






Multi-agent Architecture for Passive Rootkit Detection with Data Enrichment

Maickel Trinks¹ , João Gondim² , and Robson Albuquerque¹ 

¹ Professional Post-Graduate Program in Electrical Engineering - PPEE - Department of Electrical Engineering, University of Brasília, Brasília, DF, Brazil
maickeljosue@yahoo.com.br, robson@redes.unb.br

² Department of Computer Science, Universidade de Brasília, Brasília, DF, Brazil
gondim@unb.br

Abstract. The added value of the information transmitted in a cybernetic environment has resulted in a sophisticated malicious actions scenario aimed at data exfiltration. In situations with advanced actors, like APTs, such actions use obfuscation techniques of harmful activities as persistence assurance on strategic targets. The MADEX and NERD architectures proposed flow analysis solutions to detect rootkits that hide network traffic; however, it presents some operational cost, either in traffic volume or due to lack of aggregated information. In that regard, this work changes and improves user flow analysis techniques to eliminate impacts on network traffic, with data enrichment on local and remote bases, detection of domains consulted by rootkits and aggregation of information to generate threat intelligence, while maintaining high performance. The results show the possibility of aggregating information to data flows used by rootkits in order to have effective cyber defense actions against cybernetic threats without major impacts on the existing network infrastructure.

Keywords: Rootkit detection · data enrichment · threat intelligence · cybersecurity

1 Introduction

The last few years have brought significant changes in the cyber environment. In addition to the constant increase in users connected to the Internet [6] and migration of services to online platforms [3], the transformation caused by the COVID-19 pandemic has forced the large-scale adoption of information technology to overcome the consequences caused by the disease [5] and reduce its proliferation. As examples, there are various initiatives to adopt remote work regimes, distance learning, teleconferences, and migration of services to digital platforms, among others.

In this scenario, in addition to the growing trend of threat actors using advanced technological resources [5], rapid adaptation imposed by pandemic

made it challenging to adopt cybersecurity controls in appropriate places, making information systems even more vulnerable to cyberattacks. Morgan [10] has estimated a global cost of cybercrime of \$10.5 trillion annually in 2025, up from \$6 trillion in 2021.

The increase of sensitive information in digital media has also risen the sophistication of malicious techniques aimed at data exfiltration [13]) and digital extortion, especially with the advent of ransomware attacks. Targeted phishing (spearphishing) techniques for installing malware on targets of interest, such as Advanced Persistent Threats (APTs), partly sponsored by nation states, have been used for cyber espionage on large corporations [1].

This work aims to propose and describe the Passive Rootkit Detector With Enriched Data (PARDED) system, an architecture for detecting rootkits based on hiding network packets passively, which allows feeding other defense systems through centralization and enrichment of information (footprints). The relevance lies in the fact that rootkits, malware distinguished by their high ability to hide their presence on infected systems, can be used by various cyber actors in malicious actions. Their characteristics make them of great value to APT groups and ransomware operators. When adequately executed, attacks resulting from rootkits are highly efficient and difficult to detect. Thus, it is necessary to develop defense techniques that allow its detection and subsequent removal.

PARDED was created based on the extension of the MADEX architecture, proposed by Marques et al. [9] and NERD, presented by Terra and Gondim [14]. As a differential from previous architectures, PARDED has as its main characteristic the improvement of blocking techniques in communication network infrastructures without degrading the data transmission rate in network equipment, in addition to providing an interface for integration with other defense systems previously existing systems such as firewalls and intrusion detection systems. In addition, another important innovation is the possibility of enriching information about malicious destinations used by rootkits, notably for command and control (C2) systems, through local or external information bases. The enrichment process helps, for example, detecting the domain names used by C2 systems, which allows subsequent intelligence analysis with greater detail, maintaining the context of the evaluation object, avoiding unnecessary enrichments that pollute the analysis environments.

The organization of this work contains, in addition to this introduction, the Concepts section, which defines the critical expressions for understanding the proposal presented; the Related Work section, which briefly explains the architectures used as a basis for the project; the Proposed Architecture section, which describes the PARDED system; and the Results section, which presents data on the performance and enrichment of flows obtained in the laboratory. Finally, the conclusion exposes the work's synthesis and possible future paths.

