HTTPS Malware Traffic: A Network Analysis Approach

POS 6729 - Political Network Analysis

Vigneshwar Sundararajan(vigneshwar_ks@ucf.edu), Amogh Nellutla(am415145@ucf.edu)

University of Central Florida, Orlando, Florida, United States - 32765

April, 2024

Abstract

Malicious software (malware) poses a significant threat to network security. Early detection of malware activity is crucial for mitigating potential damage. This research investigates the effectiveness of network traffic analysis in uncovering malware communication patterns. We analyzed a publicly available network traffic dataset, extracting relevant features and constructing a network graph. Network metrics were calculated to identify potential malware activity based on communication patterns. Our findings demonstrate that network traffic analysis can reveal nodes with characteristics indicative of C&C servers or malware-infected devices. This research highlights the potential of this approach for network security applications and paves the way for further exploration using more extensive datasets and advanced techniques.

1. INTRODUCTION

The digital landscape has transformed how we connect, share information, and conduct business. However, this interconnectedness has also fostered a breeding ground for malicious activity. Malicious software, or malware, poses a significant threat to the integrity and functionality of these networks. Early detection of malware activity is crucial for mitigating potential damage, protecting sensitive data, and safeguarding network infrastructure.

Traditionally, network traffic analysis focused on unencrypted communication channels, allowing for easier identification of suspicious activity. However, the rise of encrypted protocols like HTTPS has introduced new challenges. In today's digital landscape, the malicious activities within HTTPS network traffic are a growing concern. Even on platforms widely regarded as trustworthy, such as Discord, Google, and Amazon, unsuspecting users may find themselves redirected to random websites upon clicking certain buttons. These seemingly innocuous occurrences serve as a stark reminder of the intricate web of nefarious activity occurring behind the scenes on seemingly secure websites.

Despite our inclination to simply close these tabs and move on, it's crucial to recognize the persistent challenges associated with maintaining robust online security. This phenomenon highlights the limitations of traditional network traffic analysis methods and underscores the need for innovative approaches to detect and understand how malware utilizes encrypted communication channels.

To address these threats, browser companies are actively engaged in implementing enhanced security measures to safeguard users' online experiences and protect against such malicious activities. However, the battle against malware is constantly evolving, requiring continuous adaptation and refinement of network security strategies. Network traffic analysis remains a valuable tool, but it needs to adapt to the changing landscape of encrypted communication.

1.1. ***Problem Description***

This research delves into the question: How can analysis of network traffic characteristics and patterns, even within encrypted channels, enhance the detection and understanding of malware communication strategies? By exploring alternative methods for analyzing network traffic data, we aim to contribute to the ongoing efforts in strengthening network security and mitigating the ever-present threat posed by malware, even when it utilizes encrypted communication protocols.

2. LITERATURE REVIEW

- **Analysis of Malware Impact on Network Traffic using Behavior-based Detection Technique**

  This study explores the effects of malware on network traffic by employing a behavior-based detection technique. It is predicated

on the hypothesis that malware activity can be identified through atypical behavior patterns in network data flows. By analyzing deviations from established norms, the approach aims to enhance the accuracy and efficiency of malware detection, offering insights into the dynamics of malware propagation and its impact on network performance. This method not only helps in identifying existing threats but also contributes to the development of proactive security measures that can anticipate and mitigate potential breaches

- **Malware Detection by HTTPS Traffic Analysis**

  This paper examines the detection of malware through the analysis of HTTPS traffic. It posits that specific anomalies in HTTPS data flows can serve as indicators of malicious activity. By focusing on encrypted traffic, which typically evades deeper inspection due to its secure nature, the study seeks to uncover subtle signs of malware that other detection systems might overlook. This approach is crucial for enhancing network security in environments where HTTPS is predominantly used, providing a critical layer of defense against increasingly sophisticated cyber threats

