International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 06 | Jun 2023 www.irjet.net p-ISSN: 2395-0072

A REVIEW PAPER ON API MALWARE ANALYSIS AND FORENSICS

VEESAM SAI VAMSI

Student, Department of Information Security, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India.

Abstract - A significant area of cybersecurity study is API Malware Analysis and Forensics. It is crucial to have effective defences in place to identify and stop malware assaults through APIs since they can have catastrophic effects. The goal of the article is to give a broad overview of the state of the art in API malware analysis and forensics, as well as the methods and equipment employed to identify, evaluate, and counteract API-based malware attacks.

The paper will explore the various kinds of malware that can be distributed through APIs, such as backdoors, command and control (C2) servers, and Remote Access Trojans (RATs). Additionally, an overview of the various methods for detecting malware in APIs, such as static and dynamic analysis, will be provided.

This paper's overall goal is to provide a thorough overview of API malware analysis and investigation, encompassing several methods and instruments that are used to find and examine API malware. This study also emphasizes the significance of taking proactive steps to prevent API-based malware attacks, such as routinely testing APIs for vulnerabilities, putting security protocols in place, and utilizing cutting-edge security technologies to detect and mitigate API-based malware assaults.

Key Words: Malware Analysis, Forensics Investigations, Malware Attacks.

1.INTRODUCTION

Software programmers can connect and interact with one another thanks to a collection of protocols, procedures, and tools called an application programming interface (API). Regardless of the underlying hardware and operating systems, APIs offer a standard method for data and service interchange across various software components [10] [12].

In software development, APIs play a crucial role in creating modular and scalable applications. By using APIs, developers can break down complex systems into smaller, independent components that can be developed, tested, and deployed separately. APIs also allow developers to reuse existing code and services, saving time and reducing development costs [10] [12].

APIs can be used in a variety of software applications, including web applications, mobile apps, and desktop software. They are often used to integrate different software systems, such as connecting a front-end web application to a back-end database [10] [12].

However, APIs can also be a potential security vulnerability. Malicious actors can exploit weaknesses in API design and implementation to launch attacks, such as API malware attacks. Therefore, it is important for developers and security professionals to implement proper security measures and conduct regular API security testing to ensure the integrity of their systems [10].

1.1 KEY CHARACTERISTICS OF API SECURITY THAT DISTINGUISH IT FROM TRADITION SECURITY

1. A castle with many openings and no moat: In the past, popular ports like 80 (HTTP) and 443 (HTTPS) were all that needed to be protected on traditional networks. There are several API endpoints that employ various protocols in today's web apps. Even one API can make security a challenging task because APIs often grow over time [3].

2. Incoming request formats that change frequently: In a DevOps context, APIs change quickly, and most WAFs cannot handle this level of elasticity. Traditional security tools require manual tuning and reconfiguration whenever an API changes, which is an error-prone procedure that uses up resources and time [3].

3. Clients often do not use a web browser: The majority of native and mobile applications, as well as other services and software components, have access to service or microservice APIs. Web security technologies cannot employ browser verification on these clients since they don't use browsers. Automated traffic from API endpoints is typically difficult to exclude for solutions that rely on browser verification to detect harmful bots [3].

Examining incoming requests does not guarantee detecting attacks; many API abuse attacks exploit requests that look legitimate.

1.2 THREAT OF API MALWARE ATTACKS

API malware attacks are a type of cyberattack that uses APIs to inject and execute malicious code on a targeted system. Malware is often hidden in API calls, which can then be used

to exploit vulnerabilities in the targeted system and gain unauthorized access or control.

API malware attacks can take many forms, including:[22][10][12]

- Remote code execution: Malware is injected through an API call, allowing attackers to remotely execute code on the targeted system.
- Credential theft: Malware is used to steal user credentials, such as usernames and passwords, through API calls.
- Data exfiltration: Malware is used to extract sensitive data from the targeted system through API calls.
- DDoS attacks: Through the use of APIs, malware is used to perform Distributed Denial of Service (DDoS) attacks, which bombard the targeted system with traffic and interfere with regular operations.

