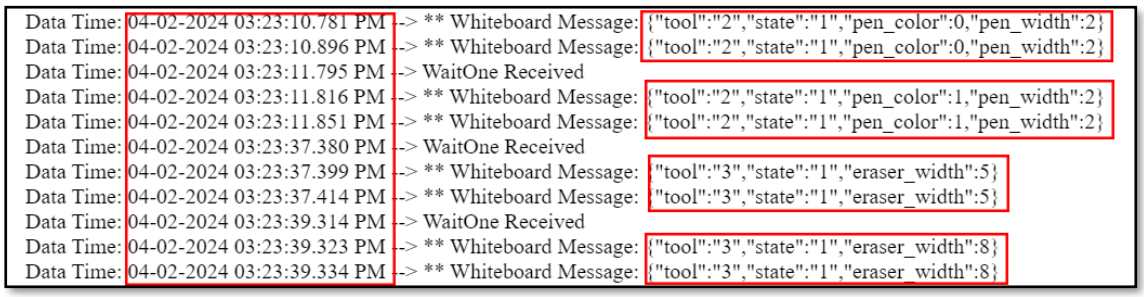


Figure 24b: Represents the *RPCSuite_digit.log* file, highlighting whiteboard activity details such as tool usage, pen color, width, state, and timestamp. The numerical values indicate the type of tool used, while state '1' denotes true.

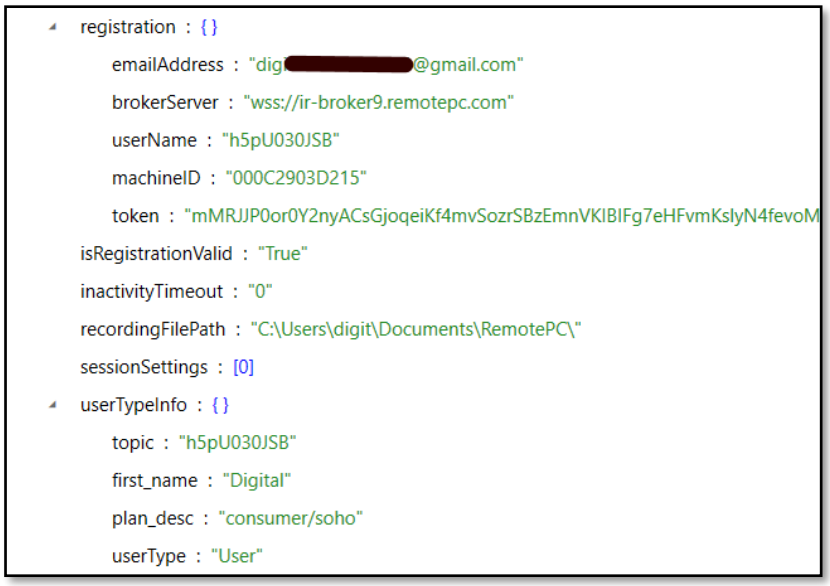


Figure 25: Represents the *settings.json* file, highlighting email address, authentication token, user type, First name, plan description, and session recording file path.

4.1.3.3 iOS

4.1.3.3.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4k.

Table 4k: Represents the digital artifacts recovered from the installation state of RemotePC on iOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
RemotePC	kSecClassGenericPassword	/Keychain/kSecClassGenericPassword	Stores RemotePC's encryption key, user identifier, app identifier, and UUID.	Free & Paid
	com.prosoftnet.remotepc.iphone.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.prosoftnet.remotepc.iphone.plist	Contains registered email address, application version, bundle ID, and iOS device name.	Free & Paid
	com.logmein.ios.gotomypc.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Documents/.RemotePCLite/ft_trace.log	Contains file transfer logs; only one shared file path was recorded.	Free & Paid
	metadata.clsrecord	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/com.crashlytics.data/com.prosoftnet.remotepc.iphone/v5/reports/active/2afebfe04a854aef94a81336bb3ae6b3/metadata.clsrecord	Contains host device details (model, machine identifier, OS build and display version, platform, locale) and application details (bundle ID, build version, display version). (Figure 26)	Free & Paid
	rm_client_app_preference.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/rm_clie	Contains the public IP address of the host device.	Free & Paid

		nt_app_preference.plist		
	trace.log	/private/var/mobile/Containers/Data/Application/<UUID>/Documents/.RemotePCLite/trace.log	It contains host and client device names, email address, public and private IP addresses (both devices), session start time, chat message typing/opening logs (no content), application state changes, session recording start and end times, file sending/receiving paths (no filenames), PC lock logs, remote sound enablement logs, and a unique username code. (Figure 27)	Free & Paid
	iTunesMetadata.plist	/private/var/containers/Bundle/Application/<UUID>/iTunesMetadata.plist	Confirms official App Store installation. Contains Apple ID, item name, artist name, purchase ID, item ID, kind, bundle version, source app, software version bundle, purchase date, and dsPersonalID.	Free & Paid
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records installation and uninstallation logs with exact timestamps, application package paths, and package names.	Free & Paid

4.1.3.3.2 Uninstallation State

After uninstalling the RemotePC apps (Free and Paid) on iOS, only two residual files, named kSecClassGenericPassword and mobile_installation.log.0 were identified. mobile_installation.log.0 file captures detailed information related to the installation and uninstallation processes of applications on iOS. Specifically, it records the exact timestamp of each event, the file path of the application package, and its corresponding package name. Although no additional user data or configuration files were found after uninstallation, the presence of mobile_installation.log.0 stands as a crucial forensic artifact for confirming the historical existence of the TeamViewer app on the device.

4.1.3.3.3 Interpretation and Forensic Significance

RemotePC on iOS produces a smaller but highly meaningful set of forensic artifacts. The kSecClassGenericPassword entry in the Keychain confirms application installation and links accounts to specific devices. The com.prosoftnet.remotepc.iphone.plist links the app to a registered email, device name, and application version, establishing user-device association. metadata.clsrecord provides technical evidence of device model, OS version, locale, and application build, validating environment, and device identity.

Session and activity insights come from trace.log, which records host/client device names, public and private IPs, session start times, chat activity indicators, session recording markers, file transfer paths, and device events. While content and file details are absent, these logs allow timeline reconstruction and correlation with network activity. rm_client_app_preference.plist further supports session attribution via the host's public IP. iTunesMetadata.plist and mobile_installation.log.0 verify App Store provenance and record installation/uninstallation events, providing historical confirmation of app presence.

After uninstallation, only kSecClassGenericPassword and mobile_installation.log.0 persist, enabling confirmation of prior use and account linkage even when other logs and preferences are removed. The absence of screenshots, video recordings, or detailed file transfer logs reflects iOS's restrictive artifact generation, limiting full session reconstruction but maintaining minimal forensic exposure.

In summary, RemotePC on iOS provides fewer artifacts than Android or Windows but retains high forensic value for user identification, device attribution, session timeline reconstruction, and confirmation of historical app usage.

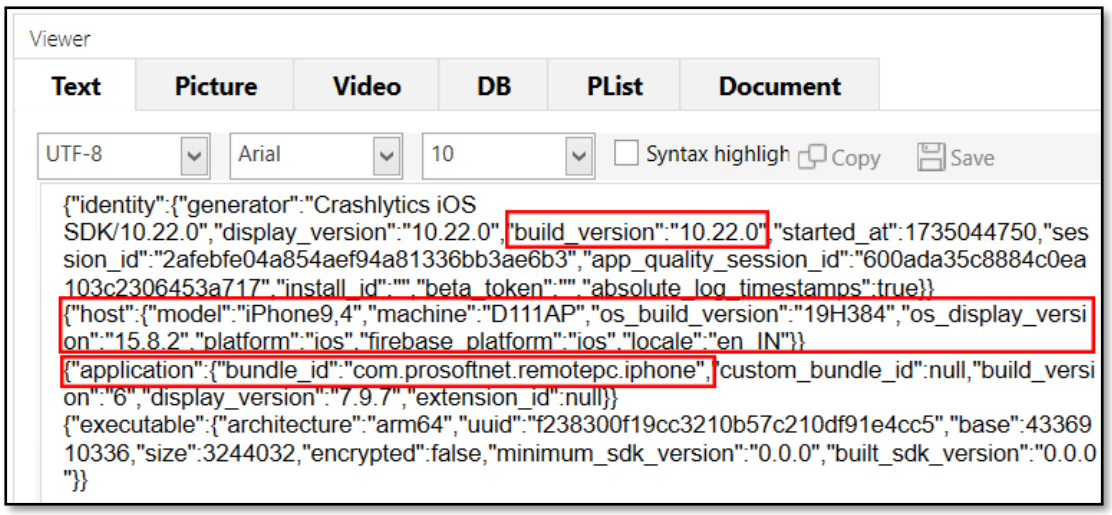


Figure 26: Represents the *metadata.clsrecord* file, highlighting artifacts such as host device model, machine identifier, OS build, display version, platform, locale, and application details, including bundle ID, build version, and display version.



Figure 27: Represents the *trace.log* file, highlighting artifacts such as client name, client public and private IP addresses, host name, model identifier, email address, host public IP address, and application version details.

4.1.3.4 macOS

4.1.3.4.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4l.

Table 4l: Represents the digital artifacts recovered from the installation state of RemotePC on macOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Remote PC	Info.plist	Applications\RemotePCSuite.app\Contents\Info.plist	Contains metadata including CFBundleName, CFBundleVersion, and CFBundleIdentifier, confirming application name, version, and unique identifier.	Free & Paid
	.dat.nosync0480.nWpx91	Library\Application Support\RPCForMac\.dat.nosync0480.nWpx91	Contains host device name, host application version, client's public IP address, and session recording file path. (Figure 28)	Free & Paid
	.HostPref.plist	Library\Application Support\RPCForMac\.HostPref.plist	Contains host device name, connection type, last logged-in user email address, session recording file path, session date, application version, subscription plan, and user's first and last names. (Figure 29)	Free & Paid
	install.log	private\var\log\install.log	Records installation events with timestamps and file paths of installation packages for RemotePC.	Free & Paid
	com.apple.finder.plist	Users\digitalsoni\Library\Preferences\com.apple.finder.plist	Contains FXRecentFolders entries referencing RemotePC, indicating recent access.	Free & Paid

			FXDesktopVolumePosition shows desktop arrangement.	
	TCC.db	Users\digitalsoni\Library\Application Support\com.apple.TCC\TCC.db & \Library\Application Support\com.apple.TCC\TCC.db	Records permissions granted to applications, including access levels for RemotePC.	Free & Paid
	LaunchDaemons (Folder)	Library\LaunchDaemons	Contains multiple .plist files with file paths, program identifiers, labels, and KeepAlive status for RemotePC.	Free & Paid
	LaunchAgents(Folder)	Library\ LaunchAgents	Contains multiple .plist files with file paths, program labels, descriptions, and KeepAlive status for RemotePC.	Free & Paid

4.1.3.4.2 Uninstallation State

After uninstalling RemotePC on macOS, all the previously identified artifacts remained present on the system, indicating that the uninstallation process does not completely remove related configuration, log, and preference files. The only exception was the Info.plist file located at *Applications\RemotePCSuite.app\Contents*, which was removed as part of the uninstallation. This persistence of residual artifacts, including user account details, configuration settings, and installation logs, underscores the potential for post-uninstallation forensic recovery and analysis.

4.1.3.4.3 Interpretation and Forensic Significance

RemotePC on macOS generates a broad spectrum of artifacts valuable for forensic investigations. Info.plist confirms the application's identity, version, and bundle identifier, providing proof of installation. User and session-related files in Library/Application/Support/RPCForMac, including .dat.nosync0480.nWpx91 and .HostPref.plist, store host/client device names, emails, subscription details, session

dates, IP addresses, and session recording paths, supporting direct user attribution, device linkage, and timeline reconstruction.

install.log provides timestamped installation metadata, assisting in validating provenance and reconstructing installation timelines. com.apple.finder.plist documents user interaction through Finder, reflecting engagement with the application. System-level artifacts like TCC.db reveal granted permissions, such as screen capture and microphone access, highlighting both the application's capabilities and potential privacy implications. Persistence files in LaunchDaemons and LaunchAgents may indicate background execution and auto-start behavior, suggesting continued accessibility for remote sessions.

Most artifacts persist after uninstallation, except the Info.plist in the Applications folder. The survival of configuration data, session metadata, user account information, and system logs allow investigators to reconstruct historical usage and user-device interactions even after the application is removed. This persistence enhances forensic value while also exposing sensitive information such as emails, subscription details, and session recording paths.

In summary, RemotePC on macOS provides a rich and diverse forensic footprint, with post-uninstallation evidence particularly useful for user attribution, device correlation, and detailed reconstruction of past session activity.

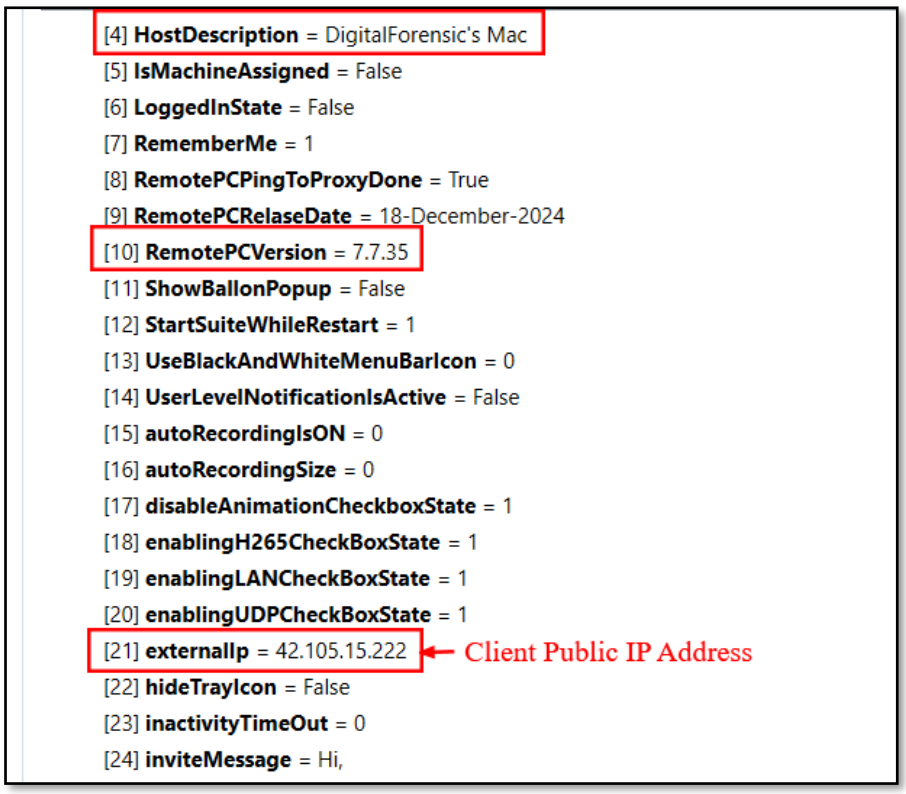


Figure 28: Represents the *.dat.nosync0480.nWpx91* file, highlighting artifacts such as host descriptions, application version, and client public IP address.

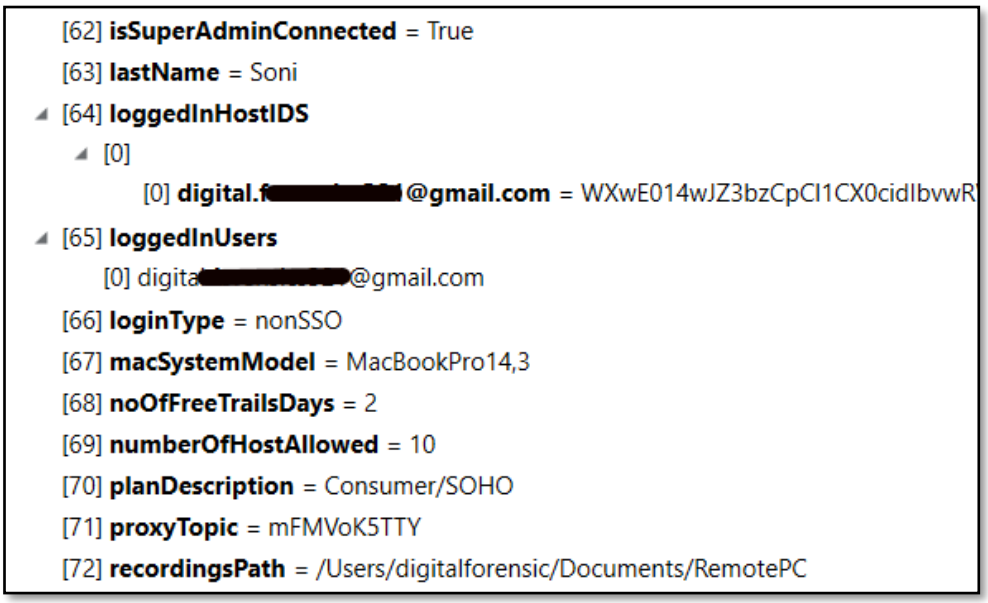


Figure 29: Represents the *.HostPref.plist* file, highlighting artifacts such as last name, email address, Mac model identifier, details of the plan, and session recording path.

