e-ISSN 2549-7472
Edumatic: Jurnal Pendidikan Informatika
Volume 10, Issue 3, **530-539.**
https://e-journal.hamzanwadi.ac.id/index.php/edumatic
DOI: 10.29408/edumatic.v10i3.36958

Session-Aware Analysis of SSH Authentication Telemetry Fidelity: Session-Boundary Credential Repetitions in Cowrie Honeypots

Dyah Ayu Pramesty Regita Putri [1,*] **, Moh. Ali Romli** [1]

[1] Universitas Teknologi Yogyakarta, Indonesia
[*] Corresponding author: Dyah Ayu Pramesty Regita Putri, Universitas Teknologi Yogyakarta, Indonesia
✉ regitaputri705@gmail.com

Copyright: © 2026 by the authors

Received: 01 October 2026 | Revised: 10 October 2026 | Accepted: 30 October 2026 | Published: 27 November 2026

Abstract
Honeypots are increasingly used not only to observe SSH brute-force attacks but also as measurement instruments whose logs become security metrics. Most prior studies, however, assume that recorded authentication events directly represent attacker activity, leaving the validity of honeypot telemetry and the influence of session context on derived metrics largely unexamined. This study evaluated the reliability, fidelity, and interpretability of Cowrie authentication telemetry by comparing the attack client's planned credential sequence with the events the honeypot recorded. We conducted a controlled experiment in a virtual environment in which Hydra executed six dictionary-attack configurations against Cowrie, and we analyzed the resulting logs by session. No events were lost, yet configurations with longer password lists produced about 3.2 percent more failed-login events than planned. Every surplus event was a credential pair that recurred after the honeypot terminated a session at its authentication-attempt limit, which doubled the apparent frequency of one password, whereas shorter password lists produced no repetition. The findings show that pipeline consistency does not guarantee measurement fidelity and that raw event counts can bias credential profiling. The study contributes a session-aware comparison approach, although the findings apply to the tested conditions and require validation in other environments.

Keywords: authentication telemetry; cowrie honeypot; measurement fidelity; session boundary; ssh brute-force

To cite this article: Putri, D. A. P. R., & Moh. Ali Romli. (2025). Session-Aware Analysis of SSH Authentication Telemetry Fidelity: Session-Boundary Credential Repetitions in Cowrie Honeypots. *Edumatic: Jurnal Pendidikan Informatika*, *10*(3), 530–539. https://doi.org/10.29408/edumatic.v10i3.36958

INTRODUCTION

The rapid expansion of internet-connected services has increased exposure to automated credential-guessing activity, including attacks targeting the Secure Shell (SSH) protocol (Agghey et al., 2021; Banoth et al., 2026; Lanka et al., 2024; Oliveira et al., 2026). Javadpour et al. (2024) and Morić et al. (2025) Describe low-interaction honeypots as widely deployed cyber deception sensors for observing such activity, whose telemetry is increasingly ingested into automated pipelines, dashboards, and security information and event management (SIEM) platforms that turn raw events into attack volumes, credential rankings, and threat indicators (Adnyana et al., 2024; González-Granadillo et al., 2021; Kelly et al., 2021). In this role, a honeypot is not merely an observation tool but a measurement instrument. The validity of a

Edumatic: Jurnal Pendidikan Informatika is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License

derived metric depends on the chain linking attacker activity, the honeypot logging mechanism, the session context of events, and the transformation of events into metrics.

