Beyond RTT: An Adversarially Robust Two-Tiered Approach for Residential Proxy Detection

Temoor Ali*, Shehel Yoosuf†‡, Mouna Rabhi*‡, Mashael Al-Sabah*, Hao Yun§
*Qatar Computing Research Institute
{tali, msalsabah}@hbku.edu.qa, moona.rabhi@gmail.com
†Hamad Bin Khalifa University
shyo09009@hbku.edu.qa
§National University of Singapore
haoyun@nus.edu.sg

Abstract*—Residential IP proxy networks have reached unprecedented scale, yet they pose significant security risks by enabling malicious activities such as fraud, web scraping, and sophisticated cyberattacks while masking traffic behind legitimate home addresses. Existing detection approaches rely primarily on cross-layer Round-Trip Time (RTT) discrepancies, but we demonstrate these methods are fundamentally flawed: simple traffic scheduling attacks can reduce detection recall from 99% to just 8%, rendering state-of-the-art techniques unreliable against basic adversarial evasion. To address this critical vulnerability, we introduce novel traffic analysis and flow-correlation features that accurately capture the characteristics of gateway and relayed traffic, moving beyond vulnerable timing-based approaches. We further develop *CorrTransform*, a Transformer-based deep learning architecture engineered for maximum adversarial resilience. This enables two complementary detection strategies: a lightweight approach using engineered features for efficient large-scale detection, and a heavyweight deep learning approach for high-assurance in adversarial settings. We validate our methods through a comprehensive analysis of Bright Data's EarnApp using 15 months of traffic data (900GB) encompassing over 110,000 proxy connections. Our two-tiered framework enables ISPs to identify proxyware devices with >98% precision/recall and classify individual connections with 99% precision/recall under normal conditions, while maintaining >92% F1 score against sophisticated attacks, including scheduling, padding, and packet reshaping where existing methods completely fail. For content providers, our approach achieves near-perfect recall with <0.2% false positive rate for distinguishing direct from proxy traffic. This work shifts proxy detection from vulnerable timing-based approaches to resilient architectural fingerprinting, providing immediately deployable tools to combat the growing threat of malicious residential proxy usage.

I. Introduction

Residential IP proxy networks have reached unprecedented scale, with major providers like Bright Data operating over 72 million residential IPs across 195 countries [1]. These networks transform ordinary home devices into proxy nodes that route external traffic through users' internet connections, creating a massive distributed infrastructure that enables clients to utilize residential IP (RESIP) addresses for web scraping, ad verification, and bypassing geo-restrictions. To expand their reach, providers increasingly embed proxy SDKs into popular mobile apps (weather, gaming, and utility applications) effectively converting millions of smartphones into proxy nodes, often without explicit user awareness [2]. This explosion has created proxy networks of unprecedented size, with providers like Bright Data [1] and Oxylabs [3] recruiting millions of IPs worldwide.

Security and privacy issues. RESIPs pose significant security and ethical risks that extend far beyond traditional proxy threats [4], [5]. Unlike traditional data center proxies, RESIPs mask malicious traffic behind legitimate home IP addresses, making them particularly attractive to cybercriminals. These services have already enabled sophisticated attacks: the Midnight Blizzard attack on Microsoft leveraged residential proxies for infiltration and evasion [6], and the DDoS attack conducted on KARAPATAN (a human rights alliance) was conducted through Bright Data's IPs [7]. Moreover, malicious actors have exploited proxyware for money laundering schemes [8], click fraud [9], cryptocurrency mining [10], and credit stuffing attacks [11]. Furthermore, these services claim to provide user anonymity [12], but major providers like Bright Data have been accused of selling data belonging to minors scraped from social media platforms [13], potentially exposing bandwidth-sharing users to legal consequences.

Service and content providers are increasingly interested in detecting proxy connections, especially those used in activities such as web scraping and automated traffic. Recently, both Meta and X filed lawsuits against Bright Data over data scraping [14], [15]. Additionally, YouTube has implemented detection mechanisms to identify and mitigate click fraud caused by bots operating through residential IP proxies [16]. While many companies deploy anti-scraping techniques to protect their platforms, proxy providers continuously develop evasion methods to bypass these defenses, creating an ongoing arms race between detection and circumvention [17].

While traffic analysis has emerged as a promising detection

‡Work done while at Qatar Computing Research Institute.

Network and Distributed System Security (NDSS) Symposium 2026
23-27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.242086
www.ndss-symposium.org

approach, existing methods suffer from a critical vulnerability. Recent research has focused on cross-layer Round-Trip Time (RTT) discrepancies as a fingerprint for proxy traffic [18]–[20]. However, we demonstrate that these RTT-based methods are fundamentally flawed: proxies can trivially evade detection through simple traffic scheduling attacks, causing detection recall to plummet from 99% to just 8%. This devastating vulnerability renders existing state-of-the-art techniques unreliable against even basic adversarial evasion, highlighting an urgent need for more robust detection approaches.

To address this critical gap, we focus our analysis on Bright Data, the world's largest RESIP provider and its EarnApp client. The Bright Data network, which originated from the popular Hola VPN service infrastructure [21], [22], represents the most significant RESIP ecosystem, yet has been excluded from previous traffic analysis studies [18], [19]. With over 100 million users reportedly opted into EarnApp [23], understanding its traffic patterns is crucial for developing effective detection methods. We deploy multiple monitoring nodes running EarnApp instances and systematically collect encrypted traffic data under realistic conditions while ensuring user privacy protection (Section X). Through this comprehensive data collection, we provide the first detailed insights into the operational characteristics of a major RESIP network, uncovering previously unknown architectural patterns, gateway behaviors, and traffic correlation structures that inform our detection approach.

In this paper, we present a robust two-tiered approach for RESIP detection that overcomes the fundamental limitations of RTT-based methods through novel correlation-based features and adversarially-resilient deep learning architectures. We address three critical research questions from different network vantage points: (RQ1) Can ISPs detect devices with RESIP activities? (RQ2) Can an ISP determine whether specific connections are relayed through proxies versus directly initiated by users? and (RQ3) Can content providers (e.g., YouTube) detect which incoming connections are proxy-based? Our solutions enable effective detection across these diverse operational scenarios. We further detail our contributions below:

- We conduct the first comprehensive traffic analysis of Bright Data's EarnApp, collecting over 900 GB of traffic data across 15 months with more than 110,000 proxy connections, revealing novel insights into the operational patterns of the world's largest RESIP network.
- We are the first to empirically show a critical vulnerability in existing detection methods, demonstrating that cross-layer RTT features, the cornerstone of prior proxy detection work [18]–[20], can be trivially defeated by simple scheduling attacks, reducing detection recall from 99% to 8%.
- We develop novel traffic analysis and correlation-based architectural fingerprinting features that exploit the intrinsic relationship between gateway and relayed traffic streams, an invariant across various proxy networks. We introduce CorrTransform, a novel Transformer-based architecture designed for resilience against adversarial manipulation through self-attention mechanisms and specialized regularization techniques.
- We design and validate a practical two-tiered detection framework for ISPs that first identifies devices running proxyware with $>$ 98% precision/recall, then distinguishes individual relayed connections from legitimate user traffic. We show the robustness of our approach: our framework achieves 99% precision and 99% recall under normal conditions, while maintaining $>$ 92% F1 score under sophisticated attacks, significantly outperforming strong baselines including DeepCoFFEA [24].
- We demonstrate effective server-side detection for content providers, achieving near-perfect recall (98%) with minimal false positives ($<$ 0.2% FPR) for distinguishing direct user connections from proxy traffic, enabling robust fraud detection. [1]

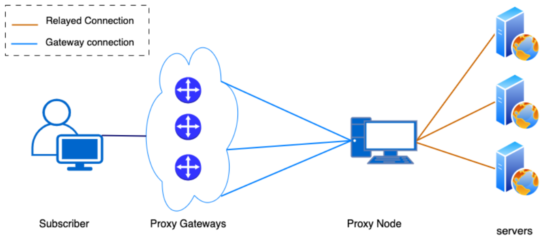


Fig. 1: RESIP architecture

In Section X, we discuss the privacy implications of our work and situate it within end-user-protection context (e.g., detecting unintended proxy participation). We also elaborate on the scope, and limitations of any deployment.

II. Background and Related Work

A. Proxy providers.

Proxy service providers have successfully sourced millions of residential IPs, as confirmed by multiple studies utilizing infiltration experiments [5]. One common recruitment strategy involves incentivizing mobile app developers to integrate proxy SDKs into their applications, enabling them to monetize their "free" third-party apps. This approach effectively transforms users' mobile devices into proxy nodes, leading to the proliferation of hundreds of proxy-enabled apps across various categories, including gaming and weather. Alternatively, some proxy networks source IPs directly from real users by offering financial incentives to run proxyware, allowing their devices to participate as nodes in the network.

When EarnApp is installed, it operates as a proxy node, establishing connections with control and gateway servers simultaneously while awaiting incoming connection requests. When a proxy network subscriber requests access to a website such as x.com, the proxy node receives the request via the gateway and initiates the connection on behalf of the subscriber. Once the three-way TCP handshake is completed, an end-to-end TLS handshake is established between the subscriber and x.com. The proxy node then forwards encrypted

[1]Code and implementation available at: https://github.com/qcri/ProxyFeatureExtraction

traffic, relaying packets upstream to x.com and downstream to the subscriber through the designated gateway. Since the TLS session is directly established between the subscriber and the destination server, the proxy node simply acts as an intermediary, forwarding encrypted traffic without decrypting or modifying its contents. Figure [1] shows how the service operates.

According to previous studies [2], [5], [18], other major RESIP providers (e.g., PacketStream, IPRoyal, Honeygain) share the same backconnect architecture as the one shown in Figure [1]. We refer to the connection between a proxy node and a gateway as a *gateway* connection, and the connection between the proxy node and a destination server (requested by the subscriber) as a *relayed* connection. Existing works [18], [19] highlight differences that are only implementation-level between Bright Data and other providers: nodes may learn gateway IPs via a fixed domain (PacketStream) or a control API (IPRoyal/Honeygain), tunnels may use standard HTTPS or a custom encrypted protocol, and DNS resolution may use plaintext UDP or DoH (Honeygain). Essentially, these variations do not alter the gateway↔node↔server relay topology. Our approach exploits architectural coupling, time-binned volume correlation between gateway and relayed flows, and their session structure, rather than superficial provider-specific details; thus, we believe our proposed methods including CorrTransform could be adapted and customized across providers.

