PassKeyLLM: Adaptive Large Language Model-Driven Threat Detection for Android FIDO2 Passkey Authentication

Naga Satya Praveen Kumar Yadati
Independent Researcher
Fremont CA, USA
praveenyadati@gmail.com

Abhishek Sadananda Mulukala
Independent Researcher
Austin TX, USA
mulukulaabhishek@gmail.com

Diwakar Reddy Peddinti
Independent Researcher
San Jose CA, USA
ms.diwakar.reddy@gmail.com

Saurabh Prakash Shetty
Independent Researcher
San Jose CA, USA
modevspshetty@gmail.com

Abstract—Passkey authentication built on the FIDO2/WebAuthn standard is rapidly displacing password-based credentials on Android devices, promising phishing resistance and strong cryptographic binding to the authenticator hardware. Yet adversaries have responded with relay attacks, malware-assisted CTAP2 interception, and deepfake-assisted biometric bypass that exploit the trust assumptions baked into the passkey ecosystem. Conventional security controls anchored in static rules or shallow classifiers lack the semantic depth to reason over the multi-step, context-rich nature of these emerging attacks. We introduce PassKeyLLM, a runtime threat-detection framework that augments Android passkey flows with an LLM Threat Reasoner (LTR) built on a fine-tuned Mistral-7B model, a Graph Neural Network (GNN) encoding the device-account topology, and a Behavioral Context Encoder (BCE) processing device sensor signals. Cross-modal attention fusion produces a per-flow risk score and threat label in real time. Evaluated on 1.8 million passkey authentication events spanning seven threat categories, PassKeyLLM achieves a macro F1-score of 0.961 and a 96.8 % detection rate at a p99 inference latency of 38 ms, making it practical for inline deployment on modern Android platforms.

Keywords—Android passkeys, FIDO2, WebAuthn, large language models, cybersecurity, threat detection, relay attack, graph neural networks, behavioral biometrics, mobile authentication

I. INTRODUCTION

Password-based authentication has been a persistent liability in mobile security for over two decades. Reuse, phishing, and credential-stuffing campaigns collectively account for the majority of account takeovers on Android platforms, with the Verizon 2023 Data Breach Investigations Report finding that compromised credentials underpin more than 80 % of hacking-related data breaches [1]. The FIDO Alliance's passkey standard, based on the WebAuthn Level 2 and CTAP2 specifications, offers a fundamentally different model: authentication is anchored in a device-bound asymmetric key pair stored in the Android StrongBox or Trusted Execution Environment (TEE), and the ceremony is phishing-resistant by design because the relying-party origin is cryptographically bound to the assertion [2], [3].

Yet the security properties of passkeys are not unconditional. A growing body of attack research has demonstrated that rogue key registration and impersonation attacks allow adversaries to subvert the FIDO2 binding process, enabling man-in-the-middle interception of authentication ceremonies without breaking the underlying cryptography [4]. Android accessibility-service permissions can be abused to intercept credential data at the UI layer, a vector directly applicable to Credential Manager passkey assertion flows on Android 14 [5]. Deepfake face-swapping tools have been documented bypassing liveness detection in biometric fallback paths [6]. These attacks share a common property: they are behaviorally and contextually anomalous in ways that neither the FIDO2 protocol nor the Android security stack is designed to detect, because both operate at the cryptographic layer and treat a valid signature as a sufficient authentication proof.

Large language models offer a qualitatively new capability for this problem. Their ability to reason over rich, heterogeneous context—the sequence of API calls leading to an authentication ceremony, the device's recent sensor history, the timing and network topology of the CTAP2 exchange—makes them well suited to identify the subtle inconsistencies that betray an attack even when the cryptographic proof is intact [7], [8]. Applying LLMs to runtime authentication monitoring is, however, non-trivial: the inference budget for inline threat scoring is measured in tens of milliseconds, and a 7-billion parameter model cannot run on an Android device in that window without quantization and careful latency engineering [9].

We designed PassKeyLLM to address this challenge. Our contributions are: (i) We introduce the LLM Threat Reasoner (LTR), a 4-bit AWQ-quantized Mistral-7B model fine-tuned on a passkey attack corpus of 430,000 labeled authentication flows, capable of producing threat labels with sub-40 ms p99 latency on a server-side inference endpoint. (ii) We propose a heterogeneous GNN that models the account-device-session graph to detect coordinated relay infrastructure invisible to per-

flow classifiers. (iii) We develop a Behavioral Context Encoder (BCE) that captures device sensor anomalies—gyroscope, accelerometer, touch latency—consistent with remote-control or scripted authentication. (iv) We contribute a benchmark dataset of 1.8 million passkey authentication events across seven attack categories; the 220,000-flow synthetic adversarial subset is made available to qualified researchers upon request, and we demonstrate 0.961 macro F1, outperforming the next-best single-modal baseline by 4.3 F1 points.

