User-agent as a Cyber Intrusion Artifact: Detection of APT Activity using minimal Anomalies on the User-agent String Traffic

Badr-Eddine Bouhlal[1,*], Tim Sonnekalb[1], Bernd Gruner[1] and Clemens-Alexander Brust[1]

[1]*German Aerospace Center (DLR), Institute of Data Science Jena, Germany*

Abstract
The detection of attacks, especially persistent intrusions, relies on a combination of various artifacts. Despite being manipulable, the user-agent string, a component of HTTP headers, has proven to be a tool for triggering alerts, thereby enhancing detection capabilities. In this paper, we perform a review and analysis of existing malicious user agent strings. We gather relevant data from different sources of threat intelligence and present a dataset of user-agent strings associated with malicious activities gathered from real incident reports. We also propose a categorization of existing user-agent string anomalies with respect to their type (e.g., syntax) and their complexity degree.

Keywords
User-agent string (UAS), Advanced persistent threat (APT), Intrusion detection, Machine learning

1. Introduction

Intrusion detection is based on a multi-factor identification method that relies on multiple indicators. One of these indicators can be the HTTP User-Agent String (UAS) component. The initial function of the UAS is to allow the server to identify the request sender and disclose technical information such as the operating system and browser version. However, the UAS presents a considerable vulnerability in the web world because it is easily manipulated, making it a tool used to carry out code injection attacks. Nevertheless, in the context of a private and highly secure corporate network, the UAS can be used in addition to identifying the request sender as one of the intrusion detection factors.

The Cybersecurity and Infrastructure Agency (CISA) released a report in 2022 [1] containing recommendations that aim at helping organizations mitigating against advanced persistent threats (APT). The report recommend checking for anomalies that may be observed in UAS. These anomalies may directly affect the syntax of the UAS or may be related to other anomalies

*Corresponding author.
badr-eddine.bouhlal@dlr.de (B. Bouhlal); Tim.Sonnekalb@dlr.de (T. Sonnekalb); Bernd.Gruner@dlr.de (B. Gruner); clemens-alexander.brust@dlr.de (C. Brust)
0009-0002-2860-3068 (B. Bouhlal); 0000-0002-0067-1790 (T. Sonnekalb); 0000-0002-4177-2993 (B. Gruner); 0000-0001-5419-1998 (C. Brust)

that can be detected by using multiple identification factors, such as multiple authentication attempts with different UASs from the same IP addresses [1].

An anomaly may appear in several forms, and a malformed UAS can have multiple explanations: indicate the use of a vulnerability scanning tool (T1595 - Active Scanning) [2, 3], or be the sign of data exfiltration in the form of a legitimate string. It can also serve as a communication channel between malware and a command and control server (command and control attack). Communication via the application layer protocol is associated with several techniques used by APT groups attempting to exploit vulnerabilities in the HTTP/HTTPS protocol [4].

Challenges. The UAS does not follow a general format. While RFC 7231 [5] defines a general format for the presentation of the UAS many benign applications do not adhere to it [6]. This variability in UAS representation makes it difficult to conceive universal classification rules that remain effective over time. This situation presents significant challenges in differentiating legitimate UAS from malicious ones, as both representations initially consist of sequences of characters, numbers, and special characters. Another issue that arises when considering automation is determining the impact of irregular and sometimes anomalous user-agent that may not necessarily represent a threat which may increase the number of false alarms.

To the best of our knowledge, no dataset containing user-agent strings associated with malicious activity and presenting a detection artifact has been published. In addition, all existing studies concerning the detection of malicious traffic from user agents are mainly based on analysis of the network traffic, due to the absence of a malicious user-agent set. We summarize the key contributions of this paper as follows:(1) We introduce a dataset that consists of a collection of 1063 malicious UASs, which is publicly available under a CC-BY 4.0 License [7], (2) we perform a review and analysis of existing malicious UASs, and (3) we propose a categorization of the different UAS anomalies.