This study builds on measurement fidelity, defined here as the degree to which recorded telemetry represents the external activity it is intended to measure. Fidelity differs from consistency, the agreement among layers that store and display the same events, and from reliability, the reproducibility of measurements under identical conditions. Telemetry can be consistent and reliable yet still misrepresent the attack. He et al. (2021) Note that logs are produced by system-specific mechanisms and therefore embed artifacts of their generating system, a concern recurring across log-analysis research (Landauer et al., 2023). For SSH, the generating mechanism is stateful. A session begins with a TCP connection and key exchange, proceeds through a user-authentication phase in which the client submits successive authentication requests, and ends when the connection closes. Ylonen and Lonvick (2006) Specify that an SSH server should limit failed attempts per session, recommend a limit of 20, and disconnect once it is exceeded. Cowrie follows this behavior in its default configuration. A long credential list is therefore split across several sessions, and each session transition is a point at which the client and server must re-synchronize the credential sequence. Consequently, the session identifier is essential: the same credential recorded twice may reflect two attacker decisions or a single decision recorded across a session boundary.

Previous honeypot research can be synthesized into three streams. The first treats honeypots as attack-observation tools and has characterized attack patterns, actors, and behaviors across deployment types (Priya & Chakkaravarthy, 2023; Sethuraman et al., 2023; Valeros et al., 2026; Var Naseri et al., 2026; Yu et al., 2024). The second focuses on telemetry analysis, processing honeypot logs into security indicators through dashboards, classifiers, and AI-assisted pipelines (Lanka et al., 2024; Prasetyo et al., 2025) or evaluating sensor performance (Pramana et al., 2026; Putri & Tahir, 2026). Both streams implicitly assume that event logs directly represent attacker activity: failed-login counts are read as attempt counts and credential frequencies as attacker preferences. The third, smaller stream concerns measurement quality. Event-correlation research shows that a security event acquires meaning largely in relation to its temporal and contextual neighbors (Kotenko et al., 2022) Log-dataset studies emphasize reproducible ground truth for evaluating detectors (Landauer et al., 2023) Network-dataset studies document recording and labeling errors (Guerra et al., 2022), and Li et al. (2026) highlight state and runtime context in SSH honeypot design. These efforts, however, address detector evaluation or honeypot realism rather than whether a honeypot authentication log reproduces the sequence an attack client actually sent.

To the best of the authors' knowledge, no study in the past five years has directly compared an attack client's planned credential sequence with session-level, honeypot-recorded authentication events. This gap matters because any misalignment propagates directly into analysis: it inflates estimated credential frequencies, distorts the inferred structure of brute-force patterns, misrepresents attacker behavior, and lowers the validity of threat intelligence and of frequency-based or machine-learning detectors trained on such data. As Apruzzese et al. (2023) caution, such data-quality problems propagate into downstream analysis. In short, previous research has used telemetry to determine what attacks occur, whereas whether telemetry accurately represents how attacks occur remains largely untested.

This study therefore aimed to evaluate the fidelity of authentication telemetry in the Cowrie honeypot by analyzing the relationship between attack-client credential sequences and server-recorded events while considering session context. It addressed three research questions. RQ1 asks whether Cowrie telemetry corresponds one-to-one with the Hydra credential sequence and, if not, whether the discrepancy is associated with SSH session transitions. RQ2 asks how the discrepancy varies with username and password list dimensions. RQ3 asks how session-boundary repetitions affect credential-frequency interpretation. The novelty lies in

shifting the evaluation of honeypot telemetry from what it records to how faithfully it records. The study offers three contributions. Theoretically, it frames honeypot telemetry as a measurement process shaped by session state. Conceptually, it distinguishes fidelity from consistency, reliability, and accuracy. Empirically, it identifies the session-boundary repetition phenomenon in Cowrie telemetry and introduces a session-aware procedure for comparing planned attack sequences with recorded events.

METHOD

This study employed a controlled experimental design, the only setting in which the attacker's intended credential sequence is known exactly and isolated from confounders such as concurrent attackers, unknown tools, and variable latency. The controlled variables were the number of parallel Hydra tasks (one), the wordlist content and order, the network endpoints, and the initial state of each run. The independent variables were the username and password list sizes, and the dependent variables were the recorded failed-login events, the discrepancy, and the session-boundary repetitions.