4.1.4 Zoho Assist

Zoho Assist was subjected to detailed forensic analysis across four major operating systems: Windows, Android, macOS, and iOS. Both installation and post-uninstallation states were examined to identify persistent artifacts. The application was tested in its free and licensed variants to evaluate any differences in artifact generation and retention. The results are organized platform-wise, presenting key evidence, including but not limited to logs, configuration files, system traces, and residual data.

4.1.4.1 Android

4.1.4.1.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4m.

Table 4m: Represents the digital artifacts recovered from the installation state of Zoho Assist on Android, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Zoho Assist - Remote Desktop	base.apk	app\com.zoho.assist-6kSZM7vKjb9d5W-T-0IWEw==\base.apk	Contains application version details and desired permissions.	Free & Paid
	iamlib.properties.xml	data\com.zoho.assist\shared_prefs\iamlib.properties.xml	Stores the Zoho Assist ID of the client device (Android).	Free & Paid
	apptics-core.xml	data\com.zoho.assist\shared_prefs\apptics-core.xml	Records the timezone and application version.	Free & Paid

	iamoauthlib.db-wal	data\com.zoho.assist\databases\iamoauthlib.db-wal	Contains the user's full name, email address of the Zoho account, timezone, and Zoho Assist ID of the Client device (Android). (Figure 30)	Free & Paid
	iamoauthlib.db	data\com.zoho.assist\databases\iamoauthlib.db	Stores Zoho Assist ID, email address, display name, first name, last name, and timezone.	Free & Paid
	apptics-core.db-wal	data\com.zoho.assist\databases\apptics-core.db-wal	Records Zoho Assist ID of the connected device, along with email address, display name, first name, last name, RAM/ROM, and timezone.	Free & Paid
	apptics-core.db	data\com.zoho.assist\databases\apptics-core.db	Contains client device model, battery status, timezone, RAM/ROM details, application version, and email address.	Free & Paid
	Screenshots	media\0\DCIM\Screenshots	This folder contains the screenshot captured using the Zoho Assist application. These screenshots' names contain the date and name of the application. **Note:** The *Free version does not support file transfer or screenshot capture.*	Paid Only
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with package name and associated email ID.	Free & Paid
Customer App - Zoho Assist	Base. apk	\app\com.zoho.assist.agent-G2knsC1KYL3s	Contains application version details and desired permissions.	Free & Paid

		NSvRYKtxng==\base.apk		
	ZA_UNATTENDED_FRAMEWORK_SHARED_PREF.xml	data\com.zoho.assist.agent\shared_prefs\ZA_UNATTENDED_FRAMEWORK_SHARED_PREF.xml	Stores the email address of the Zoho Assist account. ***Note:*** *The file was present in the free version as well, but email address was not found stored.*	Paid only
	UserDetails.xml	data\com.zoho.assist.agent\shared_prefs\UserDetails.xml	Records the email address of the Zoho Assist account.	Free & Paid
	RS_PREFS_SESSION_FILE_NAME.xml	data\com.zoho.assist.agent\shared_prefs\RS_PREFS_SESSION_FILE_NAME.xml	Contains the Zoho Assist ID.	Free & Paid
	FirebaseHeartBeatW0RFRkFVTFRd+MTo5MTAyNTY0NzA0NTI6YW5kcm9pZDpjMDhlZGYyY2Y0NzQyZjNj.xml	data\com.zoho.assist.agent\shared_prefs\FirebaseHeartBeatW0RFRkFVTFRd+MTo5MTAyNTY0NzA0NTI6YW5kcm9pZDpjMDhlZGYyY2Y0NzQyZjNj.xml	Stores the application's last date of use.	Free & Paid
	session_history_database-wal	data\com.zoho.assist.agent\databases\session_history_database-wal	Contains Zoho Assist account email, chat messages with timestamps, Zoho Assist ID, session initiation timestamps, and sent file names with timestamps (host-to-client only). ***Note:*** *The file was Present in the free version as well, but no artifact was found stored. (Figure 31)*	Free & Paid
	session_history_database	data\com.zoho.assist.agent\databases\session_history_database	Contains the email address of Zoho Assist, the session ID, and the timestamp of session	Free & Paid

			initiation. **Note:** The *file was Present in the free version as well, but no artifact was found stored.*	
	apptics-core.db	data\com.zoho.assist.agent\databases\apptics-core.db	Records client device model, battery status, timezone, RAM/ROM details, app version, and email address.	Free & Paid
	library.db	data\com.android.vending\databases\library.db	SQLite database confirming installation via the Play Store with package name and associated email ID.	Free & Paid

4.1.4.1.2 Uninstallation State

After uninstalling the three Zoho Assist apps (Free and Paid) on Android, only a few traces were left behind. The most notable ones were the library.db file and screenshot images. The library.db file is part of the Google Play Store's internal system and keeps a record of app installations, including the package names of the uninstalled apps. This makes it a valuable clue for proving that Zoho Assist was once installed on the device. In some cases, screenshots taken during remote sessions also remained, usually saved in the device's media folders depending on user settings. Other than these, no logs, settings, or user data were found, showing that Zoho Assist apps uninstall fairly cleanly on Android. Still, these few leftover items, especially the library.db entry and screenshots can be crucial for investigators trying to confirm the past use of remote access tools.

4.1.4.1.3 Interpretation and Forensic Significance

Zoho Assist on Android produces artifacts that are valuable for user attribution, device profiling, and session reconstruction. base.apk files confirm installation and requested permissions, establishing baseline proof of app presence. User-centric artifacts such as iamlib.properties.xml, apptics-core.xml, RS_PREFS_SESSION_FILE_NAME.xml, iamoauthlib.db-wal, and iamoauthlib.db store Zoho Assist IDs, email addresses, full names, time zones, and device metadata, directly linking users to devices and enabling attribution of remote sessions.

Databases including apptics-core.db and its -wal file provide detailed device information, including model, battery status, RAM/ROM, and app version, supporting reconstruction of the technical environment. session_history_database and its -wal file record session IDs, timestamps, and, in some cases, email addresses, chat traces, and filenames of transferred files, allowing timeline reconstruction and cross-correlation of session activity. UserDetails.xml and ZA_UNATTENDED_FRAMEWORK_SHARED_PREF.xml confirm account ownership and tie remote activity to specific credentials. Screenshots generated during sessions (paid versions) offer direct visual evidence of activity.

After uninstallation, most artifacts, including preferences, logs, and session histories, are removed. Residual evidence includes library.db (Play Store package names and installation timestamps) and session screenshots, which support confirmation of historical app usage. The absence of persistent session logs or credentials indicates that Zoho Assist uninstalls relatively cleanly compared to applications like RemotePC or TeamViewer.

In summary, Zoho Assist on Android provides moderate forensic value during installation, offering evidence for user-device linkage and session activity. Post-uninstallation, forensic traces are limited to Play Store metadata and screenshots, but these residual artifacts are sufficient to confirm past application use.



Figure 30: Represents the *iamoauthlib.db-wal* file, highlighting artifacts such as full name, email address of the Zoho account, and client Zoho Assist ID.



Figure 31: Represents the *session_history_database-wal* file, highlighting artifacts such as Zoho Assist account email, chat messages with timestamps, Zoho Assist ID, and sent file names with timestamps (host-to-client only).

4.1.4.2 Windows

4.1.4.2.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4n.

Table 4n: Represents the digital artifacts recovered from the installation state of Zoho Assist on Windows, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Zoho Assist Remote desktop	ACServer.log	ProgramData\ZohoMeeting\log\ACServer.log	Stores PC name, display name, username linked to email address, and timestamps. (Figure 32)	Paid only
	IPCClientLog.log	ProgramData\ZohoMeeting\log\IPCClientLog.log	Records timestamps and the Zoho Assist account email address.	Paid only
	LogFile.log	ProgramData\ZohoMeeting\log\LogFile.log	Captures timestamped entries, username linked to email address, and public IP address of the Android host device.	Paid only
	LogFileTemp.log	ProgramData\ZohoMeeting\log\LogFileTemp.log	Contains session ID of the Windows (client) device, username, username linked to email address, and timestamps of entries. (Figure 33)	Paid only
	LogFileTray.log	ProgramData\ZohoMeeting\log\LogFileTray.log	Records the session ID of the host device, the username linked to the email address, and timestamps.	Paid only
	PeerConnection.log	ProgramData\ZohoMeeting\log\PeerConnection.log	Stores timestamps and the private IP address of the Windows client device.	Paid only

	unattended.log	ProgramData\ZohoMeeting\log\unattended.log	Contains timestamps, host session ID, private IP address of the Windows (client), and username linked to email address.	Paid only
	upgrader.log	ProgramData\ZohoMeeting\log\upgrader.log	Records timestamps, host session ID, and computer name.	Paid only
	ZAAudioClient.log	ProgramData\ZohoMeeting\log\ZAAudioClient.log	Stores timestamps, Windows (client) session ID, username linked to email address, and display name.	Paid only
	ZAService.log	ProgramData\ZohoMeeting\log\ZAService.log	Contains timestamps, Windows (client) session ID, and username linked to email address.	Paid only
	ZohoURSService.log	ProgramData\ZohoMeeting\log\ZohoURSService.log	Records timestamps, host session ID, and email address configured for unattended access.	Paid only
	Connect.log	Users\<Username>\AppData\Local\ZohoMeeting\log\Connect.log	Stores timestamps, Windows (client) session ID, and username linked to email address.	Paid only
	CustomerPluginApp.log	Users\<Username>\AppData\Local\ZohoMeeting\log\CustomerPluginApp.log	Contains timestamps, session IDs of both devices, username linked to email address, and customer name. (Figure 34)	Free & Paid
	000003.log	Users\<Username>\AppData\Roaming\Zoho Assist\Session Storage\000003.log	Records session date, session key, names linked to both devices, email address, and chat messages. ***Note:*** *File was present in the free version as well, but the session key, names, email address, and chats were absent*. (Figure 35)	Free & Paid
	main.log	Users\<Username>\AppData\Roaming\Zoho Assist\logs\main.log	Contains the timestamp entries, Android device (Host) session ID, and version details of the Windows (Client) application.	Free & Paid

	Prefetch Files (Folder)	Windows\Prefetch	Logs run counts, last run time, additional execution times, and MAC timestamps. Useful for identifying total executions and associated timelines.	Free & Paid
	Window event logs	Windows\System32\winevt\Logs	Windows Event Log entries (System.evtx and Security.evtx) contain details of the user account (Security ID, Account Name, Domain, Logon ID) and the associated process execution, including Process ID and full Process Name path.	Free & Paid

4.1.4.2.2 Uninstallation State

All the previously identified artifacts were found to persist even after the uninstallation of Zoho Assist from the Windows system. The files were successfully recovered from their respective directories, indicating that the uninstallation process did not completely remove user-specific or system-level data. While the artifact content remained consistent, the hash values of a few files were found to be different, likely due to minor changes in metadata or timestamp updates during system operations.

4.1.4.2.3 Interpretation and Forensic Significance

Zoho Assist on Windows produces a comprehensive set of artifacts valuable for user attribution, session reconstruction, and system activity correlation. Log files under ProgramData\ZohoMeeting\log, including ACServer.log, IPCClientLog.log, LogFile.log, LogFileTemp.log, LogFileTray.log, and PeerConnection.log, capture usernames linked to emails, session IDs, timestamps, private/public IPs, and device information, enabling precise mapping of users to devices and reconstruction of session timelines.

Unattended access artifacts, such as unattended.log and ZohoURSService.log, document background or persistent remote sessions, including session IDs, emails, and timestamps, supporting attribution of non-interactive activity. Additional session-level logs (upgrader.log, ZAAudioClient.log, ZAService.log) track audio connections and session events, while user-specific files under AppData, including Connect.log, CustomerPluginApp.log, 000003.log, and main.log, provide evidence linking sessions

to individual accounts, including host device IDs, session dates, and chat metadata (where available).

System-level artifacts such as Prefetch files and Windows Event Logs allow investigators to correlate application activity with native system operations, reinforcing timelines and supporting the forensic chain of evidence.

Post-uninstallation, all identified artifacts persist, maintaining detailed logs of user sessions, account activity, and system interactions. This persistence is advantageous for forensic reconstruction, ensuring that historical remote access activity can be recovered even after software removal, while also highlighting potential privacy and security risks due to retained sensitive information.

In summary, Zoho Assist on Windows provides one of the richest forensic datasets among examined applications, with extensive logging and persistent artifacts that support detailed reconstruction of remote access, user attribution, and session-level activity.

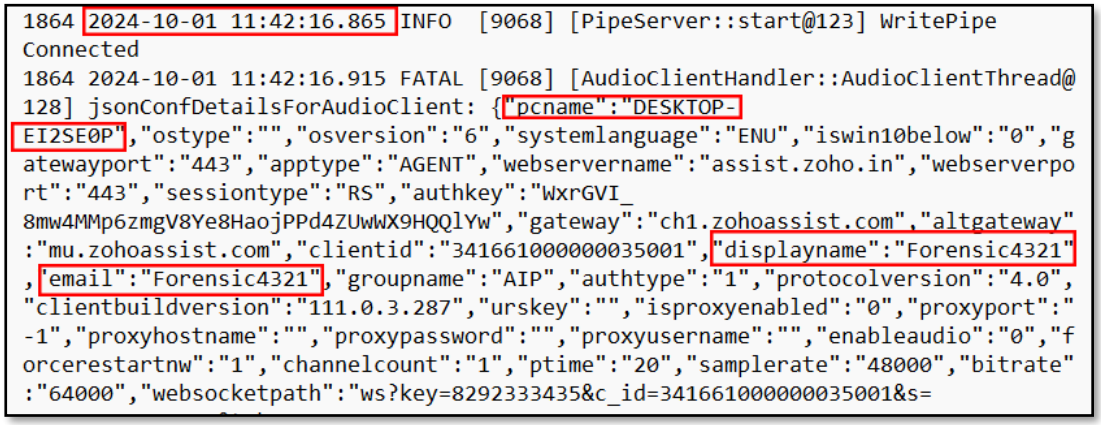


Figure 32: Represents the *ACServer.log* file, highlighting artifacts such as timestamp, pcname, displayname, and name associated with the Zoho Assist account (Email).

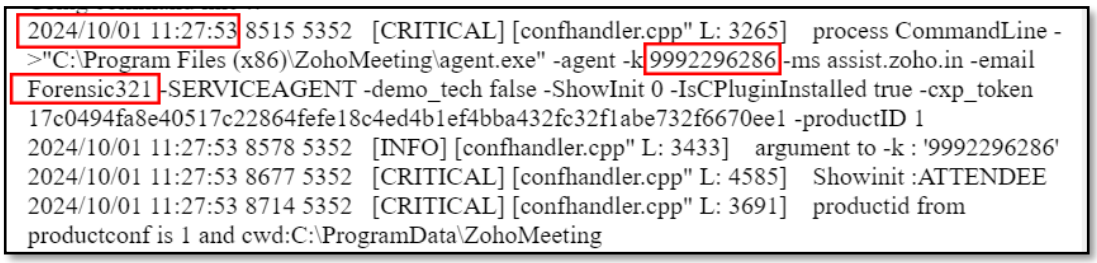


Figure 33: Represents the *LogFileTemp.log* file, highlighting artifacts such as timestamp, Zoho Assist ID of Windows (Client), and the name associated with the Zoho Assist account (Email).

