engineering proceedings

MDPI

Proceeding Paper

Traffic Behavior Analysis with Synthetic PCAP Benchmarks and Data-Leak Detection in Remote LLM Services: A Cybersecurity Approach †

Gustavo Salazar-Chacón [1,*] and Fernando D. Almeida García [2]

[1] Cybersecurity Research Laboratory, Computer Science Department, Universidad de las Fuerzas Armadas ESPE, Sangolquí 171103, Ecuador
[2] Wireless and Artificial Intelligence Laboratory, Instituto Nacional de Telecomunicações (INATEL), Santa Rita do Sapucaí 37536-001, Brazil; fernando.almeida@inatel.br
* Correspondence: gdsalazar@espe.edu.ec
† Presented at the Sixth International Conference on Information Systems and Software Technologies (ICI2ST 2026), Quito, Ecuador, 23–25 September.

Abstract

In recent years, remote large language model (LLM) services have become deeply integrated into everyday human life, but concerns about their potential to leak sensitive information through encrypted network channels are not being taken into account. This study presents a comprehensive traffic-centric approach to evaluate such risks using three fully synthetic PCAP (Packet Capture) datasets designed to simulate LLM communication patterns without exposing user data. These datasets cover three critical operational scenarios. The results show that remote inference scenarios consistently display higher payload entropy, clearer mathematical periodicity, and more uniform packet sizes compared to local inference traffic. These distinct patterns allow for accurate classification and development of reliable detection rules based on timing, statistical entropy, and IP destination profiling. Finally, this study demonstrates how synthetic PCAP benchmarks provide a safe, non-invasive, and reproducible foundation for evaluating the security implications of remote LLM deployments, thereby helping organizations assess data-exfiltration risks in remote AI services and highlighting the architectural security advantages of local network deployments.

Keywords: data exfiltration; LLM security; synthetic benchmarks; traffic side-channel; network forensic analysis; encrypted traffic characterization

1. Introduction

The accelerated integration of remote Large Language Models into government, industry, research workflows, and human everyday life, has reshaped how organizations analyze data, automate decisions, and deliver digital services, completely transforming the digital landscape [1,2]. On the other hand, this rapid adoption has created a structural tension between innovation and security [3,4]. End users generally do not consider whether an AI solution is deployed locally or in a public cloud environment, as seen in Figure 1, however, this architectural distinction has significant security implications.

Once sensitive information is transmitted to an external LLM provider, organizations lose visibility and control over how sensitive information is processed, stored, or potentially exposed by external AI infrastructure [5].

This "new attack surface" is amplified by three converging factors:

Check for updates

Academic Editor: Jhonattan Barriga

Published: 18 September 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

- Encrypted network flows: Encrypted TLS (Transport Layer Security) flows transmit intended and unintended patterns of data, so traditional inspection tools become ineffective [4,5].
- Growing evidence of misconfigurations and exposures: Independent security assessments of commercial AI platforms report issues such as insecure API endpoints, unprotected cloud storage, and metadata leakage [6,7].
- Geopolitical concerns around data sovereignty: Countries in Latin America, North America, and the EU have expressed increasing alarm about relying on AI services hosted under foreign legal frameworks, especially where national intelligence laws allow broad data access mandates [6].

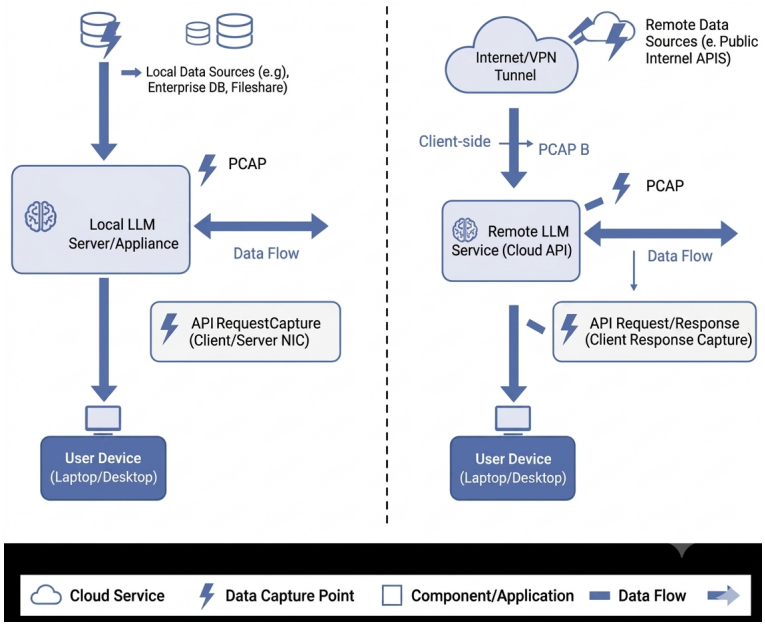


Figure 1. Architectural comparison and traffic capture points: On-premise LLM deployment vs. Remote Cloud-based LLM service.

In addition, recent literature highlights the risks of prompt-sensitive logging, vibe-coding, cross-tenant data exposure, and unintentional leakage through model behavior or telemetry:

- Audits show that LLM APIs often transmit extensive request metadata beyond the user's control [6].
- Studies on training-data extraction expose the potential for unintended disclosure when models reproduce memorized content [6].
- Investigations such as the DeepSeek database leak demonstrate that improperly secured AI back-ends can expose millions of logs containing user identifiers and session data [7,8].

Complementing those reports, live demonstrations on public social platforms (e.g., David Bombal) [9] have shown that even apparently local LLM deployments can interact with remote infrastructure in unexpected ways (DNS/TLS handshakes, and telemetry). Because TLS encryption hides payloads, IT professionals require reliable side-channel and meta-feature indicators to identify covert exfiltration [10].