To prevent API malware attacks, it is important for developers and security professionals to implement proper security measures and conduct regular API security testing.

1.3 IMPORTANCE OF API MALWARE ANALYSIS AND FORENSICS

API malware analysis and forensics are essential for detecting analyzing, and mitigating API malware attacks. Malware attacks can have serious consequences for organizations, including data theft, system downtime, financial losses, and reputational damage [10] [22].

Effective API malware analysis and forensics can help identify the source and extent of the attack, recover lost or stolen data, and prevent future attacks.

API malware analysis involves examining APIs and their associated code for signs of malware, such as unusual API calls or unexpected system behaviour. This can be a challenging task, as malware can be disguised or obfuscated to avoid detection.

API malware forensics involves conducting a detailed investigation of the attack to identify the root cause and prevent similar attacks in the future. This may involve analyzing system logs, network traffic, and other digital evidence to reconstruct the attack and determine the extent of the damage [22].

The importance of API malware analysis and forensics has only increased with the rise of APIs in software development. As more organizations rely on APIs to connect their systems and services, the potential attack surface for API malware attacks has also increased [22] [10].

In conclusion, API malware analysis and forensics are essential for protecting organizations from the damaging effects of API malware attacks. By investing in these practices, organizations can ensure the integrity and security of their APIs and prevent future attacks [22].

2. TYPES OF API MALWARE ATTACKS

There are several common types of API malware attacks that organizations need to be aware of:[8][9][12]

1. API Spoofing: API spoofing is a type of attack where attackers create a fake API that mimics a legitimate one. When users try to connect to the fake API, the attackers can steal user credentials or inject malware into the user's system.
2. API Injection: In order to run malicious code on the targeted system, API injection requires inserting erroneous code into valid API calls. This can be accomplished in a number of ways, such as by taking advantage of flaws in API inputs or by intercepting and changing API calls using man-in-the-middle attacks.
3. API Parameter Tampering: API parameter tampering is a type of attack where attackers modify parameters in API calls to gain unauthorized access or manipulate data. This can be done by intercepting and modifying API calls or by using automated tools to manipulate API inputs.
4. API Denial-of-Service (DoS) Attacks: API DoS attacks involve overwhelming an API with requests, causing it to crash or become unresponsive. This can be done through a variety of means, such as using automated tools to flood the API with requests or exploiting vulnerabilities in the API's design or implementation.
5. API Phishing: API phishing involves tricking users into connecting to a fake API that appears to be legitimate. When users enter their credentials into the fake API, the attackers can steal them for later use.
6. API Remote Code Execution (RCE): Any arbitrary code can be run on the targeted machine by taking advantage of API RCE weaknesses. This can be done in a number of ways, such as by using a malicious payload in an API call or by taking advantage of vulnerabilities in the API's input validation or authentication mechanisms.

The number of Application Programming Interfaces (APIs) deployed within organizations is multiplying.

According to this survey, 26% of businesses use at least twice as many APIs as they did a year ago, increasing attacks on APIs. APIs are integral to any application, making them a prime target for attacks [9].

2.1 EXPLANATION OF HOW API MALWARE CAN BE USED TO EXECUTE MALICIOUS CODE

API malware can be used to execute malicious code by exploiting vulnerabilities in software components that use APIs. Malware can be hidden in API calls, allowing attackers to inject and execute malicious code on a targeted system [13] [14].

One common method for executing malicious code through API malware is remote code execution (RCE). In an RCE attack, attackers use an API to send a payload that contains malicious code to the targeted system. The payload is then executed on the system, giving the attacker remote access and control over the system [13] [14].

API injection is another technique for using API malware to run malicious code. Attackers can execute the malicious code on the targeted system by injecting it into legal API calls in an API injection attack. This can be accomplished in a number of ways, such as by taking advantage of flaws in the API inputs or by intercepting and changing API calls using man-in-the-middle attacks [22].

In addition to these methods, API malware can also be used to execute malicious code through credential theft, data exfiltration, and DDoS attacks. For example, attackers may use API malware to steal user credentials through API calls and then use those credentials to execute malicious code on the targeted system [15] [10].