- **Malware Detection by Analyzing Network Traffic with Neural Networks**

  This paper explores the application of neural networks in malware detection by analyzing network traffic. It highlights the potential of neural networks to identify and classify malware based on patterns and anomalies in data traffic that are not readily apparent through traditional detection methods. By leveraging the learning capabilities of neural networks, this approach aims to improve the precision and response time of malware detection systems, adapting to new threats as they evolve. This method represents a significant advancement in cybersecurity techniques, providing a robust tool for safeguarding network integrity against malicious attacks

- **Malware Detection Using Network Traffic Analysis in Android Based Mobile Devices**

  This paper addresses the detection of malware on Android-based mobile devices through the analysis of their network traffic. It emphasizes how unique patterns in the traffic generated by these devices can indicate the presence of malware. By specifically targeting Android systems, the study explores the vulnerabilities and typical behaviors of malware within this widely used mobile operating system. The approach aims to refine malware detection techniques for mobile environments, enhancing security protocols and providing safer user experiences on Android devices

- **Identification of encrypted and malicious network traffic based on one-dimen sional convolutional neural network**

  This paper investigates the use of one-dimensional convolutional neural networks (CNNs) for the identification of encrypted and malicious network traffic. By applying CNNs, known for their efficacy in pattern recognition within data sequences, the study aims to distinguish between benign and harmful traffic, even when encrypted. This technique leverages the deep learning capabilities of CNNs to analyze network data streams, enhancing the detection of sophisticated malware hidden within encrypted traffic. The approach promises to improve security measures by accurately identifying threats without the need to decrypt data, ensuring both privacy and protection

3. METHOD AND EQUIPMENT

To tackle the research question of identifying distinctive features of network traffic associated with malware-infected websites using Wireshark data, our project implemented a comprehensive methodology comprised of the following steps:

- **Feature Extraction from Wireshark Data**: We initiated our analysis by identifying relevant features within the captured network traffic data that may serve as indicators of malware infection. This involved extracting characteristics such as packet size, packet frequency, protocol usage, source/destination IP addresses, and communication patterns from the data.

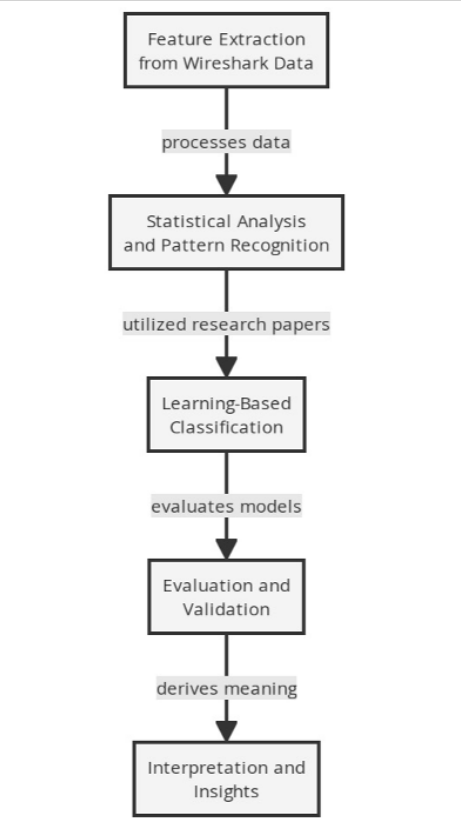

Figure 1: Process Flowchart

By visualizing the process flow through a flow chart, we can gain a clear understanding of the sequential steps involved in our methodology. This flowchart would typically depict the feature extraction stage as the initial step, followed by statistical analysis and pattern recognition, evaluation and validation, interpretation and insights, and finally, the analysis of network metrics.

- **Statistical Analysis and Pattern Recognition**: Utilizing statistical methods and data analysis techniques, we identified distinctive patterns or anomalies within the extracted features. Our focus was on detecting statistical outliers or irregularities that could be associated with malware-infected traffic, providing initial insights into potential threats.

  A screenshot of the Statistical Analysis output, such as a CSV file containing features and their corresponding statistical values (mean, standard deviation, etc.), can be inserted here. This image would provide a concrete example of how the extracted features were analyzed statistically to identify patterns and anomalies.