B. Traffic analysis

Although most internet traffic is encrypted and should appear random to an observer, side-channel fingerprints can still be extracted due to unique traffic patterns, server-side signatures, and protocol configurations. This is done using *traffic analysis*, a technique used to infer information from encrypted traffic. Traffic analysis typically involves extracting metadata such as packet sizes, sequences, inter-packet timing, and flow patterns. These features are then used to train machine learning models that classify traffic and infer additional insights, thereby diminishing the privacy protections offered by encryption. This technique has been widely applied as a privacy attack against various protocols, including DNS [25], TLS [26]–[28], and VPN [29]. A well-studied application is Website Fingerprinting (WF), where an attacker attempts to identify the website a user (using Tor or VPN) is visiting without relying on the destination IP [30]–[37].

In this work, we test our classification approach in the face of sophisticated padding and timing attacks that aim to introduce adversarial perturbations against key features such as the cross-layer RTT and flow correlation features.

Proxies. Recently, there has been growing interest in using traffic analysis to identify proxy traffic, and our research builds on and complements this expanding body of work. Xue *et al.* [38] leverage the concept of nested protocol stacks inherent to proxying and tunneling and develop an n-gram-based classification model to detect encapsulated TLS handshakes. In a later work, Xue *et al.* [20] demonstrated that the proxy architecture introduces a distinct discrepancy between TCP-level RTTs and TLS/application-level RTTs, making it a highly effective feature to fingerprint proxy connections. This measurable inconsistency serves as a discriminator, enabling the identification of traffic routed through proxies.

RESIP. For RESIP detection specifically, Chiapponi *et al.* [19] aim to detect scraping bots that are based on RESIPs. Similarly to Xue *et al.*, they empirically show that RESIPs introduce a measurable discrepancy between the TCP-level RTT and the actual network delay observed at the application level and build an RTT-based approach to detect RESIPs. More recently, Huang *et al.* [18] builds on existing literature that has explored the ecosystem of RESIPs [2], [5], [39]. They have also performed traffic analysis on three proxy providers, and show over 90% F1 and 4% FPR using as little as the first eight packets of proxied and non-proxied connections. We show the shortcomings of this approach in Section [V]. Moreover, their study excludes the largest provider, Bright Data. They also use published limited background datasets. In contrast, we collect our both the background and the RESIP traffic datasets from the same environments to reduce possible biases and ensure more realistic experiments.

III. Threat Model

We consider two types of defenders with varying network visibility: internet service providers with a comprehensive view and content providers (e.g., YouTube) with a more limited perspective. The adversary is the proxy provider (e.g., Bright Data), who seeks to make their traffic indistinguishable from that of a regular user and is capable of applying adversarial perturbations to the traffic.

A. Defenders: ISPs and Content Providers

The defenders' primary goal is to accurately identify traffic associated with residential IP proxy services being used for malicious purposes. However, their specific objectives and observational capabilities differ:

ISP. The ISP has full visibility into all incoming and outgoing traffic from a proxy node's device. Their goal is twofold: first, to identify which devices on their network are running proxyware, and second, to distinguish which specific connections from those devices are being relayed for proxy subscribers versus those initiated directly by the device owner. This allows them to monitor for potential abuse or violations of their terms of service.

Content Provider. The content provider (e.g., a social media platform or streaming service) has a more restricted view. They can only observe the incoming packets at their servers. Their goal is to determine whether an incoming connection originates directly from a legitimate user or is being relayed through a RESIP node. This is crucial for detecting fraudulent activities such as view inflation, fake engagement, and automated web scraping that undermine service integrity.

Both defenders are assumed to rely only on traffic metadata (such as packet size, timing, and IP addresses) for detection. The ISP cannot inspect packet content due to encryption, and while content providers can, we assume this information is not used.

B. Adversary: The Proxy Provider

The adversary is the proxy provider, whose main objective is to evade detection by both ISPs and content providers, ensuring their service remains operational and attractive to clients who rely on it for anonymity or to bypass restrictions. As of now, RESIPs have not been observed deploying sophisticated network-level evasion techniques, likely because ISPs have not yet implemented aggressive detection policies against them. However, we assume an advanced and motivated adversary with the following capabilities:

Traffic Manipulation. The adversary has full control over the traffic between its gateway servers and the proxy node on the user's device. They also control the outgoing traffic from the proxy node to the destination server.

Adversarial Perturbations. The adversary can implement various attacks to mimic benign user traffic by manipulating packet timing, size, and sequence. We assume they can leverage techniques like traffic scheduling and packet reshaping to disrupt detection models. The specific adversarial attacks used to evaluate our models are detailed in Section VI-C.

The adversary's primary limitation is that they cannot control or alter the legitimate background traffic generated by the user's device. Their manipulations are confined to the proxyware's own connections (gateway and relayed). The challenge for the adversary is to blend their traffic seamlessly with this uncontrollable, genuine traffic without degrading their own service's performance (e.g., by introducing excessive latency).

IV. The network behind EarnApp

To understand Bright Data's RESIP ecosystem and facilitate classification, we construct an elaborate testbed to collect the three types of traffic: background, relayed, and gateway. We then use the relayed and the gateway traffic to uncover the structure, the components, communication patterns, and the operational dynamics of the EarnApp ecosystem.

A. Data collection methodology

To collect traffic, we set up a testbed consisting of an Android emulator, which replicates a real mobile environment with predefined device models (e.g., Pixel device running Android 11). Python scripts automate browsing, enabling website visits and user interactions. We deploy Bright Data's EarnApp[2] inside the emulator, creating a reproducible setup that closely mirrors real-world conditions.

Challenges. Note that prior work on RESIP identification [18] relies on external background traffic datasets that were not collected within the same experimental setup. For accurate comparison, it is essential to capture nonproxy and proxy traffic simultaneously on the same devices and under identical network conditions. While independent browsing datasets exist and have been used in prior work, we argue that they are collected in different environments and can actually introduce discrepancies due to variations in network conditions, device types, or geographic locations. To overcome these challenges and ensure an environment consistent with RESIP setup, we synthetically generate our own background traffic within the same controlled setup, co-capturing both background and proxy traffic simultaneously from the same device.

Browsing traffic. We simulate real-world user behaviors by implementing various user profiles and usage loads. To generate browsing traffic, we curated a targeted list of domains from the Tranco list [40]. Tranco is a top sites ranking method made specifically for research and hardened against manipulation [40], and includes search engines, e-commerce platforms, and social media sites. Crucially, streaming services are also a key component of our curated list and are actively visited in our simulations as they constitute a substantial portion of overall internet traffic. After an initial filtering step (e.g. removing 404 domains), we select 700 domains from the top 1,000 Tranco list, and 300 are randomly selected from the top 1001 to 1,000,000, following previous works [41].

Using this list, we simulate client behaviors with three distinct user profiles that represent varying frequencies and patterns of web browsing. These profiles range from light to heavy and are executed using multiple tabs in Chrome[3], with randomized time gaps between site accesses to diversify the simulated browsing patterns. Lighter profiles introduce longer intervals between page loads (20 seconds to 5 minutes), mimicking casual or infrequent browsing. Medium profiles shorten these intervals to 20 seconds to 1.5 minutes, representing more regular activity. Heavy profiles simulate high-frequency browsing, with intervals ranging from 2 to 20 seconds, capturing intensive or automated access patterns. Furthermore, the browsing script performs simple scrolling actions on each opened webpage, and randomly clicks on links to other websites.

Given the upload nature of the gateway traffic, we considered the case where upload-heavy users' connections might get misclassified. Thus, we take separate measures to ensure a reasonable portion of the traffic is upstream. Previous measurements [42] [43] [44] show that upload traffic can range from 6% to 8% Internet traffic. We thus aim for our upload traffic to be around this range. To simulate this, our emulator visits upload-specific websites with a 5% probability. To emulate diverse upload traffic, our system considers three distinct content types—images, video, and text—each modeled with a different dataset. We use the LAION-400 dataset [45], and the WebVid dataset [46] for image and video uploads, respectively. For text uploads, we use a randomly-generated string. Together, these datasets model a user engaging in different upload circumstances, creating a more comprehensive traffic dataset.

Limitations. While we dedicated serious effort in simulating browsing traffic to serve as background, one limitation is that the background data is synthetic and generated through emulated client behaviors rather than live residential traces. This design enables fine-grained manipulation of user activity

[2]We use v1.323.316

[3]We tried multiple other browsers and obtained similar results.

but inevitably abstracts away some real-world variability, such as heterogeneous device mixes and background application traffic. Importantly, co-capturing proxy and background flows on the same link and varying user profiles preserves the backconnect structure central to proxy providers.

Groundtruth. The collected traffic can be categorized into three groups: gateway traffic, which involves the communication between the RESIP node and the proxy gateways; relayed traffic, representing the traffic routed through the RESIP nodes as part of the relay process; and background traffic, simulating typical user browsing behavior. We analyze the captured PCAP network traffic using Zeek [47], with a signature-based detection method implemented to identify gateway traffic. This approach focuses on distinctive patterns in server domain names linked to Bright Data services. By compiling a detailed list of domain signatures, including domain names such as "Luminati," "brdtnet," and "lumtest," gateway-related traffic is accurately identified as associated with Bright Data proxy services. Once the proxy domain list is identified, the ssl.log generated by Zeek, which records parsed TLS traffic details, is used to extract key details such as server names, IP addresses, and connection statuses. These details enable the distinction between gateway connections and other traffic (background and relayed traffic). This gateway enumeration is used only for ground-truth labeling in our study; in realistic deployments, frequent gateway rotation and limited server-side visibility make such lists too brittle to serve as a reliable standalone detection baseline.