Two Oracle VirtualBox virtual machines were connected through bridged adapters on a private local network. The attacking node (Kali Linux Rolling 2025.2, 4 vCPU, 8 GB RAM) ran Hydra 9.5, identified by Cowrie as an SSH-2.0-libssh_0.11.2 client. The target node (Ubuntu 24.04.3 LTS, 1 vCPU, 4 GB RAM) ran Cowrie 2.9.19.dev24 (commit 3a66dd2d3) on Twisted 26.4.0 with default authentication settings on TCP port 2222. A Node.js 18.19.1 log watcher ran in the target VM, and the Express.js backend (Node.js 24.12.0) and PostgreSQL 18.3 ran on the Windows host. Hydra was invoked as *hydra -L wordlist_username_Ex.txt -P wordlist_password_Ex.txt -t 1 -f <target> ssh -s 2222*, testing each username against the complete password list in order. The authors created both wordlists, and Ubuntu ranked 21st on the 31-entry password list.

Hydra output represented the planned attack sequence, and the Cowrie JSON log represented the observed telemetry, with neither treated as absolute ground truth. The Cowrie text log (cowrie.log) verified server-side session terminations, and an E0 replication with Hydra's verbose option (-V) provided an attempt-level client record. A custom Node.js agent extracted failed-login events (timestamp, session_id, source IP address, username, and password) from the JSON log and sent them through an Express.js API to PostgreSQL. This pipeline only acquired data and performed cross-layer consistency checks.

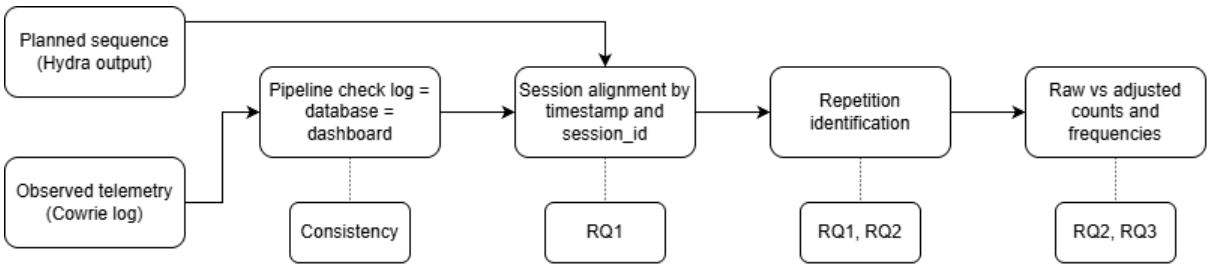

Figure 1. Research procedure linking the data sources, analytical steps, and research questions

Before each run, we archived log files, cleared database records, and restarted Cowrie so each run began from the same state. We defined six configurations (Table 1). E1 to E3 reduced only the usernames to test whether the discrepancy scales with the number of per-username sequences, and E4 and E5 reduced only the passwords to test whether it depends on sequence length. Because Hydra tested all passwords for one username consecutively, the password-list length determined whether a sequence could exceed the recommended per-session limit of 20 attempts (Ylonen & Lonvick, 2006). Lists of 5 and 10 fall below it, whereas 31 exceeds it.

Table 1. Experimental configurations used in E0 through E5

Experiments	Usernames	Passwords	Combinations
E0	55	31	1,705
E1	5	31	155
E2	10	31	310
E3	20	31	620
E4	55	5	275
E5	55	10	550

Four Cowrie event types were analyzed: session connection, login failure, login success, and session closure. We operationally defined an SSH session as the lifecycle from a connection event to a closure event, identified by its unique session_id. We calculated the discrepancy D between recorded and planned events using equation (1).

$$D = C - H \tag{1}$$

where C is the number of Cowrie failed-login events, and H is the number of Hydra-scheduled credential combinations. We calculated the relative discrepancy using equation (2).