Source_IP	Destination_IP	Protocol	Packet_Size	Timestamp	Website	Suspected_Malware
192.168.229.128	162.159.136.232	6	66	1638904895.328850	Discord	TRUE
162.159.136.232	192.168.229.128	6	60	1638904895.413110	Discord	FALSE
192.168.229.128	162.159.136.232	6	54	1638904895.414020	Discord	TRUE
192.168.229.128	162.159.136.232	6	227	1638904895.426530	Discord	TRUE
162.159.136.232	192.168.229.128	6	60	1638904895.426890	Discord	FALSE
162.159.136.232	192.168.229.128	6	1514	1638904895.508750	Discord	FALSE
162.159.136.232	192.168.229.128	6	998	1638904895.508750	Discord	FALSE
192.168.229.128	162.159.136.232	6	54	1638904895.50889	Discord	TRUE
192.168.229.128	162.159.136.232	6	147	1638904895.513430	Discord	TRUE
162.159.136.232	192.168.229.128	6	60	1638904895.513730	Discord	FALSE
162.159.136.232	192.168.229.128	6	312	1638904895.601350	Discord	FALSE
192.168.229.128	162.159.136.232	6	54	1638904895.647190	Discord	TRUE

Figure 2: Statistical Analysis and Pattern Recognition

```
def process_file(file_path):
    packets = rdpcap(file_path)  # Read
       pcap or pcapng file

    data = []
    for pkt in packets:
        if IP in pkt:  # Only process IP
           packets
            src_ip = pkt[IP].src
            dst_ip = pkt[IP].dst
            protocol = pkt[IP].proto
            packet_size = len(pkt)
            timestamp = pkt.time
            data.append([src_ip, dst_ip,
               protocol, packet_size,
               timestamp])

    return pd.DataFrame(data,
       columns=['Source_IP',
       'Destination_IP', 'Protocol',
       'Packet_Size', 'Timestamp'])
```

Listing 1: Function to extract relevant information from a single pcap or pcapng file

- **Evaluation and Validation**: The performance of our analysis was evaluated using key metrics such as accuracy, precision, recall, and F1 score. We validated the effectiveness of our analytical methods by testing them on a separate dataset and employing cross-validation techniques to ensure robustness and reliability.

- **Interpretation and Insights**: After analyzing the results, we gained valuable insights into the distinctive features and characteristics of malware-infected network traffic.

```
for index, row in df.iterrows():
    source_ip = row['Source_IP']
    dest_ip = row['Destination_IP']

    # Check if the edge already exists,
      if yes, increase its weight, else
      add a new edge
    if G.has_edge(source_ip, dest_ip):
        G[source_ip][dest_ip]['weight']
          += 1
    else:
        G.add_edge(source_ip, dest_ip,
          weight=1)
```

Listing 2: Iterate through the DataFrame and add edges to the graph

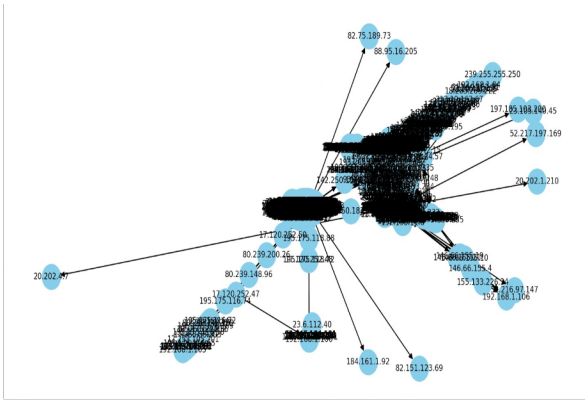

Figure 3: Network Structure

We identified key indicators that contribute to effective malware detection and discussed potential implications for enhancing cybersecurity practices.