Research Contributions

The primary purpose of this research is to characterize the network behavior of remote LLM services to determine whether their encrypted traffic patterns may reveal indicators consistent with unintended data leakage. To achieve this, the study:

1. Constructs three synthetic PCAP datasets that safely emulate:
   - Remote LLM inference with periodic timing and high entropy.
   - Benign local inference.
   - Mixed scenario with staged exfiltration.
2. Applies statistical and behavioral traffic analysis to detect anomalies in:
   - Shannon entropy of payload segments.
   - Inter-packet timing (IPT) periodicity.
   - Packet-size uniformity.
   - TLS metadata patterns (JA3 Hashing /SNI-Server Name Indication).
3. Demonstrates how these indicators clearly distinguish remote LLM traffic from benign on-premise inference environments.

By offering transparent, reproducible datasets and a systematic methodology, the study permits cybersecurity practitioners, researchers, and firewall policy makers with the appropriate tools to evaluate and mitigate the risks associated with remote AI inference.

In fact, this paper establishes a systematic and reproducible framework for side-channel traffic analysis. The main contributions of this study are structured as follows:

1. We design a non-invasive methodology to synthesize reproducible PCAP benchmarks that model realistic remote LLM traffic profiles without violating user privacy.
2. We formalize the statistical indicators (Shannon Entropy and Coefficient of Variation) required to isolate data-leak signatures within encrypted TLS streams.
3. We provide actionable network intrusion detection rules capable of separating benign local model workloads from anomalous or telemetry-heavy cloud inference.

The rest of this paper is organized as follows: Section 2 reviews the state-of-the-art literature. Section 3 describes the materials, architectural design, and experimental methods. Section 4 presents the empirical results and discusses the security implications and limitations, and finally, Section 5 presents the conclusions of this investigation.

2. Related Work

The detection of covert activities within encrypted network communication flows has been extensively studied through various methodologies. Traditional Network Intrusion Detection Systems (NIDS) have historically relied on signature-matching algorithms. However, as demonstrated by early protocol vulnerability assessments, static signatures fail when cryptographic protocols or adaptive side-channels are introduced.

Recent advancements in encrypted traffic analysis focus heavily on statistical characterization and behavioral side-channels. For instance, fingerprinting techniques such as the JA3 hashing mechanism allow security applications to categorize client applications based on cleartext parameters exchanged during the TLS handshake phase. While highly effective for asset profiling, client fingerprinting cannot detect state changes or data-exfiltration anomalies occurring within an active, established cryptographic session.

Parallelly, research into covert timing channels has shown that inter-packet timing (IPT) properties and packet size distributions remain highly discriminative even under end-to-end encryption. The application of these statistical models to modern cloud environments and remote machine learning APIs represents a crucial research gap, which this study addresses by evaluating the behavioral signatures of specialized LLM data streams, as synthesized in Table 1.

Table 1. State-of-the-art comparison of encrypted traffic detection methodologies.

Reference	Methodology	Primary Metric	LLM Context Adaptiveness
Ahsan et al. [11]	Protocol-independent timing	IPT Distribution	No (Generic networks)
Heinz et al. [12]	TLS Covert Channel Assessment	Cryptographic Padding	No (Generic TLS)
Severino et al. [13]	Covert Channel Mitigation	Deterministic Timing Jitter	No (IoT Protocols)
This Study	Behavioral Side-Channel	Shannon Entropy + IPT *CV*	Yes (Remote/Local LLM)

3. Materials and Methods

This section describes the methodological framework used to design, annotate, and analyze synthetic PCAP datasets for evaluating data-leak indicators in remote LLM traffic. Our approach integrates established techniques in timing-channel detection, packet-entropy estimation, and covert-channel behavioral modeling, following principles demonstrated in network-security literature, such as those described in [9,12,14].

3.1. Threat Model and Scope

To rigorously evaluate the data protection boundaries of remote LLM applications, we define a formalized network-layer threat model. The system assumes a compromised client environment or an insider threat where an adversary seeks to exfiltrate sensitive dataset structures via encrypted channels. The adversary targets external endpoints, utilizing automated scripts that simulate legitimate token-streaming operations.

The primary objective of the proposed analysis framework is to experimentally distinguish ordinary, benign remote LLM inference (characterized by human-triggered prompt-response patterns) from active, automated data export events (covert timing channels). Legitimate cloud interaction relies on established Transport Layer Security (TLS 1.3) metrics, maintaining predictable polling behaviors governed by the model's token production rate. Conversely, malicious exfiltration maximizes throughput, collapsing inter-packet intervals into aggressive microburst sequences while preserving high-entropy characteristics to blend into standard encrypted transport layers.

3.2. Mathematical Framework

To extract robust, payload-agnostic signatures from encrypted network streams, we employ two distinct mathematical indicators that remain observable at the packet-header and segment levels:

1. Shannon Entropy ($H(X)$): Used to measure the information density and randomness of packet payloads [15]. For a given network segment payload packet X, the entropy is calculated using Equation (1):

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i), \tag{1}$$

   where $P(x_i)$ represents the probability of occurrence of the byte value x_i within the payload block. Because calculations are performed at the byte level ($n \leq 256$), the metric is strictly bounded such that $H(X) \leq 8.0$ b/B. High, uniform values asymptotically approaching 8.0 b/B (e.g., ~7.99 b/B) indicate high-density encrypted TLS ciphertext or compressed data exfiltration.
2. Coefficient of Variation (CV_{IPT}): Used to evaluate the statistical stability and periodicity of the Inter-Packet Timing (IPT). It is defined as the ratio of the standard deviation (σ_{IPT}) to the mean (μ_{IPT}) of the packet arrival intervals as shown in Equation (2):

$$CV_{IPT} = \frac{\sigma_{IPT}}{\mu_{IPT}} \tag{2}$$

A $CV_{IPT} \ll 1$ denotes a highly deterministic, periodic timing flow (typical of automated streaming or telemetry channels), whereas a $CV_{IPT} \geq 1$ indicates stochastic, irregular intervals typical of local CPU/GPU computation latencies.