$$D\% = \left(\frac{D}{H}\right) \times 100 \tag{2}$$

We treated the discrepancy as a representational difference rather than measurement error, and the matching procedure below determined whether each surplus event was a pipeline artifact, lost or duplicated data, or a session-induced duplication. We compared the session-adjusted count, C minus the identified repetitions R, with H. To identify repetitions, we sorted failed-login events by timestamp, grouped them by session_id, and ordered the sessions by connection time. For each pair of consecutive sessions, we compared the last failed-login event of the earlier session with the first of the later one, and flagged the second occurrence as a session-boundary repetition when both shared the same username, password, and source address. Flagged events were labeled rather than deleted, preserving the raw telemetry. We then validated the session-adjusted sequence against the Hydra plan and computed repetition counts, repeated pairs, and credential-frequency distributions before and after adjustment, following the procedure in Figure 1.

Cross-layer counts tested pipeline consistency. Comparing C with H and the repetition analysis addressed RQ1; the cross-configuration comparison addressed RQ2; and the frequency analysis addressed RQ3. No human participants or production systems were involved, as all SSH activity was generated between the two experimental hosts. The wordlists, Cowrie logs, and analysis scripts are available from the corresponding author upon reasonable request.

RESULTS AND DISCUSSION

Results

In E0, the cowrie.json log, the PostgreSQL database, and the dashboard contained the same 1,760 failed-login events, showing that no events were lost or duplicated during acquisition. Hydra, however, scheduled 1,705 combinations, so the telemetry did not match the planned sequence one-to-one (RQ1). Chronological alignment identified 55 surplus events, all of which were session-boundary repetitions.

In every case, the same pair appeared as the last failed-login event of one session and as the first event of the next (Table 2). E0 produced 111 sessions: one initial connection without authentication attempts and 110 sessions with failed logins, two per username, containing 21 and 11 events, respectively. Of the 109 transitions between consecutive login sessions, 55 repeated the transitions within a username's sequence, while the 54 transitions between usernames contained none (Figure 2c). Each repeated pair combined a username with ubuntu, the 21st password. Each username was affected exactly once. In every repetition, the Cowrie text log recorded a disconnection (code 14, too many bad auths) 1.00 to 1.02 s after the 21st failed attempt. The session closed 2 to 9 ms later, and the same pair was submitted as the first attempt of the next session 0.05 to 0.08 s after that. No other session ended this way. Applying the same procedure to all six logs reproduced these counts (Figure 2c). The E0 replication reproduced the original run exactly (1,760 events in identical order, 55 repetitions, 55 disconnections). In it, Hydra logged 1,705 attempts, Ubuntu 55 times, and no re-attempts, and its sequence matched the adjusted Cowrie sequence in order, except that Cowrie recorded the 31 attempts Hydra logged with an empty username as freykirana, the local account name on the attacking host.

Table 2. Generalized session-transition sequence observed for repeated credential pairs

Order	Event	Session	Interpretation
1	session.connect	Session A	New SSH session
2	login.failed	Session A	Sequential credential attempts
3	login.failed	Session A	21st failed event (username + ubuntu)
4	session.closed	Session A	Session terminates
5	session.connect	Session B	Consecutive SSH session opens
6	login.failed	Session B	Same pair recorded again (repetition)
7	login.failed	Session B	Planned sequence continues

The discrepancy appeared only in E0 to E3, all of which used the 31-entry password list, and its relative size was constant at 3.23%, equivalent to one surplus event per 31 planned attempts (Table 3). Reducing the number of usernames changed the absolute discrepancy proportionally (5, 10, 20, and 55 events) but not the rate. E4 and E5, with 5 and 10 passwords, showed no discrepancy (RQ2).

