IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.19, No.2, April 2025, pp. 201~210
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: 10.22146/ijccs.104450

Systematic Review of High Interaction Honeypots for Microsoft SQL Server

Faiz Unisa Jazadi*[1], I Gede Mujiyatna[2]
[1,2]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]faiz.uni2003@mail.ugm.ac.id**, [2]demuji@ugm.ac.id

Abstrak

Tinjauan sistematis ini bertujuan untuk mendalami tentang honeypot interaksi tinggi untuk Microsoft SQL Server. Topik yang dibahas meliputi berbagai lingkungan honeypot (bare-metal, mesin virtual, kontainer) dan metode pemantauan (berbasis jaringan, berbasis VMM, berbasis honeypot) untuk memahami cara memantau komunikasi terenkripsi secara efektif. Fokus utamanya adalah membandingkan berbagai teknik pemantauan data untuk honeypot dengan interaksi tinggi, terutama dengan mempertimbangkan tantangan yang ditimbulkan oleh protokol terenkripsi seperti TDS yang digunakan oleh Microsoft SQL Server. Penelitian ini mengidentifikasi keterbatasan dalam penelitian saat ini dan mengusulkan penggunaan proxy MITM terenkripsi sebagai solusi potensial. Pada akhirnya, penelitian ini menyoroti perlunya penelitian lebih lanjut di bidang ini karena terbatasnya literatur yang ada tentang honeypot interaksi tinggi untuk Microsoft SQL Server.

Kata kunci*—honeypot, interaksi tinggi, microsoft sql server*

Abstract

This systematic review aims to dive into high interaction honeypots for Microsoft SQL Server. Topics covered include various honeypot environments (bare-metal, virtual machine, container) and monitoring methods (network-based, VMM-based, honeypot-based) to understand how to effectively monitor encrypted communications. The main focus is to compare different data monitoring techniques for high-interaction honeypots, especially considering the challenges posed by encrypted protocols such as TDS used by Microsoft SQL Server. This research identifies limitations in current research and proposes the use of encrypted MITM proxies as a potential solution. Ultimately, this research highlights the need for further research in this area due to the limited existing literature on high interaction honeypots for Microsoft SQL Server.

Keywords—*honeypot, high interaction, microsoft sql server*

1. INTRODUCTION

Along with the rapid development of technology, cybersecurity threats also continue to grow, targeting critical infrastructure and services [1]. The increasing adoption of technologies supported by very high connectivity, such as the Internet of Things (IoT), cloud computing, and big data also provide opportunities for attackers to carry out more sophisticated cyber attacks. One entity that is often targeted by cyber attacks is the database because of its strategic logical position in an organization's IT infrastructure [2]. Databases generally store sensitive and important data, such as customer data, financial data, and operational data [3], [4]. Cyberattacks

Received February 4th,2025; Revised Aprtil 28th, 2025; Accepted April 30th, 2025

on databases can result in significant losses for organizations, such as data theft, fraud, and financial loss [5], [6]. Databases can also be used as an entry point to attack other systems in an organization's IT infrastructure [7].

One way to observe and study cyberattacks on databases is by using honeypots. Honeypot is an information system resource that is valuable when it is misused or attacked [8]. Honeypots can be used to attract attackers and learn the tactics, techniques, and procedures used by attackers. Aside from research purposes, information obtained from honeypots can also be used as an active defense or protection mechanism [9], [10], [11]. Honeypots are generally classified based on the level of interaction with the attacker into high and low interaction [12], [13]. Low interaction honeypots generally only provide limited interaction such as protocol and network emulation, while high interaction honeypots provide more complex interaction by running real services that are vulnerable to attack. Therefore, the data collected by high interaction honeypots is richer and more varied than low interaction honeypots.