To protect against API malware attacks that execute malicious code, organizations should implement proper security measures, such as secure API design, authentication and authorization mechanisms, and monitoring API activity for suspicious behaviour. Regular API security testing and analysis can also help detect and prevent API malware attacks.[10][22]

2.2 EXAMPLES OF REAL-WORLD API MALWARE ATTACKS

There have been several real-world API malware attacks that have impacted organizations and users:

1. Facebook API Malware Attack: In 2018, a malware attack on Facebook's APIs allowed attackers to steal access tokens and gain access to over 30 million user accounts. The attackers used an API vulnerability to exploit the "View As" feature, which allowed them to steal access tokens and take over user accounts [4].
2. Twitter API Malware Attack: In 2013, a malware attack on Twitter's APIs allowed attackers to steal user data, including passwords and email addresses. The attackers used an API vulnerability to exploit a cross-site scripting (XSS) flaw in Twitter's mobile app [5].
3. Uber API Malware Attack: In 2016, a malware attack on Uber's APIs allowed attackers to steal the personal data of over 57 million users and drivers. The attackers used an API vulnerability to access a database containing user data, which they then downloaded and encrypted [6].
4. Salesforce API Malware Attack: In 2018, a malware attack on Salesforce's APIs allowed attackers to steal customer data from a number of Salesforce's customers. The attackers used an API vulnerability to access customer data, which they then used to conduct phishing attacks and other forms of fraud.
5. Equifax API Malware Attack: In 2017, a malware attack on Equifax's APIs allowed attackers to steal the personal data of over 143 million customers. The attackers used an API vulnerability to access customer data, which they then downloaded and exfiltrated [7].

3. TECHNIQUES FOR API MALWARE DETECTION.

API malware can be found using a variety of methods, such as anomaly detection based on machine learning, behaviour-based detection, and signature-based detection [2] [1].

1. Signature-based detection: This method looks for patterns or signatures of known malware within the API requests. Signatures, which are patterns taken from well-known malware, are used to spot related malware in API calls. While successful against known malware, signature-based detection may miss the presence of fresh or undiscovered threats.
2. Behaviour-based detection: This approach focuses on analyzing the behaviour of the API calls to identify potential malware. Behavior-based detection involves profiling normal API call behavior, creating a baseline, and then detecting any deviations from the baseline. Behaviour-based detection can identify new and unknown malware but may produce false positives.
3. Machine learning-Based detection: In this method, aberrant patterns in the API calls are identified using machine learning methods. To detect departures from the norm, machine learning algorithms can be trained on typical API call behaviour. This method can identify brand-new and

unidentified malware, but it could also result in false positives and false negatives.

Advantages and Disadvantages of signature-based, Machine Learning-based, and behaviour-based techniques are presented in below Table 1.

Table 1: Advantages and disadvantages of signature-based, Machine Learning-based and behaviour-based techniques.

	Advantage	Disadvantage
Signature-based	Its reduced runtime results from the fact that it only depends on the code's signature [2].	Unknown Malware can not be detected because this technique only looks for patterns from known malware signatures [2].
Behaviour-based	It is more effective in finding polymorphic malware than signature-based methods [1].	Many polymorphic viruses, often known as packers, are undetectable by it [2].
Machine Learning-based	This can analyze large volumes of API data to learn patterns and behaviours associated with malware.	It requires a significant amount of labelled training data to build accurate models.

3.1 TECHNIQUES OF API MALWARE ANALYSIS.

Analyzing API calls is a critical component of identifying and detecting malicious code. There are several techniques for analyzing API calls and identifying malicious code, including code analysis, dynamic analysis, and sandboxing.

1. Static Analysis: This technique involves examining the code within API calls to identify known malicious code patterns and behaviours. Code analysis can be performed manually or through, the use of automated tools, such as static analysis tools. Static analysis tools scan the API calls to identify any suspicious code, including known malware patterns and code obfuscation techniques.

   Several techniques can be used for static malware analysis [2].

