

Automated Privacy Policy Compliance Verification Using Dynamic Network Traffic Analysis

Vimlesh Mourya*, Maya Nair**

*Department of Computer Science, SIES College of Arts, Science & Commerce (Autonomous), Mumbai, India
Email: Vimleshfmcs2526111@asc.sies.edu.in

**Department of Computer Science, SIES College of Arts, Science & Commerce (Autonomous), Mumbai, India
Email: mayan@sies.edu.in

Abstract:

A privacy policy describes the information that an application may collect, but it cannot show what the application sends during a particular session. We built PrivacyComplianceVerifier to compare evidence from HTTP Archive (HAR) traffic with statements found in the application's privacy policy. The framework keeps repeated headers, cookies and query parameters instead of silently overwriting them. It also examines request and response bodies in formats such as JSON, multipart data, Base64, compressed text, raw Protocol Buffers and gRPC frames. Sensitive values are redacted before the evidence inventory is written. The comparison uses only the information types that can be supported by the captured traffic and the reviewed policy. URL exposure and authentication artifacts are reported separately as technical risks. We evaluated the framework using captures from 30 applications. In total, it processed 34,170 requests and detected 84,692 sensitive-data occurrences, representing 5,354 unique artifacts. Thirteen applications had at least one comparable type that was missing from the completed policy representation. Ten had all comparable observed types represented, while seven could not be fully assessed because their policy evidence was incomplete. These results show that network traffic can support a clear policy comparison, but they also reflect the limits of the tested workflow, encryption, policy wording and domain ownership.

Keywords — privacy policy, dynamic analysis, network traffic, HAR, sensitive data, policy verification, Android applications.

I. INTRODUCTION

Mobile and web applications handle many kinds of user information, including names, phone numbers, locations, device identifiers and account details. Users usually learn about this through a privacy policy. A policy, however, is only a written description. Network traffic shows what happened while the application was being used. These two sources may not match because the policy is incomplete, the capture covers only part of the application, or the implementation has changed. Comparing them manually also becomes difficult when a single capture contains thousands of requests.

Several earlier systems compare application data flows with privacy-policy text. PoliCheck checks

consistency while considering the destination entity [1], and PurPliance also studies the purposes stated in a policy [2]. OVRseen applies traffic and policy analysis to virtual-reality applications [3]. Other researchers have used program analysis, knowledge graphs and automated policy checks to identify missing disclosures [4]–[6]. This work made one issue especially clear for our project: detecting a value is not enough. The framework must also keep track of where it was found and what can reasonably be concluded from it.

In this study, we developed an explainable pipeline for analysing HAR captures. The system keeps duplicate traffic fields, detects privacy-related artifacts in several body formats, classifies the destination and stores only redacted evidence. It then

compares supported information types with policy statements that have been reviewed. The result is not presented as a legal decision. It simply reports whether an observed type is disclosed, absent from the available policy representation, or undecidable because the evidence is incomplete.

The main contributions of this work are listed below:

- A HAR extraction pipeline that preserves duplicate requests fields, including headers, cookies, query parameters and body parameters.
- A conservative detector for structured text, multipart content, encoded bodies, raw Protocol Buffers and gRPC, with redacted values in the generated reports.
- An evidence inventory and policy comparison that keeps disclosure results separate from technical artifacts and URL-risk signals.
- An evaluation of 30 applications covering 34,170 requests and 84,692 detected sensitive occurrences.

II. RELATED WORK

A. Traffic and Policy Consistency

PoliCheck compares observed data flows with privacy policies and includes the destination entity in the comparison [1]. This is important because the same data type may have a different meaning when it is sent to another recipient. PurPliance adds the purposes stated in policies to its consistency analysis [2]. OVRseen studies network traffic and policies in virtual-reality applications [3]. GNChecker examines whether Android transmissions are properly disclosed [4], while mopri combines application analysis with privacy-policy checks [5].

These systems influenced our design, but we use a narrower comparison rule. The framework does not guess the purpose of a request from network traffic. It compares the observed information type and, when the evidence supports them, the action, direction and recipient information. We chose this rule because a detailed-looking result can still be misleading when it is based on assumptions.