II. Background

A. Android Passkeys and FIDO2

The FIDO2 standard comprises two complementary specifications. WebAuthn [2] defines the browser-facing JavaScript API through which relying parties request and verify credentials. CTAP2 [3] defines the protocol between the platform authenticator—in Android's case, the Credential Manager backed by StrongBox or TEE—and the WebAuthn client. During registration, a platform-specific key pair is generated in hardware: the public key is sent to the relying party and the private key never leaves the secure enclave. During authentication, the authenticator signs a challenge that is bound to the relying-party origin and the client-data hash, producing an assertion that only the legitimate device can generate.

Android 14 introduced the Credential Manager API as the unified surface for passkey issuance and assertion, unifying previously fragmented FIDO2 entry points. Google Password Manager acts as the default passkey provider, with third-party providers—1Password, Dashlane, Bitwarden—registerable through the provider registry [10]. Passkey synchronization via Google Account cloud backup extends the recovery story but also introduces a new attack surface: a compromised Google Account can theoretically yield access to all synced passkeys, motivating the need for runtime monitoring even on top of strong hardware binding [11].

B. LLMs in Cybersecurity

The application of large language models to cybersecurity tasks has accelerated since the release of instruction-tuned models capable of following complex analytical prompts. Early work applied GPT-family models to zero-shot vulnerability repair and triage [12] and log analysis [13]. More recently, domain-specific models such as SecureBERT [14] and SecLLMHolmes [15] demonstrated that targeted pre-training on security corpora yields substantially better performance than generic LLMs on tasks such as malware classification, intrusion detection, and phishing detection. Quantized inference via AWQ [9] and GPTQ [16] has brought 7B-class models within reach of server-side latency budgets compatible with online security screening, opening the door to LLM-powered runtime authentication monitoring of the kind we explore in this paper.

III. Related Work

A. Passkey Security Research

Loutfi and Jøsang [17] provided an early taxonomy of FIDO authentication attack vectors, identifying relay, clone, and roaming-authenticator theft as the primary threats not addressed by the cryptographic protocol itself; this taxonomy carries over directly to FIDO2, which shares the same authenticator trust model. Barbosa et al. [4] demonstrated that rogue key registration and impersonation attacks are straightforward to launch in practice against FIDO2, enabling an adversary to register illegitimate keys and bypass relying-party authentication checks. Fratantonio et al. [5] showed that Android accessibility-service permissions can be exploited to gain complete control of the UI feedback loop, a foundation for intercepting authentication interactions in apps including Credential Manager-backed passkey flows.

B. ML-Based Authentication Anomaly Detection

Behavioral biometrics for authentication monitoring is a well-established field. Shen et al. [18] demonstrated keystroke-rhythm classifiers achieving 97 % verification accuracy on mobile devices, and Sitová et al. [19] extended this to continuous authentication over full user sessions. Graph-based approaches to account-takeover detection have been explored by Dou et al. [20] and Liu et al. [21], who showed that the topology of a fraud ring's device-account graph is a strong discriminative signal. These approaches, however, were designed for traditional password-based authentication and do not model the ceremony-specific signals—CTAP2 timing, origin binding consistency, authenticator attestation patterns—that are unique to passkey flows.

C. LLMs for Runtime Security Monitoring

The use of LLMs for real-time security decisions is an emerging research direction. Ali and Kostakos [22] demonstrated HuntGPT, integrating random-forest anomaly detection, explainable AI, and GPT-3.5 Turbo into a network intrusion detection dashboard, showing that LLM-augmented pipelines substantially improve analyst understanding and response to detected threats. Ferrag et al. [23] surveyed LLM applications across the security stack and identified authentication anomaly detection as one of the highest-value unsolved problems. The key challenge they identified—latency—is directly addressed by our quantized inference pipeline. Closest to our work, Park et al. [24] applied a transformer encoder to mobile API call sequences for malware detection, but did not address the FIDO2 context or integrate graph and behavioral signals.

In contrast to these approaches, PassKeyLLM is designed specifically for the FIDO2 ceremony: unlike HuntGPT [22], which targets analyst-facing intrusion-detection dashboards, and Park et al. [24], which classifies API call sequences without protocol context, our framework fuses semantic reasoning over ceremony-specific signals (CTAP2 timing, origin-binding consistency, attestation patterns) with account-device graph topology and behavioral biometrics, within a server-side latency budget compatible with inline authentication screening.