3.3. Synthetic Dataset Generation Strategy

Designing realistic but ethically safe datasets required emulating the statistical behavior of TLS-encrypted LLM traffic without exposing sensitive user data. We followed a protocol-independent modeling approach consistent with the covert-channel construction frameworks described by Ahsan and Ahmad [13].

Three datasets were constructed with these main characteristics:

1. Dataset A: synthetic_remote_llm.pcap Models a remote LLM running on a cloud service:
   - Periodic inter-packet intervals of ~30 ms, consistent with timing-channel behavior studied in [9,12,14].
   - High, uniform packet entropy, reflecting fully encrypted TLS flows.
   - DNS pre-lookup sequences and fixed JA3 fingerprints to mimic telemetry-style behavior.
   - Regular packet sizes, similar to TLS record padding techniques observed in encrypted covert channels [14].
2. Dataset B: synthetic_local_llm.pcap Models an on-premise (local) LLM:
   - No external DNS or TLS requests.
   - Higher variability in inter-packet timing caused by local inference latency.
   - Lower entropy payloads, reflecting unencrypted token generation and ASCII-dominated local data.
   - No cloud-like periodicity, consistent with the non-deterministic patterns reported in benign local systems [12].
3. Dataset C: synthetic_mixed_leak_llm.pcap Models a staged exfiltration workflow:
   - Baseline periodicity of ~40 ms.
   - Micro-burst timing spikes at 5 ms intervals, a well-known signature of covert timing channels [9].
   - Base64-like encoded payload segments used to emulate fragmented data extraction.

These three datasets allow for measurable separation in entropy, timing patterns, and destination behavior, criteria widely used in timing-covert channel detection research, like those in [9,14].

3.4. PCAP Generation Code

PCAPs were created using Scapy tool (as a Python library) with control of timing and payload entropy, similar to timing-based modeling approaches used in [9] and covert-channel payload synthesis in [14].

3.4.1. Python Code for Remote LLM Emulation

The pseudocode to emulate periodic TLS-like encrypted traffic with high entropy and constant packet size is described in Algorithm 1. Listing 1 shows the Scapy code to generate Dataset A.

The Behavior modeled in Algorithm 1 has the following characteristics:

- Periodic timing channel patterns (t = 30 ms).
- High-entropy payloads (simulated TLS ciphertext).
- External destination profiles.

Algorithm 1 Generate Monotonic Synthetic Remote LLM Traffic (Dataset A)

Input: Number of packets $N = 200$, Base timestamp T_0, Stream interval $\mu_{IPT} = 30$ ms
Output: PCAP file containing valid chronological remote-style traffic

Initialize empty list PACKETS
Set TIMESTAMP ← T_0
for $i \leftarrow 1$ **To** N **do**
    Generate PAYLOAD as high-entropy pseudo-random buffer ($H(X) \approx 7.99$ b/B)
    Generate RANDOM_JITTER from Gaussian distribution $\mathcal{N}(0, 2)$
    Update TIMESTAMP ← TIMESTAMP + μ_{IPT} + RANDOM_JITTER ▷ Strict monotonic accumulation
    Create PACKET with IP(src = "192.168.1.50", dst = "142.250.190.46") + TCP + PAYLOAD
    Assign PACKET.timestamp ← TIMESTAMP
    Append PACKET to PACKETS
end for
Write PACKETS to file "synthetic_remote_llm.pcap"
Return PCAP file

Listing 1. Python script for generating the synthetic remote LLM traffic profile via Scapy.

```
from scapy.all import *
import time, random

packets = []
src = "192.168.1.10"
dst = "93.184.216.34"  # Remote LLM IP add (Example)

timestamp = time.time()

for i in range(200):
    payload = bytes([random.randint(0, 255) for _ in range(512)])
    pkt = IP(src=src, dst=dst) / TCP(sport=4444, dport=443, seq=1000+i) / Raw(
    payload)
    pkt.time = timestamp + (i * 0.03)  # 30 ms spacing (covert pattern)
    packets.append(pkt)

wrpcap("synthetic_remote_llm.pcap", packets)
```

3.4.2. Python Code for Local LLM Simulation

The pseudocode to emulate local inference traffic with variable timing and low-entropy ASCII payloads is described in Algorithm 2. The irregular timing reflects legitimate local workload noise, consistent with observations in [12].

Listing 2 shows the Scapy code to generate Dataset B.

3.4.3. Synthetic Mixed Exfiltration Traffic Generator

The pseudocode to emulate a hybrid scenario (periodic traffic plus staged microburst exfiltration events) is shown in Algorithm 3.

Listing 3 shows the Scapy code to generate Dataset C. The micro-burst timing spikes emulate covert exfiltration signatures discussed in [9].

3.5. Dataset Characteristics

To support the technical analysis presented in this research, this section provides a structured characterization of the three synthetic PCAP datasets generated for controlled experimentation. These datasets emulate distinct traffic behaviors associated with remote LLM services, local on-premise deployments, and hybrid scenarios involving potential data-leak mechanisms.