B. Automated Privacy-Policy Analysis

Privacy policies are often long and written in broad legal language. Amaral et al. studied whether

policy completeness could be checked automatically [7]. PoliGraph represents policy statements as a knowledge graph [6], and Rodríguez Torrado et al. explored the use of large language models for policy analysis at scale [8]. The mapping study by del Alamo et al. groups this field into tasks such as classification, information extraction, readability and compliance checking [9]. Deep-learning, rule-based and machine-learning methods have also been applied to GDPR-related policy checks [10]–[12]. Viopolicy-Detector focuses on suspected GDPR violations on websites [23], while Farhadi et al. examine compliance checking for cloud and fog applications [24].

For this project, automatically extracted policy text is not treated as complete ground truth. Each policy statement keeps its source and a coverage status. If the available policy representation is only partial, a missing practice is reported as an incomplete check rather than a definite mismatch. Otherwise, a limitation of the parser could incorrectly be shown as a failure by the application.

C. Network-Traffic Analysis

TraceDroid analyses Android network traffic for privacy leakage [13], while Li et al. use self-labelling and learning to detect privacy-related traffic [14]. Surveys of encrypted traffic analysis describe both the usefulness of network features and the difficulty caused by encryption [15], [16]. METAsen combines traffic and policy analysis for metaverse systems [17]. Work based on contextual integrity asks whether a data flow is appropriate in its stated context [18]. Related studies also examine machine-learning classification [25], traffic-privacy optimisation [26], policy-based protection [27] and controls for edge data streams [28]. Together, these studies support dynamic analysis, but they also show why one capture cannot describe every behaviour of an application.

TABLE I. COMPARISON WITH SELECTED RELATED SYSTEMS

Approach	Dynamic traffic	Policy comparison	Recipient-aware	Evidence output
PoliCheck [1]	Yes	Yes	Yes	Flow/policy consistency
PurPliance [2]	App data use	Yes	Limited	Purpose consistency
OVRseen [3]	Yes	Yes	Yes	VR-labelled flows
GNChecker [4]	Yes	Yes	Notification focus	Undeclared transmission
This work	HAR requests + responses	Yes	Conservative	Redacted occurrence evidence

III. PROPOSED FRAMEWORK

PrivacyComplianceVerifier has six main stages, shown in Fig. 1. First, HTTP Toolkit captures decrypted application traffic and exports it as a HAR file. The extractor converts each HAR entry into the framework's request model. The traffic classifier then separates application-owned, third-party, Android-system and unknown destinations. After that, the detector checks supported locations in both requests and responses. The findings are normalised and stored in the privacy inventory. In the last stage, the comparator checks the observed information types against policy evidence and reports disclosure results separately from risk signals.

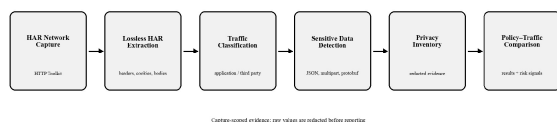


Fig. 1. PrivacyComplianceVerifier processing pipeline.

A. Lossless HAR Extraction

Using only a dictionary can remove repeated headers, cookies or query keys because a later value overwrites an earlier one. To avoid this, the extractor keeps ordered pairs as well as the older dictionary fields used by existing code. It reads headers, cookie arrays, raw cookie headers, URL parameters, request post data, HTTP Toolkit content fields and response content. MIME types, declared character sets and HAR encoding information are also preserved. When the same body parameter appears in both text and a HAR parameter array, the two representations are reconciled so that it is not counted twice.

B. Sensitive-Data Detection

The detector checks known field names and validated value patterns. It can identify items such as

email addresses, phone numbers, names, addresses, coordinates, device identifiers, IP addresses, user identifiers, authorization tokens and session cookies. Nested JSON is inspected with limits on depth and the number of key-value pairs. JSON stored inside strings, query values or cookies is parsed only when the value starts with an object or array marker and passes the safety checks. For array elements, the nearest nonnumeric parent key is also checked as an exact candidate.

Request and response bodies may contain plain text, Base64, Base64url, compressed content, multipart data, raw protobuf or gRPC-framed protobuf. Multipart sections are split only at valid line boundaries, and arbitrary binary file content is not searched as text. For raw protobuf, the scanner reads length-delimited strings but does not guess what numeric fields mean. Large text bodies are checked in overlapping chunks with a fixed limit. These restrictions were kept deliberately to reduce false positives and prevent unusually large or malformed inputs from consuming excessive resources.