2. Related Work

In this section, we will present a series of studies that have focused on the user-agent string. Almost all of these studies aim to develop methods for distinguishing regular user-agent strings from malicious ones. Their approaches focus on developing parsing methods to make this distinction. However, we have noticed that they use data that cannot be directly linked to malicious traffic (a public database for this purpose does not exist, as far as we know). Furthermore, they focus mainly on syntax errors and do not consider cases of rarity. This rarity can hide minimal anomalies that are difficult to distinguish using a syntax parser, such as fake information, e.g., the pattern of the user-agent is correct, but the version of the browser used is fake. Zhang et al. [6] examined the UASs in malicious traffic, specifically malware. The authors found that one out of every eight instances of malware traffic contained suspicious user-agent in at least one of their HTTP requests. Currently, user-agent are still being analyzed manually. However, the analysis showed that there are multiple patterns that could be used to automatically classify user-agent anomalies. They also propose an automated technique for extracting user-agent anomalies and creating signatures for malware detection.

Kheir [8] applied a rule-based methodology using regular expressions for distinguishing abnormal and normal user-agent based on the fact that user-agent have a general and fixed structure that enables their validation using regular expressions. The data used during this research was collected over two months from real traffic and contained 150 billion user-agent. Zhang et al. [6] stated that the causes of anomalies could be due to two factors, namely a malfunction during the encoding decoding of the user-agent or a malicious activity.

Zhang et al. [9] used a method that combines several steps to classify user-agent: firstly, it uses a parser based on a context-free grammar to classify them based on their representation, a standard UAS, a non-standard representation for non-standard UAS, and finally, for unrecognized representations. Then, the authors propose using an anomaly detection algorithm to separate benign UAS from malicious ones. Their study compared the Context-Free Grammar (CFG) with the User Agent parser based on regular expressions. They showed that the CFG is better suited for analyzing user-agent traffic due to its ease of adaptation and simplicity of comprehension.

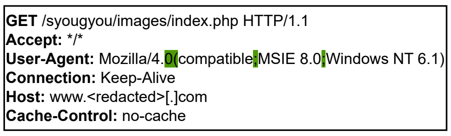
Nandakumar et al. [10] presented a novel method of parsing the user-agent strings based on Multi-Headed Attention mechanism using transformer, the method is divided into two-step, first parse the UAS to gather the information related to the device and software, then correlate the extracted information with known related Common Vulnerabilities and Exposures (CVE).

3. Illustrative Incidents: Malicious User-Agent Case Studies

Cyber-attacks, especially those carried out for espionage purposes, are designed to persist without being detected in the network. Performed by well-trained teams (APTs) using complex methods and sometimes over several well-planned stages, these attacks are not necessarily easy to analyze and sometimes difficult to determine their consequences at first glance. However, every potential indicator of malicious activity on the network must be carefully analysed and considered. These indicators, also known as network artifacts, can vary from an IP address, an URI pattern or an UAS that has not previously been observed in a defined network environment, or one that appears to be out of the ordinary [11]. In this section, we provide a concise overview of some incident reports from cyber-attack campaigns, specifically focusing on cases where the UAS field deviates from the norm. This divergence serves as a crucial factor in uncovering potential threats within the network. We perform this review of real incident to be able to analyse the type of anomalies that can be considered as artifacts of detection within the UAS (cf. Subsection 3.2), and also to perform a categorization of them (cf. Subsection 4.2).

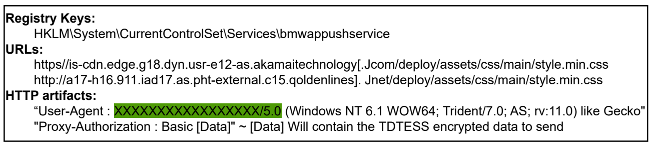
3.1. Real-life Incidents

Targeted Phishing Exploits Impacting Japanese and Taiwanese Organizations [12, 13]. APT groups have launched a mail fishing campaign with malicious word attachments targeting governmental organizations, finance, media, and high-tech sectors in Japan and Taiwan. The attack exploits a Microsoft Office EPS vulnerability CVE-2015-1701 [12], the exploit payload releases a binary, that includes an embedded sample of the IRONHALO malware. IRONHALO uses the HTTP protocol to fetch the payload from a command-and-control (C&C) server with hard-coded settings and a specific Uniform Resource Locator (URL) path [13]. This malware variant sends an HTTP request to a legitimate Japanese site with a malformed UAS with a syntax error: missing space between the different components of the UAS as shown in Figure 1a.