To distinguish relayed and background traffic, we use PCAPdroid [48] (v1.7.5), an application designed for network traffic monitoring and analysis on Android devices. We utilize PCAPdroid's root capture capability, which allows us to capture network traffic directly from the network interface. This approach provides us with a "raw" packet capture, collecting real packets as they appear on the network interface, along with the originating application (e.g. EarnApp or Chrome). We root the Android image running in the emulator using the rootAVD[4] tool (v21.4).

Collected traffic. Over a 15-month period from April 2024 to July 2025, we sporadically collected a total of 696 two-hour PCAPs, amounting to around 900 GB of network traffic. This dataset comprises of more than 120k gateway connections, 110k relayed connections, and around 6 million background connections. From this collection, we randomly selected 100 PCAPs to form the analysis dataset D_A, while the remaining PCAPs were used to construct dataset D_E. The experiments dataset D_E was randomly partitioned into training, validation, and test sets comprising 50%, 20%, and 30% of the data, respectively. Table I describes the collected dataset, and Table II summarizes the breakdown of the connections in D_E.

B. Operational Characteristics of EarnApp

For our analysis, we used the previously described D_A dataset. On average, each PCAP from D_A contains approximately 201 gateway connections and 110 relayed connections.

[4]https://gitlab.com/newbit/rootAVD

TABLE I: Summary of the collected RESIP datasets

Dataset	Description
D_A	This dataset is used for analysis. It consists of 100 two-hour PCAPs .
D_E	This dataset is used for experiments in RQ1 and RQ2. It consists of 596 two-hour PCAPs, including background, relayed, and gateway traffic.
D_S	This dataset is used for experiments in RQ3. It consists of direct and proxy-based traffic to 100 domains. The dataset contains 10,000 background connections and 1,643 relayed connections.

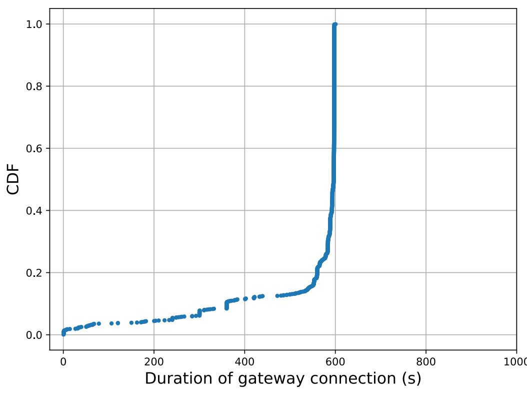


Fig. 2: CDF of gateway connection durations

This dataset provides a representative sample of network activity, allowing us to gain insights into proxy behavior.

Proxy gateways. Recall that when an EarnApp instance is deployed, it joins the network by first connecting to its gateway servers, awaiting any forwarded connection requests from network subscribers. We first extracted those gateway connections for analysis. We observed that Bright Data's proxy network consists of two types of gateway connections. The first category consists of short-lived connections, exhibiting low data transfers and short durations. They appear to primarily facilitate network management or bootstrapping. The destination domains are associated with Bright Data and include: `perr.h-cdn.com`, `perr.lum-sdk.io`, `perr.l-agent.me`. Many of these domains contain "brd" (Bright Data) and "lum-sdk". (Bright Data was previously known as Luminati Networks.) The second category of gateway connections appear to occur after the establishment of the first category and have longer durations and higher data transfers. By examining traffic patterns across multiple PCAPs from D_A, these appear to be used to upload data from the proxy node towards the subscriber. The domains associated with these data-transferring connections often follow patterns such as `[IP].brdtnet.com or [IP].luminatinet.com`).

Figure 2 presents the CDF of connection durations (in seconds) for the second category of (data-transferring) gateway connections. The CDF reveals distinct patterns in session behavior. The sharp increase near 600 seconds (10 minutes) indicates that the majority of data-transferring gateway con-

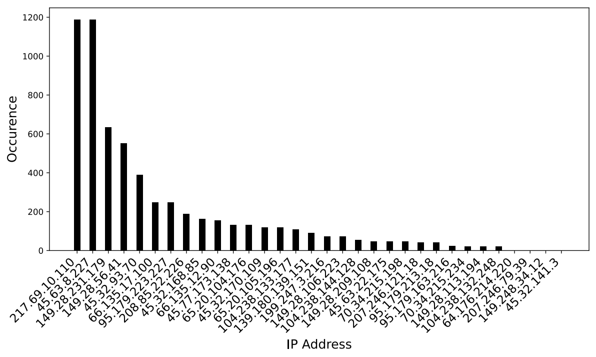


Fig. 3: Histogram of gateway IPs (used > 1)

nections (over 70%) adhere to a fixed session duration, likely enforced by predefined limits or time-based policies. This pattern suggests that most Bright Data connections remain active for around 10 minutes before being reset, and that Bright Data's infrastructure is configured to automatically rotate gateway IPs at the end of each 10-minute session. Furthermore, in the D_A dataset, we identified 72 unique (data-transferring) gateway IPs, 32 of which were reused across multiple sessions. Figure 3 shows the histogram of gateway IPs that appeared more than once. Some IPs were reused up to 1,200 times—while others were only reused a few times. This indicates a non-uniform reuse strategy, where certain gateway IPs are heavily favored.

Furthermore, we identified a pattern of periodic heartbeat or keep-alive packets exchanged between the data-transferring gateway and the proxy node. Following the TCP handshake and the data transfer, the connection remains open, during which small TLS-encrypted packets are exchanged approximately every 60 seconds. To determine their nature, we examined the TLS records for indications of standard heartbeat messages, typically indicated by TLS content type 24 (Heartbeat) as defined in RFC 6520. However, no such content type was found. In addition, we analyzed the traffic for TCP-level keepalive mechanisms, which involve the transmission of packets with the ACK flag set and zero-length payloads. Our analysis revealed no such packets. Given the absence of both TLS Heartbeat messages and TCP-level keepalive indicators, we infer that the observed periodic messages are likely part of a proprietary heartbeat or keepalive mechanism employed by the Bright Data protocol to maintain the persistent connection between the proxy gateway and node.

Furthermore, the analysis of collected traffic over a two-hour PCAP window reveals that proxy activity involves multiple simultaneous gateway connections. Figure 4 presents the CDF of the number of gateways opened within a 0.05-second window per PCAP. Our observations reveal that multiple gateway connections—typically ranging from three to six—are initiated within an extremely short time frame, often less than 0.05 seconds. Although these connections are established and active concurrently, not all of them relay data simultaneously. For instance, in cases where three gateway connections were established concurrently, only 58.6% were actively transmitting

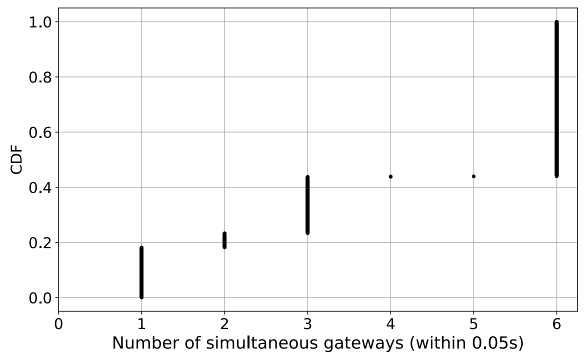


Fig. 4: CDF of the number of gateways opened within 0.05s window per PCAP

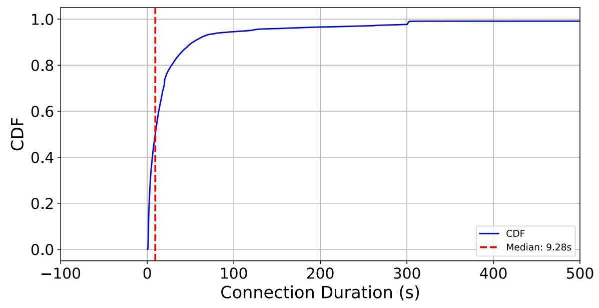


Fig. 5: CDF of relayed connection durations

data on average, while the remaining connections remained idle. This approach of initiating multiple gateway connections is likely employed to enhance the proxy system's reliability, facilitate load balancing, or enable shared utilization among multiple subscribers.

Relayed connections. Our analysis of relayed proxy connections across the D_A dataset reveals key characteristics of the network's behavior. We observed a total of 5,409 relayed connections, which accessed 246 unique domains, with an average of 54 relayed connections per PCAP. The most visited domains span various categories, including social media (e.g., TikTok, Instagram), shopping (e.g., Walmart, Kohls), and news websites, highlighting diverse online activities by proxy users. The average connection duration is 28.75 seconds. However, as shown in Figure 5, the median duration was approximately 9 seconds. The long tail shows that 10% of the connections last longer than 100 seconds. Additionally, the average volume of relayed connections was 62.89 KB. This combination of short duration and low volume suggests that most of these sessions involve smaller data transfers, such as page fetches or web scraping attempts.