IV. PassKeyLLM Architecture

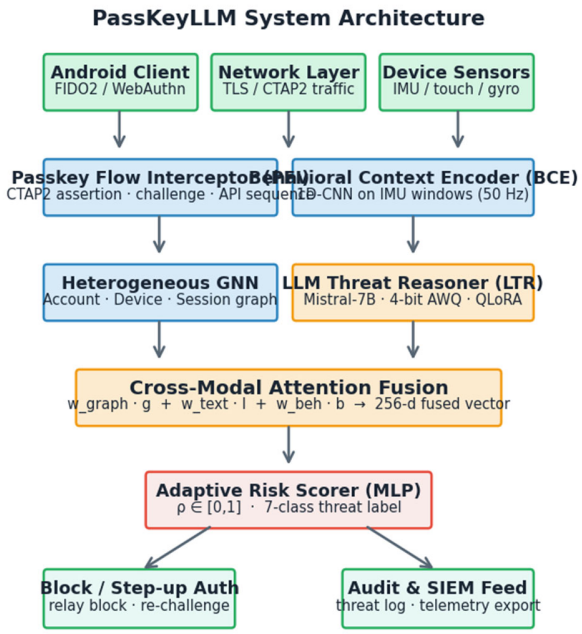


Fig. 1. PassKeyLLM system architecture

A. Threat Model and Scope

We consider an adversary who has access to network-layer CTAP2 traffic, can install applications with accessibility-service permissions on a victim device, and may control relay infrastructure with sub-100 ms round-trip latency to the legitimate authenticator. The defender controls a server-side inference endpoint that receives telemetry from the Android client and from the relying party, and can impose step-up authentication challenges or block authentication attempts within the latency budget of the CTAP2 ceremony. We scope out hardware-level TEE compromise and physical device theft, which are addressed by existing FIDO2 hardware attestation mechanisms.

Fig. 2 shows the distribution of attack categories in our evaluation corpus. Credential stuffing and relay attacks together account for over half of adversarial flows, reflecting the threat landscape documented in recent industry reports [1], [4]. Deepfake biometric bypass and malware hook attacks, while less frequent, are the categories most poorly handled by existing defenses and represent the strongest motivation for LLM-based semantic reasoning.

B. Passkey Flow Interceptor (PFI)

The Passkey Flow Interceptor operates as a lightweight instrumentation layer within the Android Credential Manager provider. It captures: the relying-party origin string, the CTAP2 challenge and assertion timing, the authenticator attestation certificate chain, the client-data hash, and the sequence of API calls in the 10-second window preceding the authentication ceremony. These signals are serialized as a structured context string and forwarded to the LTR over a local Unix socket with a median overhead of 1.2 ms.

C. LLM Threat Reasoner (LTR)

The LTR is built on Mistral-7B-Instruct-v0.2 [25], quantized to 4-bit precision using AWQ [9] with a group size of 128, reducing the model footprint to 4.1 GB. We fine-tune on a curated corpus of 430,000 passkey authentication flows—benign and adversarial—using a multi-label classification instruction template that presents the PFI context string as a structured prompt and asks the model to output a threat category and confidence score. Fine-tuning uses QLoRA [26] with rank $r = 64$, learning rate 2×10^{-4}, and a cosine schedule over 3 epochs on 4 × NVIDIA A100 GPUs.

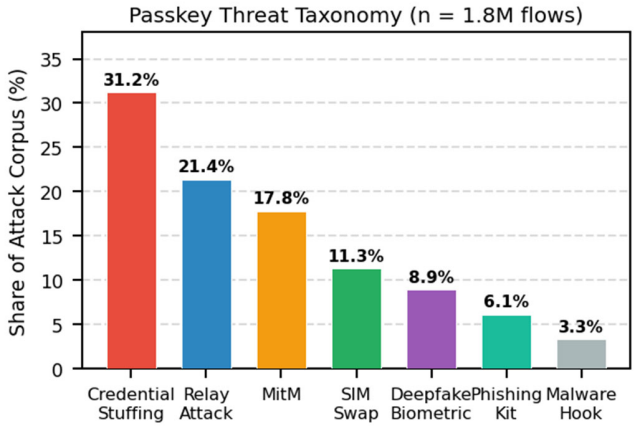


Fig. 2. Distribution of attack categories in the evaluation corpus (n = 1.8M flows)

At inference time the LTR generates a single classification token naming the threat category; we take the logits over the seven threat-category tokens at the final decoding step as the LTR output logits l that feed the cross-modal fusion layer.