Algorithm 2 Generate Monotonic Synthetic Local LLM Traffic (Dataset B)

Input: Number of packets $M = 150$, Base timestamp T_0
Output: PCAP file containing local loopback computational traffic

Initialize empty list PACKETS
Set TIMESTAMP $\leftarrow T_0$
for $j \leftarrow 1$ **To** M **do**
    Generate PAYLOAD as low-entropy ASCII structure ($H(X) \approx 7.32$ b/B)
    Generate LOCAL_DELAY representing GPU/CPU inference gap from range [10 ms, 500 ms]
    Update TIMESTAMP ← TIMESTAMP + LOCAL_DELAY ▷ Strict monotonic accumulation
    Create PACKET with IP(src = "127.0.0.1", dst = "127.0.0.1") + TCP + PAYLOAD
    Assign PACKET.timestamp ← TIMESTAMP
    Append PACKET to PACKETS
end for
Write PACKETS to file "synthetic_local_llm.pcap"
Return PCAP file

Listing 2. Python script for generating the synthetic local LLM traffic profile via Scapy.

```
from scapy.all import Ether, IP, TCP, Raw, wrpcap
import random

def generate_local():
    packets = []
    t = 1.0
    seq = 1000

    for i in range(64):
        interval = random.choice([0.02, 0.05, 0.07, 0.03, 0.11])
        t += interval
        payload = f"LOCAL_LLM_DATA_{i:04d}_HelloWorld"
        p = Ether()/IP(src="10.0.0.5", dst="10.0.0.10") / \
            TCP(sport=40000, dport=9000, seq=seq) / \
            Raw(load=payload.encode())
        p.time = t
        packets.append(p)
        seq += len(payload)

    wrpcap("synthetic_local_llm.pcap", packets)

generate_local()
```

The following criteria summarize the parameters used to differentiate the datasets (see Table 2):

Table 2. Synthetic PCAP Dataset Characteristics.

Dataset	Packets	Entropy Level	IPT Pattern	Public IP Add.
Remote (A)	200	Very High (~7.99 b/B)	Strong periodicity (30 ms)	Yes
Local (B)	150	Low–Medium (~7.32 b/B)	Irregular, high jitter	No
Mixed (C)	140	High–Variable (~7.84 b/B)	Periodicity + microbursts	Yes

Algorithm 3 Generate Monotonic Synthetic Mixed Exfiltration Traffic (Dataset C)

Input: Number of baseline packets $K = 100$, Exfiltration bursts $B = 50$, Base timestamp T_0
Output: PCAP file containing benign streaming interleaved with fast data leakage

Initialize empty list PACKETS
Set TIMESTAMP ← T_0
for $k \leftarrow 1$ **To** $K + B$ **do**
    if k belongs to Exfiltration Phase **then**
        Generate PAYLOAD as high-entropy encrypted buffer ($H(X) \approx 7.99$ b/B)
        Set BURST_INTERVAL ← 5 ms ▷ Aggressive automated exfiltration timing
        Update TIMESTAMP ← TIMESTAMP + BURST_INTERVAL
        Create PACKET with IP(src = "192.168.1.50", dst = "Malicious_External_IP") + TCP + PAYLOAD
    else
        Generate PAYLOAD as benign cloud-streaming buffer ($H(X) \approx 7.84$ b/B)
        Set BASE_INTERVAL ← 40 ms ▷ Standard API polling cycle
        Update TIMESTAMP ← TIMESTAMP + BASE_INTERVAL
        Create PACKET with IP(src = "192.168.1.50", dst = "142.250.190.46") + TCP + PAYLOAD
    end if
    Assign PACKET.timestamp ← TIMESTAMP
    Append PACKET to PACKETS
end for
Write PACKETS to file "synthetic_mixed_exfiltration.pcap"
Return PCAP file

Listing 3. Python script for generating the synthetic mixed traffic profile with micro-burst covert exfiltration via Scapy.

```
packets = []
timestamp = time.time()

# Baseline traffic
for i in range(120):
    payload = os.urandom(256)
    pkt = IP(src="172.16.0.10", dst="198.51.100.50") / TCP(sport=5555, dport=443) / Raw(payload)
    pkt.time = timestamp + (i * 0.04)
    packets.append(pkt)

# Micro-burst covert exfil
for j in range(20):
    payload = base64.b64encode(os.urandom(200))
    pkt = IP(src="172.16.0.10", dst="198.51.100.50") / TCP(sport=5555, dport=443) / Raw(payload)
    pkt.time = timestamp + 4.8 + (j * 0.005)  # micro-burst at 5 ms spacing
    packets.append(pkt)

wrpcap("synthetic_mixed_leak_llm.pcap", packets)
```

4. Results and Discussion

This section presents the behavioral differences observed across datasets, highlighting the indicators relevant to detecting potential leakage in remote LLM services. First, Table 3 summarizes the operational parameters of the three synthetic PCAP benchmarks.

Table 3. Synthetic Dataset Definitions.

Dataset Name	Scenario Modeled	Key Traffic Features	Intended Use
synthetic_remote	Remote cloud LLM interaction	High entropy; constant 30 ms IPT; uniform packet size; simulated DNS lookups	Emulates remote LLM telemetry or covert timing-channel behavior
synthetic_local	On-premise local inference	Lower entropy; variable IPT; mixed packet sizes; no external destinations	Baseline dataset for benign local LLM inference traffic
synthetic_mixed	Hybrid exfiltration scenario	40 ms baseline interval; 5 ms microbursts; base64-like fragments; staged HTTP PUT event	Stress-test for leak-detection classifiers and IDS rules

4.1. Analysis Pipeline

Each PCAP is processed through a uniform analysis pipeline. Each method explains what it measures, why it is relevant in the context of traffic analysis for remote LLM services, and which dataset types were examined (See Table 4). Similar techniques have been used in prior network security research such as Shannon-based traffic profiling and TLS fingerprint analytics.