Microsoft SQL Server is one of the popular relational database management software (RDBMS) based on market share figures [14], [15]. As one of the popular RDBMS, Microsoft SQL Server is also one of the targets of cyber attacks that are often attacked [16], [17]. However, based on the initial search on the five databases used in this study, studies on high interaction honeypot for Microsoft SQL Server are still limited [13], [18]. Therefore, this study aims to systematically review literature sources related to high interaction honeypot for Microsoft SQL Server. Specifically, research questions that will be answered in this research are as follows.

1) What are the general characteristics related to environmental settings and monitoring methods in the high interaction honeypot system? What are the advantages and disadvantages?
2) How is monitoring performed on encrypted communications?
3) How to effectively monitor attacker interaction on high interaction Microsoft SQL Server honeypot?

2. METHODS

The research question will be answered by conducting a systematic review. A systematic review is a research method that uses an explicit systematic approach to collate, critically evaluate, and synthesize findings that address a clearly formulated research question [19].

2.1 Inclusion and Exclusion Criteria

Studies selected for systematic review had to meet the following criteria inclusion criteria: (1) there is a discussion that contains practical aspects of honeypots high interaction for one or more server-side software services, e.g. Microsoft SQL Server, OpenSSH, telnet, (2) published in a scientific journal or international conference, and (3) published in an international conference, (3) published within the last 10 years. Studies that did not meet the inclusion criteria or met the exclusion criteria: (1) discussed a focus that was not in accordance with the research topic, for example, honeypots on the client-side honeypots, (2) containing honeypots for very specific domains such as honeypots for specific device firmware.

2.2 Information Sources and Search Strategy

The search will be conducted in several databases using a specific strategy to identify relevant studies. The following are the databases used along with the search strategy used.

1) Web of Science Core Collection (2024-12-05)
    a) Query: *high interaction honeypot* on all fields
    b) Filters: "article" and "proceeding paper" document type
2) IEEE Xplore (2024-12-05)
    a) Query: *high interaction honeypot* on all fields
    b) Filters: "conferences" and "journals" document type
3) ScienceDirect (2024-12-05)
    a) Query: *"high interaction" AND honeypot*
    b) Filters: "journal" and "conference" document type
4) Scopus (2024-12-05)
    a) Query: *high AND interaction AND honeypot*
    b) Filters: "article" and "conference paper" document type
5) Google Scholar (2025-01-13)
    a) Query: *mssql microsoft sql server honeypot*
    b) Sorting order: by relevance
    c) Filters: since 2014, only the first 100 results taken

2.3 Research Procedure

The search is conducted according to the specified search strategy. The keywords used in the search will be adjusted to the search features of each database. Files that have been identified through the search process will be collected and stored in the reference manager for the initial screening process in the form of removing duplicates and ineligible files. After that, the remaining files will go through a screening process based on the title and abstract. Reports that pass the screening process will be downloaded and read in full to determine their eligibility as systematic review material.

Data synthesis is carried out by identifying and extracting relevant information from reports that pass the selection. The information extracted includes the honeypot setup and the monitoring methods used. This study will then analyze and compare the characteristics of the various approaches used in designing and implementing high-interaction honeypots. The results of the analysis will be used to answer the research questions.

3. RESULTS AND DISCUSSION

At each stage of selection shown in the flowchart in Figure 1, *n* indicates the number of studies involved. At the identification phase, *n* for Google Scholar was 564, but only the first 100 results were taken. A total of 218 records were removed due to duplicates based on title and abstract. Another 110 records were removed because they did not meet the inclusion criteria and had incomplete metadata. At the screening stage, a total of 308 records were removed because they were not relevant to this study, such as discussing honeypots for the client side, honeypots for hardware, and honeypot data analysis methods. There were 5 records that could not be downloaded due to the author's limited access. A total of 17 records were selected to be downloaded and read in full. Of the 17 reports that were successfully downloaded, 4 were irrelevant because they discussed theoretical studies related to honeypots or topics outside the scope of this study. There were 2 reports that were irrelevant because they did not provide sufficient details regarding the honeypot settings used. The selection process left 11 reports to be included in this systematic review. The titles and characteristics of the included studies can be seen in Table 1.