   - File fingerprinting: This method operates at the file level, for example, by computing the file's cryptographic hash. By doing so, you can tell it apart from other files and confirm that the file hasn't been altered. This method doesn't check the file's external characteristics.
   - Extraction of hard- coded strings: This method extracts strings that are hard-coded into the binary. This is possible because programme code compiles error and status notifications into readable text. Therefore, drawing conclusions about the examined binary may be possible by looking at hard- coded error and status messages.
   - File format: Any file format's metadata could be used to gather valuable data. A magic number can be used on UNIX systems to identify the file type. Some information, including imported and exported functions, strings, icons, and menus, can be acquired for a Windows binary, which is typically in a portable executable format.
   - AV scanning: Malware binaries are examined using this method. Anti-virus (AV) scanners are likely to identify a binary as malware if it is well-known.
   - Packer detection: By utilizing packers, malware is produced in an obfuscated format, such as an encrypted or compressed file. The malware has a new appearance after packing, making it difficult to find. Although there are numerous unpackers, including PEiD2, there is no universal unpacker for all the malware that has been packaged. The main drawback of static malware analysis is this limitation.
   - Disassembly: The main function of static analysis is the disassembly of a malware binary. The machine code is converted to disassembly language using disassembly tools like IDA Pro. The programme logical functions can then be examined, and its objectives can be ascertained, using the disassembled code.

2. Dynamic Analysis: This technique involves analyzing the behaviour of API calls in real-time to identify any malicious activity. Dynamic analysis can involve monitoring network traffic, analyzing system logs, and using tools such as intrusion

detection systems (IDS). By observing the behaviour of API calls in real-time, it is possible to detect anomalies and other malicious activity.

There are two dynamic analysis approaches [2]:

- Analyzing the difference between defined points: In this method, a sample of input malware is run for a while, and if any changes are made to the system, they are examined by comparing them to the system's initial state. The course of the infection will be shown in the report created by comparing the two states. watching the runtime.
- Behaviour
- : Monitoring tools like API Monitor are used to keep an eye on malware activity while it is running.

3. Sandboxing: Sandboxing is the process of executing API calls in a regulated setting in order to observe their behaviour and spot any harmful activities. API calls can be isolated with the use of sandboxing, which keeps the overall system from being impacted. This method can be helpful for analyzing malware behaviour and spotting any odd or suspicious activity.

4. Memory Forensics: Memory forensics examines a system's memory to find any harmful behaviour or code. This method can be used to locate malware that doesn't exist in the file system but only exists in memory. Using specialized tools and methods, such as memory dump analysis and debugging tools, memory forensics can be carried out.

5. API Fuzzing: API fuzzing involves sending malformed or unexpected data to API calls to identify any vulnerabilities or weaknesses in the API. This technique can help identify potential security issues and is often used during the development and testing phases of software.

Research on some of the techniques described above is presented in Table 2.

Table 2: Research on API Malware Analysis.

Paper	Method	Accuracy
[2]	N/A	0.89
[2]	Static	0.985
[1]	Dynamic	0.998

4.API FORENSICS

Investigating and examining APIs to see if they have been exploited, misused, or compromised is known as API forensics. In order to uncover security flaws and attacks, collect evidence for use in court cases, and collect data from APIs and the systems that are connected to them, it is necessary to apply forensic techniques and technologies. API forensics is essential in the linked world of today, when systems and platforms are increasingly integrated through the use of APIs [16] [20].

4.1 API FORENSICS PROCESS

The API forensics process involves several steps, including:[21][19][16]

1. Identification: The first step in the process is identifying the APIs that are being used by the organization. This includes identifying the types of APIs, their functionality, and their associated systems and platforms.

2. Collection: Once the APIs have been identified, the next step is to collect data from them. This includes collecting logs, traffic data, and other relevant information that can help identify security breaches and attacks.

3. Analysis: The collected data is then analyzed using forensic techniques and tools. This includes analyzing traffic patterns, identifying abnormal behaviour, and determining the root cause of any security breaches or attacks.

4. Evidence Gathering: The final step in the process is to gather evidence that can be used in legal proceedings. This includes preserving the collected data in a secure manner, creating an audit trail of the investigation, and documenting the findings and conclusions.