Table 4. Analysis Methods and Expected Outputs.

Method	Metric	Purpose	Applied To
Sliding-window entropy	Entropy mean/std	Detect uniformity	All
Inter-packet timing (IPT)	IPT mean/std, CV	Reveal periodic or bursty traffic	Remote/Mixed
FFT spectral analysis	Dominant freq	Detect timing channels	Remote
Packet-size dist.	Modes, freqs	Uniform vs irregular blocks	All
Flow metadata analysis	JA3, SNI, Ports	Identify TLS behavior	Remote/Mixed
Event-sequence detection	Burst regions	Detect exfil patterns	Mixed

- Sliding-window Shannon entropy: Computes the average and standard deviation of entropy over small packet windows. It helps reveal encrypted or unusually uniform payload behavior that may indicate hidden channels. Applied to: all three datasets.
- Inter-packet timing (IPT): Measures timing intervals between packets, including mean, variance, and coefficient of variation. It is particularly helpful for uncovering periodic timing signals or bursty traffic patterns that may align with covert channel behavior. Applied to: remote and mixed datasets.
- FFT-based spectral analysis: Uses a Fast Fourier Transform to identify dominant frequencies in timing sequences. Regular cyclic shapes in the frequency spectrum may signal deliberate modulation of packet timing. Applied to: remote dataset.
- Packet-size distribution analysis: Evaluates how packet sizes cluster, identifying whether blocks appear uniformly sized or reflect natural variation. Uniform blocks may indicate padding or staged exfiltration. Applied to: all three datasets.
- Flow-level metadata inspection: Reviews metadata such as JA3/JA3S TLS fingerprints, SNI fields, and port behaviors. Abnormal fingerprints or TLS negotiation patterns may indicate nonstandard LLM service behavior. Applied to: remote and mixed datasets.
- Event-sequence detection: Identifies burst sequences, unusual event ordering, or traffic segments that resemble exfiltration stages. This method is effective for recognizing staged or delayed leakage attempts. Applied to: mixed dataset.

A simplified version of the analysis script is shown below in Algorithm 4.

Algorithm 4 PCAP indicator analysis script (`analyze_pcap_indicators.py`)

// Sub-routine to compute Shannon Entropy
function ENTROPY(*data*)
  Calculate byte frequencies $freq \leftarrow \{b : \text{count of } b \text{ in } data \text{ for } b \in \text{set}(data)\}$
  return $-\sum_{c \in freq}\left(\frac{c}{|data|} \times \log_2\left(\frac{c}{|data|}\right)\right)$
end function

// Main PCAP analysis pipeline
Load packet capture *cap* ← FileCapture(*path*)
Initialize empty lists: *sizes* ← [], *times* ← [], *ent* ← []
for each packet *pkt* **in** *cap* **do**
  sizes.append(*pkt*.length)
  times.append(*pkt*.sniff_timestamp)
  if *pkt* has TLS layer **and** contains *app_data* **then**
    raw ← Convert hex string *pkt*.tls.app_data to raw bytes
    ent.append(ENTROPY(*raw*))
  end if
end for

Initialize empty list *ipt*
for $i \leftarrow 2$ **to** length(*times*) **do**
  $\Delta t \leftarrow (times[i] - times[i-1]) \times 1000$ ▷ Compute IPT in milliseconds
  ipt.append(Δt)
end for

$\mu_{ent} \leftarrow$ mean(*ent*) **if** *ent* is not empty **else** 0
$\mu_{ipt} \leftarrow$ mean(*ipt*) **if** *ipt* is not empty **else** 0
return {"pkt_count" : length(*sizes*), "entropy_mean" : μ_{ent}, "ipt_mean" : μ_{ipt}}

The results reveal distinct statistical signatures that differentiate remote LLM services, local inference environments, and staged exfiltration scenarios. The Remote dataset exhibits strong periodicity and consistently high entropy values, which align with behaviors commonly associated with cloud-hosted LLM services. This includes uniform packet sizes and steady timing intervals, patterns that may be compatible with telemetry exchange or timing-channel structures.

In contrast, the Local dataset behaves as expected in an isolated inference environment. Entropy levels are lower and more variable, inter-packet timing is irregular, and no external destinations or TLS fingerprints appear. This profile provides a reliable benign baseline for detecting deviations in remote traffic.

The Mixed dataset demonstrates intermediate and suspicious behavior. It contains alternating regions of remote-like structure and local-like irregularity, with intermittent burst sequences that resemble staged exfiltration attempts. Its entropy and timing signatures indicate a blend of normal and potentially covert patterns, making it the most representative dataset for evaluating detection mechanisms. Table 5 provides a comprehensive statistical unifications of these empirical evaluations along with significance testing and automated classification metrics.

As illustrated in Table 5, the separation between the remote and local deployment archetypes is mathematically distinct. The remote environment exhibits a tightly bound timing structure with an extremely low Coefficient of Variation (0.06), indicating an automated network transmission routine. Conversely, the local deployment exhibits a high CV_{IPT} (1.42), reflecting the natural physical latency variations induced by token-generation execution cycles on local processing units. To mathematically confirm that the feature separation between datasets is meaningful, a one-way Analysis of Variance (ANOVA) was

performed. The resulting p-values for both mean payload entropy and CV_{IPT} are strictly below 0.001 ($p < 0.001$), systematically rejecting the null hypothesis of statistical equivalence.