2 Concepts

There is great relevance in some concepts for a better understanding of the PARDED system and the malicious actions that PARDED proposes to detect:

2.1 Advanced Persistent Threat (APT)

A high-sophisticated attack to an organization that continues its activities and yet remains undetected for an extended period, for control and long-term data collection, through a slow and stealthy approach to avoid detection [1].

2.2 Data Enrichment

Refers to the process of appending or otherwise enhancing collected data with relevant context obtained from additional sources [8]. Typically, data enrichment is achieved using external databases.

2.3 Multi-agent Systems

A specific type of distributed system, where components of the system are autonomous and selfish, seeking to satisfy their objectives. In addition, these systems also stand out for being open systems without a centralized design [7,15].

2.4 Onion Network (TOR)

Onion routing is an infrastructure for private communication over a public network. It provides anonymous connections strongly resistant to eavesdropping and traffic analysis. Applications connect through a sequence of machines called onion routers to reach a responding device. Anonymous connections hide who is connected to whom and for what purpose [12].

3 Related Works

The PARDED architecture derives from a multi-agent base composed of elements that work at different points of a communication network infrastructure and was based on previous work described below:

3.1 MADEX

Multi-Agent Data Exfiltration Detection Architecture (MADEX) [9] is a multi-agent architecture whose functionality is to detect rootkits that perform traffic obfuscation by altering the connection table of the infected terminal. The system comprises a Collector Agent, responsible for data collection, and an Auditor Agent, which detects all traffic from the infected terminal. The Auditor Agent

then interacts with the Collector to verify if the Collector has noticed the incoming traffic. The perception of legitimate communication depends on the connection being present in the operating system’s connection table. If not, it may indicate the presence of malware that hide traffic from the infected terminal, where the Collector is installed. Figure 1 represents MADEX architecture.

The system’s functioning depends on the Collector having the connection table information in a timely manner to respond the Auditor’s request. In case of a high number of packets, considering that the Collector cannot update the known connections before the polling time expires, all traffic would be marked as malicious.

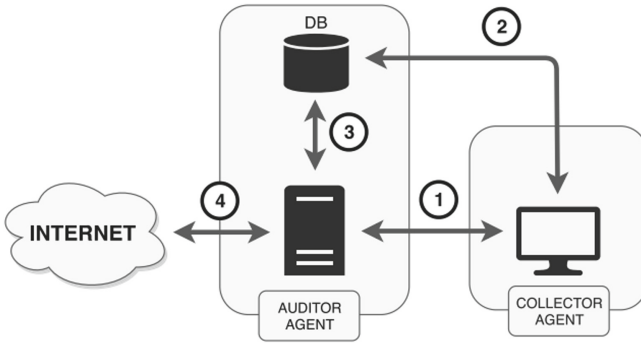


Fig. 1. MADEX architecture [9].

3.2 NERD

Network Exfiltration Rootkit Detector (NERD) [14], like MADEX, is a multi-agent architecture for rootkit detection purposes. However, instead of using the terminal connections table to check for possible obfuscation, it uses traffic capture through the *libpcap* library, avoiding constant calls to the operating system. After receiving the traffic transmitted by the terminal, the Auditor Agent interacts with the Collector to verify if the Collector perceived it. As with the MADEX architecture, traffic received at the Auditor and not perceived by the Collector may indicate the presence of malware that obfuscates detection on the infected terminal. Figure 2 represents NERD architecture.

The NERD architecture reduced false positives and detected all malicious traffic generated in lab tests. However, there was a degradation of approximately 50% in the packet transmission rate in low-performance networks (which have download rates of 50 Mbps) and about 90% in high-performance networks (with download rates close to 300 Mbps) [14], although it has shown capacity gains concerning the MADEX architecture [9].

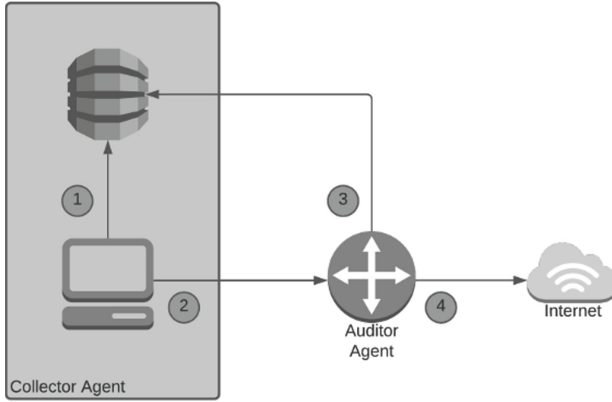


Fig. 2. NERD architecture [14].