Relayed domains. Following the footsteps of previous work [2], [5], we employed VirusTotal [49] (VT) to evaluate the maliciousness score of the domains visited by relayed connections through our deployed proxyware. We extracted all visited domains in datasets D_A and D_E (described in Table I). VT is a comprehensive online tool that aggregates

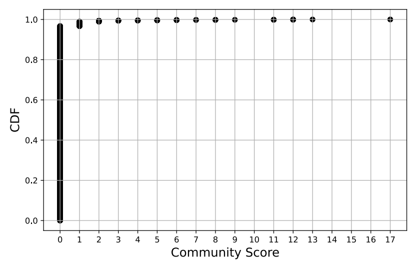


Fig. 6: CDF of the VT scores of relayed domain

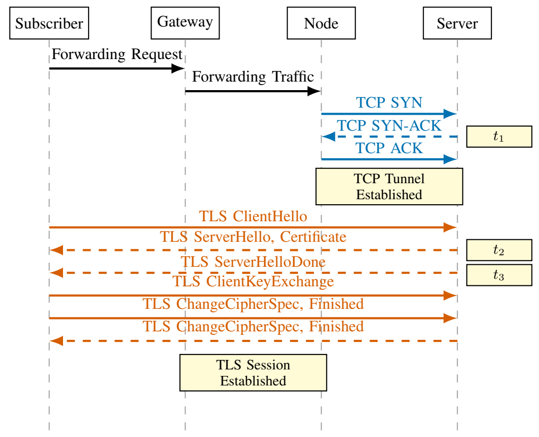


Fig. 7: TCP/TLS handshake sequence diagram illustrating the RTT differences across protocol layers in a RESIP connection. The TCP session terminates at the Node, while the TLS session extends end-to-end from Client to Server, creating a measurable timing discrepancy easily observable from RTT encapsulated in time differences such as $t_2 - t_1$ corresponding to the first two download packets.

data from multiple antivirus and security vendors, each of which may flag a domain as suspicious or malicious. The VT score is the number of vendors/engines that flagged a given domain as malicious/suspicious. Our analysis revealed that, out of 4,397 investigated relayed connections, 4.88% of the domains, i.e., 215 domains, were identified as suspicious, each having a community score of at least 1. Figure 6 shows the CDF of the VT community score of the relayed domains. The results indicate that 95% of the domains visited have a score of 0, while approximately 3% of the domains have a VT score of 1, suggesting a lower but notable suspicion level. 1% of the domains have a score higher than 3, suggesting a stronger likelihood of being linked to malicious activities. These findings are in alignment with previous research [18], where 2.16% of the domains were found to be suspicious when analyzed using VT.

V. Why RTT Features Alone Are Insufficient

Recall that previous works [19], [20] leveraged RTT as a side channel to fingerprint proxy communication. After understanding operational characteristics of the ecosystem, we show that RTT-based RESIP detection approaches are vulnerable to a simple adversarial attack and highlight the need for more robust features beyond RTT for our task. To demonstrate this feature, Figure 7 shows this characteristic of the RESIP architecture where the proxy node establishes the TCP connection with the server. The TLS connection, on the other hand, is end to end and extends to the proxy subscriber. Since the TCP handshake connection packets travel a shorter distance than TLS packets, looking at the timing features of these few initial packets of a connection enables the development of a simple but accurate detection scheme.

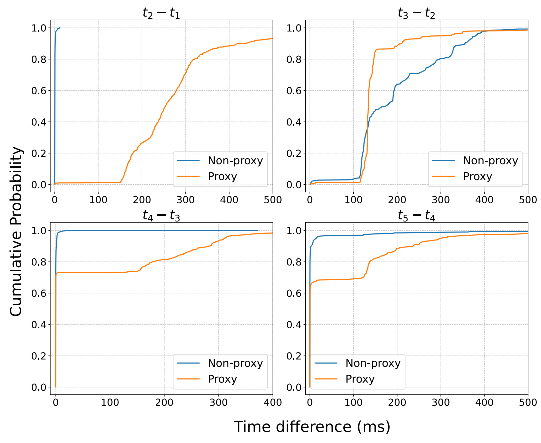


Fig. 8: CDF of inter-arrival time differences for download packets in proxied and non-proxied (direct) connections

Eight-packet classifier. Huang *et al.* propose a classifier that remarkably provides high accuracy performance using only the first eight packets[5]. After examination we observe that cross-layer RTT is implicitly calculated from their features. For each connection, Huang *et al.* calculate the difference between every two subsequent packets in both the upstream and downstream directions. Figure 8 shows the distribution of differences between every two subsequent packets in the downstream direction. For comparison, we added similar distributions for non proxy background traffic that we collected (Section IV-A). As can be seen in the figures, RTT appears to be manifested mostly in $t_2 - t_1$, which correspond to *TLS ServerHello* and *TCP SYN/ACK*, respectively, also shown in Figure 7.

Scheduling attack. RTT-based features utilizing the differences in TCP and TLS network traffic are susceptible to adversarial attacks [38]. Xue *et al.* [20] point out that traffic scheduling would fundamentally eliminate the cross-layer RTT discrepancy pattern exploited in previous work. We devised a simple adversarial approach in which a scheduler follows a fixed, predetermined timetable, regardless of the actual patterns of data being transmitted at the application layer [20]. In our case, this can be implemented by designing the proxy node to buffer the *TLS ClientHello* so that it can be sent to the server from the node as soon as the TCP tunnel is established instead of starting the transmission from the subscriber when the *TCP ACK* is sent.

[5]Huang *et al.* use four packets upstream and four packets downstream, which add up to 8 packets.

TABLE II: Summary of D_E, the dataset used for building RESIP usage detection models

Split	Background	Gateway	Relay	Total
Train	2,967,751	60,907	56,360	3,085,018
Val	1,286,045	25,398	20,129	1,331,572
Test	1,831,253	34,087	33,621	1,898,961
Total	6,085,049	120,392	110,110	6,315,551

We simulate this attack by sampling the distribution of the background traffic's timing differences between *TCP ACK* and *TLS ClientHello*. The scheduler at the proxy node then uses these sampled timings to determine when to forward the buffered *TLS ClientHello* message to the server. This approach emulates the timing characteristics of direct background traffic, effectively eliminating the discrepancy between TCP and TLS RTTs used by detection systems. This attack alone significantly drops the recall of the strongest classifier from Huang *et al.* (which uses up to 64 packets, not just the first 8) to a *mere 8% from the original 99%* on the D_E. Thus, while the model is effective in naive settings, its fatal flaw against scheduling makes it unreliable for real-world deployment.

VI. Classification Approaches

To address the limitations of RTT-based detection while meeting diverse operational requirements, we propose a novel classification framework. Our approach balances computational efficiency with detection robustness by offering two complementary strategies: (1) a light-weight approach using engineered traffic analysis and correlation features for rapid, large-scale filtering, and (2) a heavy-weight deep learning approach for maximum resilience in adversarial settings where accuracy is paramount. In practice, the lightweight classifier serves as the primary large-scale filter, while the heavy-weight CorrTransform is invoked selectively for deeper analysis of suspicious or high-risk cases (e.g. fraud or crime confirmation), ensuring that computational cost scales only with the number of uncertain cases rather than overall traffic volume.

A. Light-weight detection

For scenarios requiring fast, large-scale detection with limited resources, we develop a computationally efficient approach using carefully engineered features. To enable rapid processing, we extract features from only the first $n = 20$ packets of each connection, allowing for early classification without waiting for the full connection. Our light-weight approach combines two complementary feature sets: traffic analysis features that capture statistical patterns in packet timing and sizes, and novel correlation features that quantify the relationship between gateway and relayed streams.

Traffic Analysis features (TA features). We introduce twelve connection-level features computed over the first *n* exchanged packets. These include the inter-connection time gaps, cross-layer RTT, and statistical summaries, such as the mean, median, and mode, of sent and received traffic volume. Additionally, we also include the duration computed over the whole connection. We combine our features with other standard

Algorithm 1 Correlation Features

```
Input: $connections$        ▷ Array of connections to classify
       $gatewayConns$       ▷ Gateway traffic
Output: Correlation Features $C$ array   ▷ Correlation-based features
1:  for all $conn \in connections$ do
2:      // Get connection time range
3:      $start \leftarrow$ GET_CONN_START($conn$)
4:      $end \leftarrow$ GET_CONN_END($conn$)
5:      $gwTraffic \leftarrow$ Filter gatewayConns
        with timestamp $\in [start, end]$
6:
7:      // Bin the connection and gateway traffic based on volume
8:      $binWidth \leftarrow 0.1\,s$
9:      $connBins \leftarrow$ GET_VOL_BINS($conn, binWidth$)
10:     $gwBins \leftarrow$ GET_VOL_BINS($gwTraffic, binWidth$)
11:
12:     // Normalize the binned data using z-score
13:     $connNorm \leftarrow$ ZSCORE($connBins$)
14:     $gwNorm \leftarrow$ ZSCORE($gwBins$)
15:
16:     $sim \leftarrow$ ELEM_WISE_MULT($connNorm, gwNorm$)
17:     $C \leftarrow C \cup$ GET_METRICS($sim$)   ▷ Get mean, median, etc.
18: end for
```

traffic analysis features proposed by Hayes et al. [34]. To enhance the discriminative power of our features, we exclude features with zero standard deviation, as they do not contribute meaningfully to classification. This filtering process decreases the Hayes et al. [34] features down to 71. We categorize and define our TA-features below. For more details, Table VI in Appendix A lists the exact features within each category.

- Time-based features: These capture temporal patterns in packet transmission.
- Volume-based features: These represent the size-related characteristics of transmitted packets, capturing the overall intensity and distribution of data flow within a connection.
- Structural features: These provide additional insights into the network traffic behavior, capturing packet alternation and bidirectional flow properties.

Flow correlation features (Corr features). We expand on the previous set of features by introducing novel correlation-based features. These features build on the intuition that gateway traffic and relayed traffic must be correlated, as gateway traffic is simply forwarding relayed traffic. The algorithm is described in more detail at Algorithm 1. We elaborate on the algorithm below.

We first considered a naive per-packet correlation approach. Specifically, we attempted to match the lengths of incoming packets from a source (e.g., x.com) directly with corresponding outgoing packets on the gateway server. However, this proves unfruitful as the proxy framework splits incoming traffic across multiple packets, yielding low correlation.

We then study a different approach, where correlation is done *per time-bin* instead of per packet. Namely, for a given connection, we divide its traffic flow into time bins (e.g., 0.1 seconds), each bin characterized by its total volume of data.

During this interval, we also extract all gateway connections that fall within the same time range. Both the relayed and gateway traffic streams are binned using the same width and transformed into sequences of volume values. To enable meaningful comparison, we normalize both sequences using z-score normalization. We then compute the element-wise product of the normalized sequences, producing a similarity vector that captures how closely the gateway and relayed traffic align over time. From this vector, we extract statistical features, such as the mean and median similarity, which are then used as correlation features. We observe that a per-time-bin approach yields much better results than a per-packet one, resulting in a 81.2% F1 score and a 2.1% FPR with correlation features alone. We experimentally find 0.1 seconds to be the best size for our time bins. We also considered the case where there may be some lag between the two sides; we tried different orders of magnitude (0.01s, 0.1s, 1s) of lags, and for each we tried different ranges. While some specific instances of correlation benefited, we concluded that this misalignment on the aggregate was minimal.