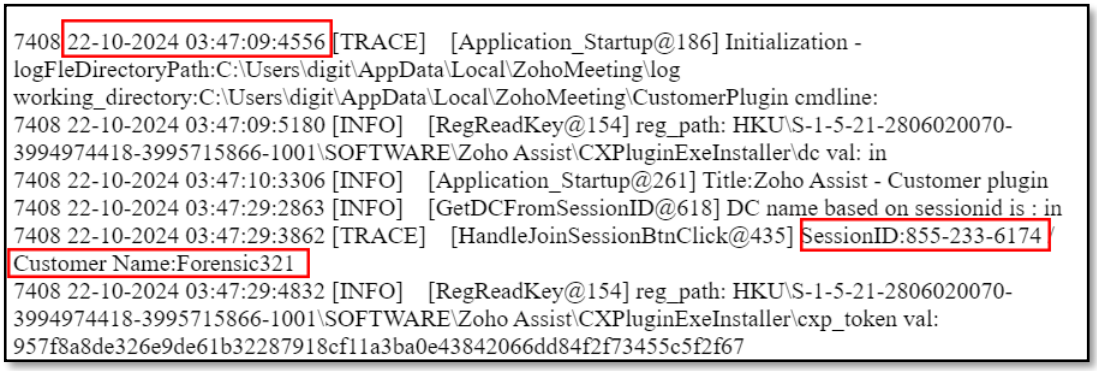


Figure 34: Represents the *CustomerPluginApp.log* file, highlighting artifacts such as timestamp, Zoho Assist ID of the Window (Host), and customer name.

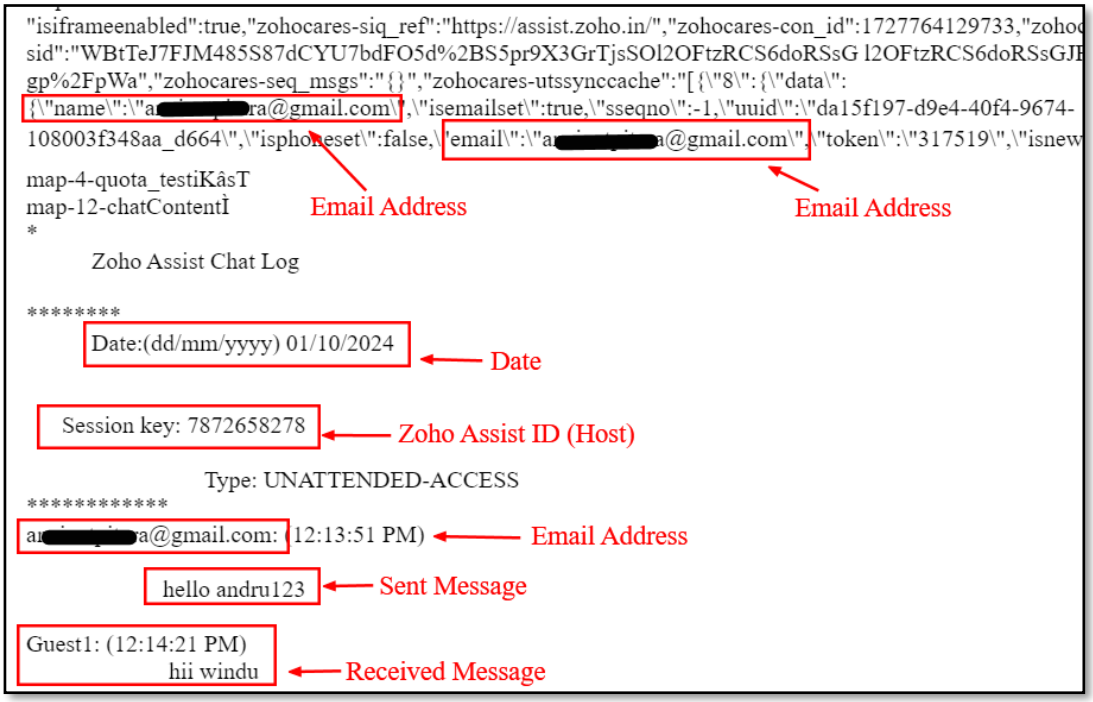


Figure 35: Represents the *000003.log* file, highlighting artifacts such as email address, Date, Zoho Assist ID of host, and sent & received messages.

4.1.4.3 iOS

4.1.4.3.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4o.

Table 4o: Represents the digital artifacts recovered from the installation state of Zoho Assist on iOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Zoho Assist	kSecClassGenericPassword	/Keychain/kSecClassGenericPassword	Stores Zoho Assist's encryption key, user identifier, app identifier, and UUID.	Free & Paid
	Shared File	/private/var/mobile/Containers/Shared/AppGroup/<UUID>/6182232128/A9B17E33-7C4A-41A2-A218-37944A475F2C/	Contains files exchanged between the host and client during a session.	Paid Only
	group.com.zoho.assist.customer.plist	/private/var/mobile/Containers/Shared/AppGroup/<UUID>/Library/Preferences/group.com.zoho.assist.customer.plist	Stores the email address linked to the Zoho Assist account. ***Note:*** *File was present in the free version as well, but the session key, names, email address, and chats were absent.*	Free & Paid
	com.zoho.assist.zohoassist.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.zoho.assist.zohoassist.plist	Contains the Session ID associated with the Zoho Assist account.	Free & Paid
	com.zoho.assist.zohoassist 2025-01-	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/ZAnalytics/Analytics/Zoho	Stores Session ID, timestamped activity logs, chat message logs (metadata only, not content), exchanged file	Free & Paid

	23_08-34-10.523.log	Assist/Logs/com.zoho.assist.zohoassist 2025-01-23_08-34-10.523.log	names, and feature/permission details. ***Note:*** *The file was present in the free version, but file transfer logs were absent since this feature is not supported. (Figure 36)*	
	Screenshots	/private/var/mobile/Containers/Data/Application/<UUID>/tmp/C175E7CC-DC6F-46A7-A10F-D477ABC02747.jpeg **&** /private/var/mobile/Media/DCIM/100APPLE/IMG_0012.JPG	Screenshots of the macOS screen captured through Zoho Assist (iOS).	Paid Only
	com.zoho.assist.customer.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.zoho.assist.customer.plist	Stores the Session ID along with iOS version and model details.	Paid Only
	com.zoho.assist.customer 2025-01-23_09-15-45.726.log	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Caches/ZAnalytics/Analytics/Assist Client/Logs/com.zoho.assist.customer 2025-01-23_09-15-45.726.log	Contains the Session ID associated with the Zoho Assist account.	Paid Only
	group.com.zoho.assist.zohoassist.widget.plist	/private/var/mobile/Containers/Shared/AppGroup/<UUID>/Library/Preferences/group.com.zoho.assist.zohoassist.widget.plist	Stores the organization name and ID.	Free & Paid
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records detailed logs of installation and uninstallation events. Includes timestamps, file paths of app packages, and package names. Helps establish a timeline for Zoho Assist installation/removal.	Free & Paid

4.1.4.3.2 Uninstallation State

After uninstalling the Zoho Assist apps (Free and Paid) on iOS, only a single residual file named mobile_installation.log.0 was identified. This file captures detailed information related to the installation and uninstallation processes of applications on iOS. Specifically, it records the exact timestamp of each installation/uninstallation event, the file path of the application package, and its corresponding package name. Although no additional user data or configuration files were found after uninstallation, the presence of mobile_installation.log.0 stands as a crucial forensic artifact for confirming the historical existence of the Zoho Assist app on the device.

4.1.4.3.3 Interpretation and Forensic Significance

Zoho Assist on iOS generates artifacts that provide strong evidence for user attribution, session activity, and application provenance. The kSecClassGenericPassword entry in the Keychain stores encryption keys, user and app identifiers, and UUIDs, linking the application to the device and account.

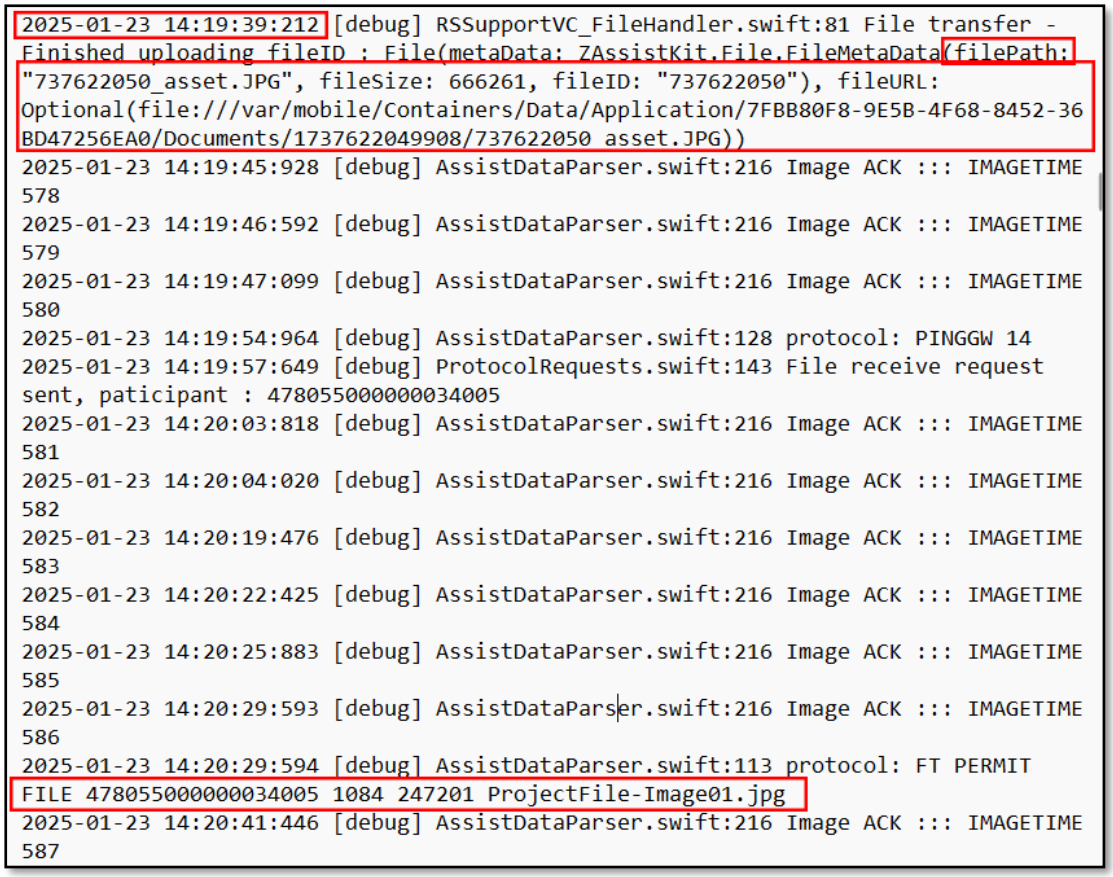
Application-specific preference files, including group.com.zoho.assist.customer.plist, com.zoho.assist.zohoassist.plist, com.zoho.assist.customer.plist, and group.com.zoho.assist.zohoassist.widget.plist, provide evidence of email addresses, session IDs, organization details, iOS version, and device model. These artifacts strongly support attribution of Zoho Assist activity to a specific user and device environment. Log files (e.g., com.zoho.assist.zohoassist YYYY-MM-DD_HH-MM-SS.log and com.zoho.assist.customer YYYY-MM-DD_HH-MM-SS.log) provide timestamped activity, session IDs, exchanged file names, and chat metadata, allowing reconstruction of session timelines even if chat content is absent.

Media artifacts, including session screenshots in paid versions, offer direct visual evidence of remote activity, enabling verification of what was shared during sessions. System-level artifacts like mobile_installation.log.0 document installation and uninstallation events, preserving a verifiable historical trail. This file is the only artifact that persists after uninstallation, allowing confirmation of prior app presence.

Free and paid versions differ in forensic yield: free versions produce identifiers, metadata, and basic logs, while paid versions generate richer evidence such as screenshots and file transfer data. After uninstallation, most user and configuration data

was removed, leaving only mobile_installation.log.0, which preserves minimal but reliable evidence for historical use.

In summary, Zoho Assist on iOS provides high-value forensic artifacts during active use, particularly for user-device linkage, session attribution, and visual evidence in paid versions. It's a clean uninstallation that limits post-removal analysis but retains minimal proof of prior installation for investigators.



```
2025-01-23 14:19:39:212 [debug] RSSupportVC_FileHandler.swift:81 File transfer -
Finished uploading fileID : File(metaData: ZAssistKit.File.FileMetaData(filePath:
"737622050_asset.JPG", fileSize: 666261, fileID: "737622050"), fileURL:
Optional(file:///var/mobile/Containers/Data/Application/7FBB80F8-9E5B-4F68-8452-36
BD47256EA0/Documents/1737622049908/737622050_asset.JPG))
2025-01-23 14:19:45:928 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
578
2025-01-23 14:19:46:592 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
579
2025-01-23 14:19:47:099 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
580
2025-01-23 14:19:54:964 [debug] AssistDataParser.swift:128 protocol: PINGGW 14
2025-01-23 14:19:57:649 [debug] ProtocolRequests.swift:143 File receive request
sent, paticipant : 478055000000034005
2025-01-23 14:20:03:818 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
581
2025-01-23 14:20:04:020 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
582
2025-01-23 14:20:19:476 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
583
2025-01-23 14:20:22:425 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
584
2025-01-23 14:20:25:883 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
585
2025-01-23 14:20:29:593 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
586
2025-01-23 14:20:29:594 [debug] AssistDataParser.swift:113 protocol: FT PERMIT
FILE 478055000000034005 1084 247201 ProjectFile-Image01.jpg
2025-01-23 14:20:41:446 [debug] AssistDataParser.swift:216 Image ACK ::: IMAGETIME
587
```

Figure 36: Represents the *com.zoho.assist.zohoassist 2025-01-23_08-34-10.523.log*file, highlighting artifacts such as timestamps and file exchange logs, including file name, size, and path.

4.1.4.4 macOS

4.1.4.4.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4p.

Table 4p: Represents the digital artifacts recovered from the installation state of Zoho Assist on macOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Zoho Assist	Info.plist	Applications\Zoho Assist Customer Plugin.app\Contents\Info.plist & Library\Application Support\ZohoAssist\Zoho.app\Contents\Info.plist & Applications\ZohoAssist\UnattendedUninstaller.app\Contents\Info.plist & Users\digitalsoni\Library\Application Support\ZohoMeetingNative\Zoho.app\Contents\Info.plist	Provides essential application metadata, including CFBundleName (app name), CFBundleVersion, and CFBundleIdentifier. These artifacts are vital for confirming application presence, exact version installed, and uniquely identifying the Zoho Assist app among other software. Correlation with system logs can validate installation and usage timeline.	Free & Paid
	000004.log	Users\digitalsoni\Library\Application Support\Zoho Assist\Local Storage\leveldb\000004.log	Contains the email address and name associated with the Zoho Assist account.	Paid Only
	000003.log	Users\digitalsoni\Library\Application Support\Zoho	Contains the email address linked to the Zoho Assist account.	Free & Paid

		Assist\Session Storage\000003.log	***Note:*** *File was present in the free version as well, but the email address was not stored.*	
	zohounattendeddetails.conf	Users\digitalsoni\Library\Application Support\ZohoMeetingNative\zohounattendeddetails.conf	Stores the email address configured with the Zoho Assist account.	Paid Only
	logFile.log	Users\digitalsoni\Library\Logs\ZohoAssist\Log\Zoho\logFile.log	Contains session ID, timestamped activity entries, chat message logs (without content), file names of transferred files, lock commands, terminal request commands, annotation entries, activity monitor entries, CMD entries, macOS device name, and details of file permissions and enabled features. (Figure 37)	Paid Only
	audit.log	Users\digitalsoni\Library\Logs\ZohoAssist\Audit\8412379734\audit.log	Stores session ID, email address, swap screen commands, and file transfer details (file name and size). Also includes entries for lock command, terminal request, activity monitor, and CMD usage.	Paid Only
	install.log	private\var\log\install.log	Provides timestamped entries of Zoho Assist installation, including file paths of installation packages. Confirms presence and deployment, and helps establish a timeline.	Free & Paid
	com.apple.finder.plist	Users\digitalsoni\Library\Preferences\com.apple.finder.plist	FXRecentFolders referencing Zoho Assist confirms recent access via Finder. FXDesktopVolumePosition shows how the app was positioned/organized on	Free & Paid

			the desktop, supporting user interaction evidence.	
	TCC.db	Users\digitalsoni\Library\Application Support\com.apple.TCC\TCC.db	Records permissions granted to Zoho Assist, including screen capture (kTCCServiceScreenCapture), full disk access (kTCCServiceSystemPolicyAllFiles), microphone (kTCCServiceMicrophone), and camera (kTCCServiceCamera). Confirms elevated access to sensitive system functions.	Free & Paid
	LaunchDaemons (Folder)	Library\LaunchDaemons	Contains multiple .plist files with file paths, program identifiers, and program labels related to Zoho Assist.	Free & Paid
	LaunchAgents(Folder)	Library\ LaunchAgents	Contains multiple .plist files with working directory, program arguments, watchpaths, and program labels.	Free & Paid

4.1.4.4.2 Uninstallation State

All the previously identified artifacts were found to persist even after the uninstallation of Zoho Assist from the macOS system. The files were successfully recovered from their respective directories, indicating that the uninstallation process did not completely remove user-specific or system-level data. While the artifact content remained consistent, the hash values of a few files were found to be different, likely due to minor changes in metadata or timestamp updates during system operations.