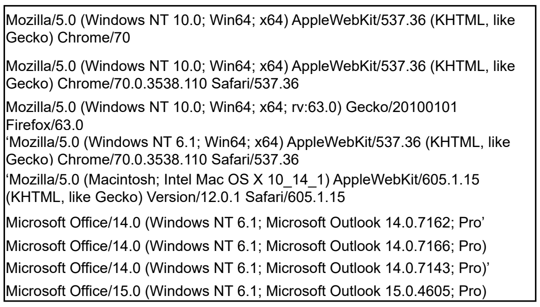
```
GET /syougyou/images/index.php HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0(compatible;MSIE 8.0;Windows NT 6.1)
Connection: Keep-Alive
Host: www.<redacted>[.]com
Cache-Control: no-cache
```

(a) IRONHALO HTTP GET request [13]



```
Registry Keys:
    HKLM\System\CurrentControlSet\Services\bmwappushservice
URLs:
    https//is-cdn.edge.g18.dyn.usr-e12-as.akamaitechnology[.Jcom/deploy/assets/css/main/style.min.css
    http://a17-h16.911.iad17.as.pht-external.c15.qoldenlines]. Jnet/deploy/assets/css/main/style.min.css
HTTP artifacts:
    "User-Agent : XXXXXXXXXXXXXXXXX/5.0 (Windows NT 6.1 WOW64; Trident/7.0; AS; rv:11.0) like Gecko"
    "Proxy-Authorization : Basic [Data]" ~ [Data] Will contain the TDTESS encrypted data to send
```

(b) CoppyKittens TDTESS indicators of compromise [14]



```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/70

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/70.0.3538.110 Safari/537.36

Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101
Firefox/63.0
'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/70.0.3538.110 Safari/537.36
'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_1) AppleWebKit/605.1.15
(KHTML, like Gecko) Version/12.0.1 Safari/605.1.15

Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7162; Pro'

Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7166; Pro)

Microsoft Office/14.0 (Windows NT 6.1; Microsoft Outlook 14.0.7143; Pro)'

Microsoft Office/15.0 (Windows NT 6.1; Microsoft Outlook 15.0.4605; Pro)
```

(c) Fancy Bear (APT28) user-agent strings [15]

Figure 1: Anomalies remarked on the UASs based on real incident related to APT Traffic

CopyKittens [14]. A cyber espionage group that mainly targets strategic organizations such as governmental organizations (defense companies, research institutions, Ministry of defense, and large IT companies), using self-developed tools that are not necessarily publicly reported. The methods of attack are complex and varied. The report [14] describes the intrusion methodology and also a set of used malware's, for example, the TDTESS which is a 64-bit . NET binary backdoor that communicates regularly with the command and control server, using basic authentication to receive new instructions. The incident analysis report presents various Indicators of compromise, among which is a malformed UAS [14], as shown in Figure 1b.

Russian GRU Conducting Global Brute Force Campaign to Compromise Enterprise and Cloud Environments [15]. Between mid-2019 and 2021, the Russian General Staff Main Intelligence Directorate (GRU) 85th Main Special Service Center (GTsSS), also known as Fancy Bear or APT28, used the Kubernetes cluster to launch large-scale anonymous brute force access attacks against government and private sector organizations, exploiting the CVE 2020-0688 and CVE 2020-17144 vulnerabilities in Microsoft Exchange, they used several protocols including HTTP. The campaign report publicly released by the NSA encompasses a detailed description of the techniques and tactics used as well as mitigation and detection methods including IP address lists and UASs, which "*are crafted to appear consistent with those sent by legitimate client software. Some of the UASs delivered in the authentication requests are incomplete or truncated versions of legitimate UASs, offering the following unique detection opportunities* [15]" (cf. Figure 1c) .

Bumblebee loader [16, 17]. A malware, employed in various campaigns by numerous threat actors, uses the Windows Management Instrumentation (WMI) framework to extract system details. Then, it establishes a connection with the C&C server at intervals of 25 seconds to receive commands to be execued. Logpoint [16] proposes two detection techniques within the proxy log files. Malware detection in the proxy log might be done either using the user agent and/or the URI. Hence, any existence of a user agent that matches the string *bumblebee* is a strong sign of the persistence of this malware. Other versions of the malware uses different evasion techniques and might change the UAS. An undefined UAS with the same number of digits should also be suspicious and might refer to the persistence of ***bumblebee*** [16, 17].