4 Proposed Architecture

The proposed architecture of Passive Rootkit Detector With Enriched Data (PARDED) implements conceptual changes foreseen in the MADEX and NERD architectures, mainly in how the Auditor Element acts to detect and block malicious communication passively. This approach avoids degradation in the network's transmission rate and provides integration with other defense systems against malware. For such behavior, the auditor was divided into three components, with the first being composed of an inline element, which has the function of mirroring the traffic as an input to the analysis of the second component, out-of-band, which receives a copy of the traffic and compares it with the data of the Collector Agent for decision making, similarly to the NERD architecture. The third element, inline, receives information from the second element about connections considered malicious and which of them should be blocked.

This modification aims to eliminate the performance bottleneck caused by the characteristics of the active Auditor Agent existing in the other architectures, allowing the treatment of packages, the storage, and the enrichment of information without traffic delays. In this way, the proposed changes result in a structure that enables intelligence analysis of rootkit behavior on the network, either through the integration of data from multiple terminals, information history, and detection of used malicious domains, or through the aggregation of new verification ways with external databases support. Creating a database with suspicious behavior characteristics also allows feeding other network defense systems, such as systems that use blocking rules by IP, domain, or YARA rules. [11].

Figure 3 illustrates the proposed architecture and the following topics explain in detail each of the elements that compose it.

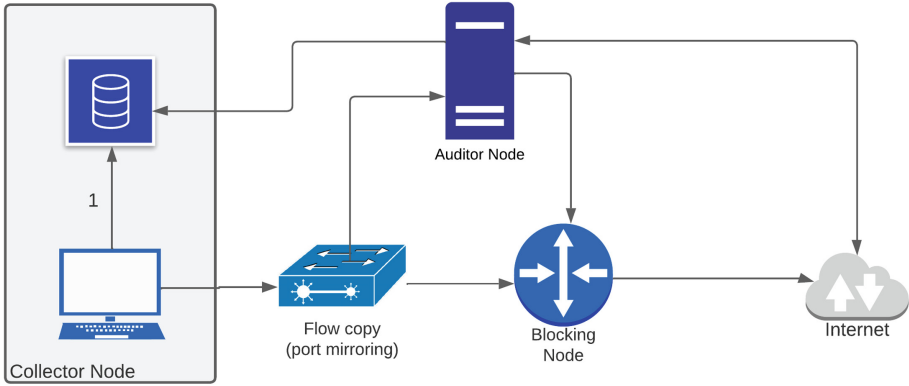


Fig. 3. PARDED - Proposed architecture.

4.1 Detection Strategy

The detection of suspicious traffic is performed by comparing the traffic observed at the terminal, obtained by the Collector Element (ColEl), and the traffic received by the Auditor Element (AudEl). Packets copied to AudEl that ColEl did not observe may indicate evasion of traffic visualization on the terminal by a malicious action of a rootkit. This detection, together with information from other endpoints and external databases, defines the action for each potentially malicious flow.

Collector Element (ColEl). It works passively through the analysis of traffic replica from the terminal. ColEl uses the approach foreseen in the NERD architecture due to the results presented, without changes, where the Collector performs the copy of the traffic through sockets linked to the active network device with the *libpcap* library. The difference presented by PARDED is the use of port mirroring to feed the Auditor Element, just like in an Intrusion Detection System (IDS) [1]. This strategy allows the auditor to passively process the information without interfering with the performance of the network.

Duplicator Element (DupEl). Its only function is to mirror the network traffic from the analyzed terminal to the Auditor Element (AudEl). This strategy allows AudEl works without interfering on network performance and to process information passively, as in an IDS. It can be any kind of hardware that duplicates information, like a network hub or a switch with port mirroring.

Auditor Element (AudEl). Unlike the MADEX and NERD architectures, AudEl does not perform the blocking function internally, allowing it to work out-of-band. This approach can perform more complex analyses before blocking traffic without impacting terminal communication performance.