4.1.4.4.3 Interpretation and Forensic Significance

Zoho Assist on macOS generates a comprehensive set of artifacts that are highly valuable for user attribution, session reconstruction, and system activity assessment. Info.plist files across application, support, and uninstaller directories confirm the app's installation, CFBundleName, version, and unique identifier, allowing investigators to verify the exact application deployed and establish installation timelines when correlated with install.log.

User account linkage is evident in 000004.log, 000003.log, and zohounattendeddetails.conf, which store email addresses and account names, providing direct attribution. Activity logs such as logFile.log and audit.log capture session IDs, timestamps, chat metadata, file transfers, commands (lock, terminal, annotations), and device information, supporting detailed reconstruction of session behavior. Advanced activity records, including swap screen commands and CMD entries, are particularly useful for investigating insider threats or unauthorized remote operations.

System-level artifacts like com.apple.finder.plist and TCC.db document user interactions and permissions (screen capture, full disk access, microphone, camera), highlighting both forensic value and potential privacy/security implications. Persistence of LaunchDaemons and LaunchAgents confirms auto-start behavior and background execution.

Post-uninstallation, the majority of artifacts, including user identifiers, session logs, configuration files, and permission records, were retained, enabling reconstruction of historical usage even after software removal. This persistence strengthens forensic opportunities while also signaling residual data exposure risks.

In summary, Zoho Assist on macOS provides a rich and durable forensic footprint. Artifacts allow investigators to confirm installation, attribute users, reconstruct session activity, and analyze system integration, with post-uninstallation traces offering strong avenues for forensic recovery.

audit.log

*** Session ID ***
2025-1-23 14:23:16 Session : 8412379734

*** Participants ***

2025-1-23 14:23:17 Participant has joined : di[redacted]1@gmail.com

*** File Transfer ***

2025-1-23 14:26:15 Viewer -> Agent : File Name : trim.C86F6315-0573-4F3A-842B-3B3A82E34B27.MOV , File Size : 480370

*** ALT TAB ***

2025-1-23 14:28:33 ALT TAB Key Pressed

*** ALT TAB ***

2025-1-23 14:29:0 ALT TAB Key Pressed

*** Power Signals ***

2025-1-23 14:30:11 Lockscreen

*** Quick Launch ***

2025-1-23 14:30:40 Quick launch to Terminal

Figure 37: Represents the *audit.log*, highlighting artifacts such as timestamps, host Zoho Assist ID, email address, file transfer details (file name and size), ALT+Tab command logs, lock screen commands, and command terminal entry.

4.1.5 Anydesk

Anydesk was subjected to detailed forensic analysis across four major operating systems: Windows, Android, macOS, and iOS. Both installation and post-uninstallation states were examined to identify persistent. The application was tested in its free and licensed variants to evaluate any differences in artifact generation and retention. The results are organized platform-wise, presenting key evidence, including but not limited to logs, configuration files, system traces, and residual data.

4.1.5.1 Android

4.1.5.1.1 Installation State

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4q.

Table 4q: Represents the digital artifacts recovered from the installation state of AnyDesk on Android, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Anydesk	base.apk	app\com.anydesk.anydeskandroid-vi0suEytFGeHClbYVfZ92Q==\base.apk	Contains application metadata such as version details and desired permissions.	Free & Paid
	Downloads (Folder)	data\com.anydesk.anydeskandroid\files\downloads	Stores files received during sessions via the AnyDesk file transfer feature.	Paid only
	Recording (Folder)	data\com.anydesk.anydeskandroid\files\storage\recordings\outgoing digit (1129473299)-Android (858373511) 0.anydesk	This folder contains the recording of the session. The name of the recording contains the status of the session (Incoming/outgoing), the Receiver device name, and the AnyDesk ID of both client & Host.	Free & Paid

	com.anydesk.anydeskandroid_preferences.xml	user_de\0\com.anydesk.anydeskandroid\shared_prefs\com.anydesk.anydeskandroid_preferences.xml	This file contains the status of session (Incoming/outgoing) and AnyDesk ID of Connected device.	Free & Paid
	Thumbnails (Folder)	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\thumbnails\25743e8ec6db7db9.png	Stores thumbnails of connected devices. Figure 38)	Free & Paid
	Chat (Folder)	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\chat	Contains files named after the AnyDesk ID of the connected device. Each file stores the username of the receiver device, chat message content, and corresponding timestamp logs. (Figure 39)	Free & Paid
	user.conf	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\user.conf	Records Android device make/model, device email address, and AnyDesk ID of the connected device.	Free & Paid
	service.conf	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\service.conf	Stores the AnyDesk ID of the Android device.	Free & Paid
	system.conf	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\system.conf	Contains the AnyDesk ID of the Android device and the license type.	Free & Paid
	file_transfer_trace.txt	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\file_transfer_trace.txt	Logs details of file transfer events, including transfer method (clipboard/file manager), date and time, transfer status (start/finish), process type (upload/download), file name, and file size. (Figure 40)	Free & Paid
	connection_trace.txt	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\connection_trace.txt	Records incoming session logs, including date, time, and client AnyDesk ID. (Figure 41)	Free & Paid
	anydesk.trace	user_de\0\com.anydesk.anydeskandroid\files\.anydesk\anydesk.trace	Comprehensive log of session activities. Includes public and private IP addresses of both devices,	Free & Paid

			Android device brand and model, application versions of both devices, permissions granted, and AnyDesk IDs of both devices. ***Note**: Screenshot records (with file names) were found during licensed sessions.*	
	Screenshots	media\0\Pictures	Contains screenshots captured via the AnyDesk feature.	Free & Paid

4.1.5.1.2 Uninstallation State

After uninstalling AnyDesk (Free and Paid) on Android, only screenshot images were recovered. No logs, settings, or user data were found, indicating that AnyDesk apps uninstall cleanly on Android.

4.1.5.1.3 Interpretation and Forensic Significance

AnyDesk on Android produces a rich set of artifacts that are highly valuable for confirming application presence, user attribution, and session activity. The base.apk establishes the installed version and desired permissions, validating the deployment environment. Configuration files such as user.conf, service.conf, and system.conf store AnyDesk IDs, license details, associated emails, and device information, providing direct links between sessions, users, and devices.

Video recordings in the Recording folder and screenshots capture on-screen activity, offering direct visual proof of remote interactions. Chat logs and file_transfer_trace.txt record messages, file names, sizes, timestamps, and transfer types, enabling detailed reconstruction of communications and data exchange. Network artifacts in connection_trace.txt and anydesk.trace document device details, AnyDesk IDs, IP addresses, timestamps, and granted permissions, supporting timeline reconstruction and potential geolocation analysis. The file named as com.anydesk.anydeskandroid_preferences.xml provides the information such as AnyDesk ID of the connected device and the status of the session (Incoming or outgoing).

Graphical artifacts such as thumbnails and screenshots complement session recordings, allowing investigators to verify user actions and correlate events. Post-uninstallation, most artifacts are removed, with only screenshots persisting, limiting post-removal forensic opportunities but still providing partial evidence of prior use.

In summary, AnyDesk on Android generates highly informative forensic artifacts during installation and usage, including identifiers, session recordings, chat and file-transfer logs, and network traces. While uninstallation cleans most traces, persistent screenshots can offer investigators residual evidence to confirm historical activity (Soni et al., 2024b).

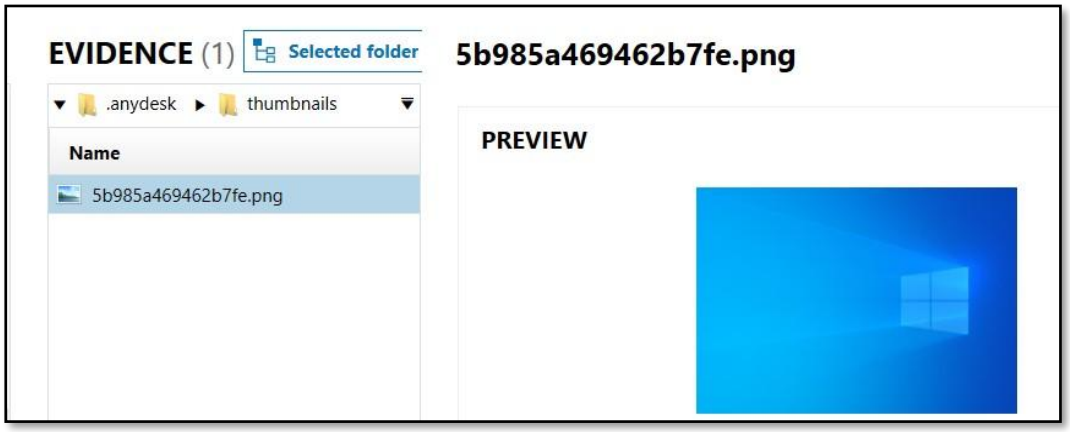


Figure 38: Represents the thumbnails of connected devices.

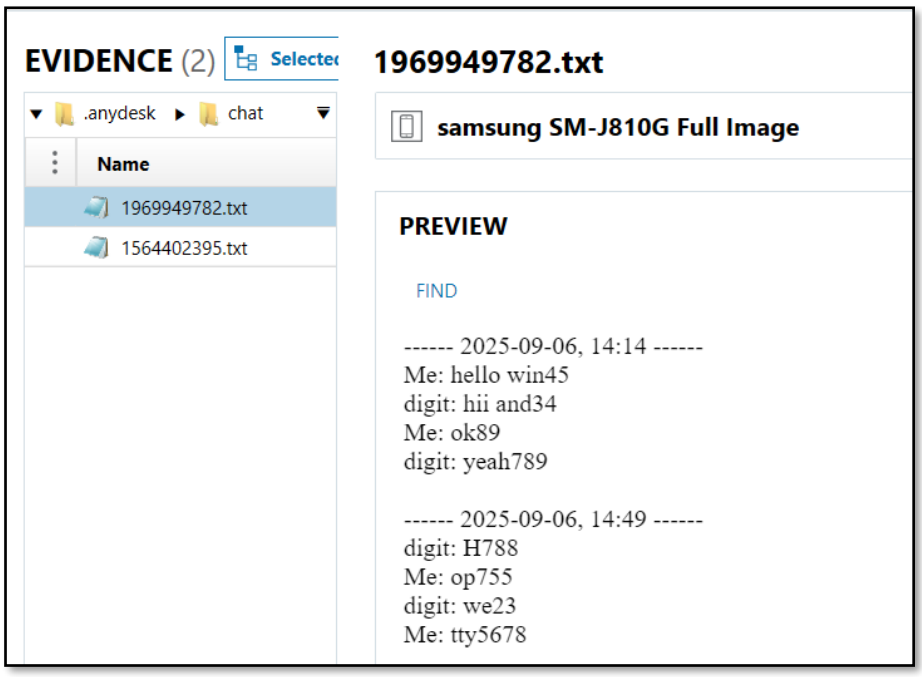


Figure 39: Represents the chat logs, highlighting artifacts such as timestamps, the name of the connected device, and sent and received messages.

```
File Manager   2025-09-06, 14:24   start    download   'ProjectFile-01.txt' (~0 B out of 258 B)
File Manager   2025-09-06, 14:24   finish   download   'ProjectFile-01.txt' (~258 B out of 258 B)
File Manager   2025-09-06, 14:24   start    upload     'Clipboarddata01.txt' (~0 B out of 82 B)
File Manager   2025-09-06, 14:24   finish   upload     'Clipboarddata01.txt' (~82 B out of 82 B)
File Manager   2025-09-06, 14:25   start    download   'ProjectFile-Video03.mp4' (~0 B out of 342.40 KiB)
File Manager   2025-09-06, 14:25   finish   download   'ProjectFile-Video03.mp4' (~342.40 KiB out of 342.40 KiB)
File Manager   2025-09-06, 14:26   start    upload     'ProjectFile-Image01.jpg' (~0 B out of 241.41 KiB)
File Manager   2025-09-06, 14:26   finish   upload     'ProjectFile-Image01.jpg' (~241.41 KiB out of 241.41 KiB)
```

Figure 40: Represents the *file_transfer_trace.txt* file, containing artifacts such as timestamps, file transfer events, transfer status (start/finish), process type (upload/download), file name, and file size.

```
Incoming   2025-09-06, 14:49   User   1969949782   1969949782
Incoming   2025-09-06, 14:54   User   1969949782   1969949782
Incoming   2025-09-06, 15:00   User   1969949782   1969949782
Incoming   2025-09-06, 15:48   User   1564402395   1564402395
```

Figure 41: Represents the *connection_trace.txt* file, highlighting artifacts such as incoming session logs, including date, time, and client AnyDesk ID.

4.1.5.2 Windows

4.1.5.2.1 Installation

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4r.

Table 4r: Represents the digital artifacts recovered from the installation state of AnyDesk on Windows, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Anydesk	Recording (Folder)	Users\<Username>\Videos\AnyDesk\outgoing samsung SM-J810G (858373511)-SYSTEM (1129473299) 0.anydesk	Contains session recordings. The recording name includes session status (incoming/outgoing), receiver device make/model, and AnyDesk IDs of both client and host.	Free & Paid
	Thumbnails (Folder)	Users\<Username>\AppData\Roaming\AnyDesk\thumbnails\414c1ca3551c142b.png	Stores thumbnails of connected devices.	Free & Paid
	Chat (Folder)	Users\<Username>\AppData\Roaming\AnyDesk\chat\858373511.txt	The file is named after the AnyDesk ID of the connected device. Contains receiver device make/model, date, and timestamps of chat messages, and message content.	Free & Paid
	user.conf	Users\<Username>\AppData\Roaming\AnyDesk\user.conf	Records the AnyDesk ID of the connected device. **Note:** *The Email address of the Windows system was found stored in the licensed version.* (Figure 42)	Free & Paid
	service.conf	ProgramData\AnyDesk\service.conf	Stores the AnyDesk ID of the Windows device.	Free & Paid

	system.conf	Users\<Username>\AppData\Roaming\AnyDesk\system.conf	Records the AnyDesk ID of the Windows device.	Free & Paid
	file_transfer_trace.txt	ProgramData\AnyDesk\file_transfer_trace.txt	Logs file transfer activity, including transfer method (Clipboard/File Manager), timestamp, transfer status (Start/Finish), process status (Upload/Download), file names, and file sizes.	Free & Paid
	connection_trace.txt	ProgramData\AnyDesk\connection_trace.txt	Records incoming session logs, such as timestamp and client AnyDesk ID.	Free & Paid
	ad_svc.trace	ProgramData\AnyDesk\ad_svc.trace	Contains public and private IP addresses of both devices and AnyDesk IDs. (Figure 43) **Note:** The *Licensed version also stores the email address of the Windows device.*	Free & Paid
	user.png (Avatar)	Users\<Username>\AppData\Roaming\AnyDesk\user.png	Stores the avatar of the user's AnyDesk account, if set.	Paid Only
	ad.trace	Users\<Username>\AppData\Roaming\AnyDesk\ad.trace	Logs comprehensive activity details, including Windows device public IP, AnyDesk IDs of both devices, screenshot request logs and save paths, as well as file transfer names and storage paths.	Free & Paid
	Screenshots	Users\<Username>\Pictures\AnyDesk	Contains screenshots captured via AnyDesk features.	Free & Paid
	Prefetch Files (Folder)	Windows\Prefetch	Stores run counts, last run times, additional execution times, and MAC timestamps. Useful for establishing frequency and timeline of usage.	Free & Paid
	Window event logs	Windows\System32\winevt\Logs	Windows Event Log entry (System.evtx) contains details of the user account and the associated process execution, including Process ID and full Process Name path.	Free & Paid

4.1.5.2.2 Uninstallation State

All the previously identified artifacts were found to persist even after the uninstallation of AnyDesk from the Windows system. The files were successfully recovered from their respective directories, indicating that the uninstallation process did not completely remove user-specific or system-level data. While the artifact content remained consistent, the hash values of a few files were found to be different, likely due to minor changes in metadata or timestamp updates during system operations.

4.1.5.2.3 Interpretation and Forensic Significance

The forensic footprint of AnyDesk on Windows is extensive, producing a diverse range of artifacts that collectively support user attribution, device linkage, and remote session reconstruction.

Artifacts related to session activity include the Recording folder (storing .anydesk session recordings) and the Screenshots folder (Users\<Username>\Pictures\AnyDesk), both of which provide direct visual evidence of remote interactions. These files allow investigators to validate session context by confirming on-screen content and device details embedded in filenames.