Table 3. Discrepancy and session-boundary repetitions across the six configurations

Exp.	Passwords (sessions) per username	Hydra planned (H)	Cowrie raw (C)	Discrepancy D (D%)	Repetitions R (per username)
E0	31 (2)	1,705	1,760	55 (3.23%)	55 (1)
E1	31 (2)	155	160	5 (3.23%)	5 (1)
E2	31 (2)	310	320	10 (3.23%)	10 (1)
E3	31 (2)	620	640	20 (3.23%)	20 (1)
E4	5 (1)	275	275	0 (0.00%)	0 (0)
E5	10 (1)	550	550	0 (0.00%)	0 (0)

The discrepancy was associated with per-username sequence length rather than total combinations: E4 (275 combinations) and E5 (550) were larger than E1 (155) and E2 (310), yet showed no repetition. The pattern coincided with the recommended per-session limit of 20 failed attempts, beyond which the server disconnects: 31-password sequences were split into

sessions of 21 and 11 attempts, whereas 5- and 10-password sequences were completed in one session per username (Figure 2a–b).

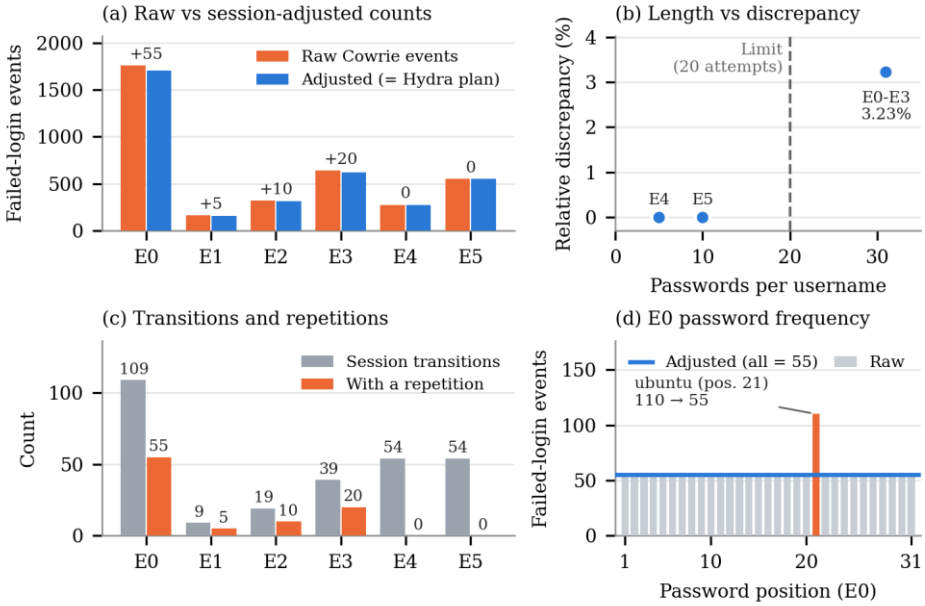

Figure 2. (a) Raw and session-adjusted failed-login counts; (b) password-sequence length versus relative discrepancy; (c) session transitions and transitions containing a repetition; (d) raw and session-adjusted password frequency in E0

In the raw telemetry of E0 to E3, every username appeared 32 times instead of 31, so username rankings were unchanged. Password frequencies, by contrast, were distorted (RQ3): in E0, ubuntu appeared 110 times, whereas each of the other 30 passwords appeared 55 times. The password ubuntu thus appeared to be the most frequently attempted, at twice the frequency of any other, although the attack client used all passwords equally. After session-aware adjustment, ubuntu returned to 55 occurrences, and the distribution became uniform, matching the planned sequence (Table 4; Figure 2d). The same proportional pattern occurred in E1 to E3, while E4 and E5 were unchanged. These patterns were observed under the tested configurations and are not assumed to hold in other environments.