The information enrichment is initially performed with the capture and store of DNS queries received by the Auditor. In this way, if a DNS request precedes malicious communication, the domain consulted by malware is detected and is included in the information to be enriched. Then, AudEl performs queries to ColEl (and other local or external systems), enabling anomalous or malicious activities detection.

For AudEl to consider the flow malicious, it must reach certain detection thresholds. The definition of initial parameters for each threshold is based on the hypotheses below:

- Six flows that are not detected by the same ColEl (same terminal) are considered malicious action. We used these premises because the NERD architecture obtained excellent false-positive results with this configuration [14].
- Two terminals whose collectors do not detect two flows to the same destination are enough for packets to be considered malicious. This premise is also based on the results obtained by NERD, added to the improbability that false positives, already rare in this architecture, occur in different collectors for the same destination IP.
- Two flows that are not detected and shares same destination, in the following situations:
 - The destination belongs to a TOR network;
 - The queried domain is considered malicious by at least two security companies;
 - The IP is considered malicious by at list four security companies.

These parameters depend on knowledge of the network infrastructure used to deploy the system. For example, if the network infrastructure allows or predicts user access to TOR network nodes, low threshold values can increase the number of false positives.

The functionalities were split modularly to allow easy evolution of the AudEl and prevent system interruption in case of failure in any step, as shown in Fig. 4. In this way, the system becomes generalizable and adaptable.

After receiving a data packet, the Initial Analysis System checks if it is from an already known flow (it's in Temporary Base). This strategy prevents the AudEl from performing unnecessary queries to the ColEl, making the analysis more efficient by reducing the traffic generated between auditor and collector and reducing processing in the Auditor and the Collector Elements. When AudEl does not know the flow, it consults ColEl. If ColEl has perceived the packet, it is considered legitimate traffic and stores the flow on the Temporary Base; otherwise, it is considered suspicious and the Auditor stores (in the Connection Database) the relevant flow data. This data, along with enrichment information and data from the other ColEls, is used to decide if the flow will be considered malicious.

The Enrichment System verifies, through the Connection Database, which data has not yet been enriched and queries the configured bases. The enrichment system is implemented in such a way that permits external base addition without layout changes.

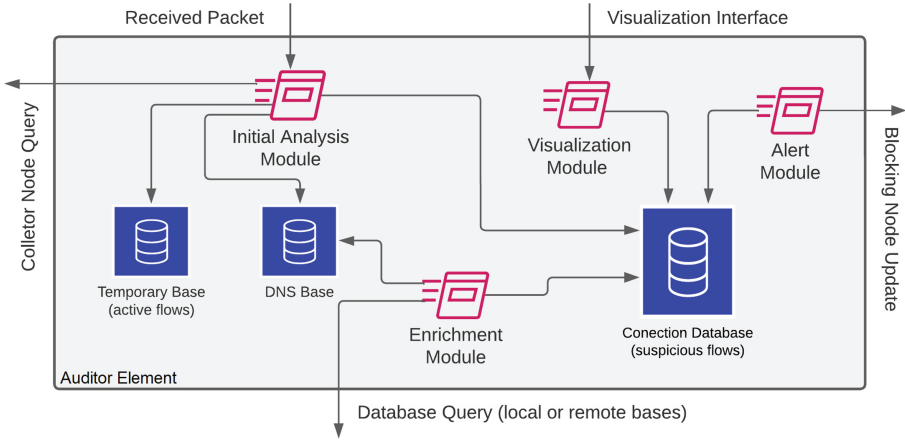


Fig. 4. PARDED - Auditor Element.

The Visualization System allows a security analyst to check the flows marked as suspicious and their characteristics, in addition to an overview of the tool’s operation, facilitating its use in data monitoring environments and integration with other security systems.

Through the Connection Database, the Alert Module verifies which destinations have reached the configured thresholds for a flow to be considered malicious and informs the Blocking Element. The Alert Module is scalable; it can send warnings to different blocking devices.

Blocking Element (BlkEl). It is an inline device that checks whether it should block the received network packet based on the warning received by the AudEl. This element can be any device that allows blocking rules update or YANA rules update remotely, such as routers and firewalls.

The elapsed time between detection of the first suspicious packet, enrichment of the information, sending block notification by the Alert Module, and the effective blocking action by the BlkEl depends on several factors, such as ColEl query time, remote base query time for enrichment, updating the connection base and reaching the thresholds defined by the auditor.