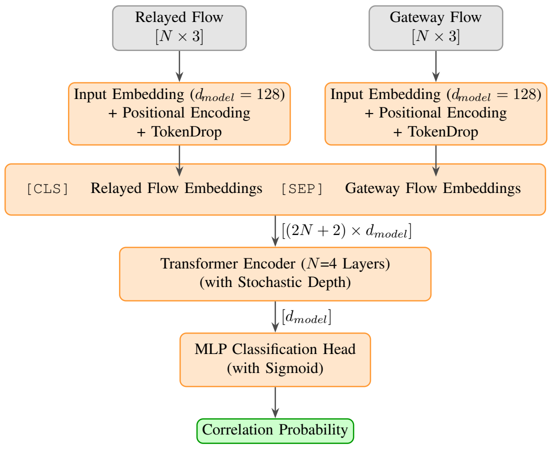
For our classification tasks where the previous traffic analysis or flow correlation features are used, we primarily employ XGBoost due to its efficiency, interpretability, and strong performance on network traffic features. For RQ3's server-side detection scenario, where content providers require maximum detection capability with a more limited feature set, we additionally validate our approach using AutoGluon's ensemble methods [50].

B. Heavy-weight detection

While the feature-engineering methods described above provide an excellent balance of performance and efficiency for most detection scenarios, high-stakes environments with sophisticated adversaries may require additional robustness. In cases where attackers have detailed knowledge of detection systems and can deploy advanced evasion techniques, a more resilient approach becomes necessary. For these critical scenarios where maximum detection accuracy is paramount and computational resources are available, we introduce a heavy-weight classification strategy that complements our light-weight approach.

This heavy-weight approach leverages deep learning to automatically discover complex, non-obvious patterns that are difficult for adversaries to identify and manipulate. Rather than relying on specific engineered features that can be targeted, our approach learns from raw traffic sequences to capture subtle, long-range dependencies between packets. For these models, we use the first $n = 50$ packets of a connection, providing richer context essential for identifying nuanced correlations that remain robust under adversarial conditions.

We present ***CorrTransform***, a novel Transformer-based architecture designed to classify whether relayed and gateway flows are correlated. We chose a Transformer because its self-attention mechanism is uniquely suited to identify the subtle, non-local dependencies between packets—a characteristic we



Relayed Flow $[N \times 3]$
Gateway Flow $[N \times 3]$
Input Embedding ($d_{model} = 128$) + Positional Encoding + TokenDrop
Input Embedding ($d_{model} = 128$) + Positional Encoding + TokenDrop
[CLS] Relayed Flow Embeddings [SEP] Gateway Flow Embeddings
$[(2N+2) \times d_{model}]$
Transformer Encoder (N=4 Layers) (with Stochastic Depth)
$[d_{model}]$
MLP Classification Head (with Sigmoid)
Correlation Probability

Fig. 9: The architecture of our proposed CorrTransform model.

hypothesize is critical for robust classification, especially in adversarial settings.

Input Representation. For both deep learning models, each connection is transformed into a sequence of feature vectors. Each packet in the sequence is represented by a vector containing three features: (i) packet size, (ii) inter-arrival time, and (iii) packet direction.

CorrTransform Architecture. The architecture of CorrTransform, illustrated in Figure 9, processes the two flows by concatenating them into a single input sequence. This sequence is structured as: [CLS] [relay_flow] [SEP] [gateway_flow], where the 'CLS' and 'SEP' tokens serve as classification and separator markers, respectively.

Each packet in the input is first projected into a 128-dimensional embedding space, and we add sinusoidal positional encodings to inform the model of the packet order. The core of the model is a 4-layer Transformer encoder, with each layer containing 4 self-attention heads. To enhance robustness, we employ two distinct regularization techniques: Stochastic Depth [51], which randomly bypasses entire encoder layers during training to improve gradient flow, and TokenDrop, which randomly nullifies entire packet embeddings, forcing the model to learn from the global context rather than over-relying on specific local features.

C. Attacks

To assess robustness, we evaluate the models under normal conditions and four adversarial attack scenarios designed to mimic evasion techniques described below, along with their costs:

1) **Scheduling:** This timing attack, detailed in Section V, disrupts RTT-based features by manipulating packet schedules to mimic benign traffic, directly targeting the weakness of the SLT baseline. As this attack only delays a single handshake packet, its impact on the user is minimal.
2) **Targeted Padding:** This attack alters the packet size distribution by adding padding to influential initial packets, aiming to morph the traffic's statistical signature to evade size-based detection [52]. However, the padding leads to

increased data usage and decreased goodput for the user, becoming especially noticeable in cases where the uplink is the bottleneck.

3) **Packet Reshaping:** This structural attack leverages the proxy's control to randomly split larger packets into smaller ones, fundamentally altering packet-length sequences to poison size, count, and sequence-based features [33]. This increases per-flow overhead by inflating packet counts and header processing, which can degrade goodput and latency on constrained links.
4) **Inter-Packet Delay (IPD) Jitter:** This attack introduces small, random delays between packets to distort the flow's timing rhythm, targeting classifiers sensitive to inter-packet delay patterns. This has an obvious impact on latency and jitter, becoming especially noticeable for uses like streaming and video games

VII. Experiments: ISP & Server Detection

A. RQ1: Can ISPs detect devices with RESIP activities?

In this section, we use the D_E dataset to investigate an ISP's capability to detect RESIP traffic by identifying its gateway connections using connection-level features. We demonstrate a two different classification strategies: a simple binary classifier to robustly detect the presence of proxyware, and a more granular multiclass classifier to show the challenges that motivate RQ2.

Binary Classification Approach. We first investigate a binary classifier to reliably identify the presence of proxyware on a node's device by distinguishing its unique gateway connections from all other traffic (a combined class of background and relayed connections). This initial filtering step is inherently robust because gateway connections possess fundamental operational characteristics (e.g., high upload volumes, long durations) that are difficult for an adversary to alter without crippling the proxy service itself. Attempts to apply attacks like target padding or packet reshaping are ineffective in hiding these characteristics. As such, we evaluate this model under normal, non-adversarial conditions to establish its baseline effectiveness.

Multiclass Classification Approach. Next, we develop a multiclass classifier that distinguishes between background, gateway, and relay connections to establish a baseline for a more granular, all-in-one classification model. However, this finer-grained approach exposes vulnerabilities. We demonstrate this by applying the simple scheduling attack (detailed in Section V) to the multiclass model. This attack specifically targets timing-based features to make relayed traffic indistinguishable from benign traffic, testing the model's resilience.

Training. We train XGBoost models using the D_E dataset. For feature extraction, we chose the first $n = 20$ packets, as we did not observe noticeable performance benefits for larger values of n. A smaller value of n allows for an earlier, quicker, and more practical classification, which can help ISPs filter out devices running proxies. We use all the timing, volume, and structural features described in Table VI, including cross-layer RTT. Note that the duration feature is computed at the connection level, independent of n, as we assume it is readily available in connection logs. It is also important to note that the correlation features are not applicable in this context as they require the identification of relayed and gateway traffic first to correlate against. We split the datasets into training (50%), validation (20%), and testing (30%). Gateway connections only represent ~2% of our traffic. This configuration is intentionally unbalanced to reflect real-world scenarios where gateway traffic is a small fraction of total traffic.

TABLE III: RQ1 classification performance

Dataset	Precision(%)	Recall(%)	FPR(%)
Gateway			
Multiclass	99.35	97.86	0.01
$\text{Multiclass}_{Attack}$	99.33	97.86	0.01
Binary	99.42	98.08	0.01
Relay			
Multiclass	97.00	93.84	0.05
$\text{Multiclass}_{Attack}$	95.78	65.83	4.22

Results. Table III shows the performance of our classification models. Our binary classifier proves highly effective, identifying gateway connections with 99.42% precision and 98.08% recall at a negligible 0.01% FPR. This confirms that an ISP can reliably detect devices running proxyware by focusing on the distinct footprint of gateway traffic.

The multiclass classifier also performs well under normal conditions, identifying gateway connections with 99.35% precision and 97.86% recall. However, the challenge of granular classification becomes apparent when detecting the relayed class, which achieves 97.00% precision and 93.84% recall.

This vulnerability is magnified under the scheduling attack. While gateway detection remains almost perfect, the recall for the relayed class plummets to 65.83%, with the FPR increasing significantly to 4.22%. This result is critical: it demonstrates that while we can easily tell which devices are running proxyware, simple adversarial timing attacks can severely hinder our ability to identify which specific connections are being relayed. This finding strongly motivates our two-tiered approach and highlights the need for the more robust, correlation-based features we explore in RQ2 to reliably identify relayed connections, especially in adversarial environments.

Permutation-based analysis reveals that the most important features for detecting gateways are related to the uploaded volume and the duration of the connection. Figure 10 visualizes these features, highlighting the distinct patterns of gateway traffic (high transfer volumes and long durations) compared to background and relayed traffic. Note that the dashed lines represent the background distribution partitioned into three equal thirds to show how they overlap with the relay, resulting in less accuracy for the relay class. Our results demonstrate that an ISP can easily identify EarnApp gateway connections, and can further increase detection confidence by setting a threshold for the number of gateway connections observed from a single device.