Communication traces are found in the Chat folder (AppData\Roaming\AnyDesk\chat), where text files named after AnyDesk IDs preserve chat messages, timestamps, and device information. These enable investigators to reconstruct communication flow during remote sessions.

Configuration and identification files, such as user.conf, service.conf, and system.conf record device and session-specific AnyDesk IDs, with licensed versions additionally storing the Windows system email address. The user.png avatar (present only in paid versions) further ties an AnyDesk account to a specific user identity.

Network and session metadata are captured in several artifacts. connection_trace.txt preserves incoming connection logs, including timestamps and client IDs. file_transfer_trace.txt provides detailed documentation of file transfer activity, including method, status, timestamps, filenames, and file sizes. ad_svc.trace and ad.trace store critical data such as public and private IP addresses, AnyDesk IDs,

screenshot request logs, and file storage paths, creating a comprehensive picture of session activity and network interactions.

Supporting artifacts include the Thumbnails folder, which stores images of connected devices, and Prefetch files (Windows\Prefetch), which record execution frequency, run counts, and MAC timestamps, allowing investigators to establish timelines of application usage. Additionally, Windows Event Logs (System.evtx) provide system-level evidence of process execution, including user account associations, process IDs, and execution paths, strengthening correlation between user activity and application execution.

Importantly, forensic persistence is observed even after application removal. All major artifacts—including recordings, configuration files, chat logs, file transfer traces, network logs, thumbnails, and screenshots—were found to remain intact post-uninstallation. While some file hashes showed minor variations, likely due to metadata updates, the evidentiary content persisted, ensuring continued investigative value.

In summary, AnyDesk on Windows generates one of the most comprehensive sets of digital artifacts among remote access applications. Its combination of visual evidence (recordings, screenshots), textual logs (chat, file transfer), configuration files (user.conf, system.conf, service.conf), and network traces (connection_trace.txt, ad_svc.trace, ad.trace) provides a multi-dimensional evidence base for reconstructing sessions, attributing users, and validating remote activity. The persistence of these artifacts post-uninstallation further enhances its forensic significance, enabling investigators to reconstruct historical use even after attempts at removal (Soni et al., 2024b).

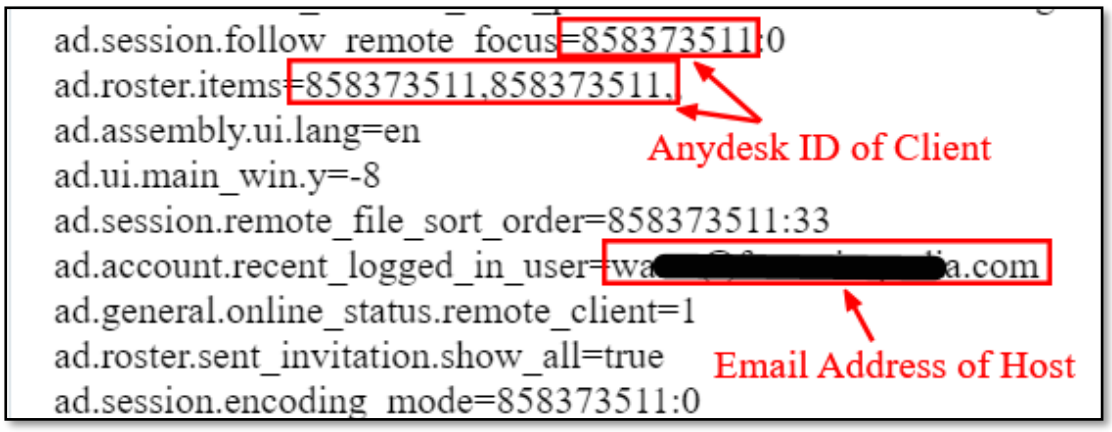


Figure 42: Represents the *user.conf* file, highlighting artifacts such as the AnyDesk ID of Client and email address of the host device.

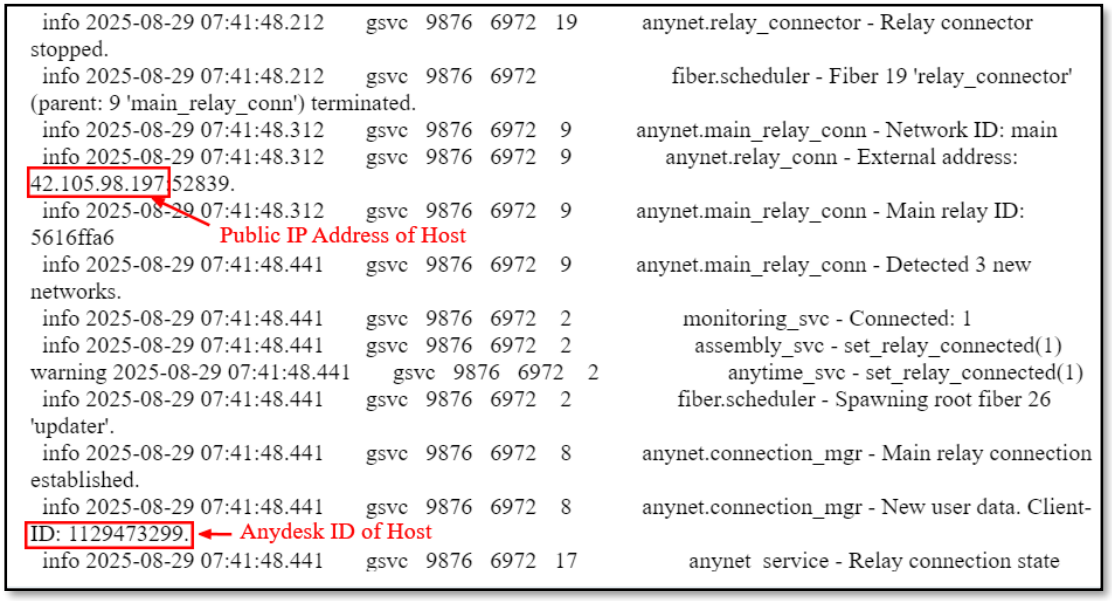


Figure 43: Represents the *ad_svc.trace* file, highlighting artifacts such as timestamps, the host public IP address, and the AnyDesk ID.

4.1.5.3 iOS

4.1.5.3.1 Installation

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4s.

Table 4s: Represents the digital artifacts recovered from the installation state of AnyDesk on iOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
Anydesk Remote Desktop	Chat (Folder)	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/.anydesk/chat/1128343375.txt	Contains chat logs associated with the connected device's AnyDesk ID. Records include connected device username, chat message content, and timestamp of received messages.	Free & Paid
	Thumbnails (Folder)	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/.anydesk/thumbnails/440cfb7ecee9f0f6.png	Contains the thumbnail of the connected device.	Free & Paid
	user.conf	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/.anydesk/user.conf	Records the AnyDesk ID of the connected device.	Free & Paid
	service.conf	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/.anydesk/service.conf	Records the Anydesk ID of the iOS device.	Free & Paid

	system.conf	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/.anydesk/system.conf	Records the device's own AnyDesk ID and license type.	Free & Paid
	com.anydesk.AnyDesk.plist	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Preferences/com.anydesk.AnyDesk.plist	Stores the AnyDesk ID of the iOS device along with the application download date and time in UTC.	Free & Paid
	anydesk.trace	/private/var/mobile/Containers/Data/Application/<UUID>/Library/Application Support/anydesk.trace	Detailed log of session activities, including public and private IP addresses of both devices, application versions, permissions granted, AnyDesk IDs of both devices, clipboard activity (without content), session timeout logs, and session start/end with recording initialization logs. (Figure 44a and 44b)	Free & Paid
	mobile_installation.log.0	/private/var/installd/Library/Logs/MobileInstallation/mobile_installation.log.0	Records application installation and uninstallation events, including precise timestamps, application package file paths, and package names. Confirms installation or removal of AnyDesk.	Free & Paid

4.1.5.3.2 Uninstallation State

After uninstalling the AnyDesk (Free and Paid) on iOS, only a single residual file named mobile_installation.log.0 was identified. This file captures detailed information related to the installation and uninstallation processes of applications on iOS. Specifically, it records the exact timestamp of each event, the file path of the application package, and its corresponding package name. Although no additional user data or configuration files were found after uninstallation, the presence of mobile_installation.log.0 stands as a crucial forensic artifact for confirming the historical existence of the AnyDesk app on the device.

4.1.5.3.3 Interpretation and Forensic Significance

AnyDesk on iOS generates artifacts that provide critical evidence for user identification, session activity reconstruction, and device linkage. Chat logs capture connected device usernames, full message content, and timestamps, enabling investigators to reconstruct conversations during remote sessions, which can be crucial in cases involving social engineering, insider threats, or unauthorized access. The Thumbnails directory preserves visual identifiers of connected devices, supporting device attribution within sessions.

Configuration files such as user.conf, service.conf, and system.conf documents AnyDesk IDs and license information, directly linking local and remote devices. The com.anydesk.AnyDesk.plist file records the device's AnyDesk ID along with download timestamps, providing timeline verification that can be cross-referenced with system logs. The anydesk.trace file serves as a rich forensic resource, logging IP addresses, device identifiers, session permissions, clipboard interactions, and session start/end markers, while also documenting recording events. These details facilitate the precise reconstruction of user actions and the assessment of security risks.

System-level logs, particularly mobile_installation.log.0, capture installation and uninstallation events, providing verifiable evidence of the application's presence even when user-generated artifacts are deleted.

Following uninstallation, nearly all artifacts are removed, leaving only the installation log. Despite its limited content, this residual file confirms historical application usage, an essential aspect in cases where attempts have been made to conceal activity.

In summary, AnyDesk on iOS produces a forensic footprint that, although less persistent than on Android or Windows, offers significant evidentiary value through chat logs, session metadata, device identifiers, and installation records. The surviving artifacts ensure investigators can validate prior usage and reconstruct aspects of session activity when available.

```
   info 2025-09-05 06:30:31.949         ad    svc    346  10300    7
anynet.main_relay_conn - Network ID: main
   info 2025-09-05 06:30:31.961         ad    svc    346  10300    7
anynet.relay_conn - External address: 42.105.10.32:16881.
   info 2025-09-05 06:30:31.971         ad    svc    346  10300    7
anynet.main_relay_conn - Reporting system information.
   info 2025-09-05 06:30:31.971         ad    svc    346  10300    7
anynet.main_relay_conn - Main relay ID: 2f2a9920
   info 2025-09-05 06:30:31.972         ad    svc    346  10300    7
anynet.main_relay_conn - Detected 0 new networks.
   info 2025-09-05 06:30:31.972         ad    svc    346  10300   13
anynet_service - Relay connection state changed: Connected.
   info 2025-09-05 06:30:31.990         ad    svc    346  10300    6
anynet.connection_mgr - Main relay connection established.
   info 2025-09-05 06:30:31.990         ad    svc    346  10300    6
anynet.connection_mgr - New user data. Client-ID: 1311406769.
```

Figure 44a: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, the public IP address, and AnyDesk ID of the client.

```
glwrap - GL Version: 20 (0)
   info 2025-09-05 06:46:30.666         ad    svc    350  11072   59
app.session - Starting session recording.
   info 2025-09-05 06:46:31.357         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Connect failed (result_timeout)).
   info 2025-09-05 06:46:31.614         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Connecting (lport 49170, attempt
2).
   info 2025-09-05 06:46:46.618         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Connect failed (result_timeout)).
   info 2025-09-05 06:46:46.872         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Connecting (lport 49170, attempt
3).
   info 2025-09-05 06:47:01.878         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Connect failed (result_timeout)).
  error 2025-09-05 06:47:01.878         ad    svc    350  11072   68
anynet.punch_connector - [49.15.105.111:21351] Too many retries.
   info 2025-09-05 06:47:01.878         ad    svc    350  11072
fiber.scheduler - Fiber 68 'tcp_punch' (parent: 63) terminated.
   info 2025-09-05 06:47:01.878         ad    svc    350  11072   63
anynet.punch_connector - All connect fibers have failed.
   info 2025-09-05 06:47:06.288         ad    svc    350  11072   59
app.session - Stopping session recording.
   info 2025-09-05 06:47:06.883         ad    svc    350  11072   63
anynet.punch_connector - Quit timeout has expired. Connect failed.
```

Figure 44b: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, the public IP address, and AnyDesk ID of client.

4.1.5.4 macOS

4.1.5.4.1 Installation

The application was installed and actively used, after which artifacts were extracted before uninstallation. The details of the recovered artifacts, including their file paths, content, and license type (indicating whether the artifact originated from the free or licensed version), are presented in Table 4t.

Table 4t: Represents the digital artifacts recovered from the installation state of AnyDesk on macOS, including corresponding file names, storage locations, Content, and license-specific availability.

Application Name	File Name(s)	File Path	Content	License Type
AnyDesk	Info.plist	Applications\AnyDesk.app\Contents\Info.plist	Provides metadata such as CFBundleName, CFBundleVersion, and CFBundleIdentifier. Useful for confirming the exact application installed, its version, and uniquely identifying it among other apps on the system. Supports correlation with system logs and activity records.	Free & Paid
	Chat (Folder)	Users\digitalforensics\.anydesk\chat\1138165049.txt	Stores chat logs named after the AnyDesk ID of the connected device. Includes receiver device name, timestamp, and chat content.	Free & Paid
	user.conf	Users\digitalforensics\.anydesk\user.conf.bak & Users\digitalforensics\.anydesk\user.conf	Records the AnyDesk ID of the connected device.	Free & Paid
	system.conf	Users\digitalsoni\.anydesk\system.conf	Records the device's own AnyDesk ID and license type.	Free & Paid

	Thumbnails (Folder)	Users\digitalsoni\.anydesk\thumbnails	Contains thumbnail images of connected devices.	Free & Paid
	anydesk.trace	private\var\log\anydesk.trace	Session activity log. Records public and private IP addresses of both devices, AnyDesk IDs, version details, file-sharing directory paths, remote reboot requests, screenshot request names, and saved paths. (Figure 45)	Free & Paid
	Screenshot	/Users/digitalsoni/Desktop	Screenshots captured via AnyDesk feature are stored on the desktop.	Free & Paid
	Recording (Folder)	Users\digitalsoni\.anydesk\AnyDesk\outgoing samsung SM-J810G (1146523170)-digitalsoni (1564402395) 0.anydesk	Contains session recordings. File names indicate session type (incoming/outgoing), receiver device make/model, and AnyDesk IDs of both client and host. (Figure 46)	Free & Paid
	com.apple.finder.plist	Users\digitalsoni\Library\Preferences\com.apple.finder.plist	Presence of FXRecentFolders referencing AnyDesk confirms recent interaction via Finder. FXDesktopVolumePosition reveals desktop arrangement, supporting evidence of user activity.	Free & Paid
	LaunchDaemons (Folder)	Library\LaunchDaemons	Contains multiple .plist files related to AnyDesk, including program arguments and labels.	Free & Paid
	LaunchAgents(Folder)	Library\ LaunchAgents	Contains .plist files related to AnyDesk. These files reference the installed application directory path.	Free & Paid

4.1.5.4.2 Uninstallation State

All the previously identified artifacts except Info.plist (Applications\AnyDesk.app\Contents\Info.plist) were found to persist even after the

uninstallation of AnyDesk from the macOS system. The files were successfully recovered from their respective directories, indicating that the uninstallation process did not completely remove user-specific or system-level data. While the artifact content remained consistent, the hash values of a few files were found to be different, likely due to minor changes in metadata or timestamp updates during system operations.

4.1.5.4.3 Interpretation and Forensic Significance

AnyDesk on macOS produces a rich and comprehensive set of artifacts that are highly valuable for user attribution, session reconstruction, and timeline establishment. Chat logs capture full message content, device names, and timestamps, providing direct evidence of user interactions. The Thumbnails directory visually links sessions to specific endpoints, supporting device-level attribution.

Configuration files such as user.conf and system.conf record AnyDesk IDs of local and connected devices, along with license information, enabling precise mapping of sessions to accounts and machines. The anydesk.trace file logs extensive session metadata, including IP addresses, version details, file paths, remote reboot and screenshot requests, and storage locations, allowing near-complete reconstruction of remote access activity.

Multimedia artifacts, including session recordings and screenshots, provide visual confirmation of user actions during live sessions. These serve as direct evidence of on-screen activity and corroborate other system and user-level logs. System-level artifacts such as com.apple.finder.plist and entries in LaunchDaemons and LaunchAgents further support evidence of application usage and persistence across system reboots.