5 Results

The tests were carried out to verify the architecture’s viability and the execution time of each stage of the package analysis process by the Auditor Element. AudEl and ColEl were directly connected without using any intermediary network elements. Softwares run on Linux Debian 11 Operating System. The configuration of the Collector Element is as described in NERD architecture [14]. AudEl uses

a relational database (PostgreSQL) for the Connection Database. The prototype used in the tests reflects the architecture proposed in this work and is fully functional.

The development of the Initial Analysis System, belonging to AudEl, was in C language, as it is the most critical module. Since they don't have the same performance characteristics, the Enrichment, Alert, and Visualization Modules were made in Python3 language, facilitating the compatibility with external databases and blocking elements and graphics libraries for visualization. The tests were performed through traffic simulation, with packets sent by the terminal containing a Collector Element copied to the Auditor Element by port mirroring technique.

Two test scenarios were designed. The first, without the active presence of a rootkit, where all data streams are legitimate (not obfuscated), was carried out to verify that all transmitted packets would be correctly processed and also the performance of each system element. The test, containing between 600 and 1200 data streams with 10 packets each, was designed to obtain the average packet processing time after 5 complete executions.

The second, with the active presence of the rootkit, where legitimate (unobfuscated) and malicious (obfuscated) flows were generated, was carried out to verify that all transmitted packets would be correctly processed, the detection rate of malicious flows, and the performance of each system element, for both malicious and legitimate flows. The test, containing between 500 and 1000 legitimate data streams and 100 malicious streams with 10 packets each, was designed to obtain the average packet processing time after 5 executions.

Both scenarios were repeated with additional data to verify performance with high load. The transmission rate chosen for the execution of the tests was 50 Mbps, the same used in the NERD architecture during tests. The load (packet flooding) was made with the Iperf3 software.

The results took into account only TCP flows and "DNS Response" packets. There was no performance change in the network since the AudEl does not interfere with the inline elements; there was only a slight increase in the transmission rate (less than 2%) generated by the queries made to ColEl and external databases.

5.1 Legitimate FLoWs

In this scenario, the rootkit was not active (without traffic obfuscation). In simulated data transmission, all packets received in AudEl were perceived by ColEl; thus, there were no false positives. On average, packets from unknown flows by AudEl (which requires ColEl queries) were processed in 0.5 s. The Initial Analysis System processed other packets in the flow (in the Temporary Base and representing 95% of the total throughput) in less than 0.0012 ms each. The "DNS response" packets, locally treated to detect possible domains used by rootkits, were processed in approximately 0.004 ms.

Table 1 shows the obtained results with legitimate flows only.

Table 1. System Response Time (without rootkit active operation).

Process Type	Flow	Average Packet Count	Process Time (ms)	Process Time with 50 Mbps load (ms)
Without ColEl verification	In Temporary Base	55,393	0.00110	0.00100
	DNS Response	2030	0.00407	0.00734
With ColEl verification	Legitimate Flows	983	516.78	551.05

5.2 Suspicious Flows

In this scenario, the rootkit was active and obfuscated traffic was transmitted, corresponding to approximately 2% of no-load traffic (100 flows of 10 packets each). In the case of packages that required queries to ColEl (and were considered suspicious), the response time was, on average, 1.6 s. The increase in processing time is due to the treatment and updating of the base of suspicious connections (Connection Database). There was no significant change in the processing time of DNS Response packets or packets present in the Temporary Base.

The results were considered satisfactory for out-of-band solutions that depend on other factors, such as access to external bases and network protocols, such as TCP. The performance of the ColEl is similar to that presented by NERD [14]. That was expected because this project maintains the exact characteristics of the Collector. Table 2 shows the obtained results with suspicious flows.

Table 2. System Response Time (with rootkit active operation).

Process Type	Flow	Average Packet Count	Process Time (ms)	Process Time with 50 Mbps load (ms)
Without ColEl verification	In Temporary Base	54,058	0.00114	0.00100
	DNS Response	1,329	0.00370	0.00730
With ColEl verification	Legitimate Flows	684	536.15	551.05
	Suspicious Flows	100	1,642.64	1,709.62

5.3 Enrichment System

The local enrichment tests were performed with a local database of nodes from the TOR network. It uses a database obtained from the website dan.me.uk [4], locally stored. The Enrichment System processes these queries (local database) in approximately 42 ms each. The remote enrichment tests were performed in real-time using the IP and Domain detection base provided by the Virustotal platform [2]. As this test depends on factors not controlled by the AudEl, such

as the internet connection used, in addition to the processing time and response of the researched system, process time results showed more significant variance. On average, it took about 1.2s to query the Enrichment System on this base, as shown in Table 3.