LYCEUM middle east campaign [18]. An APT group that targets organizations in strategic sectors, including oil and gas. Campaigns were reported in South Africa in 2018 and in the Middle East in 2019. The group primarily uses password-sparing or brute-force attacks to gain access to an organization to obtain credentials. Then, using compromised accounts, they send spearphishing emails containing malicious attached Excel files containing the Danbot malware. Danbot malware is a first-stage access trojan (RAT) that uses DNS and the HTTP protocol for communication. The Danbot HTTP request contains two anomalies: An ampersand (&) after operating system values in the UAS (**`Mozilla/5.0 (Windows NT 10.0; &) Gecko/20100101 Firefox/64.0`**), and a misspelling of 'Encoding' in the accept-encoding header [18].

Quasar: Open-Source Remote Administration Tool [19]. Is a RAT open source used for Windows Operating systems, hosted publicly on GitHub, specially dedicated for being used for legitimate purposes. In addition, various APT threat groups are using Quasar to conduct cyber espionage campaigns. Quasar enables remote control, keylogging, file transfer and enables the user to collect information about the host system. During the client connection's setup, the client tries to determine its geolocation, including its Wide Area Network (WAN) IP address. This is achieved by sending an HTTP GET request to the Uniform Resource Locator (URL) `ip-api[.]com/json/` with the following User-Agent string: **`Mozilla/5.0 (Windows NT 6.3; rv:48.0) Gecko/20100101 Firefox/48.0`**. "*This User-Agent string mimics a Mozilla Firefox 48 browser running on Windows 8.1. This User-Agent string would likely stand out as unique in a corporate network environment, and its presence could be a high-confidence indication of Quasar activity*" as stated by the Cybersecurity and Agency [19].

3.2. Analysis of the Incidents

By analyzing the various real incidents presented in the previous section, we note that the irregularities in the HTTP traffic (in our case, UASs), represents especially syntax errors or rarity of occurrence, represent a serious sign of a potential threat.

Indeed, UAS patterns vary significantly and do not necessarily follow a general structure that could be generalized to all device types (software, hardware). However, some syntax anomalies could represent a "red flag" and should be carefully examined. These syntax errors vary from grave errors, where the user-agent does not match any pattern of a correct UAS, to non-defined strings, as in the case of Bumblebee, or the TeamTNt group, where they use the UAS field to execute an injection code operation **`curl –referer $REFERER –user-agent TNTcurl $CURLPARA $GETFROM -o $PUTITTO`** [20]. Another example of using the UAS as a tool of code injection is the recent exploit of the Log4j vulnerability by the APT35 [21]. In addition we mention also syntax errors such as misspellings, absences, or even the existence of strings that should not exist in the correct UAS (as in the cases of CopyKittens and LYCEUM).

Another category of anomalies concerns not the syntax of the UAS, where it may be completely accurate, but the existence of a rare UAS, that can be suspicious especially in corporate networks. The detection of such anomalies depends on the analysis of the overall traffic, whereas a simple analysis of the syntax would not allow the detection, as in the case of Quasar malware and Fancy Bear (APT28). A rule-based detection method based on syntax would not be useful for the detection of fake UASs. The latter may take the form of a syntactically correct UAS. However, with a fake browser version, for example, the Metamorfo [22] malware uses a UAS with Mozilla/3. 0, the existence of such a version 3.0 must be suspicious.

4. Dataset Construction

In this section we present the construction process of our dataset of UASs associated with malicious traffic, focusing on attacks related to APTs. An essential part of building this dataset is categorizing anomalies in UASs.

4.1. Data Sources and Data Collection

For creating the dataset, we relied on different sources. Firstly, security reports on real incidents (cf. Subsection 3.1) which are collected by reputable organizations such as MITRE ATT@CK [23] and the Cybersecurity and Infrastructure Security Agency (CISA) [24]. These reports permit valuable insights into the latest cyber threats, offering detailed information on tactics, techniques, and procedures (TTPs) employed by malicious actors. The second source of our data consisted of contributions published by security professionals and experts in security blogs and community forums, for example, Microsoft [25] and Cisco Talos [26]. Subsequently, the UASs identified as artifacts of malicious activity were obtained from the Sigma open-source rules. These entries are regularly identified and logged as part of blacklists [1]. In addition, the dataset includes data gathered from the open-source project Apache Bad Bot Blocker [2].