TABLE IV: RQ2 classification performance

Attack Setting	Features	Precision(%)	Recall(%)	F1(%)	FPR(%)
Without Attack	TA	97.76	89.68	93.55	0.03
	Corr	82.51	79.92	81.20	2.10
	TA + Corr	98.34	95.85	97.08	0.03
	CorrTransform	99.63	99.62	**99.62**	0.01
	DeepCoFFEA	94.42	92.52	93.46	0.10
Scheduling	TA	95.60	44.55	60.77	0.03
	Corr	62.30	30.98	41.39	2.40
	TA + Corr	97.44	59.69	74.03	0.02
	CorrTransform	99.50	85.57	**92.01**	0.01
	DeepCoFFEA	97.97	74.97	84.94	0.03
Targeted Padding	TA	96.32	53.64	68.90	0.03
	Corr	81.60	77.17	79.32	2.20
	TA + Corr	98.20	88.17	92.91	0.02
	CorrTransform	99.65	99.07	**99.36**	0.01
	DeepCoFFEA	97.46	89.07	93.08	0.04
Packet Reshaping	TA	89.55	17.32	29.03	0.03
	Corr	79.05	62.87	70.04	2.10
	TA + Corr	96.75	47.49	63.70	0.02
	CorrTransform	99.80	96.51	**98.12**	0.01
	DeepCoFFEA	96.56	89.61	93.00	0.06
Inter-Packet Delays	TA	97.89	91.14	94.35	0.03
	Corr	74.31	63.19	68.30	2.70
	TA + Corr	98.03	92.61	95.24	0.02
	CorrTransform	99.70	95.54	**97.58**	0.01
	DeepCoFFEA	96.97	85.05	90.62	0.05
Scheduling + Packet Reshaping	TA	87.08	19.64	32.05	0.05
	Corr	50.32	19.55	28.16	0.30
	TA + Corr	95.20	24.91	39.48	0.02
	CorrTransform	99.87	62.98	77.24	0.01
	DeepCoFFEA	99.01	82.88	**90.23**	0.001
Scheduling + Inter-Packet Delays	**TA**	96.45	80.38	**87.68**	0.05
	Corr	57.95	27.02	36.85	0.30
	TA + Corr	98.28	72.97	83.75	0.02
	CorrTransform	99.67	75.57	85.96	0.01
	DeepCoFFEA	93.21	70.43	80.23	0.01
Targeted Padding + Packet Reshaping	TA	93.92	45.04	60.88	0.05
	Corr	77.88	67.82	72.50	0.30
	TA + Corr	98.29	72.14	83.21	0.02
	CorrTransform	99.39	98.77	**99.08**	0.001
	DeepCoFFEA	95.83	60.66	74.30	0.05
Packet Reshaping + Inter-Packet Delays	TA	95.43	60.82	74.29	0.05
	Corr	72.15	55.99	63.05	0.40
	TA + Corr	98.46	76.20	85.91	0.02
	CorrTransform	99.51	90.42	**94.75**	0.001
	DeepCoFFEA	98.36	65.66	78.75	0.02

B. RQ2: Can an ISP identify relayed connections?

Following the identification of devices potentially running proxyware via gateway connection detection (RQ1), the next critical step for an ISP (the second tier of our approach) is to determine the nature of other connections originating from or passing through such devices. Specifically, RQ2 addresses whether an ISP, with its network-wide visibility, can distinguish between connections directly initiated by the user on the device and connections being relayed through the device for external proxy subscribers. Accurately identifying relayed connections is crucial for attributing network activity correctly and detecting potential misuse of the user's bandwidth for activities like scraping, fraud, or other malicious purposes.

To address this question, we frame the problem as a binary classification task: classifying a given connection observed by the ISP as either *Relayed* or *Background*. First, we test the light-weight approach, which uses an XGBoost model on our engineered TA and Corr features from the first 20 packets. Second, we assess the heavy-weight approach, which employs our novel CorrTransform and the DeepCoFFEA baseline on the first 50 packets.

We use the dataset D_E (Table II) to train models to distinguish 'Relayed' from 'Background' connections. We evaluate four set of features: our proposed TA features, our novel Corr features, and a combination of both (TA + Corr). We further evaluate two DL-based correlation approaches: our novel CorrTransform, and a state-of-the-art baseline; DeepCoFFEA [24]. DeepCoFFEA is a CNN-based model that has demonstrated strong performance in the related problem of correlating Tor traffic. By modifying it for our RESIP detection context, we establish a robust baseline for comparison.

To further assess robustness, we evaluate single-attack settings (Scheduling, Targeted Padding, Packet Reshaping, and Inter-Packet Delay), and include four representative two-attack combinations. For completeness, the full matrix of pairwise,

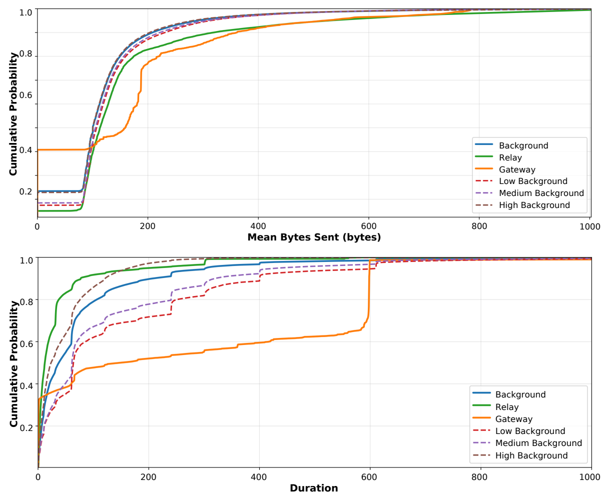


Fig. 10: CDF of the two most important features for gateway classification

three-way, and four-way combinations appears in Appendix B.

Results. The classification performance across all models and scenarios is detailed in Table IV. Under normal conditions (Without Attack), all feature-driven methods prove highly effective. The feature-based TA + Corr model reaches a 97.08% F1 score, while the deep learning models CorrTransform and DeepCoFFEA achieve strong 99.62% and 93.46% F1 scores, respectively, demonstrating their capability in a non-adversarial setting.

The central story, however, unfolds under adversarial pressure. Recall the scheduling attack introduced in Section V, designed to disrupt RTT features, which caused the classifier presented by Huang *et al.* to collapse. Our own TA features are similarly impacted, with recall dropping to 44.55%. In contrast, the introduction of our novel correlation features provides a dramatic improvement. The TA + Corr model demonstrates significant resilience by maintaining a 74.03% F1 score, a substantial recovery compared to the TA-only model. While CorrTransform proves to be the most robust overall solution, sustaining an impressive 92.01% F1 score, DeepCoFFEA also demonstrates robustness, maintaining an F1 score of 84.94% and indicating it is less reliant on the RTT feature. The Packet Reshaping attack devastates the TA model (recall drops to 17.3%), but CorrTransform remains highly effective with a 98.12% F1 score. This pattern of the CorrTransform's robustness continues across the other adversarial scenarios. Similarly, against Targeted Padding and Inter-Packet Delays, CorrTransform consistently outperforms the baseline and our feature-based methods, achieving F1 scores of 99.36% and 97.58%, respectively. This highlights the architecture's ability to learn deeper, non-local correlations that are independent of any single, engineered feature. Instead, the model learns a more holistic and redundant feature representation, making it highly effective against a diverse set of attacks.

In the combined attacks scenario, CorrTransform continues to provide the most consistent performance across the different combinations; it remains above 80% in nearly all combined settings (see further results in Appendix B). However, for the Scheduling + Packet Reshaping combined attacks, we notice a dip in recall, affecting the overall F1 to 77.24%, suggesting a plausible mitigation strategy against CorrTransform. DeepCoFFEA does better in this attack scenario with an F1 of 90.23%. This reflects a difference in how each model learns. CorrTransform captures complex patterns across long packet sequences, a strength this specific attack disrupts. DeepCoFFEA's simpler, more local approach happens to be more robust here. However, CorrTransform still remains the more reliable approach, offering more consistency and better detection performance overall.

While DeepCoFFEA is effective in some settings, it falls short in most others, particularly against Scheduling, and consistently underperforms against CorrTransform across all single-attack settings. Its performance also degrades under several combined attacks, such as Targeted Padding + Packet Reshaping, where its performance falls below 75% F1 score. Notably, our engineered TA+Corr features prove more effective than the DeepCoFFEA deep learning baseline against both the Without Attack and Inter-Packet Delay attack settings, highlighting the power of our carefully designed traffic analysis and correlation metrics. Across all scenarios, CorrTransform provides the most consistent and robust defense.

Takeaways. The results in Table IV clearly illustrate the strengths of each strategy. The light-weight TA+Corr model provides a remarkable balance of performance and efficiency, making it a practical choice for large-scale filtering. However, when faced with sophisticated attacks like Packet Reshaping, its limitations become apparent. In contrast, the heavy-weight CorrTransform demonstrates superior resilience and consistency across diverse adversarial scenarios, justifying its additional computational cost for use cases where robustness is non-negotiable.

C. RQ3: Can a server distinguish between relayed and direct connections?

The goal of our server-side analysis is to test whether lightweight, single-ended features can distinguish relayed from direct connections in a realistic deployment with severe class imbalance. At this vantage point, detection relies on single-ended timing summaries and aggregate flow statistics; pushing these enough to evade typically requires aggressive traffic shaping that materially degrades throughput and user experience. Our adversarial analysis in RQ1 (Table III) already shows how scheduling-style attacks can affect timing-based features at the ISP vantage, and we expect analogous effects at the server. In this section, we therefore focus the server-side evaluation on baseline separability and robustness under class imbalance, reflecting realistic classification scenarios.

From a server's perspective, distinguishing between *direct* and *relayed* connections is crucial for maintaining the integrity of online services. Platforms such as YouTube rely on accurate user engagement metrics for recommendations, ad revenue, and content moderation. However, RESIPs can be used to manipulate these systems—for instance, artificially inflating views, likes, or engagement by masking the true origin of traffic. To mitigate such risks, servers seek to identify whether incoming traffic originates directly from users or is relayed through a residential proxy. In this section, we investigate whether a server can reliably distinguish between relayed and originated connection from a device by analyzing the network traffic patterns.

Dataset. To investigate whether a server can distinguish between relayed and directly originated connections, we construct a new dataset D_S composed of two types of traffic extracted and collected as follows. For relayed traffic, we use the experimental dataset D_E introduced in Section IV-A. To ensure the quality of the domain set, we removed suspicious entries by filtering out domains with a VT score greater than 3 [6]. From the remaining domains, we selected 100 domains to create the desired dataset. For relayed traffic, we filter the original proxy traffic, retaining only the connections targeting the selected 100 domains.