Table 3. Enrichment System actions.

Process Type	Database Description	Response Time (ms)
Local Base Query	Onion (TOR) node	42.18
Remote Base Query	VirusTotal Platform	1,241.14

5.4 Visualization

The Visualization System was structured to illustrate the data of the flows considered suspicious, the enrichment performed by the system, the status of the Collector Element, and the queried databases, in addition to related data to each suspicious flow. The interface presents two modules: general data and suspicious flow data. In general data, it graphically presents the query results in the external and local databases, besides the solution's working status and a summary of all flows detected, enriched, and marked as malicious, as shown in Fig. 5. In suspicious flow data, each flow is detailed with information from the source terminal, the destination IP, blocking information, and enrichment data, besides the domains used to obtain the destination IP by the source terminal, as shown in Fig. 6. By clicking on a flow, it is possible to see detailed information about each enrichment, like the ones shown in Fig. 7 and Fig. 8.

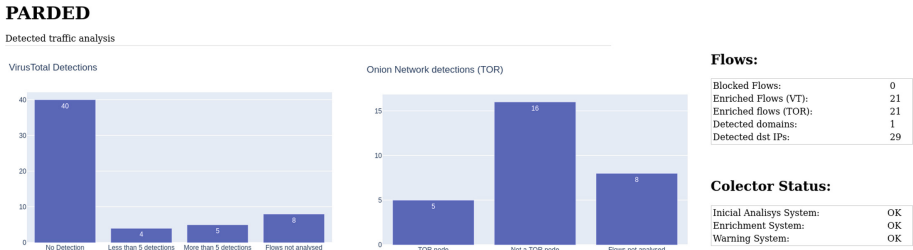


Fig. 5. PARDED - Auditor Element (general data).

5.5 Conclusions

In this work, the developed architecture could detect rootkits that use communication obfuscation techniques in the infected terminal, in a passive, scalable, generalizable, and adaptable way, without impacting the performance of the

Fluxos suspeitos:

Id	Source IP	Destination IP	VT Enriched	TOR Enriched	Blocked	Domain List
32	192.168.190.10	208.84.264.50	true	true	false	
39	192.168.190.10	23.108.123.73	true	true	false	ruycallback.771658.com
40	192.168.190.10	199.232.114.132	true	true	false	debian.map.fastlydns.net
41	192.168.190.10	40.117.131.251	true	true	false	packages.microsoft.com
49	192.168.190.10	201.16.50.19	true	true	false	r3.o.lencr.org
50	192.168.190.10	2.16.15.19	true	true	false	r3.o.lencr.org
51	192.168.190.10	184.18.32.48	true	true	false	ocsp.sectigo.com
52	192.168.190.10	108.158.121.180	true	true	false	ocsp.scalb.amazontrust.com
53	192.168.190.10	104.18.21.226	true	true	false	ocsp.globaltrust.com
54	192.168.190.10	217.23.15.46	false	false	false	

Fig. 6. PARDED - Auditor Element (suspicious flow data).

Malicious Detection	Domain	IP	Date
0		23.108.123.73	2022-07-17T00:07:06.564395
0	ruycallback.771658.com		2022-07-17T00:07:07.706740

Fig. 7. PARDED - Enrichment data from an external base.

network infrastructure. Furthermore, it adds data enrichment and incorporates information derived from multiple endpoints, allowing integration with other existing defense systems, such as IDS and firewalls. This way, it adds a new technique for real-time monitoring of malicious actions to organizations.

The detection of domain names used by suspicious flows, with enrichment data from local and external sources, in addition to the historically and visually presentation, adds threat intelligence and allows further analysis in greater detail, maintaining the context of the object of the evaluation, avoiding unnecessary enrichments that pollute the analysis environments.

About 95% of packets were processed in less than 0.0012 ms, even on heavily loaded systems, without any legitimate packet loss. Since they are executed in parallel, the enrichment data do not interfere with the legitimate data flow. These results prove the possibility of using PARDED in existing network infrastructures without significant impacts.