Table 4. Raw and session-adjusted credential frequencies

Exp.	Count per username (raw / adjusted)	Ubuntu (raw / adjusted)	Each other password	Password ranking (raw)
E0	32 / 31	110 / 55	55	Ubuntu first
E1	32 / 31	10 / 5	5	Ubuntu first
E2	32 / 31	20 / 10	10	Ubuntu first
E3	32 / 31	40 / 20	20	Ubuntu first
E4	5 / 5	No change	55	Unchanged
E5	10 / 10	No change	55	Unchanged

Discussion

The findings show that Cowrie authentication telemetry did not fully match the planned Hydra credential sequence, despite consistent data across the logging, database, and dashboard layers. In configurations E0–E3, which used 31 passwords per username, recorded failed-login

events exceeded the planned combinations by 3.23%. By contrast, E4 and E5, which used shorter password lists, showed no discrepancy. All surplus events resulted from repeated credential pairs across session boundaries. These repetitions also skewed password-frequency distributions, making ubuntu appear twice as frequently as other passwords, even though the planned attack sequence used each password equally.

These findings distinguish pipeline consistency from measurement fidelity: agreement among data collection layers does not necessarily mean that recorded events accurately represent the intended activity. He et al. (2021) emphasized that logs reflect the mechanisms of the systems that generate them. In this study, session context was essential to identifying discrepancies between planned credential sequences and recorded authentication events. Session identifiers and event timestamps should therefore be treated as integral to interpreting authentication telemetry, not merely as supplementary logging attributes.

Previous honeypot research has used authentication logs to characterize attacks, develop security indicators, and support data-driven detection (Lanka et al., 2024; Prasetyo et al., 2025). These findings complement this work by showing that recorded event frequencies may not always match the planned sequence of credential attempts. This concern aligns with research on security-event correlation and dataset quality, which highlights the importance of contextual interpretation and reliable data for downstream analysis (Guerra et al., 2022; Kotenko et al., 2022). While studies have focused primarily on attack characterization or detection, this study examines how closely a controlled attack sequence matches honeypot-recorded telemetry.

The repeated credential pairs appeared at session boundaries, after Cowrie disconnected sessions for exceeding the authentication-attempt limit. RFC 4252 recommends limiting failed authentication attempts within a session and disconnecting once that limit is exceeded (Ylonen & Lonvick, 2006). This protocol context plausibly explains the observed pattern. However, because we did not trace the underlying Hydra or SSH client-library routine, the precise mechanism behind the repeated submissions remains unconfirmed. In addition, replacing an empty username with the attacking host's local account name suggests a potential discrepancy between client-side attributes and server-side telemetry that warrants further investigation.

This study contributes a session-aware method for evaluating SSH honeypot telemetry fidelity by comparing planned credential sequences with recorded authentication events. The results provide controlled empirical evidence that session transitions can systematically create discrepancies and distort estimates of credential frequency. The contribution is to show that we must distinguish a logging pipeline's consistency from the fidelity of its measurements, rather than assuming every recorded event represents a distinct planned credential attempt.

Honeypot analytics should preserve raw telemetry and report any session-adjusted metrics separately. Retain session identifiers, timestamps, and credential attributes to support transparent validation and reproducibility. Suspected repetitions should not be deleted automatically, since identical credentials may reflect genuine repeated activity. Instead, any adjustments should follow explicit, documented criteria. This approach may improve the interpretability of dashboards, SIEM outputs, and frequency-based security analytics (González-Granadillo et al., 2021), although its effect on operational detection performance requires further evaluation.

The findings are limited to one attack tool, a controlled virtual environment, fixed wordlists, and a single parallel task. The authors did not vary the authentication threshold or password-list order independently; they replicated only E0, and they did not trace the client-side mechanism behind the repetition. Future research should compare multiple SSH clients and honeypots, vary session limits and credential ordering independently, and use protocol-level tracing to investigate the mechanism. Further evaluation should also establish whether session-boundary repetitions occur in real-world telemetry and whether correcting them changes threat-intelligence outputs or machine-learning detection performance.