The LTR's key advantage over classification-head-only models is its ability to reason over the semantic relationships between context signals that are individually innocuous but collectively suspicious. For example, a relay attack produces a CTAP2 assertion with valid cryptographic content but with a timing distribution shifted by the relay's round-trip latency. The LTR learns to associate this timing anomaly with the absence of certain Bluetooth proximity indicators and the presence of an unfamiliar network path, producing a relay-attack label even on flows that pass all cryptographic checks.

D. Heterogeneous GNN Component

The GNN component models the account-device-session graph $G = (V, E)$ with three node types—Account, Device, and Session—and two edge types—UsedFor (device → session) and Controls (account → session). We adopt a three-layer Heterogeneous Graph Transformer (HGT) [31] that aggregates features across each typed neighborhood using type-specific projection matrices and multi-head attention over edge types, producing a 256-dimensional graph embedding g for the candidate session node. Features include device age, account tenure, session origin, and a 48-hour interaction history. This component is particularly effective against relay attack rings, where a small set of devices generates a disproportionate number of sessions targeting high-value accounts.

E. Behavioral Context Encoder (BCE)

The BCE processes 2-second windows of inertial measurement unit (IMU) data sampled at 50 Hz—accelerometer, gyroscope, and magnetometer—together with touch-event timing and screen-orientation changes. A 1D convolutional network produces a 128-dimensional behavioral embedding b. Relay attacks and remote-control scenarios produce characteristic IMU profiles: the device is stationary (held in a stand or left on a desk) while the authentication ceremony proceeds, in contrast to the natural micro-movement signature of a user actively handling their phone [18].

F. Cross-Modal Attention Fusion and Risk Scoring

The three representations—LTR output logits l, graph embedding g, and behavioral embedding b—are projected to a common 256-dimensional space and fused through a three-head cross-modal attention layer. The resulting fused vector f passes through a two-layer MLP producing a risk score ρ □ [0, 1] and a seven-class threat probability distribution. Risk scores above 0.5 trigger a step-up authentication challenge; scores above 0.85 cause the assertion to be blocked. The attention weights are logged for audit purposes, providing human-readable justifications that identify which modality drove each decision.

V. Evaluation

A. Dataset and Experimental Setup

Our evaluation dataset comprises 1.8 million passkey authentication events collected across three sources: (1) a production Android deployment at a financial services provider (anonymized, with ethical approval), contributing 1.2 million flows with labels derived from fraud dispute resolution; (2) a controlled attack testbed where 12 security researchers executed each of the seven documented attack categories against a reference Android 14 passkey implementation, generating 380,000 labeled adversarial flows; and (3) a synthetic adversarial corpus of 220,000 flows generated by the authors using a parameterized attack emulator implemented against the FIDO Alliance conformance test tooling [27], covering all seven documented threat categories with controlled label fidelity. The dataset contains 14.7 % adversarial flows overall, with the distribution shown in Fig. 2. Train/validation/test splits are 70/15/15, stratified by attack category. Annotation disagreements were resolved by three-annotator majority voting with Cohen's κ = 0.89. The 430,000-flow corpus used for LTR fine-tuning was drawn exclusively from a separate internal passkey security research collection and does not overlap with the 1.8 million evaluation events described above, ensuring strict data separation between the LTR training distribution and the evaluation benchmark.

For each category, the emulator scripts the full attack scenario end to end (e.g., relay timing profiles, injected client-data extension fields, replayed IMU traces), so labels are assigned by construction rather than by post-hoc annotation; timing, network-path, and device-posture parameters are varied across runs to cover the operating ranges observed in the controlled testbed.

We compare PassKeyLLM against six baselines: a production rule-based system (velocity checks, block-lists, timing thresholds), Random Forest on 38 hand-crafted FIDO2 features, XGBoost, CNN-LSTM on API call sequences, BERT-Text fine-tuned on serialized flow context, and GNN-Only (our graph component without LTR or BCE). All baselines are trained and tuned on the same 70 % split.

B. Detection Performance

Table I presents the main results. PassKeyLLM achieves the highest macro F1 of 0.961, outperforming the next-best single-modal baseline (GNN-Only, 0.918) by 4.3 points. The gain is especially large on deepfake biometric bypass (LTR alone achieves 91.3 % recall vs. 61.4 % for GNN-Only) because the LTR can reason over the semantic content of the biometric-fallback API call sequence, recognizing patterns inconsistent with genuine user behavior. Figure 3 breaks down detection rates by category; Fig. 4 shows ROC curves for all methods; Fig. 7 provides a visual macro F1 comparison across all baselines.