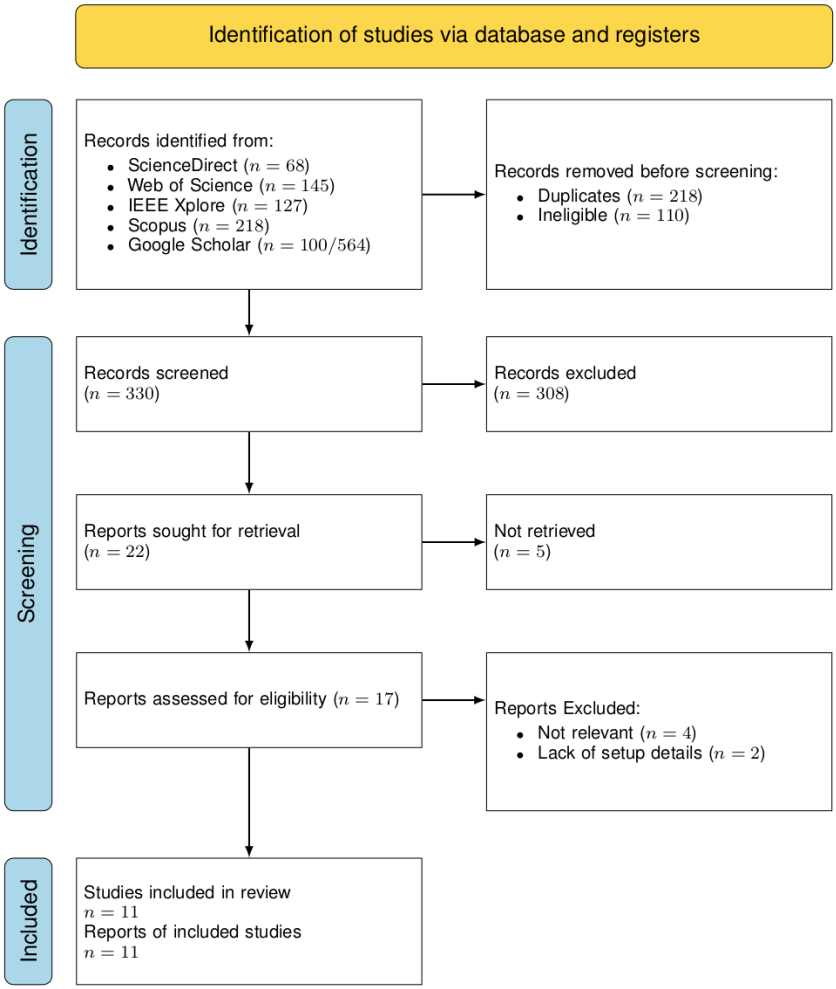

Figure 1 Study selection flow diagram

Table 1 Included studies and their characteristics

Ref.	Title	Characteristics
[20]	Towards virtual honeynet based on LXC virtualization	Hybrid honeypot architecture using LXC containers in comparison to VMs
[21]	CloudHoneyCY - An Integrated Honeypot Framework for Cloud Infrastructures	Framework for hybrid honeypot management on cloud infrastructures
[22]	Taming the IPv6 Address Space with Hyhoneydv6	Hybrid honeypot architecture with IPv6 support
[23]	Implementing High Interaction Honeypot to Study SSH Attacks	High interaction SSH honeypot monitored using system call hook via LKM
[24]	Creation and Integration of Remote High Interaction Honeypots	Integration of remote high interaction honeypot into an organization's internal network using VPN
[25]	Planning and Implementation of Honeypot System - Building of a bogus	Honeypot based on bogus Microsoft SQL Server monitored using proxy
[26]	Creation of a High-Interaction Honeypot System based-on Docker containers	Dynamic high interaction honeypot setup with Docker containers