4. 2 CHALLENGES OF API FORENSICS

API forensics has its share of difficulties. The complexity of contemporary API architectures, which might include several levels and components, is one of the main obstacles. This makes it challenging to spot and isolate security flaws, attacks. The intricacy of the forensic investigation process is further increased by the fact that many third-party services serve as the foundation for APIs [19] [21].

The absence of standardised forensic methods and tools for API forensics is another difficulty. While certain tools are accessible, they are frequently proprietary and rarely used. This makes it challenging for researchers to successfully share data and work together [19] [21].

4.3 TECHNIQUES FOR API FORENSICS

API forensics involves the collection, preservation, and analysis of evidence related to API-based attacks. The

following are some common techniques for API forensics: [16][18][19][20][21]

1. Memory Dump Analysis: To find any malicious activity, memory dump analysis includes retrieving the memory from a machine. This method can be used to locate malware that doesn't exist in the file system but only exists in memory. It is possible to analyze memory dumps using specialized tools and methods, such as the Volatility Framework.

2. Log AnalysisLog analysis involves looking through system logs for any odd activity that might be connected to API-based attacks. System logs may contain important details about the behaviour of API calls, such as the date and time of execution, the source and destination addresses, and the nature of the operation. Both automatic tools, like Log Parser, and human analytic methods can be used to perform log analysis.

3. Network Traffic Analysis: Network traffic analysis involves monitoring the traffic between systems to identify any suspicious activity related to API-Based attacks. This technique can be used to identify the source and destination of API calls, as well as the type of data being transferred. Network traffic analysis can be performed using specialized tools, such as Wireshark.

4. System Profiling: System profiling involves collecting information about the configuration of the system to identify any potential vulnerabilities or weaknesses. This technique can be used to identify any software or hardware configurations that may be susceptible to API-based attacks. System profiling can be performed using automated tools, such as OSSEC, or manual analysis techniques.

5. Evidence Collection: Evidence collection involves the proper collection and preservation of evidence related to API-based attacks. This includes the collection of system logs, memory dumps, and network traffic data. Proper evidence collection is critical to ensuring that the evidence collected is admissible in court and can be used to support legal action, if necessary.

4.4 API FORENSICS TOOLS

There are several tools available for performing API forensics. The following are some commonly used tools: [16][17]

1. Volatility Framework: API forensics can be performed using the open-source memory forensics framework Volatility. It enables the detection of malicious behaviour connected to API-based attacks through the study of system memory dumps.

2. Wireshark: Wireshark is a network traffic analysis tool that can be used to monitor and analyze network traffic related to API-based attacks. It can capture and decode network traffic in real-time, allowing for the analysis of the data transferred during API calls.

3. Log Parser: Log Parser is a log analysis tool that can be used to parse and analyze system logs related to API-based attacks. It can extract relevant information from log files and provide insight into the behaviour of API calls.

4. OSSEC: OSSEC is a host-based intrusion detection system that can be used for system profiling and to detect and prevent API-based attacks. It can monitor system logs, file changes, and network traffic and alert administrators when suspicious activity is detected.

5. Fiddler: Fiddler is a web debugging proxy tool that can be used to analyze and debug HTTP traffic related to API-based attacks. It can capture and analyze HTTP traffic between clients and servers, allowing for the identification of any malicious activity.

5.FUTURE STUDY:

In terms of potential future research directions in API malware analysis and forensics, machine learning-based techniques could be developed to detect and classify API-based attacks with high accuracy. There is also a need for automated systems that can detect API-based attacks in real-time and respond appropriately. Additionally, techniques for analyzing encrypted traffic could be developed to identify any suspicious activity related to API-based attacks.

Emerging technologies such as cloud computing and the Internet of Things (IoT) can have an impact on API security. As more applications are developed for these technologies, it becomes increasingly important to ensure that their APIs are secure. Future research could focus on developing security solutions for these emerging technologies.

In terms of implications for software development and cybersecurity best practices, API security should be a top priority. Developers should be trained in secure coding practices and follow established security guidelines. Regular security assessments and testing should be performed to identify any vulnerabilities in API-based applications.