C. Evidence and Privacy Inventory

Every normalised finding keeps the artifact type, privacy category, source, direction, key, domain and traffic type. The detected raw value is excluded from the privacy inventory. Separate occurrences are retained because their frequency is useful, but a fingerprint prevents the same value from being counted again when two parser paths reach the same underlying field. Request counts, domains and category lists are also retained for compatibility with the earlier output format.

D. Policy-Traffic Comparison

For comparison, observed evidence is summarised using its category, artifact type, action or direction, traffic-recipient scope, destination, source and risk flags. Policy statements are normalised separately and keep their source and review details. Application-owned traffic is not automatically treated as a policy-defined first party. Recipient scope is used only when the available evidence supports that mapping.

The dashboard uses three direct comparison outcomes. A comparable type is marked Disclosed

when a reviewed policy statement clearly covers it. It is marked Not Disclosed only when the policy representation is complete for that scope and no matching type is present. In all other cases, the framework reports Policy Check Incomplete. Authorization tokens and session cookies are still shown as technical evidence, but they do not create policy gaps. Security artifacts, API credentials and unknown categories are also excluded from automatic mismatch decisions. A personal value or credential found in a URL is reported separately as a risk signal, even if the policy discloses that type of information.

IV. RESEARCH METHODOLOGY

A. Application Selection and Capture

The dataset contains 30 commonly used mobile and web applications from categories such as communication, social media, shopping, food delivery, entertainment, productivity, navigation and employment. It also includes open-source applications. I captured the traffic with HTTP Toolkit while carrying out common actions available in each application, including login, search, profile access, browsing, messaging, media access and location-related functions. The exact steps could not be identical for every application because their features differ, and some traffic could not be decrypted.

The captures are activity-based samples rather than exhaustive tests. Information that did not appear may still be processed during an action I did not perform, by a background task, on another device or through traffic that the capture tool could not read. For this reason, Not Observed means only that the item was absent from the recorded session.

B. Policy Evidence Preparation

For each application, I reviewed the official privacy-policy page and represented the relevant statements in JSON. Existing category fields were retained, while clear statements were converted into canonical information-type practices with their sources. The policy's publication date and the date of manual review were recorded separately. Recipient domains and actions were added only when the policy text stated them. If the extraction was incomplete, the policy was marked partial so that

missing JSON detail would not create a false non-disclosure.

C. Measurements

The evaluation records the number of requests, traffic-ownership classes, sensitive occurrences, unique artifacts, comparable outbound findings, attributable inbound findings, category totals, policy decisions and URL-risk signals. An occurrence count includes repeated network events. A unique-artifact count groups repeated appearances of the same redacted value. Policy results are counted by distinct comparable information type rather than by every occurrence. Without this separation, a token or identifier sent many times could dominate the final result.

V. IMPLEMENTATION

The framework is written in Python. Request extraction, traffic classification, sensitive-data detection, normalisation, inventory generation, policy parsing, practice matching, risk assessment, reporting and dashboard aggregation are handled by separate modules. Keeping these stages separate helped during development because detector changes could be tested without changing the comparator, and the result display could be revised without removing the underlying evidence.

Safety limits are applied throughout the detector. They cover JSON depth, stringified-JSON recursion, decoded size, multipart boundary length, decompressed output, protobuf parsing and free-text scanning. Email matching first limits the number of candidates instead of applying one unrestricted expression to an entire body. Character-set decoding is performed only on bytes, so a Python string that is already decoded is not encoded and decoded again. Several of these checks were added after regression tests revealed failure cases in real captures.

At the time of evaluation, the automated suite contained 244 passing tests. The regression tests include duplicate headers and cookies, repeated query parameters, post-data merging, Base64 and Base64url, UTF-16 bodies, compressed content, large-text chunk boundaries, malformed multipart input, binary form parts, raw protobuf, multi-frame gRPC and repeated occurrences. Inventory evidence,

semantic mapping and policy comparison are also tested. The framework produces JSON reports, aggregate CSV and JSON files, a command-line summary and a local dashboard used for presenting the results.