Post-uninstallation, almost all artifacts remain intact, except for the installation Info.plist. The persistence of configuration files, logs, session recordings, and screenshots preserves substantial evidentiary value, allowing forensic investigators to reconstruct remote access activities and link them to specific users and devices.

In summary, AnyDesk on macOS generates one of the most artifact-rich forensic footprints among the platforms studied. Its combination of textual, visual, and system-level evidence ensures robust opportunities for attribution, session reconstruction, and timeline validation, even after the application has been uninstalled.

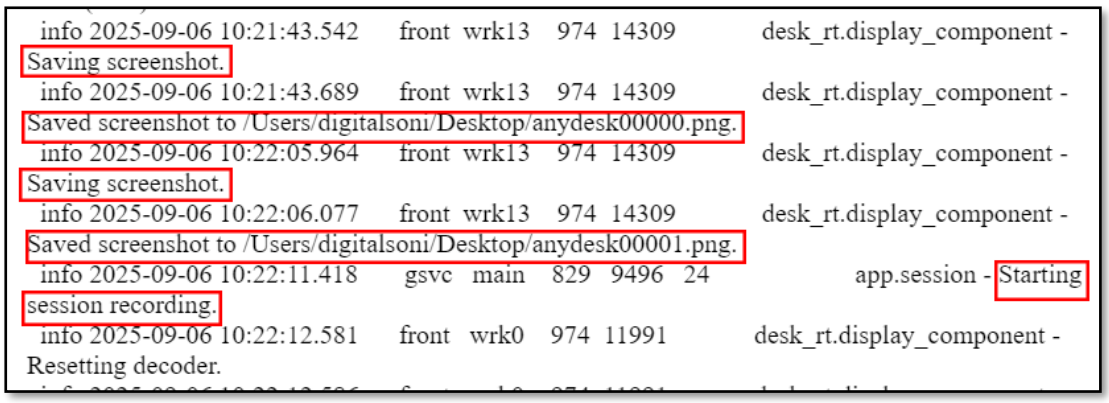


Figure 45: Represents the *anydesk.trace* file, highlighting artifacts such as timestamps, screenshot logs with saved file paths, and session recording logs.

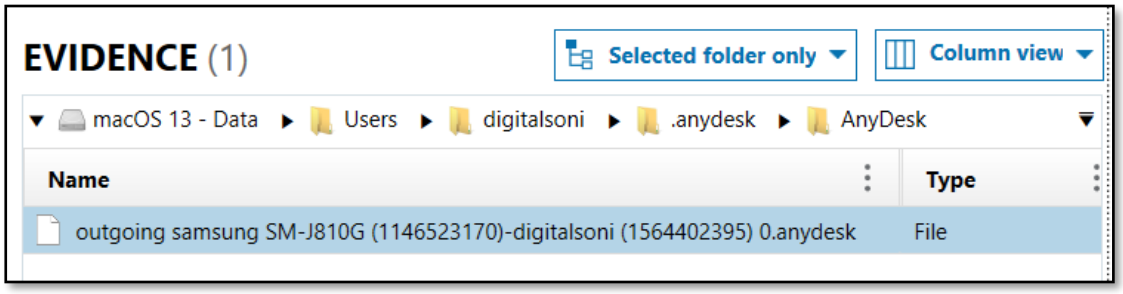


Figure 46: Represents the session recording file, whose name consists of the connection type, host device name (make/model), host AnyDesk ID, client device name, and client AnyDesk ID, formatted as: *outgoing samsung SM-J810G (1146523170)-digitalsoni (1564402395) 0.anydesk*.

4.2 Discussion

The forensic analysis of artifacts generated by the selected RAAs, such as TeamViewer, LogMeIn (GoToMyPC), RemotePC, Zoho Assist, and AnyDesk across Android, Windows, iOS, and macOS, reveals a diverse range of digital evidence, influenced by multiple factors such as application variants (e.g., free vs. licensed), operating systems, and installation/uninstallation states.

The result of this study demonstrated that Remote Access Applications (RAAs) produce a diverse range of artifacts, including but not limited to logs, configuration files, databases, media files (e.g., screenshots and session recordings), and system-level traces (e.g., prefetch files and event logs). These artifacts are crucial for investigators to investigate and reconstruct the incidents of unauthorized access, as these artifacts capture user identifiers (e.g., email addresses, usernames, device names), session metadata (e.g., timestamps, IP addresses, host/client IDs), and activity details (e.g., file transfers, chats, clipboard, screenshots, lock screen commands, app uninstallation). For instance, TeamViewer's TeamViewer15_Logfile.log and TVLog.html, along with AnyDesk's ad.trace and anydesk.trace files, emerged as a rich source for the identification of IP addresses, device details, and session commands across platforms, enabling timeline reconstruction and device attribution (Kerai & Vekariya, 2016; Manson, 2022; Pañeda et al., 2024; Soni et al., 2024a; Soni et al., 2024b).

Significant variations were observed between free and licensed versions of Remote Access Applications (RAAs)(Table 4u), with licensed variants typically producing more comprehensive and forensically valuable artifacts. For TeamViewer, paid versions generated unique artifacts such as TVLogOld.html (Android), which includes audio call metadata, and session recording files like samsung_SM-J810G_ffa0c9cad54ec054 (1 078117778)_2024-10-0915.09.08.tvs (Windows) and Session name_2025-02-06 11.22.23.tvs (macOS), which are absent in free versions. Zoho Assist's paid versions were particularly rich in artifacts, producing logs such as ACServer.log, IPCClientLog.log, LogFile.log, LogFileTemp.log, LogFileTray.log, PeerConnection.log, unattended.log, upgrader.log, ZAAudioClient.log, ZAService.log, ZohoURSService.log, and Connect.log (Windows), as well as 000004.log, zohounattendeddetails.conf, logFile.log, and audit.log (macOS). On iOS, Zoho Assist's

paid versions included Shared File, Screenshots, com.zoho.assist.customer.plist, and com.zoho.assist.customer 2025-01-23_09-15-45.726.log, which either contained richer data or were absent in free versions. AnyDesk's licensed versions contributed to the user.png avatar file (Windows) for user identification and the Downloads folder (Android) for file transfer details, both limited to licensed installations. These additional artifacts, such as screenshots, session recordings, detailed logs, and user-specific files, provide critical evidence for reconstructing session activities, attributing users, and analyzing system interactions, significantly enhancing and contributing to the forensic investigations. In contrast, RemotePC and LogMeIn (GoToMyPC) exhibited no such difference in artifact generation between their free and licensed versions, as both variants produced similar types of configuration files, logs, and system metadata (e.g., unattended.json for GoToMyPC, com.prosoftnet.rpcnew.xml for RemotePC), limiting their forensic differentiation. This difference underscores the enhanced investigative potential of licensed versions of TeamViewer, Zoho Assist, and AnyDesk, particularly in capturing dynamic session data and user interactions across platforms.

Table 4u: Forensic Artifacts Identified in Licensed Versions of Remote Access Applications Across Operating Systems

RAA	Android	Windows	iOS	macOS
TeamViewer	TVLogOld.html (audio call metadata), Screenshots	Email address present in TeamViewer artifacts, richer session logs; .tvs session recording generated in licensed version	kSecClassGenericPassword additionally stores device email address in licensed version	Session_name_YYYY-MM-DD_HH.MM.SS.tvs (session recording file)
GoToMyPC	No Difference identified	No Difference identified	No Difference identified	No Difference identified
RemotePC	No Difference identified	No Difference identified	No Difference identified	No Difference identified

Zoho Assist	Screenshots, session_history_database, session_history_database-wal, ZA_UNATTENDED_FRAMEWORK_SHARED_PREF.xml,	ACServer.log, IPCClientLog.log, LogFile.log, LogFileTemp.log, LogFileTray.log, PeerConnection.log, unattended.log, upgrader.log, ZAAudioClient.log, ZAService.log, ZohoURSService.log, Connect.log	Shared file folder, group.com.zoho.assist.customer.plist, com.zoho.assist.customer.plist, com.zoho.assist.customer 2025-01-23_09-15-45.726.log, group.com.zoho.assist.zohoassist.widget.plist and screenshots available through licensed features	000004.log, zohounattendeddetails.conf, logFile.log, audit.log, enhanced session logs and unattended-access artifacts
AnyDesk	Downloads folder artifacts generated through licensed file-transfer functionality	user.png, email address stored within user.conf and ad_svc.trace in licensed version	No Difference identified	No Difference identified

This licensing-based divergence is better understood as a feature-driven rather than a tier-driven effect. TeamViewer, Zoho Assist, and AnyDesk reserve their most session-intensive capabilities, such as audio-call recording, unattended access, screen recording, and tray-resident background services, for paid tiers, and each of these capabilities requires the application to maintain state across the lifetime of a session rather than simply relaying input and display data. Maintaining that state necessarily produces a corresponding artifact: a recording must be written to storage, an unattended-access session must record its authorization state, and a background service must log its own activity to remain auditable. The forensic richness of the paid tier is therefore a structural consequence of which features are licensed, not of licensing itself. This is corroborated by the absence of a comparable split for RemotePC and GoToMyPC, where the free and licensed tiers differ chiefly in account-level entitlements, such as the number of permitted devices or support priority, rather than in

the underlying session-handling logic; because the software performs the same stateful operations regardless of tier, it generates the same category of artifacts regardless of tier. This distinction suggests that artifact richness in RAAs should be predicted from an application's feature architecture rather than treated as an inherent property of its free or paid status, a principle that is likely to generalize to other RAAs and to the embedded remote-support modules discussed later in this section.

Operating system differences significantly impacted artifact persistence after the uninstallation of Remote Access Applications (Table 4v). Windows and macOS retained a wealth of artifacts due to their deep system integration. For instance, Zoho Assist on Windows preserved all logs, such as session and activity logs, along with Prefetch files and Windows Event Logs. Similarly, AnyDesk on Windows kept session recordings, chat logs, configuration files, and system traces. On macOS, Zoho Assist retained post-uninstallation artifacts such as user data, session logs, and system files like LaunchDaemons, LaunchAgents, and install logs, while AnyDesk preserved similar artifacts, except for the Info.plist file. The persistence of artifacts after uninstallation of the application from their respective device, supported by artifacts including but not limited to macOS's LaunchDaemons or Windows' Prefetch files, provides rich evidence for reconstructing user activity and sessions even after software removal. In contrast, iOS and Android exhibited cleaner uninstalls, leaving minimal traces. On iOS, apps like Zoho Assist, AnyDesk, and TeamViewer left only mobile_installation.log.0, which logs installation details, while GoToMyPC also retained a Keychain entry. On Android, TeamViewer and Zoho Assist left library.db and screenshots, and AnyDesk preserved only screenshots. These limited mobile artifacts, mostly installation logs or media files, reflect stricter data cleanup but still offer proof of prior app usage for forensic analysis.

Table 4v: Comparative Persistence of Forensic Artifacts After Uninstallation Across Remote Access Applications and Operating Systems

Sr. No.	Remote Access Applications	Android	Windows	iOS	macOS
1	TeamViewer	Moderate	High	Low	Moderate
2	GoToMyPC	Low	Moderate	Low	Low
3	RemotePC	Low	High	Low	Moderate
4	Zoho Assist	Moderate	High	Low	Moderate
5	AnyDesk	Moderate	High	Low	Moderate

Key: High = A substantial number of forensic artifacts remained recoverable, Moderate =Some significant artifacts remained available after uninstallation, Low = Very limited residual artifacts remained after uninstallation

This persistence gradient across operating systems is best explained by differences in how each platform's architecture governs application lifecycle and data ownership, rather than by chance variation in uninstaller design. Windows and macOS are designed around a multi-user, file-system-exposed model in which applications are permitted to write configuration files, logs, and caches outside their own installation directory, into locations such as AppData, ProgramData, Application Support, and system-level logging stores (e.g., Prefetch, Windows Event Logs, LaunchDaemons, LaunchAgents, and install.log). Because uninstallers on these platforms are conventionally scoped to remove only the application's primary installation folder and registry or property-list entries explicitly registered with the package manager, artifacts deposited in these peripheral, OS-managed locations fall outside the uninstaller's defined scope and are therefore left behind by default rather than through any deliberate forensic safeguard. iOS and Android, by contrast, are built on a sandboxed, mobile-first security model in which each application is confined to its own container (e.g., /data/data/<package> on Android or the app's UUID-scoped directory on iOS), with limited ability to write persistent data outside that boundary. When such an application is uninstalled, the operating system reclaims the entire sandbox as a single unit, which removes most user-generated artifacts together with the application itself; the few traces that do survive

(e.g., mobile_installation.log.0, Keychain entries, or Play Store library records) persist precisely because they are written by the operating system or an OS-level service rather than by the RAA itself, placing them outside the sandbox that gets reclaimed at uninstallation. This explains why persistence is not simply a property of the application but an emergent outcome of the interaction between an application's write behaviour and the host platform's data-ownership boundaries. A similar logic underlies the licensing-based differences observed for TeamViewer, Zoho Assist, and AnyDesk: paid tiers unlock features that are inherently stateful and session-bound (call recording, unattended access, screen-recording, and tray-level service logging), and any feature that must persist state across a session necessarily writes a corresponding artifact, whereas RemotePC and GoToMyPC's free and licensed tiers differ mainly in account entitlements rather than in session-handling features, which is why their artifact sets remain materially similar regardless of licensing status.

Comparative analysis of Remote Access Applications reveals distinct differences in the forensic value of their artifacts (Table 4w). AnyDesk, Zoho Assist, and TeamViewer produced the most comprehensive and forensically significant artifacts across all platforms, capturing detailed logs of IP addresses, chat messages, file transfers, and media files like screenshots and session recordings. For instance, AnyDesk's anydesk.trace and file_transfer_trace.txt, Zoho Assist's logFile.log and audit.log, and TeamViewer's TeamViewer15_Logfile.log and TVLog.html provide robust evidence of session activities, user interactions, and network data, enabling precise reconstruction of remote access events. These applications also generated rich visual evidence, such as AnyDesk's thumbnails and recordings, Zoho Assist's screenshots (paid versions), and TeamViewer's screenshots and .tvs session recordings (e.g., samsung_SM-J810G_ffa0c9cad54ec054 (1078117778)_2024-10-09 15.09.08.tvs on Windows, Session name_2025-02-06 11.22.23.tvs on macOS), enhancing attribution and activity verification. In contrast, LogMeIn (GoToMyPC) and RemotePC produced fewer session-specific artifacts, primarily focused on configuration files (e.g., GoToMyPC's unattended.json, RemotePC's com.prosoftnet.rpcnew.xml) and system metadata like installation logs and browser data. This emphasis on static system configuration over dynamic user or session data, such as limited or absent logs for chats, file transfers, or

screen captures, restricts the investigation and reconstruction of the incident in forensic investigations. Consequently, AnyDesk, Zoho Assist, and TeamViewer offer a superior forensic artifact range for tracing unauthorized access and reconstruction of session timelines compared to the more limited evidentiary output of other RAAs like LogMeIn and RemotePC. The same session-feature-to-artifact relationship observed across these dedicated RAAs is likely to extend to cloud-based collaboration platforms that embed remote-support functionality within a broader enterprise suite, such as the remote-control modules bundled with Microsoft Teams, Cisco Webex, or Zoom. Because these tools execute their remote-session logic within an existing application sandbox rather than as a stand-alone installed program, their artifacts would be expected to inherit the access-and-logging behaviour of the host platform and browser or client process, rather than generating dedicated installation- and uninstallation-state traces of the kind documented for AnyDesk, TeamViewer, Zoho Assist, RemotePC, and GoToMyPC in this study; this distinction in artifact origin, rather than mere coverage, is what differentiates dedicated RAAs from embedded enterprise remote-support features and is examined further as a direction for future work.

Table 4w: Comparative Extraction of Forensic Artifacts Across Remote Access Applications and Operating Systems

Sr. No.	**Remote Access Applications**	**Android**	**Windows**	**iOS**	**macOS**
1	TeamViewer	High	High	High	High
2	LogMeIn (GoToMyPC)	Low	Moderate	Low	Moderate
3	RemotePC	High	Moderate	Low	Low
4	Zoho Assist	High	High	High	High
5	AnyDesk	High	High	Moderate	High

Key: High = Recovery of a broad range of forensic artifacts,
Moderate = Recovery of a moderate range of forensic artifacts,
Low = Recovery of a low range of forensic artifacts

The findings of this study are constructed upon prior studies on RAAs forensics, which were focused on individual applications or platforms. For example, Manson (2022)

conducted a forensic analysis of TeamViewer on Windows, identifying similar logs and configuration files (e.g., Connections.txt), but noted limited persistence post-uninstallation, in contrast to the results of this study, where all artifacts were found to persist on Windows. This difference may arise from version differences, underscoring the need for updated analyses as software evolves (Boztas et al., 2025). Similarly, Pañeda et al. (2024) examined AnyDesk and TeamViewer on Windows for traces of file exfiltration, correctly noting that AnyDesk logs (anydesk.trace, ad_svc.trace) do not include the names of transferred files. However, their claim that TeamViewer logs do not store the destination IP address contrasts with the findings of this study, which confirmed that the TeamViewer15_Logfile.log contains both host and client IP addresses.