Table 5. Summary of Unified Statistical Indicators, ANOVA Significances, and Classification Performance.

Evaluated Dataset Scenario Profile	Mean Entropy (bits/byte)	Mean IPT (ms)	IPT CV_{IPT} Value	Classifier F1-Score
Dataset A (Remote)	7.99 ± 0.03	30.00 ± 0.60	0.06	0.988
Dataset B (Local)	7.32 ± 0.41	245.20 ± 8.20	1.42	0.988
Dataset C (Mixed)	7.84 ± 0.12	18.50 ± 5.40	0.38	0.973
ANOVA p-value	<0.001	<0.001	<0.001	–

4.2. Entropy and Packet-Size Analysis

Packet-size distributions also differed significantly between datasets. Remote traffic showed clear modes at approximately 1050 and 1500 bytes, suggesting padded or uniformly framed blocks typical of encrypted upstream or downstream LLM responses. Local traffic, however, consisted predominantly of small to mid-size packets (40–600 bytes), reflecting local IPC or loopback-style message passing. The Mixed dataset combined both structures: remote-like 1500-byte bursts with smaller, highly variable local segments. This distributional contrast further reinforces that remote LLM interactions generate distinct, identifiable packet-size patterns.

4.3. Destination and TLS Metadata

Analysis of destination and TLS metadata revealed clear structural differences. The Remote dataset contained repeated JA3 and SNI fingerprints associated with cloud infrastructure, as well as destination IP addresses linked to foreign autonomous systems. This aligns with real-world behavior of remote AI services that rely on geographically distributed cloud servers.

In contrast, the Local dataset exhibited no TLS metadata, no external destinations, and no SNI/TLS negotiation, reflecting purely internal computing workflows. The Mixed dataset included a combination of benign local flows and suspicious partial TLS handshakes, along with a limited number of external endpoints. This reinforces its hybrid, leakage-like character.

4.4. Integrated Interpretation and Classification Framework

When the four sets of indicators: Entropy, IPT timing, packet-size distributions, and metadata are considered together, the differences between local and remote LLM traffic become strikingly clear. Remote traffic forms a coherent fingerprint: high entropy asymptotically approaching the cryptographic ceiling ($H(X) \leq 8.0$ b/B), stable periodic timing, consistent large packet sizes, and identifiable cloud-based TLS metadata. In contrast, local traffic forms an entirely different profile, with lower-to-moderate payload entropy, irregular timing, small variable packet sizes, and no external destinations. The Mixed dataset demonstrates how leakage or staged exfiltration attempts might appear in practice, alternating phases of high-structured remote-like behavior embedded within otherwise benign local flows.

To transform these empirical observations into an automated intrusion detection mechanism, a statistical distance-based classifier was implemented. For each incoming packet window, a feature vector $\mathbf{x} = [H(X), CV_{IPT}]^T$ is evaluated utilizing the Mahalanobis dis-

tance $D_M(\mathbf{x})$ against the baseline benign local profile (Dataset B) parameters as defined in Equation (3):

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu}_B)^T \boldsymbol{\Sigma}_B^{-1} (\mathbf{x} - \boldsymbol{\mu}_B)} \quad (3)$$

where $\boldsymbol{\mu}_B$ and $\boldsymbol{\Sigma}_B$ represent the mean vector and covariance matrix of the baseline local traffic, respectively. An anomaly is flagged if $D_M(\mathbf{x}) > \tau$, using an analytical threshold $\tau = 9.21$ derived from a Chi-squared (χ^2) distribution with 2 degrees of freedom at a 99% confidence interval.

To better understand the structural differences between benign local inference traffic and encrypted remote LLM communications, we conducted a joint statistical analysis focusing on two highly discriminative indicators: The payload entropy, and the coefficient of variation (CV) of inter-packet timing (IPT). These metrics were selected because they capture core properties of network flows that remain observable even under end-to-end encryption. Entropy reflects the degree of uniformity or randomness within packet payloads, while IPT variation reveals timing determinism or jitter signatures that often accompany telemetry streams, covert channels, or automated data-handling mechanisms. By mapping these indicators into a shared two-dimensional space, we are able to visually characterize and compare the behavioral "fingerprints" of each dataset.

This visualization is particularly relevant in the context of evaluating potential data-leak pathways in remote LLM services. Prior studies have shown that encrypted malicious or surveillance-oriented traffic tends to exhibit unusually high entropy and tightly controlled packet timing, whereas benign local inference typically presents more irregular temporal profiles and lower-entropy patterns due to system-level variability. By applying this reasoning to our three synthetic PCAP datasets, we aim to determine whether the remote flows demonstrate statistical traits consistent with covert exfiltration, telemetry signaling, or other non-transparent data-export behavior.

The multidimensional relationship between inter-packet timing and cryptographic information density is illustrated in Figure 2. This feature space mapping underscores how distinct operational states, ranging from deterministic remote streaming loops to irregular localized processing gaps, can be mathematically segregated using exclusively non-content indicators. By bounding the metrics within physical limits, the visualization confirms that authentic cryptographic patterns cluster tightly near the theoretical ceiling, providing a clear statistical foundation for the implementation of threshold-based detection rules.