For direct traffic, we apply the same methodology and testbed described in Section IV-A. We collect direct connections traffic by visiting the selected 100 domains, i.e., without routing through a residential proxy, using the browsing script, capturing realistic traffic that includes scrolls, clicks, and searches. The captured background PCAP traffic is analyzed using Zeek to extract key network attributes, including server names, IP addresses, and port information.

Training. We train an AutoGluon classifier to distinguish between background and relayed connections using a D_S consisting of 10,000 background connections and 1,643 relayed connections. Unlike the ISP scenarios where observers have full network visibility and access to correlation features, servers must rely solely on features extracted from incoming connections. We extract a comprehensive set of features, detailed in Section VI-A, from the first 20 packets exchanged. Notably, correlation features are unavailable in this scenario as servers cannot observe gateway traffic. After removing features with zero standard deviation, we obtain a refined feature set of 83 features. AutoGluon's ensemble approach helps maximize detection performance despite this constrained feature space by combining multiple models that capture different traffic characteristics.

Subsequently, we divide the dataset into training (50%), validation (20%), and testing (30%) subsets. To address the class imbalance, wherein background connections substantially outnumber relayed connections, we implement a downsampling strategy. This limits the majority class (background) to four times the size of the minority class (relayed), yielding a balanced training set consisting of 3,944 background connections and 986 relayed connections.

Results. The model's performance on the entire test dataset D_S^{test}, which consists of 328 relayed and 2,000 background connections, is as follows: a precision of 98.76% and a recall of 98.15%. The FPR is consistently low at 0.2%. These results indicate that the classifier performs well overall, accurately distinguishing between relayed and direct connections with minimal misclassification of background traffic. Using AutoGluon's permutation-based feature importance, the results indicate that the most important features are time related, with the top contributors being the 25th percentile of outbound traffic (i.e., the time by which 25% of outbound packets have been exchanged), the 25th percentile of the time taken for inbound traffic, the average interarrival time of inbound packets, the 75th percentile of total traffic, and the 75th percentile of outbound traffic. Note that these features are well-established in traffic analysis and work well in the context of RESIP.

[6]Previous work utilizing VT such as [53] often considers domains malicious if their VT score is larger than 3. Domains with a VT score less than 3 are often false positives.

TABLE V: RQ3 classification performance

Dataset	Precision(%)	Recall(%)	FPR(%)
$D_S^{test}5$	96.07	98.0	0.2
$D_S^{test}3$	93.65	98.33	0.2
$D_S^{test}1$	83.33	100	0.2
$D_S^{test}0.5$	71.42	100	0.2

To evaluate the classifier's robustness under different levels of class imbalance, we construct four test datasets with varying relayed-to-background ratios: 5%, 3%, 1%, and 0.5%, denoted as $D_S^{test}5$, $D_S^{test}3$, $D_S^{test}1$, and $D_S^{test}0.5$, respectively. These datasets consist of 2,000 background samples and varying numbers of relayed samples (100, 60, 20, and 10 for the respective ratios). The classification performance across the different test datasets is presented in Table V. The model demonstrates strong overall performance in distinguishing between direct and relayed connections, with high precision, high recall, and a consistently low FPR across all test datasets. However, as the proportion of relayed traffic decreases, precision decreases. This decline in precision is not due to an increase in false positives, as the FPR remains constant across all datasets, but rather a result of the decrease in true positives as fewer relayed connections are present in the datasets. Furthermore, even when precision drops to 71.42% in $D_S^{test}0.5$, recall is 100%, meaning all relayed instances are indeed being identified correctly as relayed.

VIII. Limitations and Future Work

While our study provides significant insights into the characteristics and detection of RESIP traffic and introduces robust correlation-based features resilient to adversarial timing perturbation attacks, it is subject to certain limitations that open avenues for further research. First, we restricted data collection to Bright Data and did not test against other providers that may use different protocols, such as a control API for gateway domain discovery. However, our features and models are not specific to those details, but rather target provider-agnostic invariants: (i) persistent encrypted tunnels between node and gateway, (ii) concurrent gateway and relayed flows during sessions. Reported cross-provider differences (e.g., fixed gateway domain vs. control-API discovery) [18] should not affect these invariants. Practically, porting requires only updating gateway discovery heuristics and applying our feature extraction pipeline to the newly extracted traffic; the TA/Corr feature sets and the CorrTransform architecture may require adaptation, but the high-level approach remains unchanged. Empirical validation on other providers is left for future work.

We demonstrated the vulnerability of widely used RTT features to basic scheduling attacks and assessed our features against different perturbations, but critically examining the trade-off between their effectiveness and the potential hit to service utility (e.g., latency), is an important next step. Our data collection was geographically localized, meaning variations in network conditions (latency, jitter) globally could impact the performance of timing-dependent features; evaluating generalizability across diverse network environments is needed. Beyond technical detection, a significant challenge remains in distinguishing willing RESIP hosts from users who unwittingly become victims of *proxyjacking* [54], often through bundled SDKs, necessitating research into behavioral or network indicators of nonconsensual participation. Additionally, exploring robust, privacy-preserving features that RESIP providers themselves could use to detect malicious usage internally represents a promising avenue for enhancing platform integrity. Finally, as with machine-learning-based approaches, our models are prone to concept drifts and will require periodic retraining. However, continuous learning approaches can be used to reduce retraining costs.

IX. Conclusion

This paper addressed a critical vulnerability in residential IP proxy detection: the fundamental weakness of existing RTT-based methods against simple adversarial attacks. We demonstrated that state-of-the-art detection techniques, which rely on cross-layer timing discrepancies, can be trivially defeated through basic traffic scheduling, reducing detection accuracy from 99% to just 8%. To overcome this limitation, we introduced a robust detection framework based on novel correlation features that exploit the intrinsic relationship between gateway and relayed traffic streams, a fundamental architectural property that adversaries cannot easily manipulate without degrading service quality.

Our comprehensive evaluation using 15 months of data from the world's largest RESIP network validates the practical effectiveness of our approach. The results demonstrate that robust proxy detection is achievable across diverse operational scenarios: ISPs can reliably identify both proxyware devices and individual relayed connections, while content providers can distinguish direct user traffic from proxy connections with near-perfect accuracy. Critically, our correlation-based methods maintain strong performance even when adversaries deploy sophisticated evasion techniques, providing the reliability needed for real-world deployment. CorrTransform further showcases how deep learning architectures can be designed for inherent robustness against adversarial manipulation.

Beyond technical contributions, this work shifts the proxy detection paradigm from vulnerable timing-based approaches to resilient architectural fingerprinting. Our findings provide immediately deployable tools for ISPs and content providers to combat the growing threat of malicious RESIP usage, from click fraud and web scraping to sophisticated cyberattacks. As residential proxy networks continue to expand and evolve, the robust detection capabilities presented here offer a critical foundation for maintaining security and accountability in an increasingly proxy-mediated internet ecosystem.

X. Ethical Considerations

Since we deploy proxyware and perform traffic analysis on its connections, it is important to ensure we are not compromising users' privacy or data. In terms of anonymity, the proxies we deployed forward HTTPS requests to domains requested by subscribers of the proxyware system. While the destination servers appear in our PCAPs, the subscriber IP is not visible to us (as requests come through gateways), nor is their traffic or interaction with the destination server, since the traffic is encrypted and the TLS connection is established directly between the subscriber and the destination server. The data extracted from our setup is stored in a secure server within our institution and is only accessible by the research team cleared and approved by our Institutional Review Board (IRB). As for data processing, all processing is done to derive useful features based on packet timings and sizes. Although we query VirusTotal for destination addresses to understand malicious use, this is generally accepted and has been done previously in related work [2], [5], [18].

Another potential concern is that our deployed proxy nodes could be misused by attackers for spamming or other malicious activities. Indeed, we have observed access to certain malicious domains, which may indicate malware-related activity. However, similar observations have been reported in prior research [2], [5], [18], where proxies were deployed and analyzed to better understand this ecosystem and contribute findings to the research community. We have not received any complaints during our data collection, and we acknowledge that such risks are inherent in any proxy deployment study. Nonetheless, we believe that the insights gained from our research far outweigh these limited potential risks.

One concern shared with much of the existing traffic analysis literature (Section II-B) is that our work could be misused by ISPs to censor proxy traffic, which is not our intended purpose. We advocate for advancing traffic analysis as a defensive tool to combat fraudulent and suspicious activities that impact both content providers, ISPs, and proxy operators, who can face legal risks in case their proxies were used for illicit activities. Our work can also help users detect non-consensual residential proxy traffic on their devices. By enhancing security and accountability, our approach aims to create a safer environment for legitimate users while also offering responsible proxy operators a layer of protection should their infrastructure be exploited for malicious purposes.

Acknowledgment

This work was supported by the Qatar Research, Development and Innovation Council (QRDI) under Academic

Research Grant ARG01-0525-230338.

Appendix

A. *Feature Breakdown for Traffic Analysis (TA)*

Table VI lists the complete set of features used for TA.

TABLE VI: Network traffic features used for classification

Feature Category	Category-Based Features
Time-Based Features	Connection duration, round-trip time (RTT), inter-packet arrival time(max, avg, std, 75th percentile) for inbound, outbound, and total traffic, time percentiles stats (25th, 50th, 75th, and 100th) for inbound, outbound, and total traffic, average and standard deviation of packet ordering, packet transmission rate (average, median, min, and max packets per second), statistics for the first and last 30 packet.
Volume-Based Features	Mean, median, and mode of total traffic volume, mean, median, and mode of bytes sent and received traffic.
Structural Features	Packet concentration metrics (average and standard deviation over 20 packet windows), Median and maximum packet concentration, summation of total packets, percentage of incoming and outgoing packets, sum of alternating packet concentration, and packet rate.