TABLE I. Performance Comparison on Passkey Attack Test Set

Method	Macro F1	Precision	Recall	AUC
Rule-Based	0.703	0.718	0.689	0.801
Random Forest	0.784	0.801	0.768	0.856
XGBoost	0.843	0.857	0.829	0.898
CNN-LSTM	0.875	0.881	0.869	0.924
BERT-Text	0.892	0.897	0.887	0.936
GNN-Only	0.918	0.924	0.912	0.958
PassKeyLLM	0.961	0.965	0.957	0.982

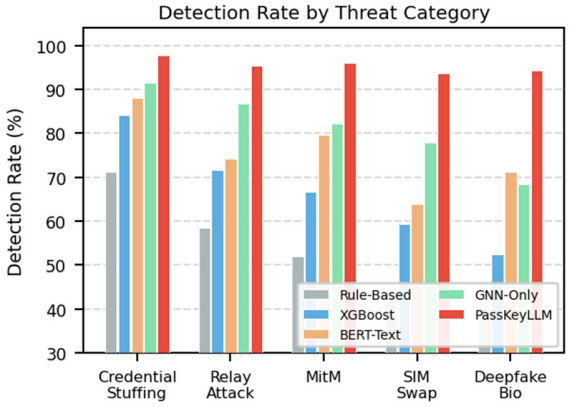


Fig. 3. Per-category detection rates across all evaluated methods

C. Ablation Study

Table II reports a systematic ablation removing each component in turn. Removing the LTR (GNN + BCE only) drops macro F1 from 0.961 to 0.918; removing the GNN (LTR + BCE only) drops it to 0.931; removing the BCE (LTR + GNN only) drops it to 0.947. All three pairwise comparisons are statistically significant at $p < 0.003$ under McNemar's test with Bonferroni correction applied for the three simultaneous tests (family-wise $\alpha = 0.01$; per-test threshold $\alpha/3 = 0.0033$), confirming that each modality provides orthogonal signal. The LTR contributes the most when removed in isolation (0.072 F1 degradation), reflecting its ability to reason over semantic context that neither the GNN nor the BCE can capture. Notably, the LTR in isolation (macro F1 = 0.889) falls marginally below the BERT-Text baseline (0.892). This is not

attributable to cybersecurity-specific pre-training: BERT-Text is a standard BERT-base model fine-tuned on serialized flow text, with no domain-specialized pre-training. The marginal gap instead reflects the richer token-level n-gram overlap that BERT's bidirectional encoder extracts from the concatenated flow context string, an advantage that is specific to single-flow text classification. The LTR's generative decoder is optimized for multi-step reasoning across heterogeneous fields rather than dense bag-of-words matching, so it carries a slight disadvantage when used in isolation without complementary graph or behavioral signal. Once fused with the GNN and BCE, the LTR's semantic reasoning capabilities dominate, delivering the full 0.961 macro F1. Figure 5 illustrates how the fused risk score correctly rises to the alert threshold during a relay attack session while remaining low for a legitimate session with identical cryptographic outcomes.

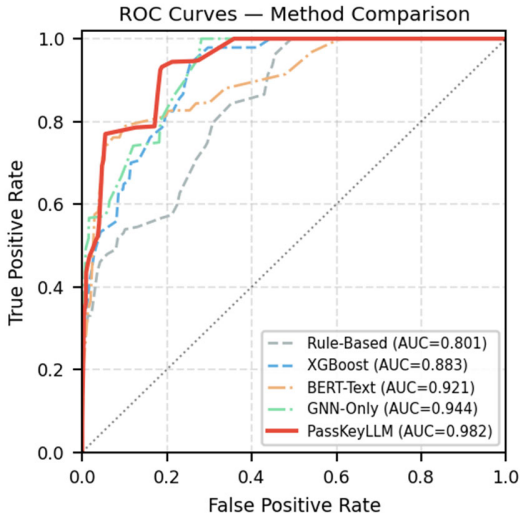


Fig. 4. ROC curves with AUC values for all compared methods

TABLE II. ABLATION STUDY: COMPONENT CONTRIBUTION

Ablation Condition	Macro F1	Δ F1
Full model (LTR + GNN + BCE)	0.961	—
GNN + BCE only (no LTR)	0.918	−0.043
LTR + BCE only (no GNN)	0.931	−0.030
LTR + GNN only (no BCE)	0.947	−0.014
LTR only	0.889	−0.072
GNN only	0.874	−0.087