[27]	Analysing Attackers and Intrusions on a High-Interaction Honeypot System	Analyzing SSH attacks by utilizing modified versions of OpenSSL and OpenSSH
[28]	SSHkex: Leveraging virtual machine introspection for extracting SSH keys and decrypting SSH network traffic	Lower overhead VMI approach for SSH session key extraction
[29]	An Improved Honeypot Model for Attack Detection and Analysis	Honeypot model with planned attack path according to MITRE ATT&CK framework
[30]	Fifteen Months in the Life of a Honeyfarm	Attack analysis of a large scale Dionaea honeypot deployment

2.3 What are the general characteristics related to environmental settings and monitoring methods in the high interaction honeypot system? What are the advantages and disadvantages?

2.3.1 Environmental Setup

As summarized in Table 2, all studies included in this research used a high interaction honeypot that involved the attacker's interaction with a real operating system or application. In general, high interaction honeypot environments can be grouped into three types, namely bare-metal, virtual machines, and containers. The bare-metal environment was used in the [21] study using Raspberry Pi as the host. In addition to bare-metal, virtual machine environments in the form of KVM and full emulation were also used in several studies. User-space container environments like LXC and Docker were also found in several studies. All studies involved a physical or virtual isolation mechanism.

Table 2 Grouping of honeypot environment setup in the included studies

Environment		Ref.
Bare-metal	Raspberry Pi	[21]
Virtual machine	KVM	[22], [24]
	Full emulation	[22]
	Other	[23], [24], [25], [28]
Container	LXC	[20], [29]
	Docker	[26]

Study [20] found that using LXC containers is more efficient than virtual machines in implementing a hybrid honeynet. LXC containers allow for faster and more efficient honeypot deployment than virtual machines, even though they are not fully isolated [31]. Study [21] used Raspberry Pi due to the cost-effectiveness and ease of isolation at scale. Study [22] used a hybrid setup using a collection of virtual machines running on KVM and full emulation. In [22], the time required to handle an attacker request was 1.5 seconds for KVM and 2.5 seconds for full emulation. In [26], Docker containers enable dynamic and efficient honeypot deployment by using the attacker's IP address as a reference to create new containers. This approach is useful to ensure that data between attackers does not overlap.

2.3.1 Monitoring methods

Determining the monitoring method is one of the challenges in setting up a high interaction honeypot [18]. In general, the monitoring methods found in the reviewed studies can be grouped into three categories based on the monitoring location, namely network, VMM, and honeypot as seen in Table 3.

Table 2 Grouping of honeypot monitoring methods in the included studies

Monitoring Methods		Ref.
Network	Packet Sniffing	[21], [25], [26], [27], [28]
	Intrusion Detection System (IDS)	[20]
	Extended Detection and Response (XDR)	[29]
	MITM Proxy	[25], [30]
VMM	Memory dump	[22]
	Virtual Machine Introspection	[28]
Honeypot	Log file	[20]
	XDR Agent	[29]
	auditd, strace	[26]
	Modified program	[23], [27]

On the network level, the monitoring methods found are packet sniffing, security devices such as IDS or XDR, and with proxies. Packet sniffing is used to collect network traffic data that passes through the honeypot. This method is difficult to detect because it is generally hidden from the attacker's environment. This method is also effective for recording the entire spectrum of the attacker's network interactions with the honeypot [32]. However, the data collected is still raw data of all packets passing through the network. This data can grow rapidly in size and requires additional processing stages to be interpreted [18].

Study [20], [29] uses security devices in the form of Intrusion Detection System (IDS) and Extended Detection and Response (XDR) to detect incidents or security events found including attack attempts by attackers. The data collected by IDS/XDR is a security event that occurs within the honeypot environment. The Wazuh XDR used in [29] has an agent installed on the honeypot container, allowing for more detailed monitoring of security incidents. IDS/XDRs have the advantage of identifying threats or creating more structured event feeds compared to packet sniffing. However, IDS/XDR capabilities depend on the threat fingerprint database used and tend to be less flexible in detecting unidentified threats .