4.2. User-Agent String Anomaly Categorization

Our categorization encompasses a wide range of anomalies observed in UASs, based on analyzing anomalies related to real-life incidents, we define the following two main categories.

4.2.1. The type of anomalies

The anomaly type, can be used as a detection hint and for this we define three different cases:

Syntax errors. These are the syntax anomalies that can be distinctive in a user-agent, for example, the missing space in the case of IRONHALO or the existence of an invalid string part, as is the case with CopyKittens [14] (**`XXXXXXXXXXXXXXX/5.0`** instead of **`Mozilla/5.0`**). In this case, the UAS has a correct pattern (format) but contains syntax errors.

Unknown string. In this category, we will classify all UASs that do not contain any pattern or part of a pattern of a correct UAS. These UASs are a collection of random strings (characters, digit or special characters) and may contain the name of the malware itself or any other string like the *bumblebee* for example.

[1]Sigma - Generic Signature Format for SIEM Systems:https://github.com/SigmaHQ/sigma#sigma---generic-signature-format-for-siem-systems

[2]Apache ultimate bad bot blocker https://github.com/mitchellkrogza/apache-ultimate-bad-bot-blocker/tree/master

Other. This category contains UASs without syntax anomaly, but are associated with a malware, APTs, or other malicious activities. Intrusion detection from these user agents may be due to their rarity in the traffic. They may stand out as anomalous in network traffic because of their rarity, or simply because their signatures should not exist in the specific network traffic.

4.2.2. The anomaly complexity degree

The second classification criterion distinguishes between high anomalies and low anomalies. This criterion is designed to evaluate the expected ability of a machine learning model to detect anomalous UASs at two levels of difficulty: the detection of a user-agent that differs totally syntactically from a normal pattern, and between the detection of a simple syntax anomaly. The two categories are defined as follows:

Low anomaly. Low anomaly formats represent user agent strings with minor or subtle deviations from normal (standard) formats. These may be a small syntax error, missing components, or the presence of unknown elements in an otherwise normal structure, as in the case of IRONHALO, where the anomaly is a missing space. In this category, we also include the UASs previously classified as *Other*.

High anomaly. Formats or patterns involving irregular, unusual, or entirely new structures in user agent strings that deviate significantly from regular formats. These anomalies can include random unknown strings, special characters, or unique identifiers with no correspondence with known, legitimate user agents, like the example of the **bumblebee**, where the string *bumblebee* do not represent a legitimate (or a part of) UAS. Another example is: **sample (unknown version) CFNetwork/596.5 Darwin/12.5.0 (x86_64) (iMac8%2C1)** [27], which includes many syntax errors and deviates systematically from a normal UAS. This UAS exhibits several anomalous characteristics, including elements like *sample (unknown version).* Additionally, the presence of the identifier *(iMac8%2C1)* does not correspond to known UAS or a part of a legitimate UAS and the percentage sign is not typically part of a standard UAS.

5. Dataset Description and Usage

Description. During data collection, we systematically inspect malicious UASs and collect additional information: The *type of anomaly*, the *degree of complexity*, the *APT associated with the malicious UAS*, if the information is available, the *sources* (e.g., citations of the resources from where the UAS was gathered), and the *string pattern* that distinguishes between two cases: if the abnormal UAS matches the entire string (match_string) or if it involves a regular expression (regx). The Regular expression describes a string part that might be included in a syntactically correct UAS but might be a sign of malicious activities. The data is provided in a CSV format, each row represents a UAS with corresponding information. Table 1 depicts an example of the dataset structure and Table 2 shows the distribution of the malicious UASs per category.

Usage. The dataset contains two types of UAS entries that must be treated differently: *match_string*, which represents malicious UASs that can be used directly, and *regx*, which presents only string parts which must be applied to correct UASs to generate malicious ones, these regular expressions define the potential location of these parts within a correct UAS. In addition to the set of malicious UASs (abnormal), we provide a set of 1000 of the most frequently

UAS	Anomaly Type	Anomaly Complexity	Related Apt	Ressources	String Pattern
UAS-1	syntax error	low	APT34	URL of document/online resources	match_string

Table 1
Description of the dataset structure

	Type of anomaly		
Complexity degree	Unknown string	Syntax errors	Other
High anomaly	738	77	-
Low anomaly	-	34	214