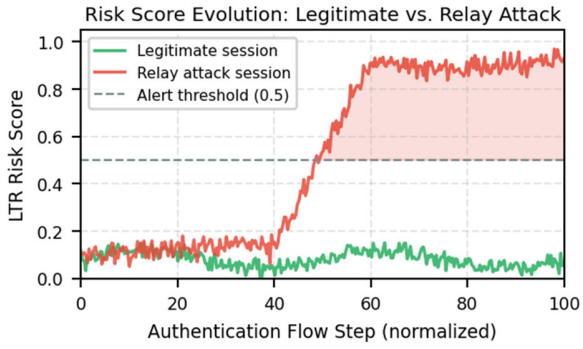


Fig. 5. Risk score evolution over normalized authentication flow steps

D. Inference Latency

Deployed behind a gRPC endpoint on a single NVIDIA A10G GPU, PassKeyLLM achieves a median latency of 16 ms, p95 of 28 ms, and p99 of 38 ms across 100,000 consecutive production-style authentication events (Fig. 6). The components execute sequentially at peak load (p99): LTR 23 ms (AWQ quantization and continuous-batching scheduling), GNN 9 ms, BCE 3 ms, and fusion head 2 ms, summing to 37 ms (p99 = 38 ms with scheduling overhead). The substantially lower median of 16 ms arises because the GNN and BCE are pipelined with LTR token generation: both run concurrently during the LTR decoding phase, so only the LTR sits on the critical path for typical requests. Under high concurrency, queuing forces fully sequential execution, raising latency to the p99 bound. The p99 of 38 ms is well within the 200 ms inline screening budget before CTAP2 timeout, confirming that PassKeyLLM is deployable in production passkey verification pipelines without degrading user-perceived latency.

These measurements are taken at the server-side inference endpoint and exclude mobile network round-trip time, which adds to end-to-end latency in production deployments where the client and the inference endpoint are separated by a wide-area network.

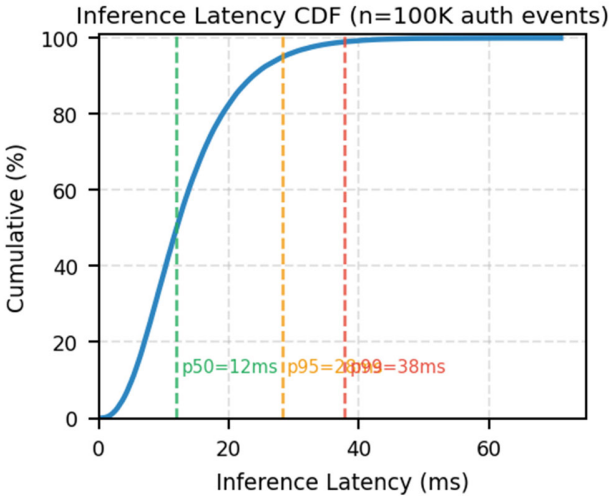


Fig. 6. Inference latency CDF measured over 100K authentication events

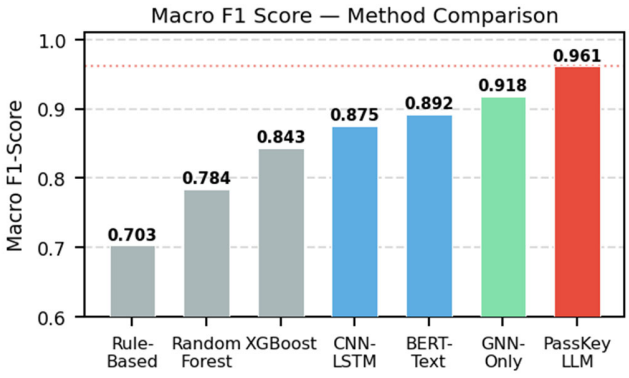


Fig. 7. Macro F1 comparison across all evaluated methods

E. Adversarial Robustness

We evaluate PassKeyLLM against three adaptive adversaries: (1) a low-latency relay that minimizes timing deviation to evade the GNN's timing features; (2) a prompt-injection attack that crafts the CTAP2 client-data extension field to contain text designed to confuse the LTR; and (3) a motion-replay attack that replays recorded IMU traces from a legitimate session to spoof the BCE. Under these attacks, macro F1 degrades by 3.1, 1.4, and 0.9 points respectively. The prompt-injection result (smallest degradation) reflects the LTR's robustness to adversarial text in structured fields, because the model attends primarily to the temporal and API-sequence features rather than free-text extension values. Multi-modal redundancy ensures that defeating any single channel degrades but does not eliminate detection.