VI. RESULTS AND DISCUSSION

A. Overall Dataset

The 30 application captures contained 34,170 HAR requests. From these requests, the framework detected 84,692 sensitive occurrences and 5,354 unique artifacts. It classified 44,490 occurrences as comparable outbound evidence and 34,178 as attributable inbound evidence. The difference between the occurrence and artifact totals is large because the same identifiers and account details often appear in many requests.

TABLE II. EMPIRICAL DATASET SUMMARY

Measure	Value
Applications	30
HAR network requests	34,170
Sensitive occurrences	84,692
Unique sensitive artifacts	5,354
Comparable outbound occurrences	44,490
Attributable inbound occurrences	34,178
URL-risk signal records	35
Automated tests	244 passing

B. Observed Privacy Categories

Personal Information was the most frequent category, with 27,073 occurrences. Authentication followed with 22,717. Location produced 17,306 occurrences and Device Identifier produced 14,138. Security, API Credentials and Network Information appeared less often. These numbers show what the detector found in the captures; they do not, on their own, show whether a privacy policy is adequate.

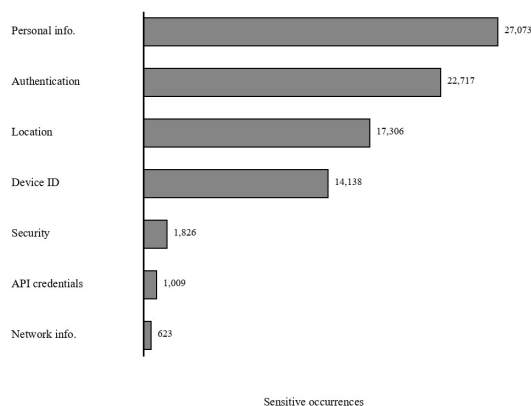


Fig. 2. Sensitive occurrences by privacy category.

C. Policy Comparison Results

The final presentation results contained 70 disclosed information types. Another 24 comparable observed types were not present in policy evidence that had been reviewed as complete for the relevant scope. Ten types remained undecided because the available policy check was incomplete. At application level, 13 applications had at least one potential non-disclosure. Ten had every comparable observed type disclosed, and seven remained incomplete.

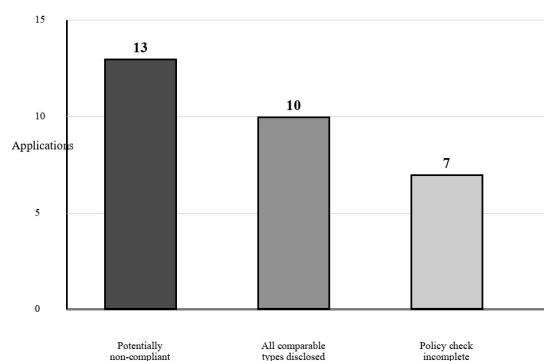


Fig. 3. Application-level policy-traffic result distribution.

In this project, Potentially Non-Compliant is a technical result label, not a legal judgment. The label is used when a comparable information type appears in application-owned outbound traffic but is missing from policy evidence reviewed as complete for that scope. Similarly, a Disclosed result does not prove complete compliance. It only means that the observed information type is represented in the policy evidence used by the framework.

D. URL Risk Signals

Personal data in a URL was found in 19 applications, while credentials in a URL were found in 16, giving 35 application-level signal records. These signals are shown separately from policy disclosure because a policy may mention a data type even when the request location is risky. URLs can appear in browser history, server logs, proxies, analytics tools and screenshots. For that reason, a URL signal requires technical review but is not automatically treated as a policy mismatch. Capturing and storing network traffic can also introduce privacy risks, so the reports contain redacted values instead of the original data [19].

E. Truecaller Case Study

The Truecaller capture illustrates why disclosure and transport risk are different questions. It contained 184 requests, 216 sensitive occurrences and eight unique artifacts. Outbound application traffic included phone numbers, names, email addresses and authorization tokens. The reviewed policy covered Phone, Name and Email, so the framework marked those comparable types as disclosed. The Authorization Token was kept as technical evidence rather than treated as a missing policy item. However, two items of personal data appeared in URLs and produced a medium risk signal. In other words, the policy can disclose the information while the way it appears in traffic still needs review.