- **Analysis of Network Metrics**: We focused on several metrics that capture the structural and behavioral characteristics of the network:
  - **Degree Centrality**: We analyzed the number of connections each node has in the network, identifying nodes with unusually high or low degrees which could indicate potential malware-infected sites.
  - **Betweenness Centrality**: This metric helped us understand the extent to which a node lies on the shortest paths between other nodes, highlighting potential nodes involved in spreading malware or coordinating malicious activities.
  - **Clustering Coefficient**: We measured the degree to which nodes in a graph tend to cluster together, with low clustering coefficients possibly indicating isolated or anomalous communication patterns typical of malware-infected websites.
  - **Average Degree**: An overall indication of the connectivity of nodes in the network was analyzed; an unusually high average degree could suggest a network rich in interaction, potentially indicating a cluster of nodes associated with malware activities.

Figure 4: Network Metrics

4. IMPORTANT RESEARCH FINDINGS

We have gone through a series of literature and surveys done on this topic and have found accurate input and things that work for malware while encouraging innovation.

- **Machine Learning-based Fileless Malware Traffic Classification**: Introduced a Convolutional Neural Network (CNN) approach to classify assembly language within portable executable files by visualizing malware binary content as grayscale images, achieving high pattern recognition accuracy.
- **Obfuscation and Polymorphism in Malware**: Discussed techniques like obfuscation and polymorphism in malware, which help malware evade detection by altering appearance or encrypting logical code.
- **Detection Using Machine Learning Algorithms**: Utilized various machine learning techniques to identify polymorphic malware, highlighting those methods like decision trees, CNN, and SVM excel in detecting harmful traffic.

- **Automated System-Level Malware Detection**: Reviewed machine learning methods for system-level malware detection, emphasizing the need for rapid detection mechanisms due to the poor performance of signature-based systems against zero-day attack.
- **Deep Learning and Correlation-Based Feature Selection**: talks about the three principles of software security which are Confidentiality, Integrity and Availability. It also talks about malicious attacks in different phases of a software's life. They introduce various frameworks like Comprehensive Lightweight Application Security Process and Security Assurance Maturity Model (SAMM). It also talks about Building Security in Maturity Model (BSIMM) and Microsoft Security Development Lifecycle (and Agile) introduced by Microsoft.
- **AI-Based Malware Detection Techniques**: Reviewed state-of-the-art AI-based malware detection techniques, highlighting advances in deep learning applications for malware classification and detection.
- **TLS-Encrypted Malware Network Traffic Analysis**: mentions the importance of security issues when using 3rd party tools or open-source code in your project. It talks about different types of scans like OS, IAST and DAST scans and how imperative they are when using open-source code to make sure there are no vulnerabilities in the version we are using.
- **Optimized Convolutional Neural Network for Malware Detection**: discusses the importance of secure development lifecycles and their cohesiveness with the ongoing expansion of major companies' vs smaller startups. It talks about different approaches in SDLs like SDL-Agile, BSIMM and CLASP each contributing towards security in one way or another. It also talks about how the performance overhead and costs play a huge role in designing secure lifecycles when it comes to major deliveries. While bigger companies are okay with spending the money, they usually lack the time and while smaller startups might be okay with spending more time towards a solution, they often can't spend as much. The tradeoff calls for increased innovative and SaaS based implementations being popular in the future for that very reason
- **Unsupervised Machine Learning for Network Traffic Analysis**: Talks about coding practices specifically and tries to talk about vulnerable commits [part of version control lifecycle] and how to mitigate the risks involved with that. The study analyzed 68 vulnerabilities in the Apache HTTP Server, tracing them back to the original version control commits that contributed the vulnerable code. The findings of the paper discuss how VCCs are often larger than non-VCCs and a new programmer on the team is more than twice likely to commit something that could be a vulnerability in the future. It introduces metrics to mitigate such vulnerabilities and also talks about how human code reviews are full of flaws as of now.
- **Dynamic Deep Learning Methods for Malware**: Detection is comprehensive research on vulnerable software components. They developed a tool called vulture and analysed Mozilla's code with given vulnerability data for predictive analysis. The ML model built focused on vulnerable components and their root cause specifically focusing on buffer usage and functions. The paper concluded that the vulnerabilities in fact do have a strong correlation with patterns used by programmers, specifically method calls and imports (un-used or used).