Study [30] used Cowrie as a honeypot. Cowrie is a honeypot for SSH and Telnet which supports low to high interaction mode [33]. Though not used in the study, Cowrie's high interaction mode works using a proxy approach. The proxy approach in Cowrie works by creating two SSH/Telnet connections: (1) between the attacker and the proxy, (2) between the proxy and the honeypot/host. In any mode, Dionaea is able to record the attacker's login activity, shell activity, and file transfer activity. Study [25] also used a MITM proxy approach to create a bogus Microsoft SQL Server instance. The MITM proxy approach can capture richer data features than

IDS/XDR on encrypted protocols such as SSH. However, aside from added latency, the implementation of this method is tightly coupled to a particular application protocol.

In [22], [28], monitoring is conducted at the Virtual Machine Monitor (VMM) level. Study [28] used a Virtual Machine Introspection (VMI) technique called SSHKex to extract session keys during the SSH key exchange process. The extracted keys were then used to decrypt TLS/SSL packets captured through packet sniffing. This method requires VMI facilities on the VMM and in-depth knowledge of the data structures to be extracted from memory. The VMI method adds overhead in the form of pause time on the VM which can affect the VM's quality of service.

Study [22] used memory dumps at specific times to analyze the impact of attacker interactions on the VM. The VMM approach allows for in-depth analysis of the impact of attacker interactions on the VM. However, the collected data may require complex extraction processes to be interpretable which is inefficient at scale [34]. This method also cannot be used for real-time monitoring as the memory is only dumped after the attacker session has timed out and thus it only reflects the consequences of the attacker's procedure.

Study [20] used a combination of IDS and Apache 2 log collection. The information obtained from the log files can reflect attacker activity at the application level, but is limited to the logging features provided by the application. Study [26] used strace and auditd on the Docker host to monitor attacker activity on the honeypot container. This method allows for more detailed monitoring at the operating system level, but is only applicable in Linux environments. Study [23] used the Loadable Kernel Module (LKM) to capture the arguments of several important system calls such as *read()* on Linux. This method allows for more detailed monitoring at the operating system level, but requires modifications to the kernel that can potentially compromise operating system stability.

2.3 How is monitoring performed on encrypted communications?

SSH monitoring methods are discussed in [27], [28] and Tabular Data Stream (TDS) in [25]. Signature-based monitoring methods such as IDS cannot fully capture encrypted SSH communications [27], [28]. In monitoring SSH connections, [28] uses a combined approach of passive packet capture and Virtual Machine Introspection (VMI) to extract session keys during the key exchange process. Study [27] modified the OpenSSH and OpenSSL source code to allow session keys to be stored. Another approach is to use a man-in-the-middle proxy like the high interaction feature in Cowrie used by [30]. In a similar fashion, [25] successfully monitored TDS 7.x connections through a reverse proxy by forcing the attacker to use an unencrypted channel as encryption is not mandatory. However, this weakness is not possible on TDS 8.0.

2.3 How to effectively monitor attacker interaction on high interaction Microsoft SQL Server honeypot?

Study [21] used Dionaea for a low interaction (emulation) honeypot and did not support encryption. Study [25] designed and implemented a man-in-the-middle proxy to a bogus Microsoft SQL Server in Python. However, the implemented proxy does not support the encryption feature in the TDS protocol. In its specification, Tabular Data Stream protocol versions 7 and 8 support TLS/SSL encryption [35]. Techniques for extracting encryption keys that are very close to implementation details such as VM introspection, program modification, are not legally possible as Microsoft SQL Server is a closed-source commercial product. Traditional packet sniffing approaches can be used but will not work if TDS encryption is used. Audit log-based monitoring can be used to see changes to the database but cannot directly reflect the

attacker's communications. A man-in-the-middle proxy approach similar to [25] but supports encryption like Cowrie[33] is a candidate for an effective monitoring method.