Kerai and Vekariya (2016) examined GoToMyPC on Windows, reporting that the application did not log details on the local computer and no logs were stored in Windows event logs, in contrast to the present study, where the System.evtx file was found on the Windows system, containing relevant event log data. The artifacts identified offer substantial value for digital forensic investigations, particularly in cybercrime cases involving unauthorized access, data exfiltration, or insider threats. For example, persistent logs post-uninstallation (e.g., in Windows Event Logs or macOS install.log) can counter suspect denials by proving historical RAA presence.

A blind study was conducted on Remote Access Applications (RAAs) across Android, Windows, iOS, and macOS to validate the results of this study. It covered all selected RAAs, including TeamViewer, LogMeIn, RemotePC, Zoho Assist, and AnyDesk. The blind testing successfully confirmed the presence of most artifacts identified in the main study, including logs (e.g., TeamViewer15_Logfile.log, anydesk.trace), configuration files (e.g., user.conf, com.prosoftnet.rpcnew.xml), system logs (e.g., mobile_installation.log.0, prefetch files), user identifiers (e.g., email addresses, session IDs), session metadata (e.g., IP addresses, timestamps), and media files (e.g., screenshots, session recordings). Minor variations in the detection of artifacts were observed, but these did not compromise the overall validity of the findings. This blind testing validated the consistency and forensic significance of the identified artifacts, confirming their reliability in real-case investigations.

CHAPTER 5

SUMMARY AND CONCLUSION

5.1 Summary

Remote Access Applications (RAAs) have become the most important part of modern computing environments. The ability of RAAs to provide real-time connectivity across geographical boundaries has positioned them as indispensable tools for legitimate use in business, education, and IT support. However, the important features such as remote access and screen mirroring make them attractive for productivity and efficiency, and have also made them a weapon of choice for cybercriminals. Fraudsters and threat actors are increasingly misusing RAAs for unauthorized system access, remote financial scams, and data exfiltration, leaving investigators with the challenge of proving their usage and establishing accountability of incidents. This dual-use nature of RAAs highlights their importance in digital forensic research.

The present research was initiated to conduct a systematic forensic analysis of selected RAAs, namely TeamViewer, AnyDesk, RemotePC, GoToMyPC, and Zoho Assist. These applications are widely adopted and frequently referenced in cybercrime incidents. As earlier studies focused on a single application and were limited to one platform, this study undertook a cross-platform and cross-license comparative examination of artifacts in both installation and uninstallation states. The objective of this study was to identify the range of artifacts generated, evaluate their persistence across different operating systems and licensing states, and determine their forensic relevance.

The study was guided by the following objectives:

1. To determine and compare a range of artifacts extracted from selected licensed Remote Access Applications and their free version.
2. To compare the extracted artifacts of Remote Access Applications on devices with different operating systems.
3. To conduct a forensic analysis of the artifacts generated by remote access applications after installation and uninstallation from a device.

4. To conduct blind study on extraction of artifacts from selected Remote Access Applications to determine real case scenario with minimum one-fourth of the total samples.

A structured methodology was employed to achieve the above objectives. Each selected application was installed and actively used on multiple operating systems under controlled experimental conditions. Activities, including but not limited to remote access, file transfers, chats, and screen sharing, were carried out in both free and licensed versions. Following this, artifacts were extracted using forensic suites including Magnet AXIOM, MD-NEXT, MD-RED, DB Browser for SQLite, and PECmd.

A total of 100 samples were analyzed, comprising structured experiments and blind samples. The structured experiments were designed to capture artifacts across different application states and license types, while the blind samples were conducted to validate the findings of the present study.

The findings of this study revealed that RAAs produce a rich and diverse set of artifacts. On Windows systems, artifacts were abundant and persistent across all RAAs. TeamViewer generated rolloutfile.tv13 (unique device ID), Connections.txt (session metadata), and TeamViewer15_Logfile.log (IP addresses, device identifiers, file transfers, and timestamps). AnyDesk created user.conf (usernames and connection details), service.conf (session settings), and ad.trace logs recording connections and errors. RemotePC produced HostPref.ini and HostPref.plist, containing device associations and login identifiers, along with logs under ProgramData\RemotePC\Logs. GoToMyPC retained g2cc.json files in AppData documenting session IDs, user email addresses, and timestamps, while Zoho Assist produced ACServer.log and ZohoAgent.conf, which captured controller IDs and connection events. Even after uninstallation, prefetch files and Windows Event Logs persisted for all applications, establishing proof of historical usage.

In macOS environments, structured artifacts were consistently observed in plist files and Application Support directories. TeamViewer created com.TeamViewer.plist and detailed logs under ~/Library/Logs/TeamViewer/. AnyDesk stored identifiers in

com.philandro.anydesk.plist and connection metadata in ad_svc.trace. RemotePC maintained host configuration in RemotePC.plist, while Zoho Assist generated persistent session logs in ZohoAssistDaemon.log. GoToMyPC saved preferences and identifiers in com.citrix.gotopc.plist. These plist and log artifacts proved particularly resilient, surviving uninstallation and retaining user identifiers, session history, and application metadata.

On Android devices, multiple RAAs generated SQLite databases, configuration files, and logs in app-specific directories. TeamViewer produced TVLog.html, recording IP addresses, timestamps, and device details, while global.conf and client.conf preserved cryptographic keys and usernames. AnyDesk generated anydesk.trace and service.conf, which contained session logs and configuration details. RemotePC left traces in shared_prefs XML files and logs under /data/data/com.remotepc/, whereas Zoho Assist stored artifacts in ZohoAssist.xml and ZohoAssistService.log. GoToMyPC created g2viewer.db databases containing account names, session IDs, and activity logs. Screenshots were recoverable across applications in /DCIM/Screenshots and the library.db file from the Play Store consistently confirmed installation history, even post-uninstallation.

In iOS systems, forensic artifacts were more restrictive due to sandboxing, but critical evidence was still recoverable. TeamViewer artifacts included com.TeamViewer.Settings.plist, Connections.txt, and Keychain entries storing encryption keys and user identifiers. AnyDesk created adtrace.log files along with preference lists capturing client IDs. RemotePC stored artifacts in HostPref.plist, which contained hostnames, account IDs, and connection timestamps. Zoho Assist logs retained controller identifiers, session IDs, and timestamps within ZohoAssist.plist, while GoToMyPC left behind session records in g2m.plist and installation history in mobile_installation.log.0. Screenshots in /Media/DCIM/ and system-level logs provided further corroborative evidence of usage.

The comparison of free and licensed versions highlighted that free variants produced limited identifiers and session traces, often restricted to basic IDs, timestamps, or minimal logs, whereas licensed versions generated enriched evidence, including extended log entries, persistent configuration files, and session recordings (.tvs for

TeamViewer, and extended activity logs in RemotePC and Zoho Assist). These differences significantly enhanced the ability to reconstruct timelines and attribute activities.

Blind testing validated the repeatability of artifact detection across all RAAs as majority of artifacts were consistently detected in unsupervised samples, proving their evidentiary reliability in real-world investigations.

In conclusion, the present study established that all examined RAAs leave behind multi-layered forensic footprints across platforms. Persistence of these artifacts beyond uninstallation ensures their utility in proving historical usage and reconstruction of incident timeline. Hence, the study successfully achieved all its objectives by systematically identifying and comparing forensic artifacts from both free and licensed versions of RAAs, examining their variations across multiple operating systems, such as Windows, macOS, Android, and iOS platforms, and documenting their persistence in both installation and uninstallation states. Lastly, the incorporation of blind testing further validated the consistency and reliability of the recovered artifacts. This comprehensive framework not only fills critical gaps in existing forensic research but also strengthens investigative methodologies, enhancing the capacity of digital forensics to tackle crimes involving the misuse of remote access technologies.

5.2 Conclusion

- Remote Access Applications (RAAs) consistently generated a diverse range of forensic artifacts across all tested operating systems (Windows, macOS, Android, and iOS), license types (free and licensed), and operational states (installation, usage, and uninstallation). These artifacts collectively provide a reliable foundation for reconstructing timeline of remote access activities and hold substantial significance for digital forensic investigations. Comparative evaluation of licensed and free versions revealed notable differences. Licensed editions were observed to contain more features and consequently generated a larger volume of artifacts, including session recordings, detailed log entries, user authentication

data, and additional identifiers such as registered email addresses and device association details.

- The cross-platform analysis confirmed that Windows yielded the richest forensic artifacts, supported by extensive system logging and open file structures. macOS demonstrated remarkable artifact persistence and structured evidence storage in plist and log files. Android environments produced meaningful evidence through shared_prefs XMLs, SQLite databases, and configuration logs, while iOS artifacts were more restricted but still provided critical identifiers, preference files, and installation logs, particularly useful for confirming application presence and usage history.
- A key outcome of this research is the persistence of artifacts even after the uninstallation of RAAs. In Windows systems, residual evidence such as Prefetch files, event logs, and leftover log files was consistently available, allowing investigators to reconstruct prior usage. In macOS, persistence was even more pronounced, with artifacts such as plist files, Application Support folders, and logs remaining intact post-uninstallation, making macOS one of the most valuable environments for persistence analysis. In contrast, Android and iOS devices displayed limited post-uninstallation remnants, yet preference files, SQLite records, and installation logs still provided sufficient traces to establish prior presence and activity of the applications.
- The blind study validates the findings of the present research, confirming the reliability and reproducibility of the artifact extraction and analysis across different operating systems and RAAs.
- The present study holds significant importance and clear superiority over previous works due to its integrated, systematic, and multi-dimensional forensic examination framework. Unlike earlier studies that focused on a single application, a single operating system, or limited usage states, this research simultaneously evaluates multiple widely used Remote Access Applications across four major platforms, free and licensed versions, and both installation and post-uninstallation conditions. The incorporation of blind sample validation further enhances the

scientific rigor and real-world reliability of the findings. By documenting artifact locations, persistence behavior, and evidentiary significance in a structured manner, this research fills critical gaps in existing literature and provides a validated forensic framework that benefits both academic researchers and digital forensic practitioners. The outcomes offer direct operational value to investigators, law enforcement agencies, and judicial processes by improving evidence recovery efficiency, strengthening legal admissibility, and reducing the risk of evidentiary oversight in cybercrime investigations.

- All objectives of the present study have been successfully achieved, and the findings are expected to support future research, aid in the development of forensic tools, and assist practitioners in real-world investigations involving remote access applications.

5.3 Limitations and Future Recommendations

Mobile devices used in this study were rooted or jailbroken to enable complete artifact acquisition; however, in practical investigations, only a small proportion of devices are available in such modified states. The scope of the research was limited to five RAAs and did not include other emerging remote access applications that may demonstrate different artifact generation and persistence patterns. Finally, the continuous release of updated versions of RAAs may influence artifact structures and storage mechanisms.

Future research should extend this work by examining cloud or browser-based remote access and dual-nature collaboration applications, which are increasingly adopted in both enterprise and personal contexts because of their portable use. Many other digital forensic techniques such as volatile memory forensics and network traffic analysis with artifact-based examinations can be adopted, which would further allow for a more holistic understanding of artifact generation and persistence. The development of semi-automated forensic tools that can identify, extract, and correlate artifacts related to RAA would significantly aid digital forensic investigations. Finally, longitudinal studies that monitor the evolution of artifacts across application updates and OS can also be adopted to ensure forensic methodologies remain adaptive to rapid technological change.

BIBLIOGRAPHY

Altschaffel, R., Clausing, R., Kraetzer, C., Hoppe, T., Kiltz, S., & Dittmann, J. (2013). Statistical Pattern Recognition Based Content Analysis on Encrypted Network: Traffic for the TeamViewer Application. 2013 Seventh International Conference on IT Security Incident Management and IT Forensics. https://doi.org/10.1109/imf.2013.19

AnyDesk. (2025). *AnyDesk: the Fast remote desktop application*. https://anydesk.com/en/try-anydesk-today

Azhar, M. a. H. B., Timms, J., & Tilley, B. (2022). Forensic Investigations of Google Meet and Microsoft Teams – Two Popular Conferencing Tools in the Pandemic. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 20–34. https://doi.org/10.1007/978-3-031-06365-7_2

Bal, Y., & Bulgur, N. E. (2023). Remote work. In *Advances in human resources management and organizational development book series* (pp. 374–391). https://doi.org/10.4018/978-1-6684-9172-0.ch019

Bowling, H., Seigfried-Spellar, K., Karabiyik, U., & Rogers, M. (2023). We are meeting on Microsoft Teams: Forensic analysis in Windows, Android, and iOS operating systems. Journal of Forensic Sciences, 68(2), 434–460. https://doi.org/10.1111/1556-4029.15208

Cybersecurity Ventures Magazine (2025). *Cybercrime to cost the world $10.5 trillion annually by 2025*. Cybercrime Magazine. https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/

DB Browser for SQLite. (2025) https://sqlitebrowser.org/

ET Bureau. (2019, March 16). RBI, CID warn of frauds using remote access app. *The Economic Times*. https://economictimes.indiatimes.com/news/politics-and-nation/rbi-cid-warn-of-frauds-using-remote-access-app/articleshow/68403703.cms?from=mdr

Free Press Journal. (2024, December 14). Indore Cyber Crime roundup: ₹2.45 lakh lost in AnyDesk scam, three booked for tender fraud, & OLX seller. *Free Press*

Journal. https://www.freepressjournal.in/indore/indore-cyber-crime-roundup-245-lakh-lost-in-anydesk-scam-three-booked-for-tender-fraud-olx-seller-duped-of-20000

GMDSOFT. (2025, August 18). *Mobile Device Forensic Software | Android Forensic Tool*. GMDSOFT ». https://www.gmdsoft.com/product/mobile-forensic-software/

GoToMyPC. (2025). *Remote desktop software & Remote computer access.* https://get.gotomypc.com/

Hamid, I., & Rahman, M. M. H. (2024). A Comprehensive Literature Review on Volatile Memory Forensics. *Electronics*, *13*(15), 3026. https://doi.org/10.3390/electronics13153026

Hammazahmed. (2024, August 9). *Digital forensics tutorial: Analyzing prefetch files with pecmd.exe*. Towards Dev. https://towardsdev.com/digital-forensics-tutorial-analyzing-prefetch-files-with-pecmd-exe-64665767f7ec

Heidelberg. (2025). TeamViewer for Remote Service. https://www.heidelberg.com/us/media/local_media/services___whitepapers/whitepaper-teamviewer.pdf

Hoehe, M. R., & Thibaut, F. (2020). Going digital: how technology use may influence human brains and behavior. *Dialogues in Clinical Neuroscience*, *22*(2), 93–97. https://doi.org/10.31887/dcns.2020.22.2/mhoehe

Hur, U., Kim, G., Kang, S., & Kim, J. (2023). Forensic analysis for multi-platform Cisco Webex. Forensic Science International Digital Investigation, 47, 301659. https://doi.org/10.1016/j.fsidi.2023.301659

India, T. O. (2024, December 14). Stay alert, fake apps give access to your phone: Cops. *The Times of India*. https://timesofindia.indiatimes.com/city/kolkata/stay-alert-fake-apps-give-access-to-your-phone-cops/articleshow/116323902.cms

Iqbal, F., Khalid, Z., Marrington, A., Shah, B., & Hung, P. C. (2022). Forensic investigation of Google Meet for memory and browser artifacts. Forensic Science International: Digital Investigation, 43, 301448. https://doi.org/10.1016/j.fsidi.2022.301448

Ismail, A., Hajjar, M., & Hajjar, H. (2008). REMOTE ADMINISTRATION TOOLS: A COMPARATIVE STUDY. *Journal of Theoretical & Applied Information Technology*, *4*(2). https://www.jatit.org/volumes/research-papers/Vol4No2/Remote%20Administration%20Tool%20(RAT),%20LAN,%20WAN,%20control,%20WMI,%20RPC,%20Web.pdf

Jiang, M., Gou, G., Shi, J., & Xiong, G. (2019). I Know What You Are Doing with Remote Desktop. International Performance, Computing, and Communications Conference. https://doi.org/10.1109/ipccc47392.2019.8958721