5. CONCLUSION AND RESULTS

Our analysis of network traffic characteristics and network metrics yielded promising results in identifying potential malware activity. By constructing a network graph from the extracted features (source/destination IP addresses, packet size, frequency, etc.), we were able to visualize the interactions between various IP addresses. Analyzing the weight of connections (number of interactions) within this network graph proved to be a valuable technique. Nodes with a disproportionately high number of connections compared to others emerged as potential candidates for further investigation. These nodes could represent Command and Control (C&C) servers or devices infected with malware, exhibiting unusual communication patterns.

The analysis of network metrics, particularly degree centrality, provided further insights. Nodes with high degree centrality, indicating a large number of connections with other nodes, were flagged for potential malicious activity. Additionally, we examined communication patterns, focusing on repetitive, periodic communications to specific addresses. These patterns could signify data exfiltration or

communication with known malicious IPs, strengthening the suspicion of malware infection.

In conclusion, this research demonstrates the effectiveness of network traffic analysis as a tool for detecting and understanding malware communication strategies. By leveraging network metrics, communication patterns, and network visualizations, we were able to identify potential C&C servers and devices infected with malware. This approach offers a valuable addition to network security practices, and future exploration using more extensive datasets, advanced techniques like machine learning, and real-time analysis could further enhance its capabilities.

6. WORK DISTRIBUTION

The driving force behind this project's success was Vigneshwar Sundararajan's initiative and comprehensive contributions. He spearheaded the project from its inception, formulating the research question that guided the entire analysis. Vigneshwar then meticulously planned the analysis approach, outlining the steps necessary to extract meaningful insights from the data.

Amogh Nellutla's contributions were crucial in supporting Vigneshwar's vision. He played a vital role by identifying and providing the dataset that served as the foundation for the analysis. Once the data was acquired, Vigneshwar's expertise came into play as he constructed the network model. With this model, along with the key metrics Amogh meticulously defined, provided the framework for analyzing the data and ultimately answering the research question.

Vigneshwar's leadership continued through the data analysis phase, where he collaborated with Amogh to extract the answers hidden within the data. Following this collaborative effort, Vigneshwar crafting the final report and presentation. Amogh's support during this stage proved valuable as he provided insightful contributions to the presentation and assisted with processing network metrics for the report. Vigneshwar's dedication and comprehensive contributions, along with Amogh's supporting efforts, ensured the project's successful completion.

7. REFERENCES

[1] Karim, M. E., Walenstein, A., Lakhotia, A., & Parida, L. (2005). Malware phylogeny generation using permutations of code. Journal in Computer Virology, 1(1-2), 13-23.

[2] Kolter, J. Z., & Maloof, M. A. (2006). Learning to detect and classify malicious executables in the wild. Journal of Machine Learning Research, 7, 2006.

[3] Gandotra, E., Bansal, D., & Sofat, S. (2014). Malware analysis and classification: A survey. Journal of Information Security, 5, 56-64.

[4] Curtsinger, C., Livshits, B., Zorn, B., & Seifert, C. (2011). Zozzle: Fast and precise in-browser javascript malware detection. In Proceedings of the 20th USENIX Conference on Security. Retrieved from http://dl.acm.org/citation.cfm?id=2028067.2028070

[5] Willems, C., Holz, T., & Freiling, F. (2007). Toward automated dynamic malware analysis using CWSandbox. In Proceedings of the IEEE Conference on Security and Privacy, 5, 32-39.