4. CONCLUSIONS

In this study, researchers tried to explore the environmental settings and monitoring methods for high interaction honeypots. This study also discusses the analysis of data monitoring method options for high interaction honeypot for Microsoft SQL Server. Eleven studies show that the environment for high interaction honeypot must involve an isolation mechanism, including creating a dedicated machine as a honeypot, using a virtual machine, or using a container. Isolation is necessary because high interaction honeypots are intentionally designed to allow attackers to interact directly with the real operating system or application [8]. The monitoring methods used in high interaction honeypots can be grouped into three categories, namely network, VMM, and honeypot. Network monitoring methods such as packet sniffing, IDS/XDR, and MITM proxy are effective methods for monitoring unencrypted data. VMM monitoring methods such as VMI and memory dump allow for in-depth analysis of the impact of attacker interactions on the VM. Honeypot monitoring methods such as log files, XDR agents, and modification programs allow for more detailed monitoring at the application or operating system level. Monitoring encrypted data such as SSH or TDS requires a special approach such as VMI or MITM proxy. Ultimately, the environment setup and monitoring methods for a high interaction honeypot depend on the needs and goals of the honeypot research or implementation being conducted.

Research on high interaction honeypot for Microsoft SQL Server is still limited. Surveys revealed that all honeypots for Microsoft SQL Server found were low-interaction honeypots [13], [18]. The emulative nature of low-interaction honeypots limits the ability to collect rich attacker data, limiting it to authentication credentials, handshake data, or connection metainformation [36], [37]. The MITM proxy approach used in [25] allows for richer data collection. However, the MITM proxy implementation used in the study did not support the encryption feature of the TDS protocol. After exploring the general characteristics of data monitoring on honeypots, this study suggests that an effective candidate monitoring method for Microsoft SQL Server is to use a MITM proxy that supports encryption such as the high interaction mode in Cowrie [13], [33].

ACKNOWLEDGEMENTS

This work was partially supported by the Department of Computer Science and Electronics, Universitas Gadjah Mada under the Publication Funding Year 2025.

REFERENCES

[1] CrowdStrike, “CrowdStrike Global Threat Report 2024,” CrowdStrike, Inc., 2024.
[2] G. Pestana and S. Sofou, “Data Governance to Counter Hybrid Threats against Critical Infrastructures,” *Smart Cities*, vol. 7, no. 4, pp. 1857–1877, Jul. 2024, doi: 10.3390/smartcities7040072.
[3] A. Nikiforova, “Data security as a top priority in the digital world: preserve data value by being proactive and thinking security first,” Mar. 18, 2023, *arXiv*: arXiv:2206.06814. doi: 10.48550/arXiv.2206.06814.
[4] Huawei Technologies Co., Ltd., *Database Principles and Technologies – Based on Huawei GaussDB*. Singapore: Springer Nature Singapore, 2023. doi: 10.1007/978-981-19-3032-4.
[5] Check Point, “Check Point Cyber Security Report,” Check Point Software Technologies, Ltd., 2024.
[6] K. Swani, L. Labrecque, and E. Markos, “Are B2B data breaches concerning? Consequences