TABLE III. TRUECALLER OUTBOUND OBSERVATIONS

Artifact	Count	Source	Dashboard treatment
Authorization Token	107	Header	Technical evidence
Phone	67	Query / gRPC	Disclosed
Name	8	Request body	Disclosed
Email	4	Request body	Disclosed
Personal data in URL	2 signals	Query/URL	Risk review

F. External Dataset Experiment

We also ran small experiments using traffic from AntShield, PARROT 2025 and OVRseen. The aim was to test whether external traffic could be converted into a form accepted by the framework, not to claim benchmark-level accuracy. One converted OVRseen RecRoom sample contained seven requests, ten sensitive occurrences and three unique artifacts. An AntShield Facebook sample contained 99 requests, 234 occurrences and nine unique artifacts. These trials showed that the detector can process external HTTP evidence when it can be represented faithfully as HAR. The original dataset labels were treated as dataset-provided evidence, not as unquestionable ground truth.

VII. LIMITATIONS AND THREATS TO VALIDITY

A. Capture Coverage

The main limitation is the coverage of the recorded workflow. A manual session cannot activate every feature, background task, account state, error case, regional option or advertisement.

Therefore, Not Observed does not mean that the application never processes the information. Repeating the experiment with scripted actions, longer sessions and more than one account would make the captures more reproducible.

B. Encryption and Representation

Some applications use certificate pinning, custom encryption, QUIC or binary formats that hide part of the message meaning. The raw protobuf scanner can extract strings without a schema, but it deliberately does not assign meanings to numeric fields. The framework also does not semantically inspect Brotli or Zstandard content, bodies above its safety limits, images, PDFs or video. These restrictions may cause missed findings. Removing them without suitable controls, however, could increase false positives or create unsafe memory and processing costs. Research on encrypted traffic shows that useful classification is possible, but the actual content remains difficult to interpret without decryption [15], [16].

C. Policy Interpretation

Another limitation is the difference between policy language and traffic terminology. Policies use broad descriptions, while requests contain engineering terms such as field names and headers. Matching the two requires judgment. For example, the framework does not automatically treat Device Information as Device Identifier or Passwords as Authentication. A policy may also describe an item using wording that has not been entered in the project's JSON representation. Such cases are reported as incomplete checks. Previous studies describe the same difficulty when converting natural-language policies into structured data [7]–[9]. Work on privacy-aware auditing [29] and mobile privacy risk [30] also shows that the quality and context of the evidence affect the final result.

D. Traffic Ownership and Generalisation

Traffic ownership was classified using application configuration and known domain rules. Even with these rules, content-delivery networks, shared cloud services, redirects and embedded services can be ambiguous. An application-owned domain is therefore not automatically equivalent to a first party defined in a policy. The 30 selected applications

provide useful case studies, but they cannot represent every application used in India or elsewhere. The results also depend on the tested version, date, account and network conditions.

E. Ethics and Data Handling

HAR files may contain personal information and credentials. For this reason, the raw captures are kept outside the public Git repository. Generated reports use redacted values, and this paper includes only totals and non-sensitive examples. A later version of the project should add automatic secret scanning and a clear retention procedure for raw captures.

This study was completed for academic research. Its results are technical observations from specific application versions tested under controlled conditions. They describe the data-handling behaviour visible in those captures; they are not a legal judgment or final proof of non-compliance. Different versions, user actions, devices and network conditions may produce different results.

VIII. CONCLUSION AND FUTURE WORK

This paper described PrivacyComplianceVerifier, a framework that compares privacy-related network evidence with application privacy policies. The project began with HAR extraction and was gradually extended to handle duplicate fields, structured bodies and selected binary formats. It keeps the source of each finding and separates policy results from transport-related risks. Across the 30 tested applications, the framework made repeated personal, authentication, location and device information visible without publishing the original values. The final results included disclosed types, possible disclosure gaps, incomplete policy checks and risky placements of information in URLs.

The framework cannot prove legal compliance, and it cannot observe activity outside the captured session. Future work should use repeatable interaction scripts and improve policy extraction while keeping human review. Schema-aware protobuf analysis could be added when message definitions are available. The detector should also be evaluated for precision and recall using a suitable labelled dataset. Finally, traffic ownership needs further study for applications that rely on shared

infrastructure. These steps would strengthen the evidence while keeping uncertain findings separate from confirmed results.