Table 2
Distribution of the collected malicious UASs per category

seen user agents during November 2023, parsed by the Whatismybrowser API parser [3]. These 1000 entries will be considered as a reference for normal user agents, containing no syntax anomalies. This allows using the dataset to train/test machine learning models on the detection of malicious UASs. We performed a similarity check between the two sets (normal and abnormal), and we noticed that there are eight common entries. An example entry is the malicious UAS representing the pattern used by the Google's bot crawler **`Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)`**, normally legitimately used for indexing purposes. The explanation for this finding is the fact that some malwares use complicated evasion techniques and they might use the most common and widely used UAS patterns. Moreover, some malwares mimics the UAS of the system on which it is installed to blend into the network traffic and avoid to be detected, as in the case of the *FatDuke malware* used by APT29, also known as *Cozy Bear* [28]. Such UASs fall under the category *Other*, where the UAS does not exhibit any syntax errors. However, the detection artifacts might be due to their rarity of existence within a traffic of legitimate UASs. Another example is the *Quasar malware* that uses a legitimate UAS that mimics a *Mozilla Firefox 48* browser running on *Windows 8.1*, which could be suspicious, especially in corporate networks [19].

6. Conclusion and Future Work

This study highlights the significant role of the UAS as an indicator of attack detection, and especially persistent intrusions. We present a dataset of 1063 UAS associated with malicious activities, the majority of which were collected manually from real incident reports. These UASs exhibit anomalies in syntax or other aspects, providing a basis for detecting persistent activities within network traffic. As future work, we plan to further extend the dataset and use it as training and test set to evaluate different machine learning models on their capabilities to automatically detect anomalies related to UASs.

[3]Whatismybrowser https://www.whatismybrowser.com/

Acknowledgments

We extend our sincere gratitude to Whatismybrowser, Raape, Ulrich and Möbius, Max for their invaluable assistance in collecting UASs for this study.

References

[1] Cybersecurity and Infrastructure Security Agency (CISA), Cybersecurity advisory: Impacket and exfiltration tool used to steal sensitive information from defense industrial base organization, 2022. URL: https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-277a, accessed: April 15, 2023.

[2] Center for Threat-Informed Defense, Microsoft azure security control mappings to mitre att&ck®, 2023. URL: https://center-for-threat-informed-defense.github.io/security-stack-mappings/Azure/README.html, accessed: April 20, 2023.

[3] M. Attck, Active scanning, 2022. URL: https://attack.mitre.org/techniques/T1595/, accessed: April 20, 2023.

[4] M. Attck, Application layer protocol: Web protocols, 2020. URL: https://attack.mitre.org/versions/v7/techniques/T1071/001/, accessed: April 21, 2023.

[5] R. T. Fielding, J. F. Reschke, Hypertext transfer protocol (http/1.1): Semantics and content, 2014. URL: https://www.rfc-editor.org/rfc/rfc7231, accessed: April 11, 2023.

[6] Y. Zhang, H. Mekky, Z.-L. Zhang, R. Torres, S. ju Lee, A. Tongaonkar, M. Mellia, Detecting malicious activities with user-agent-based profiles, International Journal of Network Management 25 (2015) 306 – 319. URL: https://api.semanticscholar.org/CorpusID:13959125.

[7] B.-E. Bouhlal, Dataset of malicious user-agent strings, 2024. URL: https://doi.org/10.5281/zenodo.10700806. doi:10.5281/zenodo.10700806.

[8] N. Kheir, Behavioral classification and detection of malware through http user agent anomalies, Journal of Information Security and Applications 18 (2013) 2–13. URL: https://www.sciencedirect.com/science/article/pii/S2214212613000331. doi:https://doi.org/10.1016/j.jisa.2013.07.006, sETOP'2012 and FPS'2012 Special Issue.

[9] Y. Zhang, H. Mekky, Z.-L. Zhang, R. Torres, S.-J. Lee, A. Tongaonkar, M. Mellia, Detecting malicious activities with user-agent-based profiles, International Journal of Network Management 25 (2015) 306–319. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.1900. doi:https://doi.org/10.1002/nem.1900.

[10] D. Nandakumar, S. Murli, A. Khosla, K. Choi, A. Rahman, D. Walsh, S. Riede, E. Dull, E. Bowen, A novel approach to user agent string parsing for vulnerability analysis using mutli-headed attention, 2023. arXiv:2306.03733.