of buyer's or firm's data loss on buyer and supplier related outcomes," *Ind. Mark. Manag.*, vol. 119, pp. 43–61, May 2024, doi: 10.1016/j.indmarman.2024.03.007.
[7] The MITRE Corporation, "Exploitation of Remote Services, Technique T1210 - Enterprise," MITRE ATT&CK®. Accessed: Dec. 02, 2024. [Online]. Available: https://attack.mitre.org/versions/v16/techniques/T1210/
[8] L. Spitzner, "Honeypots: catching the insider threat," in *19th Annual Computer Security Applications Conference, 2003. Proceedings.*, Las Vegas, Nevada, USA: IEEE, 2003, pp. 170–179. doi: 10.1109/CSAC.2003.1254322.
[9] Q. Sun *et al.*, "Research and Application of High Interaction Deception Defense and Traceability Based on RASP Technology," presented at the 2024 2ND INTERNATIONAL CONFERENCE ON MOBILE INTERNET, CLOUD COMPUTING AND INFORMATION SECURITY, MICCIS 2024, 2024, pp. 48–52. doi: 10.1109/MICCIS63508.2024.00016.
[10] X. Yang, J. Yuan, H. Yang, Y. Kong, H. Zhang, and J. Zhao, "A Highly Interactive Honeypot-Based Approach to Network Threat Management," *Future Internet*, vol. 15, no. 4, 2023, doi: 10.3390/fi15040127.
[11] W. Bythwood, A. Kien, I. Vakilinia, and IEEE, "Fingerprinting Bots in a Hybrid Honeypot," in *State University System of Florida*, 2023, pp. 76–80. doi: 10.1109/SoutheastCon51012.2023.10115143.
[12] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A Survey of Honeypots and Honeynets for Internet of Things, Industrial Internet of Things, and Cyber-Physical Systems," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 4, pp. 2351–2383, 2021, doi: 10.1109/COMST.2021.3106669.
[13] N. Ilg, P. Duplys, D. Sisejkovic, and M. Menth, "A survey of contemporary open-source honeypots, frameworks, and tools," *J. Netw. Comput. Appl.*, vol. 220, p. 103737, Nov. 2023, doi: 10.1016/j.jnca.2023.103737.
[14] DB-Engines, "Ranking of the most popular relational database management systems worldwide, as of June 2024," Statista, Inc. Accessed: Nov. 29, 2024. [Online]. Available: https://www.statista.com/statistics/1131568/worldwide-popularity-ranking-relational-database-management-systems/
[15] A. Akhtar, "Popularity Ranking of Database Management Systems," 2023, *arXiv*. doi: 10.48550/ARXIV.2301.00847.
[16] R. Zdonczyk, "Honeypot Recon: Global Database Threat Landscape | Trustwave." Accessed: Dec. 08, 2024. [Online]. Available: https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/honeypot-recon-global-database-threat-landscape/
[17] V. Narayan, A. Raj, and V. Muskan, "Exploitation of SQL Common Language Runtime Assemblies: A Novel Attack Vector for Compromising Microsoft SQL Server Environments," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kamand, India: IEEE, Jun. 2024, pp. 1–6. doi: 10.1109/ICCCNT61001.2024.10725944.
[18] S. C. Sethuraman, T. G. Jadapalli, D. P. V. Sudhakaran, and S. P. Mohanty, "Flow based containerized honeypot approach for network traffic analysis: An empirical study," *Comput. Sci. Rev.*, vol. 50, p. 100600, Nov. 2023, doi: 10.1016/j.cosrev.2023.100600.
[19] M. J. Page *et al.*, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *BMJ*, vol. 372, p. n71, Mar. 2021, doi: 10.1136/bmj.n71.
[20] N. Memari, S. J. B. Hashim, and K. B. Samsudin, "Towards virtual honeynet based on LXC virtualization," in *2014 IEEE Region 10 Symposium*, 2014, pp. 496–501. doi: 10.1109/TENCONSpring.2014.6863084.
[21] H. Gjermundrod and I. Dionysiou, "CloudHoneyCY - An Integrated Honeypot Framework for Cloud Infrastructures," in *University of Nicosia*, I. Raicu, O. Rana, and R. Buyya, Eds., 2015, pp. 630–635. doi: 10.1109/UCC.2015.110.
[22] S. Schindler, B. Schnor, T. Scheffler, and IEEE, "Taming the IPv6 Address Space with Hyhoneydv6," in *University of Potsdam*, 2015, pp. 113–118.
[23] M. Zemene and P. Avadhani, "Implementing High Interaction Honeypot to Study SSH

Attacks," in *Andhra University*, J. Mauri, S. Thampi, M. Wozniak, O. Marques, D. Krishnaswamy, S. Sahni, C. Callegari, H. Takagi, Z. Bojkovic, M. Vinod, N. Prasad, J. Calero, J. Rodrigues, X. Que, N. Meghanathan, R. Sandhu, and E. Au, Eds., 2015, pp. 1898–1903.