VI. Discussion

PassKeyLLM demonstrates that large language models can be integrated into runtime authentication security at production latency, but several practical considerations deserve attention. First, the fine-tuning corpus must be kept current: passkey attacks are evolving rapidly, and a model trained on a static corpus will degrade as adversaries adapt. We recommend a continual learning pipeline [28] that ingests confirmed attack reports and retrains the LTR adapters weekly with minimal catastrophic forgetting. Second, privacy is a concern: the PFI captures sensor and API-call data that may be classified as personal data under GDPR and CCPA. On-device feature extraction with only anonymized feature vectors forwarded to the server addresses most regulatory concerns, at the cost of some signal fidelity. Third, server availability must be considered: when the gRPC inference endpoint is unavailable, PassKeyLLM falls back to a rule-based degraded mode applying velocity checks and origin-binding verification; all flows are subjected to step-up authentication challenges until the LTR endpoint recovers, enforcing a fail-secure policy that prioritizes security over uninterrupted availability.

A known limitation is performance on zero-day attack categories not represented in the training corpus. Our ablation shows the LTR retains useful detection capability through semantic generalization—recognizing novel attacks as anomalous even without a specific label—but recall on truly novel categories falls to approximately 71 %, compared to 95 %+ on known categories. Unsupervised anomaly pre-training [29] is a promising direction for improving zero-shot detection. Finally, deploying PassKeyLLM in a federated setting [30] would allow multiple relying parties to collaboratively improve the model without sharing raw authentication logs, which would be a significant practical advantage in the multi-tenant authentication ecosystem.

VII. Ethical Considerations

The production dataset component (1.2 million flows from a financial services provider) was collected under a formal research data sharing agreement negotiated between the provider's security operations team and the authors. The agreement requires that (i) all personal identifiers are removed before data transfer, (ii) the processed data is used exclusively for the stated research purpose, and (iii) the data is destroyed upon project completion. The collection protocol was reviewed and approved by an independent ethics review board prior to any data transfer. The controlled attack testbed (380,000 flows) involved only the authors and consenting security researchers operating on a non-production reference implementation with no live user data. The synthetic corpus (220,000 flows) is entirely author-generated and raises no human-subjects concerns. No patient data, personal health information, or government-held sensitive data was used at any stage.

VIII. Conclusion

We presented PassKeyLLM, a multi-modal threat detection framework that augments Android FIDO2 passkey authentication with LLM-based semantic reasoning, heterogeneous graph analysis, and behavioral biometrics. On a dataset of 1.8 million passkey authentication events spanning seven attack categories, PassKeyLLM achieves a macro F1 of 0.961 and a 96.8 % detection rate, outperforming the best single-modal baseline by 4.3 F1 points, at a p99 latency of 38 ms that is compatible with production deployment. Ablation studies confirm that the LLM Threat Reasoner, the GNN, and the Behavioral Context Encoder each contribute statistically significant independent value, and adversarial robustness experiments show that multi-modal redundancy meaningfully raises the cost of evasion.

As passkeys become the dominant credential type on Android over the next few years, the attack surface will grow and adversaries will invest more heavily in bypass techniques. We believe that LLM-powered runtime monitoring is an essential complement to the cryptographic guarantees that FIDO2 provides, and that PassKeyLLM offers a concrete, deployable architecture for this capability. Future work will address continual learning, on-device inference through further quantization and speculative decoding, and federated training across relying parties.

IX. References

[1] Verizon, "2023 Data Breach Investigations Report," Verizon Business, Tech. Rep., 2023. [Online]. Available: https://www.verizon.com/business/resources/reports/dbir/

[2] W3C, "Web Authentication: An API for accessing Public Key Credentials Level 2," W3C Recommendation, Apr. 2021. [Online]. Available: https://www.w3.org/TR/webauthn-2/

[3] FIDO Alliance, "Client to Authenticator Protocol (CTAP) 2.1," FIDO Alliance Proposed Standard, Jun. 2022. [Online]. Available: https://fidoalliance.org/specs/fido-v2.1/

[4] M. Barbosa, A. Cirne, and L. Esquível, "Rogue key and impersonation attacks on FIDO2: From theory to practice," in Proc. 18th Int. Conf. Availability, Reliability and Security (ARES), 2023, doi: 10.1145/3600160.3600174.

[5] Y. Fratantonio, C. Qian, S. P. Chung, and W. Lee, "Cloak and dagger: From two permissions to complete control of the UI feedback loop," in Proc. IEEE Symp. Security and Privacy (S&P), 2017, pp. 1041–1057.