However, as the system detects suspicious flows passively and depends on information collected and processed after the first packet from the analyzed data stream, at least the initial packets of suspicious flows will not be blocked, even if treated as malicious. Future works can optimize the threshold values used as a baseline for blocking packets, define ciphertext packets (such as DNS-over-HTTPS) strategies, integrate auxiliary detection techniques and databases, and verify the payloads transmitted by suspicious flows in malicious hash bases.

Name	Flags	Version	Date
Unnamed	FGHRSDVX	Tor 0.4.5.8	2022-07-17T00:36:24.120958

Fig. 8. PARDED - Enrichment data from a local base.

Acknowledgements. M.T. gratefully acknowledge Mateus Berardo de Souza Terra for providing the source code of the Collector Element used in the Network Exfiltration Rootkit Detector (NERD). R.A. gratefully acknowledges the technical and computational support of the Laboratory of Technologies for Decision Making (LATTITUDE) of the University of Brasília, the General Attorney’s Office (Grant AGU 697.935/2019), the General Attorney’s Office for the National Treasure - PGFN (Grant 23106.148934/2019-67), and the National Institute of Science and Technology in Cyber Security - Nucleus 6 (grant CNPq 465741/2014-2); The authors thankfully acknowledge the support of Brazilian Intelligence Agency - ABIN grant 08/2019.

References

1. Basin, D.: The cyber security body of knowledge. University of Bristol, ch. Formal Methods for Security, version (2021). <https://www.cybok.org>
2. Chronicle Security: Virustotal. <http://www.virustotal.com>. Accessed 15 July 2022
3. Costa, H., Nicoletti, G., Pisu, M., Von Rueden, C.: Are digital platforms killing the offline star? Platform diffusion and the productivity of traditional firms. OECD Economics Department Working Papers, vol. 1, no. 1682, pp. 1–27 (2021). <https://doi.org/10.1787/18151973>
4. Dan: Tor Node List. <https://www.dan.me.uk>. Accessed 11 July 2022
5. ENISA: The year in review - ENISA Threat Landscape (2020). <https://www.enisa.europa.eu/publications/year-in-review/@@download/fullReport>
6. ITU: Individuals using the Internet (2021). <https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx>. Accessed 04 July 2022
7. Julian, V., Botti, V.: Multi-agent systems. *Appl. Sci.* **9**(7) (2019). <https://doi.org/10.3390/app9071402>
8. Knapp, E.D., Langill, J.T.: Exception, anomaly, and threat detection. *Ind. Netw. Secur.* 323–350 (2015). <https://doi.org/10.1016/B978-0-12-420114-9.00011-3>
9. Marques, R.S., et al.: A flow-based multi-agent data exfiltration detection architecture for ultra-low latency networks. *ACM Trans. Internet Technol.* **21**(4), 1–30 (2021). <https://doi.org/10.1145/3419103>
10. Morgan, S.: Cybercrime To Cost The World \$10.5 Trillion Annually By 2025. Cybersecurity Ventures, p. 1 (2020). <https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/>. Accessed 20 July 2022
11. Naik, N., et al.: Embedded YARA rules: strengthening YARA rules utilising fuzzy hashing and fuzzy rules for malware analysis. *Complex Intell. Syst.* **7**(2), 687–702 (2021). <https://doi.org/10.1007/s40747-020-00233-5>
12. Reed, M., Syverson, P., Goldschlag, D.: Anonymous connections and onion routing. *IEEE J. Sel. Areas Commun.* **16**(4), 482–494 (1998). <https://doi.org/10.1109/49.668972>
13. Sowinski, D.: The Growing Danger of Data Exfiltration by Third-Party Web Scripts. <https://securityintelligence.com/posts/growing-danger-data-exfiltration-third-party-web-scripts/>. Accessed 04 July 2022
14. Terra, M.B., Gondim, J.J.: NERD: a network exfiltration rootkit detector based on a multi-agent artificial immune system. In: 2021 Workshop on Communication Networks and Power Systems, WCNPS 2021 (2021). <https://doi.org/10.1109/WCNPS53648.2021.9626241>
15. Wooldridge, M.: An Introduction to Multiagent Systems. Wiley, England (2009)