[11] CSNP, Tryhackme - pyramid of pain room, Dec 5, 2022. URL: https://www.csnp.org/post/tryhackme-pyramid-of-pain-room, accessed: on: [02.01.2024].

[12] R. W. Genwei Jiang, Dan Caselden, The eps awakens, 2015. URL: https://web.archive.org/web/20170613151054/https://www.fireeye.com/blog/threat-research/2015/12/the_eps_awakens.html, accessed: [15.12.2023].

[13] F. T. I. Ryann Winters, The eps awakens - part 2, 2015. URL: https://web.archive.

org/web/20151226205946/https://www.fireeye.com/blog/threat-research/2015/12/the-eps-awakens-part-two.html, accessed: [15.12.2023].

[14] C. C. Security, Operation wilted tulip, 2017. URL: https://www.clearskysec.com/wp-content/uploads/2017/07/Operation_Wilted_Tulip.pdf, accessed: [15.12.2023].

[15] Cybersecurity, I. S. Agency, Russian gru conducting global brute force campaign compromise enterprise and cloud environments, 2021. URL: https://media.defense.gov/2021/Jul/01/2002753896/-1/-1/1/CSA_GRU_GLOBAL_BRUTE_FORCE_CAMPAIGN_UOO158036-21.PDF, accessed on [18.12.2023].

[16] Logpoint, Buzz of the bumblebee – a new malicious loader, https://www.logpoint.com/wp-content/uploads/2022/05/buzz-of-the-bumblebee-a-new-malicious-loader-threat-report-no-3.pdf, 2022. Accessed on [18.12.2023].

[17] K. Merriman, P. Trouerbach, This isn't optimus prime's bumblebee but it's still transforming, April 28, 2022. URL: https://www.proofpoint.com/us/blog/threat-insight/bumblebee-is-still-transforming, accessed on [18.12.2023].

[18] Secureworks, Lyceum takes center stage in middle east campaign, 2019. URL: https://www.secureworks.com/blog/lyceum-takes-center-stage-in-middle-east-campaign, accessed: on: [02.01.2024].

[19] Cybersecurity, I. S. Agency, Quasar open-source remote administration tool, 2019. URL: https://www.cisa.gov/news-events/analysis-reports/ar18-352a, accessed on [18.12.2023].

[20] David Fiser and Alfredo Oliveira, Tracking the Activities of TeamTNT: A Closer Look at Cloud-Focused Malicious Actor Group, . URL: https://documents.trendmicro.com/assets/white_papers/wp-tracking-the-activities-of-teamTNT.pdf, Accessed on: [18.12.2023].

[21] C. point, Apt35 exploits log4j vulnerability to distribute new modular powershell toolkit, https://research.checkpoint.com/2022/apt35-exploits-log4j-vulnerability-to-distribute-new-modular-powershell-toolkit/, 2022. Accessed on: [18.12.2023].

[22] Xiaopeng Zhang, Another Metamorfo Variant Targeting Customers of Financial Institutions, https://www.fortinet.com/blog/threat-research/another-metamorfo-variant-targeting-customers-of-financial-institutions, 2020. Accessed on: [18.12.2023].

[23] M. Corporation, MITRE ATTCK, Accessed: on: [02.01.2024]. URL: https://attack.mitre.org/.

[24] Cybersecurity and Infrastructure Security Agency (CISA), CISA, Accessed: on: [02.01.2024]. URL: https://www.cisa.gov/.

[25] Microsoft Corporation, Microsoft Security Blog, Accessed February 2024. URL: https://www.microsoft.com/en-us/security/blog/.

[26] Cisco Talos, Cisco Talos Blog, Accessed February 2024. URL: https://blog.talosintelligence.com/.

[27] Unit 42 Palo Alto Networks, Unit 42: Xagentosx – sofacy's xagent macos tool, 2017. URL: https://unit42.paloaltonetworks.com/unit42-xagentosx-sofacys-xagent-macos-tool/, accessed: on: [19.02.2024].

[28] T. D. ESET: Matthieu Faou, Mathieu Tartare, Operation ghost the dukes aren't back they never left, 2019. URL: https://web-assets.esetstatic.com/wls/2019/10/ESET_Operation_Ghost_Dukes.pdf, accessed: on: [05.01.2024].