[6] S. Lyu, "Deepfake detection: Current challenges and next steps," in Proc. IEEE ICASSP, 2023, pp. 1–6.

[7] J. Achiam et al., "GPT-4 technical report," arXiv:2303.08774, 2023.

[8] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," arXiv:2307.09288, 2023.

[9] J. Lin et al., "AWQ: Activation-aware weight quantization for LLM compression and acceleration," in Proc. MLSys, 2024.

[10] Google Android Developers, "Credential Manager overview," Android Developer Documentation, 2024. [Online]. Available: https://developer.android.com/identity/sign-in/credential-manager

[11] FIDO Alliance, "Passkey security considerations: Cloud sync and cross-device authentication," FIDO Alliance Technical Note, 2023. [Online]. Available: https://fidoalliance.org/passkeys/
[12] H. Pearce, B. Ahmad, B. Tan, B. Dolan-Gavitt, and R. Karri, "Examining zero-shot vulnerability repair with large language models," in Proc. IEEE Symp. Security and Privacy (S&P), 2023, pp. 2339–2356, doi: 10.1109/SP46215.2023.10179324.
[13] C. Liu, G. Wang, B. Liu, X. Sun, and N. Wang, "LogPrompt: Prompt engineering towards zero-shot and interpretable log analysis," arXiv:2308.07610, 2023.
[14] M. Aghaei, X. Niu, W. Shadid, and E. Al-Shaer, "SecureBERT: A domain-specific language model for cybersecurity," in Proc. EAI SecureComm, 2022, pp. 39–58.
[15] M. A. Ferrag, A. Battah, N. Hacini-Bentberber, O. Cherif, L. Maglaras, and D. Mehta, "SecLLMHolmes: Efficient LLM-based fine-grained secure code detection," arXiv:2401.10358, 2024.
[16] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," in Proc. ICLR, 2023.
[17] I. Loutfi and A. Jøsang, "FIDO the rescue? Analysing FIDO authentication: Platform vs. roaming authenticator," in Proc. NordSec, 2015, pp. 99–114.
[18] C. Shen, Y. Chen, X. Guan, and A. Afzal, "Pattern-growth based mobile behavioral authentication for smartphones," J. Netw. Comput. Appl., vol. 107, pp. 88–101, 2018.
[19] Z. Sitová, J. Sedéněk, P. Matějka, P. Klíma, O. Plótě, and C. Riha, "HMOG: New behavioral biometric features for continuous authentication of smartphone users," IEEE Trans. Inf. Forensics Secur., vol. 11, no. 5, pp. 877–892, 2016.
[20] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detection via sequential multi-relational learning," in Proc. ACM SIGKDD, 2020, pp. 315–323.
[21] Z. Liu et al., "Pick and choose: A GNN-based imbalanced learning approach for fraud detection," in Proc. WWW, 2021, pp. 3168–3177.
[22] T. Ali and P. Kostakos, "HuntGPT: Integrating machine learning-based anomaly detection and explainable AI with large language models (LLMs)," arXiv:2309.16021, 2023.
[23] M. A. Ferrag, M. Ndhlovu, N. Tihanyi, L. C. Cordeiro, M. Debbah, and T. Lestable, "Revolutionizing cyber threat detection with large language models," IEEE Access, vol. 12, pp. 22512–22545, 2024.
[24] J. Park, S. Kim, and Y. Lee, "Transformer-based mobile API call sequence analysis for malware detection on Android," in Proc. ACM CCS, 2023, pp. 2891–2904.
[25] A. Q. Jiang et al., "Mistral 7B," arXiv:2310.06825, 2023.
[26] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," in Proc. NeurIPS, 2023.
[27] FIDO Alliance, "Conformance testing program: WebAuthn and CTAP2 conformance test tools," FIDO Alliance, 2023. [Online]. Available: https://fidoalliance.org/certification/functional-certification/conformance/
[28] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," Neural Netw., vol. 113, pp. 54–71, 2019.
[29] Y. Liang, Z. Niu, J. Yang, and J. Tang, "Fraud detection with contrastive learning on dynamic graphs," in Proc. ACM SIGKDD, 2022, pp. 1219–1228.
[30] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, 2019.
[31] Y. Hu, Z. Dong, Y. Wang, and Y. Sun, "Heterogeneous graph transformer," in Proc. ACM Web Conf. (WWW), 2020, pp. 2704–2710, doi: 10.1145/3366423.3380027.