Kerai, P. (2010). Tracing VNC And RDP Protocol Artefacts on Windows Mobile and Windows Smartphone for Forensic Purpose. International Cyber Resilience Conference. https://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1006&context=icr

Kerai, P. (2012). Remote Access Forensics for Vnc and Rdp on Windows Platform. Edith Cowan University. https://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1080&context=adf

Kerai, P., & Vekariya, V. M. (2016). An exploration of artefacts of remote desktop applications on Windows. *Australian Digital Forensics Conference*. https://doi.org/10.4225/75/58a54f83180cc

Khalid, Z., Iqbal, F., Al-Hussaeni, K., MacDermott, A., & Hussain, M. (2022a). Forensic Analysis of Microsoft Teams: Investigating Memory, Disk and Network. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 583–601. https://doi.org/10.1007/978-3-031-06371-8_37

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). https://doi.org/10.1109/csnet52717.2021.9614647

Khalid, Z., Iqbal, F., Kamoun, F., Khan, L. A., & Shah, B. (2022b). Forensic investigation of Cisco WebEx desktop client, web, and Android smartphone applications. Annales Des Telecommunications. https://doi.org/10.1007/s12243-022-00919-6

Klasen, L., Fock, N., & Forchheimer, R. (2024). The invisible evidence: Digital forensics as key to solving crimes in the digital age. *Forensic Science International*, *362*, 112133. https://doi.org/10.1016/j.forsciint.2024.112133

Lahaie, C., & Leberfinger, D. (2013). TeamViewer Forensics. The Senator Patrick Leahy Center for Digital Investigation Champlain College. https://www.champlain.edu/documents/lcdi/archive/team-viewer-forensics.pdf

Magnet Forensics. (2025). *Magnet AXIOM | Digital Forensic Software | Magnet Forensics*. https://www.magnetforensics.com/products/magnet-axiom/

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, 301107. https://doi.org/10.1016/j.fsidi.2021.301107

Mallick, A. I., & Nath, R. (2024). Navigating the Cyber security Landscape: A Comprehensive Review of Cyber-Attacks, Emerging Trends, and Recent Developments. *World Scientific News*, *190*(1), 1–69. https://worldscientificnews.com/wp-content/uploads/2024/01/WSN-1901-2024-1-69-1.pdf

Manson, J. (2022). Remote Desktop Software as a forensic resource. *Journal of Cyber Security Technology*, *6*(1–2), 1–26. https://doi.org/10.1080/23742917.2022.2049560

NDTV. (2023, January 17). Man loses rs 5 lakh in online fraud after downloading anydesk app. *NDTV*. https://www.ndtv.com/india-news/thane-man-loses-rs-5-lakh-in-online-fraud-3698703

Paligu, F., & Varol, C. (2022). Microsoft Teams desktop application forensic investigations utilizing IndexedDB storage. Journal of Forensic Sciences. https://doi.org/10.1111/1556-4029.15014

Paneda, X. G., Melendi, D., Corcoba, V., Pañeda, A. G., García, R., & García, D. (2024). Forensic analysis of file exfiltrations using AnyDesk, TeamViewer and Chrome Remote desktop. Electronics, 13(8), 1429. https://doi.org/10.3390/electronics13081429

Raghavan, S. (2012). Digital forensic research: current state of the art. *CSI Transactions on ICT*, *1*(1), 91–114. https://doi.org/10.1007/s40012-012-0008-7

RemotePC. (2025). *Remote Desktop by RemotePC - top-rated Remote Access Software*. https://www.remotepc.com/

Sahut, J. M., & Lissillour, R. (2022). The adoption of remote work platforms after the Covid-19 lockdown: New approach, new evidence. *Journal of Business Research*, *154*, 113345. https://doi.org/10.1016/j.jbusres.2022.113345

Saidi, R. M., Ahmad, S. A., Noor, N. M., & Yunos, R. (2013). Windows registry analysis for forensic investigation. International Conference on Technological Advances in Electrical, Electronics and Computer Engineering. https://doi.org/10.1109/taeece.2013.6557209

Sixo Online APK Analyzer. (2025). https://www.sisik.eu/apk-tool.

Soni, N., Kaur, M., & Aziz, K. (2024a). Decoding digital interactions: An extensive study of TeamViewer's Forensic Artifacts across Windows and android platforms. *Forensic Science International Digital Investigation*, *51*, 301838. https://doi.org/10.1016/j.fsidi.2024.301838

Soni, N., Kaur, M., & Bhardwaj, V. (2024b). A forensic analysis of AnyDesk Remote Access application by using various forensic tools and techniques. *Forensic Science International Digital Investigation*, *48*, 301695. https://doi.org/10.1016/j.fsidi.2024.301695

StatCounter Global Stats. (2025). *Operating system market* share worldwide-StatCounter Global Stats. https://gs.statcounter.com/os-market-share#monthly-202502-202502-bar

Statista. (2025, April 1). *Worldwide digital population 2025*. https://www.statista.com/statistics/617136/digital-population-worldwide/

TeamViewer. (2025, August 29). *TeamViewer—The Digital Workplace platform*. https://www.teamviewer.com/

Tiwari, N. S. K. (2025). Forensics Analysis of Browser-Based GoToMeeting clients: Uncovering memory and browser artefacts. Journal of Information Systems

Engineering & Management, 10(20s), 483–495. https://doi.org/10.52783/jisem.v10i20s.3172

VMware Workstation Player, 2025. VMware. https://www.vmware.com/in/products/workstation-player/workstation-player-evaluation.html.

Zoho Assist. (2025). *Best Remote Access Software-Remote support software*. https://www.zoho.com/assist/

List of Publications

1. Soni, N., Kaur, M., & Aziz, K. (2024). Decoding digital interactions: An extensive study of TeamViewer's Forensic Artifacts across Windows and android platforms. *Forensic Science International Digital Investigation*, *51*, 301838. https://doi.org/10.1016/j.fsidi.2024.301838

2. Soni, N., Kaur, M., & Bhardwaj, V. (2024). A forensic analysis of AnyDesk Remote Access application by using various forensic tools and techniques. *Forensic Science International Digital Investigation*, *48*, 301695. https://doi.org/10.1016/j.fsidi.2024.301695

List of Conferences

1. Presented a paper titled *"Forensic Analysis of the AnyDesk Remote Access Application with the Help of Forensic Tools and Techniques"* at the National Conference on Forensic Advancements & Technological Explorations (FATE), held on 25–26 November 2022, organized by the Department of Forensic Science, Lovely Professional University, Punjab.
2. Presented a paper titled *"Uncovering Digital Forensic Artifacts in RemotePC: A Deep Dive into Screen Mirroring Fraud on Android Platform"* at the Forensic ज्ञान-2024 International Conference, held on 14–15 November 2024 at Vivekananda Global University, Jaipur.
3. Awarded Best Presentation for the paper titled *"Cross-Platform Digital Forensic Investigation of AnyDesk Remote Access Tool on macOS and iOS"* at NYAYARTH 2025 – National Conference on Law, Language, and Forensic Science: A Synergistic Approach to Justice, held on 8–9 July 2025 at Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore

Forensic Science International: Digital Investigation 48 (2024) 301695

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

A forensic analysis of AnyDesk Remote Access application by using various forensic tools and techniques

Nishchal Soni [a], Manpreet Kaur [a,*], Vishwas Bhardwaj [b]

[a] Department of Forensic Science, School of Bioengineering and Biosciences, Lovely Professional University, Phagwara, Punjab, 144001, India
[b] State Forensic Science Laboratory, Rajasthan, India

ARTICLE INFO

Keywords:
Disk forensics
Digital artifacts
Remote access application
AnyDesk
Digital forensics

ABSTRACT

This study delves into a forensic analysis of the AnyDesk Remote Access application, focusing prominently on disk forensic acquisitions. We aim to assess the security and privacy features of AnyDesk, uncovering insights vital for forensic investigators and potential adversaries. The recovery of artifacts from Android Mobile and Window-based PC devices, employing acquisition techniques, plays a pivotal role in forensic analysis. The study underscores the significance of log files, housing crucial details like user IDs, dates, transfer times, and file movements. Manual scrutiny of the extracted data establishes user connections and reveals user-centric information, encompassing wallpapers, chat logs, AnyDesk-IDs, and transferred files. As the data lacks encryption, artifacts are easily comprehensible and interlinked. AnyDesk-related files, including session recordings, media files, and documents, undergo successful extraction via forensic methods. Root permissions on the Android phone emerge as a critical asset, facilitating the identification of more reliable and concealed data. In contrast, on the PC, all files related to AnyDesk were identified through a combination of automatic and manual examination. In essence, this study provides profound insights into AnyDesk's security and privacy features, underscored by the instrumental role of forensic acquisitions in pinpointing and extracting pertinent data.

1. Introduction

The ubiquitous presence of electronic devices and advancements in technology have revolutionized the field of forensic science. The rise of digital technology has brought about a new era of connectivity and innovation, transforming the way we access information, conduct business, and interact with one another. This transformation has led to significant advancements in the field of forensic science, including the specialized area of digital forensics, which plays a critical role in modern investigations (Beebe, 2009).

Digital forensics focuses on the investigation and analysis of electronic devices and digital data. It involves the process of recovering, preserving, and analyzing digital evidence, such as emails, text messages, internet activity, metadata, etc., using specific tools and techniques. Digital forensics experts extract data from various devices like computers, smartphones, tablets, and servers, utilizing specialized software and hardware tools for analysis. These experts aim to identify patterns, trends, and abnormalities in the data (Raghavan, 2012).

The field of digital forensics has become increasingly important as digital crimes continue to grow. Criminals are becoming more advanced in exploiting digital technologies for illicit activities, such as cyberbullying, identity theft, hacking, and online scams. Remote access applications (RAAs) have been utilized by criminals for several years to commit cybercrimes.

One such RAA is AnyDesk, a cross-platform remote desktop software application that allows users to access their computers remotely using an AnyDesk ID. AnyDesk is known for its lightweight and user-friendly design, making it a popular choice for both personal and professional use. It offers advanced features such as file transfer, remote printing, and session recording, making it a versatile tool for remote access and support. AnyDesk provides businesses and organizations with the ability to offer remote support to employees, customers, and clients without physical access to the computer (AnyDesk, n.d.)

RAAs like AnyDesk provide convenience and flexibility for remote work, especially during events like the COVID-19 pandemic that have necessitated remote working arrangements (Lissillour and Sahut, 2023). However, it's important to recognize that criminals may also exploit RAAs for illegal purposes (Guide to Securing Remote Access Software | CISA, 2023). Remote access applications can be used as malware, like Remote Access Trojans (RATs), which enable unauthorized access and

* Corresponding author.
E-mail address: dr.kaurmanpreet2790@gmail.com (M. Kaur).

https://doi.org/10.1016/j.fsidi.2024.301695
Received 22 September 2023; Received in revised form 24 January 2024; Accepted 25 January 2024
Available online 6 February 2024
2666-2817/© 2024 Elsevier Ltd. All rights reserved.

Forensic Science International: Digital Investigation 51 (2024) 301838

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

Decoding digital interactions: An extensive study of TeamViewer's Forensic Artifacts across Windows and android platforms

Nishchal Soni [a], Manpreet Kaur [a,*], Khalid Aziz [b]

[a] School of Bioengineering and Biosciences, Lovely Professional University, Phagwara, Punjab, 144001, India
[b] School of Computer Application, Lovely Professional University, Phagwara, Punjab, 144001, India

ARTICLE INFO

Keywords:
Disk forensics
Memory forensics
Registry forensics
Android forensics
Remote access application
TeamViewer

ABSTRACT

The pervasive influence of digital technology has ushered in a new era of connectivity, reshaping the landscape of forensic science and challenging investigators to adapt to evolving methods of digital interaction. Remote access applications (RAAs) like TeamViewer have become integral tools for facilitating remote collaboration and support across various platforms. However, the widespread adoption of such applications has also led to an increase in cybercrimes, underscoring the critical need for meticulous forensic analysis. This study presents a comprehensive examination of TeamViewer's forensic artifacts across Windows and Android platforms, employing advanced forensic techniques such as registry analysis, disk forensics, memory forensics, and Android forensics. By meticulously dissecting digital evidence and uncovering valuable insights into user interactions, configuration settings, and session dynamics, this research aims to enhance understanding of remote access activities and empower forensic investigators with the tools needed to combat cybercrimes effectively. The findings highlight the forensic significance of each investigative approach and underscore the importance of continuous innovation in the field of digital forensics.

1. Introduction

In the rapidly evolving digital landscape of the 21st century, our dependence on technology has surged, profoundly influencing how we conduct business, communicate, and engage with the world. At the vanguard of this digital transformation are Remote Desktop Software (RDS) solutions like TeamViewer, which have fundamentally reshaped the paradigm of remote access, facilitating effortless connectivity and collaborative endeavors across expansive geographical. Notably, in 2020 alone, TeamViewer experienced an extraordinary growth of 75 %, highlighting its pivotal role in enabling remote work, technical support, and virtual collaboration amidst the global shift towards digitalization boundaries (TeamViewer - Investor Relations,).

TeamViewer stands as a dominant force in the realm of Remote Desktop Software (RDS), offering comprehensive support across all major Operating Systems and capturing a significant market share, comprising over a third of the market. Its widespread adoption extends to prestigious entities, with over 90 % of Fortune 500 companies incorporating TeamViewer into their operations. In critical sectors like healthcare, TeamViewer is a trusted ally, relied upon for its seamless connectivity and robust features. In fact, data sources often designate TeamViewer as the undisputed market leader, boasting a market share that doubles that of its closest competitors (Manson, 2022).

However, with the convenience afforded by RDS comes a myriad of security implications that cannot be overlooked. The inherent nature of these tools, necessitating deep integration into system functionalities, renders them susceptible to exploitation by malicious actors seeking unauthorized access to sensitive data and critical infrastructure. High-profile cyberattacks, such as the widely publicized breach of the Colonial Pipeline, serve as stark reminders of the vulnerabilities inherent in remote access solutions and the urgent need for robust forensic investigation protocols (Kerner, 2022).

Against this backdrop of heightened cybersecurity threats and the escalating sophistication of cybercriminal tactics, the role of digital forensics in uncovering, analyzing, and interpreting digital evidence has never been more crucial (Alghamdi, 2021). Digital investigators face the daunting task of unraveling the intricate digital trails left behind by RDS applications like TeamViewer, discerning between benign user activity and malicious intent to reconstruct the sequence of events leading up to a cybersecurity incident.

Our research endeavors to shed light on this complex forensic landscape by conducting a comprehensive analysis of the forensic

* Corresponding author.
E-mail address: dr.kaurmanpreet2790@gmail.com (M. Kaur).

https://doi.org/10.1016/j.fsidi.2024.301838
Received 6 March 2024; Received in revised form 25 August 2024; Accepted 28 September 2024
Available online 4 October 2024
2666-2817/© 2024 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

Certificate No.261490

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

Certificate of Participation

This is to certify that **Prof./Dr./Mr./Ms. Nishchal Soni** of **Lovely professional University** has participated in **Oral Presentation** on the topic entitled **Forensic analysis of the Anydesk remote access application with the help of forensic tools and techniques** in **National Conference on 'Forensic Advancements & Technological Explorations' (FATE)** held on 25-26 November 2022 organized by Department of Forensic Science, School of Bio-engineering and Biosciences, in association with Patiala Bureau of Identification, Punjab, organized at Lovely Professional University, Punjab.

Date of Issue : 03-01-2023
Place: Phagwara (Punjab), India

Prepared by
(Administrative Officer-Records)

Dr. Jyoti Dalal
Organizing Secretary, FATE, 2022

Dr. Neeta Raj Sharma
Convener, FATE, 2022



SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE

1st NATIONAL CONFERENCE- NYAYARTH, 2025

On

Law, Language and Forensic Science: A Synergistic Approach to Justice

Certificate

This is to certify that...Mr. NISHCHAL SONI.....

of .....LOVELY PROFESSIONAL UNIVERSITY.....has presented a paper titled "CROSS-PLATFORM DIGITAL FORENSIC INVESTIGATION OF ANYDESK REMOTE ACCESS TOOL ON MACOS & iOS..."

His/Her paper has been adjudged as the Best Paper ..IN FORENSIC SCIENCE.....

in the National Conference (Nyayarth 2025) jointly organized by Shri Vaishnav Institute of Law and Shri Vaishnav Institute of Forensic Science on 8-9 July 2025.

Dr. Swati Dubey Mishra	Dr. T.N. Prasad	Dr. Yogesh C. Goswami
Convenor	Convenor	Vice Chancellor
Coordinator, SVIFS - SVVV, Indore	Director, SVIL - SVVV, Indore	SVVV, Indore