CONCLUSION

This study evaluated whether Cowrie authentication telemetry faithfully reflects the credential sequence of an SSH dictionary attack and found that it does not always: the discrepancy was not a pipeline error, but evidence that session context shapes how attack activity is represented and can bias credential-frequency profiles. The study contributes the perspective that honeypot telemetry is a measurement process requiring session-aware analysis, a procedure for comparing planned attack sequences with recorded events, and a recommendation to keep raw telemetry separate from session-adjusted metrics. Unlike much previous honeypot research, which uses telemetry to describe the attacks that occur, this work evaluates how faithfully telemetry represents them. Because the experiment used one tool with specific configurations in a controlled environment, its generalizability is limited. Future work should trace the client routine, extend the analysis to other honeypots, tools, and network conditions, and develop a benchmark for evaluating honeypot telemetry fidelity. Overall, the study suggests that trustworthy honeypot analytics require not only reliable data collection but also session-aware interpretation of what the collected events represent.

REFERENCES

Adnyana, I. G., Dirgayusari, A. M., & Atmaja, K. J. (2024). Data visualization for building a cyber attack monitoring dashboard based on honeypot. *Sinkron: Jurnal dan Penelitian Teknik Informatika, 8*(4), 2510–2518. https://doi.org/10.33395/sinkron.v8i4.14144

Agghey, A. Z., Mwinuka, L. J., Pandhare, S. M., Dida, M. A., & Ndibwile, J. D. (2021). Detection of username enumeration attack on SSH protocol: Machine learning approach. *Symmetry, 13*(11), 2192. https://doi.org/10.3390/sym13112192

Apruzzese, G., Laskov, P., Montes de Oca, E., Mallouli, W., Brdalo Rapa, L., Grammatopoulos, A. V., & Di Franco, F. (2023). The role of machine learning in cybersecurity. *Digital Threats: Research and Practice, 4*(1), 1–38. https://doi.org/10.1145/3545574

Banoth, R., Addula, S. R., Godishala, A. K., Sannapu, R., Sajja, G. S., Kumar, D., Kasula, V. K., & Tumma, C. (2026). Evolving IoT botnet threats and practical honeypot observation: A summary review and experimental study. *Journal of Cybersecurity and Privacy, 6*(3), 82. https://doi.org/10.3390/jcp6030082

González-Granadillo, G., González-Zarzosa, S., & Diaz, R. (2021). Security information and event management (SIEM): Analysis, trends, and usage in critical infrastructures. *Sensors, 21*(14), 4759. https://doi.org/10.3390/s21144759

Guerra, J. L., Catania, C. A., & Veas, E. E. (2022). Datasets are not enough: Challenges in labeling network traffic. *Computers & Security, 120*, 102810. https://doi.org/10.1016/j.cose.2022.102810

He, S., He, P., Chen, Z., Yang, T., Su, Y., & Lyu, M. R. (2021). A survey on automated log analysis for reliability engineering. *ACM Computing Surveys, 54*(6), 1–37. https://doi.org/10.1145/3460345

Javadpour, A., Ja'fari, F., Taleb, T., Shojafar, M., & Benzaïd, C. (2024). A comprehensive survey on cyber deception techniques to improve honeypot performance. *Computers & Security, 140*, 103792. https://doi.org/10.1016/j.cose.2024.103792

Kelly, C., Pitropakis, N., Mylonas, A., McKeown, S., & Buchanan, W. J. (2021). A comparative analysis of honeypots on different cloud platforms. *Sensors, 21*(7), 2433. https://doi.org/10.3390/s21072433

Kotenko, I., Gaifulina, D., & Zelichenok, I. (2022). Systematic literature review of security event correlation methods. *IEEE Access, 10*, 43387–43420. https://doi.org/10.1109/ACCESS.2022.3168976

Landauer, M., Onder, S., Skopik, F., & Wurzenberger, M. (2023). Deep learning for anomaly detection in log data: A survey. *Machine Learning with Applications, 12*, 100470. https://doi.org/10.1016/j.mlwa.2023.100470

Landauer, M., Skopik, F., Frank, M., Hotwagner, W., Wurzenberger, M., & Rauber, A. (2023). Maintainable log datasets for evaluation of intrusion detection systems. *IEEE Transactions on Dependable and Secure Computing, 20*(4), 3466–3482. https://doi.org/10.1109/TDSC.2022.3201582