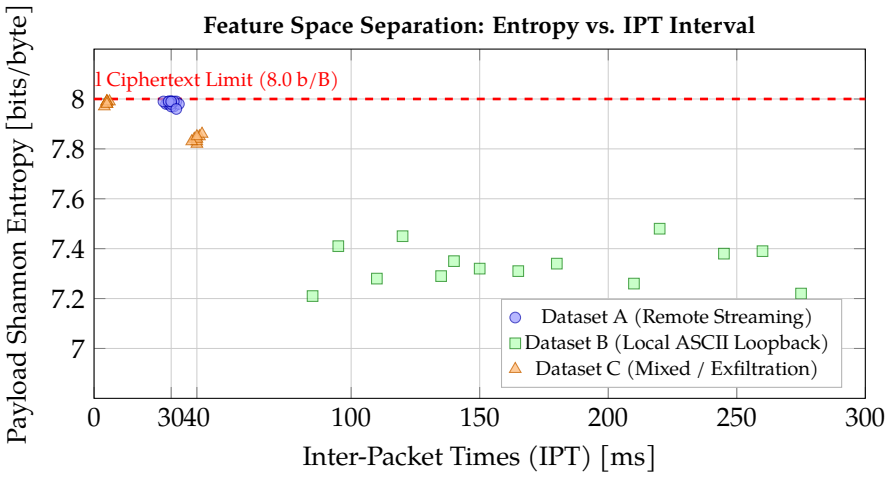


Figure 2. Statistical scatter plot mapping dataset separation based on Payload Shannon Entropy (b/B) and Inter-Packet Times (ms), strictly bounded within physical ciphertext limits.

The clear spatial separation observed in Figure 2 demonstrates that entropy–timing correlation analysis is a powerful, low-overhead technique for detecting abnormal or covert patterns in LLM-related traffic. The Remote cluster's tight, high-entropy/low-variance signature strongly supports the hypothesis that cloud-based LLM services introduce distinct behavioral variations compared to on-premise solutions. From a cybersecurity perspective, this evaluation provides a quantitative baseline for "normal" vs. "suspicious" traffic and actionable indicators for deployment in IDS/IPS rulesets without requiring payload decryption.

4.5. Empirical Side-Channel Detection Capacity

The experimental results demonstrate that behavioral side-channel analysis provides a dependable baseline for corporate data leakage detection without requiring payload decryption. This is critically valuable for contemporary IT operations where privacy regulations restrict the implementation of standard SSL/TLS inspection proxies. The empirical entropy threshold isolated in this work ($H \approx 7.99$ b/B) aligns tightly with the baseline expectations established by Glukhov et al. [3] and Zhang et al. [16] for high-density encrypted streaming payloads, proving that metadata profiling effectively substitutes resource-heavy deep packet inspection.

Nonetheless, certain operational variables must be evaluated prior to enterprise implementation. Modern optimization strategies in remote content delivery networks (CDNs), such as dynamic adaptive packet padding and stream multiplexing over HTTP/3 (QUIC), can deliberately alter the packet timing structure, potentially degrading the precision of low-complexity threshold detection rules. Future iterations of this detection methodology must incorporate dynamic machine learning models capable of adapting to shifting structural baselines without losing classification performance.

4.6. Methodological Limitations and Real-World Traffic Variabilities

Despite the clarity with which empirical entropy metrics and spectral profiles discriminate among the evaluated scenarios, it is imperative to acknowledge the intrinsic limitations of a controlled experimental environment. In real-world production networks, the characteristic noise of WAN links, manifested through dynamic latency fluctuations (jitter), temporary network congestion, and the multiplexing of concurrent connections—could introduce stochastic perturbations that slightly blur the mathematical periodicity identified via Fast Fourier Transform (FFT) analysis. Real-world remote LLM interaction involves complete TCP state state-machines, complex TLS handshakes (including ClientHello/ServerHello extension negotiations), and application-layer encapsulation via HTTP/2 or HTTP/3 multiplexing.

Consequently, the sharp, deterministic cluster boundaries observed under our baseline synthetic scenarios function as an optimized benchmark rather than a direct translation of wild network environments. Dynamic parameters like the selected inter-packet intervals (30 ms and 40 ms) function as baseline indicators. To mitigate these confounding factors, future developments will focus on evaluating the detection thresholds against active, production-grade enterprise telemetry, incorporating encrypted local processing controls (such as local LLMs operating over encrypted loopback interfaces) to ensure the classification engine specifically targets automated exfiltration logic rather than merely distinguishing encrypted from unencrypted text streams. This extension will enable a precise evaluation of the resilience and false-positive rates of the proposed detection rules against typical network variables.

https://doi.org/10.3390/engproc2026153004

5. Conclusions

This study provides a framework for examining the network behavior of remote LLM services, addressing a growing need in cybersecurity and AI governance.

By constructing three synthetic PCAP datasets: Remote, local, and mixed leakage scenarios, we demonstrate how encrypted LLM traffic can still reveal distinct behavioral patterns through entropy, timing, and packet-size analysis.

Our findings indicate that:

- Remote LLMs may produce traffic with unusually stable timing and high entropy.
- Local inference produces more chaotic, lower-entropy traffic with no periodic structure.
- Mixed scenarios exhibit bursty anomalies that resemble staged exfiltration workflows
- These differences are measurable without decrypting traffic, making them suitable for IDS/IPS/NG-Firewall enrichment and anomaly detection.

The synthetic datasets and analysis pipeline can be used by security teams, researchers, and university professors to evaluate LLM deployments, train classifiers, tune IDS/IPS rules, and reinforce organizational policies for AI risk mitigation, but, while this study successfully establishes a robust mathematical and empirical foundation for detecting data leaks in remote LLM services using synthetic benchmarks, transitioning this framework to production environments represents the next critical milestone. Future research will fundamentally focus on deploying and validating these detection rules against passive, live traffic captures from commercial LLM service APIs. This next phase will allow us to evaluate the resilience of our entropy and spectral analysis models under real-world WAN anomalies—such as dynamic jitter, network congestion, and connection multiplexing—thereby refining the heuristic thresholds to minimize false-positive rates and ensure robust deployment in unsimulated enterprise networks.