REFERENCES

- [1] B. Andow, S. Y. Mahmud, J. Whitaker, W. Enck, B. Reaves, K. Singh, and S. Egelman, "Actions Speak Louder than Words: Entity-Sensitive Privacy Policy and Data Flow Analysis with PoliCheck," in Proc. 29th USENIX Security Symposium, 2020.
- [2] D. Bui, Y. Yao, K. Shin, J.-M. Choi, and J. Shin, "Consistency Analysis of Data-Usage Purposes in Mobile Apps," in Proc. ACM SIGSAC Conf. Computer and Communications Security, 2021, doi: 10.1145/3460120.3484536.
- [3] R. Trimananda, H. Le, H. Cui, J. Tran Ho, A. Shuba, and A. Markopoulou, "OVRseen: Auditing Network Traffic and Privacy Policies in Oculus VR," in Proc. 31st USENIX Security Symposium, pp. 3789–3806, 2022.
- [4] M. Fan, J. Shi, Y. Wang, L. Yu, X. Zhang, H. Wang, W. Jin, and T. Liu, "Giving without Notifying: Assessing Compliance of Data Transmission in Android Apps," in Proc. IEEE/ACM Int. Conf. Automated Software Engineering, 2024, doi: 10.1145/3691620.3695528.
- [5] C. Ziepel, S. Escher, S. Rehms, and S. Köpsell, "mopri—An Analysis Framework for Unveiling Privacy Violations in Mobile Apps," arXiv, 2026, doi: 10.48550/arxiv.2602.03671.
- [6] H. Cui, R. Trimananda, and A. Markopoulou, "PoliGraph: Automated Privacy Policy Analysis using Knowledge Graphs," arXiv, 2022, doi: 10.48550/arxiv.2210.06746.
- [7] O. Amaral, S. Abualhaija, D. Torre, M. Sabetzadeh, and L. Briand, "AI-Enabled Automation for Completeness Checking of Privacy Policies," IEEE Trans. Software Engineering, 2021, doi: 10.1109/TSE.2021.3124332.
- [8] D. Rodríguez Torrado, I. Yang, J. M. del Álamo, and N. Sadeh, "Large language models: a new approach for privacy policy analysis at scale," Computing, 2024, doi: 10.1007/s00607-024-01331-9.
- [9] J. M. del Álamo, D. S. Guamán, B. García, and A. Díez, "A systematic mapping study on automated analysis of privacy policies," Computing, 2022, doi: 10.1007/s00607-022-01076-3.
- [10] T. A. Rahat, M. Long, and Y. Tian, "Is Your Policy Compliant? A Deep Learning-based Empirical Study of Privacy Policies' Compliance with GDPR," in Proc. Workshop on Privacy in the Electronic Society, 2022, doi: 10.1145/3559613.3563195.
- [11] S. Liu, B. Zhao, R. Guo, G. Meng, F. Zhang, and M. Zhang, "Have You been Properly Notified? Automatic Compliance Analysis of Privacy Policy Text with GDPR Article 13," in Proc. Web Conf., 2021, doi: 10.1145/3442381.3450022.
- [12] R. El Hamdani, M. Mustapha, D. Restrepo Amariles, A. Troussel, S. Meeüs, and K. Krasnashchok, "A combined rule-based and machine learning approach for automated GDPR compliance checking," in Proc. Int. Conf. Artificial Intelligence and Law, 2021, doi: 10.1145/3462757.3466081.
- [13] H. Cui, G. Meng, Y. Zhang, W. Wang, D. Zhu, T. Su, X. Zhang, and Y. Li, "TraceDroid: A Robust Network Traffic Analysis Framework for Privacy Leakage in Android Apps," 2022, doi: 10.1007/978-3-031-17551-0_35.
- [14] Y. Li, H. Cui, J. Sun, Y. Zhang, Y. Li, G. Meng, and W. Wang, "Automated Privacy Network Traffic Detection via Self-labeling and Learning," in Proc. IEEE Symp. Computers and Communications, 2022, doi: 10.1109/ISCC55528.2022.9912471.
- [15] M. Shen, K. Ye, X. Liu, L. Zhu, J. Kang, S. Yu, Q. Li, and K. Xu, "Machine Learning-Powered Encrypted Network Traffic Analysis: A Comprehensive Survey," IEEE Commun. Surveys & Tutorials, 2023, doi: 10.1109/COMST.2022.3208196.
- [16] E. Papadogiannaki and S. Ioannidis, "A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures," ACM Computing Surveys, 2021, doi: 10.1145/3457904.