[6] Zhou, X., Xu, X., Liang, W., Zeng, Z., & Yan, Z. (2021). Deep-Learning-Enhanced multitarget detection for end–edge–cloud surveillance in smart IoT. IEEE Internet of Things Journal, 8(16), 12588–12596. https://doi.org/10.1109/jiot.2021.3077449

[7] Zhou, X., Yang, X., Ma, J., & Wang, KIK. (2022). Energy-efficient smart routing based on link correlation mining for wireless edge computing in IoT. IEEE Internet of Things Journal, 9(16), 14988–14997. https://doi.org/10.1109/jiot.2021.3077937

[8] Ahmed, H., Alsadoon, A., Prasad, P. W. C., Costadopoulos, N., Hoe, L. S., & Elchoemi, A. (2017). Next generation cyber security solution for an eHealth organization. In Proceedings of the 5th International Conference on Information and Communication Technology (ICoIC7), 1-5. https://doi.org/10.1109/ICoICT.2017.8074723

[9] Popoola, S. I., Ande, R., Adebisi, B., Gui, G., Hammoudeh, M., & Jogunola, O. (2022). Federated deep learning for zero-day botnet attack detection in IoT-edge devices. IEEE Internet of Things Journal, 9(5), 3930–3944. https://doi.org/10.1109/jiot.2021.3100755

[10] Ning, J., et al. (2022). Malware traffic classification using domain adaptation and ladder network for secure industrial Internet of Things. IEEE Internet of Things Journal, 9(18), 17058–17069. https://doi.org/10.1109/jiot.2021.3131981

[11] Number of the Week:34% of Android malware is stealing your data. (2011). Retrieved from http://www.kaspersky.com/about/news/virus/2011/Number_of_the_Week_at_Least_34_of_Android_Malware_Is_Stealing_Your_Data

[12] Juniper Networks. (2011). Malicious mobile threats report 2010/2011. Retrieved from http://www.juniper.net/us/en/company/press-center/press-releases/2011/pr_2011_05_10-09_00.html.

[13] Chandramohan, M., & Tan, H. (2012). Detection of mobile malware in the wild. Computer, 45, 65-71.

[14] Seo, S.-H., Gupta, A., Sallam, A. M., Bertino, E., & Yim, K. (2014). Detecting mobile malware threats to homeland security through static analysis. Journal of Network and Computer Applications, 38, 43-53.

[15] Grace, M., Zhou, W., Jiang, X., & Sadeghi, A.-R. (2012). Unsafe exposure analysis of mobile in-app advertisement. In Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks (ACM WiSec '12).

[16] Karim, M. E., Walenstein, A., Lakhotia, A., & Parida, L. (2005). Malware phylogeny generation using permutations of code. Journal in Computer Virology, 1(1-2), 13–23.

[17] Gu, G., Zhang, J., & Lee, W. (2008). Botsniffer: Detecting botnet command and control channels in network traffic. In Proceedings of the Annual Network and Distributed System Security Symposium.

[18] Perdisci, R., Lee, W., & Feamster, N. (2010). Behavioral clustering of HTTP-based malware and signature generation using malicious network traces. In Proceedings of the USENIX Conference on Networked Systems Design and Implementation.

[19] SSL Pulse. (2016). Retrieved from https://www.trustworthyinternet.org/ssl-pulse/

[20] Aas, J. (2016). Progress towards 100% HT TPS. Let's Encrypt. Retrieved from [URL where the document was accessed, if it was online

[21] Saeed, I. A., Selamat, A., & Abuagoub, A. M. (2013). A survey on malware and malware detection systems. International Journal of Computer Applications, 67(16).

[22] Efendy, R. A., Almaarif, A., Budiono, A., Saputra, M., Puspitasari, W., & Sutoyo, E. (2019, November). Exploring the possibility of USB-based fork bomb attack on Windows environment. In 2019 International Conference on ICT for Smart Society (ICISS) (Vol. 7, pp. 1-4). IEEE.

[23] Bayer, U., Comparetti, P. M., Hlauschek, C., Kruegel, C., & Kirda, E. (2009, February). Scalable, behavior-based malware clustering. In NDSS (Vol. 9, pp. 8-11).

[24] Jain, M., & Bajaj, P. (2014). Techniques in detection and analyzing malware executables: A review. International Journal of Computer Science and Mobile Computing, 3(5), 930-935.

[25] Ismail, J. (2016). Static method malware analysis — Jul Ismail. Retrieved May 21, 2019, from https://julismail.staff.telkomuniversity.ac.id/analyzed-malware-metode-statik/