Lanka, P., Gupta, K., & Varol, C. (2024). Intelligent threat detection—AI-driven analysis of honeypot data to counter cyber threats. *Electronics, 13*(13), 2465. https://doi.org/10.3390/electronics13132465

Li, X., Li, N., Li, Z., Yan, L., Ma, D., Cao, H., Wang, X., & Liu, Y. (2026). LLM-SSHH: An LLM-powered SSH honeypot framework via state snapshot. *Applied System Innovation, 9*(5), 101. https://doi.org/10.3390/asi9050101

Morić, Z., Dakić, V., & Regvart, D. (2025). Advancing cybersecurity with honeypots and deception strategies. *Informatics, 12*(1), 14. https://doi.org/10.3390/informatics12010014

Oliveira, R., Gomes, T., Pinho, D., & Almeida, F. (2026). An integrated honeypot and LLM-based framework for near real-time detection and behavioral analysis of malicious activities. *Journal of Cybersecurity and Privacy, 6*(5), 147. https://doi.org/10.3390/jcp6050147

Pramana, Y. T., Huwae, R. B., & Jatmika, A. H. (2026). Perbandingan sumber daya honeypot berbasis Cowrie dan OpenCanary terhadap serangan SSH brute-force dan port-scanning. *Jurnal Teknologi Informasi, Komputer, dan Aplikasinya (JTIKA), 8*(1), 54–65. https://doi.org/10.29303/jtika.v8i1.541

Prasetyo, A. A., Herianto, Yahya, & Syamsiyah, N. (2025). Detection of SSH brute force attacks using Naïve Bayes classification on Cowrie honeypot logs in a virtualized environment. *Journal Technology Information and Data Analytic, 2*(1), 88–91. https://doi.org/10.70491/tifda.v2i1.88

Priya, V. D., & Chakkaravarthy, S. S. (2023). Containerized cloud-based honeypot deception for tracking attackers. *Scientific Reports*, *13*(1), 1437. https://doi.org/10.1038/s41598-023-28613-0

Putri, A. M., & Tahir, M. (2026). Performance evaluation of a Cowrie-based honeypot integrated with a Suricata intrusion prevention system in a school wireless network. *Journal of Embedded Systems, Security and Intelligent Systems, 7*(1), 49–65. https://doi.org/10.59562/jessi.v7i1.2605

Sethuraman, S. C., Jadapalli, T. G., Sudhakaran, D. P. V., & Mohanty, S. P. (2023). Flow based containerized honeypot approach for network traffic analysis: An empirical study. *Computer Science Review, 50*, 100600. https://doi.org/10.1016/j.cosrev.2023.100600

Valeros, V., Sladić, M., & Garcia, S. (2026). *Ghost without shell: Measuring non-interactive SSH attacks on honeypots*. arXiv. https://doi.org/10.48550/arXiv.2606.28006

Var Naseri, M., Welch, I., Haseeb, J., & Mansoori, M. (2026). Q-Cowrie: An adaptive honeypot to analyse attackers' behaviour. *International Journal of Information Security, 25*(2), 60. https://doi.org/10.1007/s10207-026-01221-5

Wahyudi, F., & Utomo, L. T. (2021). Perancangan security network intrusion prevention system pada PDTI Universitas Islam Raden Rahmat Malang. *Edumatic: Jurnal Pendidikan Informatika, 5*(1), 60–69. https://doi.org/10.29408/edumatic.v5i1.3278

Ylonen, T., & Lonvick, C. (2006). The Secure Shell (SSH) authentication protocol. Retrieved October 11, 2026, from RFC Editor website: https://www.rfc-editor.org/info/rfc4252/

Yu, T., Xin, Y., & Zhang, C. (2024). HoneyFactory: Container-based comprehensive cyber deception honeynet architecture. *Electronics, 13*(2), 361. https://doi.org/10.3390/electronics13020361