- [17] B. Yu, Y. Liu, S. Ren, Z. Zhou, and J. Liu, "METAsen: analyzing network traffic and privacy policies in Web 3.0 based Metaverse," *Digital Communications and Networks*, 2023, doi: 10.1016/j.dcan.2023.11.006.
- [18] A. Markopoulou, R. Trimnanda, and H. Cui, "A CI-based Auditing Framework for Data Collection Practices," *arXiv*, 2023, doi: 10.48550/arxiv.2303.17740.
- [19] M. Mastroianni, F. Palmieri, M. Ficco, R. Kozik, and M. Choraś, "Privacy risk analysis and metrics in capturing and storing network traffic," in *Proc. Int. Conf. Control Systems and Computer Science*, 2023, doi: 10.1109/CSCS59211.2023.00097.
- [20] T. Heino, R. Carlsson, S. Rauti, and V. Leppänen, "Assessing discrepancies between network traffic and privacy policies of public sector web services," in *Proc. Int. Conf. Availability, Reliability and Security*, 2022, doi: 10.1145/3538969.3539003.
- [21] H. Atapattu, W. Fernando, J. P. A. K. Somasiri, P. M. K. Lokuge, A. Senarathne, and M. Tissera, "A Sensitive Data Leakage Detection and Privacy Policy Analyzing Application for Android Systems (PriVot)," in *Proc. Int. Conf. Advancements in Computing*, 2021, doi: 10.1109/ICAC54203.2021.9671075.
- [22] D. Sánchez, A. Viejo, and M. Batet, "Automatic Assessment of Privacy Policies under the GDPR," *Applied Sciences*, vol. 11, 2021, doi: 10.3390/app11041762.
- [23] H. Ou, Y. Fang, Y. Guo, W. Guo, and C. Huang, "Viopolicy-Detector: An Automated Approach to Detecting GDPR Suspected Compliance Violations in Websites," in *Proc. Int. Symp. Research in Attacks, Intrusions and Defenses*, 2022, doi: 10.1145/3545948.3545952.
- [24] M. Farhadi, G. Pierre, and D. Miorandi, "Towards automated privacy compliance checking of applications in Cloud and Fog environments," in *Proc. Int. Conf. Future Internet of Things and Cloud*, 2021, doi: 10.1109/FiCloud49777.2021.00010.
- [25] I. A. Alharbi, C. Zou, and R. N. Alharbi, "Encrypted Network Traffic Analysis and Classification Utilizing Machine Learning," *Sensors*, vol. 24, 2024, doi: 10.3390/s24113509.
- [26] L. Chaddad, A. Chehab, and A. Kayssi, "OPriv: Optimizing Privacy Protection for Network Traffic," *Journal of Sensor and Actuator Networks*, vol. 10, 2021, doi: 10.3390/jsan10030038.
- [27] G. Baldini, J. L. Hernández-Ramos, S. Nowak, R. Neisse, and M. Nowak, "Mitigation of Privacy Threats due to Encrypted Traffic Analysis through a Policy-Based Framework and MUD Profiles," *Symmetry*, vol. 12, 2020, doi: 10.3390/sym12091576.
- [28] B. Sedlak, I. Murturi, P. K. Donta, and S. Dustdar, "A Privacy Enforcing Framework for Data Streams on the Edge," *IEEE Transactions on Emerging Topics in Computing*, 2024, doi: 10.1109/TETC.2023.3315131.
- [29] M. Barati, G. Aujla, J. T. Llanos, K. A. Duodu, O. Rana, M. Carr, and R. Ranjan, "Privacy-Aware Cloud Auditing for GDPR Compliance Verification in Online Healthcare," *IEEE Transactions on Industrial Informatics*, 2021, doi: 10.1109/TII.2021.3100152.
- [30] D. Blanco-Aza, A. Robles-Gómez, R. Pastor Vargas, L. Tobarra, P. Vidal-Balboa, and M. Méndez-Suárez, "Comprehensive AI-driven privacy risk assessment in mobile apps and social networks," *Cluster Computing*, 2025, doi: 10.1007/s10586-025-05624-2.