[24] M. Valicek, G. Schramm, M. Pirker, S. Schrittwieser, and IEEE, "Creation and Integration of Remote High Interaction Honeypots," in *St. Polten University of Applied Sciences*, 2017, pp. 50–55. doi: 10.1109/ICSSA.2017.21.

[25] V.-M. A. Mäntysaari, "Planning and Implementation of Honeypot System - Building of a bogus Microsoft SQL Server," Bachelor's Thesis, Turku University of Applied Sciences, 2020. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/353709/Planning%20and%20implementation%20of%20honeypot%20system.pdf

[26] J. Buzzio-Garcia, "Creation of a High-Interaction Honeypot System based-on Docker containers," in *2021 Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4)*, 2021, pp. 146–151. doi: 10.1109/WorldS451998.2021.9514022.

[27] M. Knöchel, S. Wefel, and IEEE, "Analysing Attackers and Intrusions on a High-Interaction Honeypot System," in *Martin Luther University Halle Wittenberg*, 2022, pp. 433–438. doi: 10.1109/APCC55198.2022.9943718.

[28] S. Sentanoe and H. Reiser, "SSHkex: Leveraging virtual machine introspection for extracting SSH keys and decrypting SSH network traffic," *Forensic Sci. Int.-Digit. Investig.*, vol. 40, Apr. 2022, doi: 10.1016/j.fsidi.2022.301337.

[29] M. Abbas-Escribano, H. Debar, and ACM, "An Improved Honeypot Model for Attack Detection and Analysis," in *IMT - Institut Mines-Telecom*, 2023. doi: 10.1145/3600160.3604993.

[30] C. Munteanu, S. J. Saidi, O. Gasser, G. Smaragdakis, and A. Feldmann, "Fifteen Months in the Life of a Honeyfarm," in *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, Association for Computing Machinery, 2023, pp. 282–296. doi: 10.1145/3618257.3624826.

[31] L. Baresi, G. Quattrocchi, and N. Rasi, "A qualitative and quantitative analysis of container engines," *J. Syst. Softw.*, vol. 210, p. 111965, Apr. 2024, doi: 10.1016/j.jss.2024.111965.

[32] V. S. D. Priya and S. S. Chakkaravarthy, "Containerized cloud-based honeypot deception for tracking attackers," *Sci. Rep.*, vol. 13, no. 1, p. 1437, Jan. 2023, doi: 10.1038/s41598-023-28613-0.

[33] M. Oosterhof, *cowrie/Cowrie*. (Dec. 14, 2024). Python. Cowrie. Accessed: Dec. 15, 2024. [Online]. Available: https://github.com/cowrie/cowrie

[34] A. S. Bozkir, E. Tahillioglu, M. Aydos, and I. Kara, "Catch them alive: A malware detection approach through memory forensics, manifold learning and computer vision," *Comput. Secur.*, vol. 103, p. 102166, Apr. 2021, doi: 10.1016/j.cose.2020.102166.

[35] Microsoft Corporation, "[MS-TDS]: Tabular Data Stream Protocol," Microsoft Corporation, v20241119, Nov. 2024. Accessed: Dec. 26, 2024. [Online]. Available: https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-tds/b46a581a-39de-4745-b076-ec4dbb7d13ec

[36] T. Favale, D. Giordano, I. Drago, and M. Mellia, "What Scanners do at L7? Exploring Horizontal Honeypots for Security Monitoring," in *2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, Jun. 2022, pp. 307–313. doi: 10.1109/EuroSPW55150.2022.00037.

[37] T. Sochor, M. Zuzcak, and P. Bujok, "Analysis of attackers against windows emulating honeypots in various types of networks and regions," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2016, pp. 863–868. doi: 10.1109/ICUFN.2016.7537159.