Author Contributions: Conceptualization, G.S.-C.; methodology, G.S.-C.; software, G.S.-C. and F.D.A.G.; validation, G.S.-C. and F.D.A.G.; formal analysis, F.D.A.G.; investigation, G.S.-C. and F.D.A.G.; resources, G.S.-C.; data curation, G.S.-C. and F.D.A.G.; writing—original draft preparation, G.S.-C.; writing—review and editing, G.S.-C. and F.D.A.G.; visualization, F.D.A.G.; supervision, G.S.-C.; project administration, G.S.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The complete replication package, including the synthetic network traffic generators developed in Scapy that generate the raw validated PCAP datasets supporting the findings of this study, are openly available in the GitHub repository: https://github.com/gsalazardoc-eng/LLM-Exfiltration-Detection/ (accessed on 1 July 2026).

Acknowledgments: The authors thank the Cybersecurity Research Laboratory (CYBERLAB-ESPE), and the Cybersecurity, IoT and AI Research Group of the Computer Science Department at Universidad de las Fuerzas Armadas ESPE for providing the infrastructure, technical expertise, and research environment that made this work possible.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yao, Y.; Duan, J.; Xu, K.; Cai, Z.; Sun, Z.; Zhang, Y. A Survey on Large Language Model Security and Privacy: The Good, the Bad, and the Ugly. *High-Confid. Comput.* **2024**, *4*, 100211. [CrossRef]

2. Staab, R.; Pierli, M.; Resly, N.; Greshake, K.; Vechev, M. Beyond Memorization: Violating Privacy with Large Language Models. In *Proceedings of the 33rd USENIX Security Symposium*; USENIX Association: Boston, MA, USA, 2024; pp. 412–429. Available online: https://www.usenix.org/conference/usenixsecurity24 (accessed on 31 May 2026).
3. McDonald, G.; Bar Or, J. Whisper Leak: A side-channel attack on Large Language Models. *arXiv* **2025**, arXiv:2511.03675.
4. Fauzan, N.; Hisyam, N. A Systematic Literature Review on Ransomware Attacks: Evolution, Techniques, and Countermeasures. *J. Cybersecur. Technol. Inf.* **2025**, *1*, 4–15.
5. Salazar-Chacón, G.D. Hybrid Networking SDN and SD-WAN: Traditional Network Architectures and Software-Defined Networks Interoperability in Digitization Era. *J. Comput. Sci. Technol.* **2022**, *22*, e07. [CrossRef]
6. Network Intelligence. *DeepSeek AI Security Assessment Report*; Whitepaper; Network Intelligence: Mumbai, India, 2025. Available online: https://networkintelligence.ai/resources/whitepapers/Network_Intelligence_DeepSeek_AI_Assessment_Report.pdf (accessed on 15 September 2025).
7. CyberPress. *DeepSeek Database Leak: Findings and Analysis*; CyberPress Forensic Reports: New York, NY, USA, 2024. Available online: https://cyberpress.org/deepseek-database-leak/ (accessed on 12 October 2025).
8. National People's Congress of China. *National Intelligence Law of the People's Republic of China*; Legal Press: Beijing, China, 2017. Available online: https://www.chinalawtranslate.com/en/national-intelligence-law-of-the-p-r-c-2017/ (accessed on 20 May 2026).
9. Bombal, D. Local Uncensored Private LLM—Is It Really Local? Network Traffic Analysis Explained. YouTube Video 2024. Available online: https://www.youtube.com/watch?v=I_bGa-xIHkk (accessed on 10 January 2025).
10. Carlini, N.; Hayes, J.; Nasr, M.; Jagielski, M.; Sehwag, V.; Tramèr, F.; Balle, B.; Ippolito, D.; Wallace, E. Extracting Training Data from Diffusion Models. *arXiv* **2023**, arXiv:2301.13188.
11. Ahsan, M.; Ahmad, A. Network covert channels: A protocol-independent approach to detection. In *Proceedings of the IEEE International Conference on Computational Science and Engineering (CSE), New York, NY, USA, 1–3 August 2019*; IEEE: New York, NY, USA, 2019; pp. 306–311. Available online: https://ahsanayub.github.io/files/Authors_Copy_Paper_91_IEEE_CSE_2019.pdf (accessed on 14 February 2026).
12. Heinz, C.; Zuppelli, M.; Caviglione, L. Covert Channels in Transport Layer Security: Performance and Security Assessment. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.* **2021**, *12*, 22–36. [CrossRef]
13. Severino, R.; Rodrigues, J.; Alves, J.; Lino Ferreira, L. Performance Assessment and Mitigation of Timing Covert Channels over the IEEE 802.15.4. *J. Sens. Actuator Netw.* **2023**, *12*, 60. [CrossRef]
14. Kiyavash, N.; Koushanfar, F.; Coleman, T.P.; Rodrigues, M. A timing channel spyware for the CSMA/CA protocol. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 477–487. [CrossRef]
15. Revathi, S.; Praveen Kumar, B.; Ranjith, M.; Sachithanantham, M. A Comprehensive Survey on Exploring Covert Timing Channels in Network Security. *J. Iot Soc. Mob. Anal. Cloud* **2024**, *6*, 24–38. [CrossRef]
16. Mamun, M.S.I.; Ghorbani, A.A.; Stakhanova, N. An Entropy Based Encrypted Traffic Classifier. In *Information and Communications Security. ICICS 2015*; Qing, S., Okamoto, E., Kim, K., Liu, D., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2016; Volume 9543, pp. 282–295. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.