6.CONCLUSION:

The paper explored the topic of API malware analysis and forensics, providing an overview of APIs and their role in software development, the threat of API malware attacks, and the importance of API malware analysis and forensics. It also discussed common types of API malware attacks, techniques for API malware detection and analysis, techniques for analyzing API calls and identifying malicious code, techniques for API forensics, and tools used for API forensics.

The paper highlights the importance of API security and the need for regular security assessments and testing to identify vulnerabilities in API-based applications. Developers should follow established security guidelines and be trained in secure coding practices. Incident response plans should also be in place to quickly respond to any API-based attacks. Researchers should focus on developing automated systems that can detect API-based attacks in real-time, techniques for analyzing encrypted traffic, and security solutions for emerging technologies such as cloud computing and IoT.

Future research should focus on developing machine learning-based techniques for API malware analysis and forensics, automated systems that can detect API-based attacks in real-time, techniques for analyzing encrypted traffic, and security solutions for emerging technologies such as cloud computing and IoT. Additionally, more research is needed on the effectiveness of different API malware detection and analysis techniques, and on the development of best practices for API security.

7. REFERENCE:

[1] Youngjoon Ki,1 Eunjin Kim,2 and Huy Kang Kim1, A Novel Approach to Detect Malware Based on API Call Sequence Analysis, 1 Korea University, 145 Anam-ro, Seongbuk-gu, Seoul 137-713, Republic of Korea .

[2] Fahad Mira, A Review Paper of Malware Detection Using API Call Sequences, School of Computer Science and Technology University of Bedfordshire, Luton, UK.

[3] https://brightsec.com/blog/api-security/

[4] https://techcrunch.com/2018/09/28/everything-you-need-to-know-about-facebooks-data-breach-affecting-50m-users/

[5] https://venturebeat.com/security/twitter-breach-api-attack/

[6] https://techcrunch.com/2017/11/21/uber-data-breach-from-2016-affected-57-million-riders-and-drivers/

[7]https://en.wikipedia.org/wiki/2017_Equifax_data_breach

[8] https://www.radware.com/cyberpedia/application-security/api-attack/

[9] https://www.dpsolutions.com/blog/common-attack-types-on-apis

[10]https://www.techtarget.com/searchapparchitecture/definition/application-program-interface-API

[11] Qiao, Y., Yang, Y., He, J., Tang, C. and Liu, Z. (2014), CbmCBM: free, automatic malware analysis framework using api API call sequences

, in ‚Knowledge Engineering and Management".

[12] Goertzel, K.M., 2009. Tools on Anti Malware. Technical Information Center.

[13] Acosta, J. C., Mendoza, H. and Medina, B. G. (2012), An efficient common substrings algorithm for on-the-fly behavior-based malware detection and analysis, in „Proceedings -– IEEE Military Communications Conference MILCOM"

[14] Ki, Y., Kim, E. and Kim, H. K. (2015), A Novel Approach to Detect Mal-ware Based on API Call Sequence Analysis", International Journal of Dis-tributed Sensor Networks.

[15] Sundarkumar, G.G. and Ravi, V., 2013, December. Malware detection by text and data mining. In 2013 IEEE International Conference on Computational Intelligence and Computing Research.

[16] Vassil Roussev, Andres Barreto and Irfan Ahmed, API-BASED FORENSIC ACQUISITION OF CLOUD DRIVES.

[17] Cloud forensicseTool development studies & future outlook , Vassil Roussev* , Irfan Ahmed, Andres Barreto, Shane McCulley, Vivek Shanmughan

[18] API Tracing Tool for Android-Based Mobile Devices ,Seonho Choi, Michael Bijou, Kun Sun, and Edward Jung.

[19] Live Forensics on RouterOS using API Services to Investigate Network Attacks, Muhammad Itqan Mazdadi ,Imam Riadi Ahmad Luthfi.

[20] Mike Bond and Ross Anderson, API-Level Attacks on Embedded Systems.

[21] https://indiaforensic.com/api-forensics-security/

[22] https://en.wikipedia.org/wiki/API