B. *Additional Results*

We expand on the analysis in Table VII with six combined-attack settings, extending beyond the single and four representative cases in the main text. Across these broader combinations, CorrTransform continues to show the highest and most stable performance, while TA+Corr remains a strong lightweight option that consistently improves over TA under timing distortion. Corr alone degrades when both timing and structure are altered, and the CNN baseline is occasionally competitive in specific regimes.

When multiple attacks are layered, detection sometimes improves rather than weakens. We suspect this occurs because stacked perturbations introduce unintended regularities or inconsistencies that models can exploit, such as abnormal burst patterns or exaggerated packet-timing differences. For example, adding IPD jitter can make TA's non-correlational features more discriminative by amplifying inter-arrival gaps that differ between relayed and background flows. Similarly, for deep models, heavy combinations may push traffic further from the benign patterns they were trained to distinguish, effectively making anomalies more visible. These are likely contributing factors rather than confirmed mechanisms, but they align with the observed increase in separability under more complex attack mixes.

TABLE VII: RQ2 classification performance — Combined Attack Settings

Attack Setting	Features	Precision(%)	Recall(%)	F1 (%)	FPR(%)
SC + TP	TA	94.82	54.17	68.95	0.05
	Corr	54.49	23.53	32.86	0.3
	TA + Corr	97.76	55.77	71.02	0.02
	CorrTransform	99.7	72.77	84.13	0.01
	DeepCoFFEA	97.74	77.85	**86.67**	0.03
TP + IPD	TA	96.89	92.14	94.45	0.05
	Corr	71.91	55.85	62.87	0.4
	TA + Corr	98.68	93.39	95.96	0.02
	CorrTransform	99.67	95.51	**97.54**	0.01
	DeepCoFFEA	98.19	71.51	82.75	0.02
SC + TP + PR	TA	85.85	17.69	29.33	0.05
	Corr	50.20	19.48	28.06	0.3
	TA + Corr	94.76	22.71	36.63	0.02
	CorrTransform	98.84	74.07	**84.68**	0.01
	DeepCoFFEA	95.41	65.87	77.93	0.06
SC+ TP + IPD	TA	95.89	69.05	80.29	0.05
	Corr	57.32	26.36	36.11	0.3
	TA + Corr	98.16	67.93	80.29	0.02
	CorrTransform	99.32	84.40	91.26	0.01
	DeepCoFFEA	95.59	94.77	**95.18**	0.08
SC + PR + IPD	TA	92.23	34.59	50.31	0.05
	Corr	56.60	25.22	34.89	0.3
	TA + Corr	96.46	34.22	50.52	0.02
	CorrTransform	98.99	76.34	**86.20**	0.01
	DeepCoFFEA	96.64	60.99	74.78	0.04
TP + PR + IPD	TA	95.09	56.49	70.87	0.05
	Corr	53.12	71.14	60.82	0.4
	TA + Corr	98.31	73.27	83.97	0.02
	CorrTransform	85.57	99.66	**92.08**	0.01
	DeepCoFFEA	96.82	73.09	83.29	0.04
SC + TP + PR + IPD	TA	90.96	29.32	44.35	0.05
	Corr	55.78	24.36	33.92	0.3
	TA + Corr	96.18	31.64	47.61	0.02
	CorrTransform	98.80	78.79	87.66	0.01
	DeepCoFFEA	98.64	89.11	**93.63**	0.02

Legend: SC = Scheduling; TP = Targeted Padding; PR = Packet Reshaping; IPD = Inter-Packet Delays.

References

[1] "Bright Data: How the world collects public web data," https://brightdata.com, 2025.

[2] X. Mi, S. Tang, Z. Li, X. Liao, F. Qian, and X. Wang, "Your phone is my proxy: Detecting and understanding mobile proxy networks," in *Network and Distributed System Security Symposium (NDSS)*, 2021.

[3] "Oxylabs:Easy Access to Web Data at Scale ," https://oxylabs.io, 2025.

[4] E. Khan, E. Chiapponi, M. Verkleij, A. Sperotto, R. Van Rijswijk-Deij, and J. Van Der Ham-De Vos, "A first look at user-installed residential proxies from a network operator's perspective," in *2024 20th International Conference on Network and Service Management (CNSM)*, 2024, pp. 1–9.

[5] X. Mi, X. Feng, X. Liao, B. Liu, X. Wang, F. Qian, Z. Li, S. Alrwais, L. Sun, and Y. Liu, "Resident evil: Understanding residential ip proxy as a dark service," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1185–1201.

[6] D. Jones, "Midnight blizzard attack seen as another sign of microsoft falling short on security," https://www.cybersecuritydive.com/news/midnight-blizzard-hack-microsoft-security/705416/, 2024.

[7] P. Paganini, "ISRAELI FIRM 'BRIGHT DATA' (LUMINATI NETWORKS) ENABLED THE ATTACKS AGAINST KARAPATAN," https://securityaffairs.com/121641/hacking/luminati-networks-ddos-karapatan.html, 2021.

[8] R. Lakshmanan, "U.S. Dismantles World's Largest 911 S5 Botnet with 19 Million Infected Devices," https://thehackernews.com/2024/05/us-dismantles-worlds-largest-911-s5.html, 2024.

[9] ABCproxy, "Unveiling the Proxy Click Fraud: How to Detect and Prevent It," https://www.abcproxy.com/blog/unveiling-the-proxy-click-fraud--how-to-detect-and-prevent-it.html, 2024.

[10] E. Montalbano, "'Proxyjacking' Cybercriminals Exploit Log4j in Emerging, Lucrative Cloud Attacks," https://www.darkreading.com/cloud-security/cybercriminals-can-earn-potentially-200k-monthly-exploiting-log4j-in-proxyjacking-attacks, 2023.

[11] B. Toulas, "FBI warns of residential proxies used in credential stuffing attacks," https://www.bleepingcomputer.com/news/security/fbi-warns-of-residential-proxies-used-in-credential-stuffing-attacks/, 2022.

[12] A. Zanini, "Anonymous proxy: Definition and how it works," https://brightdata.com/blog/proxy-101/anonymous-proxy-definition.

[13] M. Newman, "Bright data accused of scraping minors' information from instagram," https://finance.yahoo.com/news/bright-data-accused-scraping-minors-074036797.html, 2023.

[14] L. Kolodny and H. Field, "Elon Musk's X loses lawsuit against Bright Data over data scraping," https://www.cnbc.com/2024/05/10/elon-musks-x-loses-lawsuit-against-bright-data-over-data-scraping.html, 2024.

[15] S. Perez, "Meta drops lawsuit against web-scraping firm Bright Data that sold millions of Instagram records," https://techcrunch.com/2024/

02/26/meta-drops-lawsuit-against-web-scraping-firm-bright-data-that-sold-millions-of-instagram-records/, 2024.

[16] J. Smith, “Teaching you to create explosive videos with YouTube bots and IP Proxies,” https://medium.com/@w908683127/teaching-you-to-create-explosive-videos-with-youtube-bots-and-ip-proxies-6c7d9428a791, 2024.

[17] S. Tripathi, “Most Popular Anti-Scraping Techniques in 2025,” https://brightdata.com/blog/web-data/anti-scraping-techniques, 2024.

[18] R. Huang, D. Zhao, X. Mi, and X. Wang, “Shining light into the tunnel: Understanding and classifying network traffic of residential proxies,” 2024. [Online]. Available: https://arxiv.org/abs/2404.10610

[19] E. Chiapponi, “Detecting and Mitigating the New Generation of Scraping Bots,” https://theses.hal.science/tel-04443915v1/file/CHIAPPONI_Elisa_these_2023.pdf, 2023.

[20] D. Xue, R. Stanley, P. Kumar, and R. Ensafi, “The discriminative power of cross-layer RTTs in fingerprinting proxy traffic,” in *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2025.

[21] “Access a Borderless Internet,” https://hola.org/premium?ref=navbar, 2025.

[22] D. Goodin, “Hola! VPN’s free (and insecure) peer-to-peer network,” Ars Technica, June 2015. [Online]. Available: https://arstechnica.com/information-technology/2015/06/hola-vpn-used-to-perform-ddos-attacks-violate-user-privacy/

[23] EarnApp, “Earn money from both inactive and active app users with earnapp,” https://earnapp.com/blog/earn-money-from-both-inactive-and-active-app-users-with-earnapp, 2021, accessed: 2025.

[24] S. E. Oh, T. Yang, N. Mathews, J. K. Holland, M. S. Rahman, N. Hopper, and M. Wright, “Deepcoffea: Improved flow correlation attacks on tor via metric learning and amplification,” in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 1915–1932.

[25] S. Siby, M. Juarez, C. Díaz, N. Vallina-Rodriguez, and C. Troncoso, “Encrypted DNS $\rightarrow$ privacy? A traffic analysis perspective,” in *Network and Distributed System Security Symposium (NDSS)*, 2020.

[26] Q. Sun, D. R. Simon, Y.-M. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu, “Statistical identification of encrypted web browsing traffic,” in *Proceedings 2002 IEEE Symposium on Security and Privacy*. IEEE, 2002, pp. 19–30.

[27] B. Miller, L. Huang, A. D. Joseph, and J. D. Tygar, “I know why you went to the clinic: Risks and realization of https traffic analysis,” in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2014, pp. 143–163.

[28] R. Schuster, V. Shmatikov, and E. Tromer, “Beauty and the burst: Remote identification of encrypted video streams,” in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1357–1374. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/schuster

[29] D. Xue, R. Ramesh, A. Jain, M. Kallitsis, J. A. Halderman, J. R. Crandall, and R. Ensafi, “OpenVPN is open to VPN fingerprinting,” in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 483–500. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/xue-diwen

[30] D. Herrmann, R. Wendolsky, and H. Federrath, “Website fingerprinting: Attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier,” in *Proceedings of the ACM Conference on Computer and Communications Security*, 2009, pp. 31–42.

[31] T. Wang and I. Goldberg, “Improved website fingerprinting on tor,” in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, 2013, pp. 201–212.

[32] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg, “Effective attacks and provable defenses for website fingerprinting,” in *23rd USENIX Security Symposium (USENIX Security 14)*, 2014, pp. 143–157.

[33] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, and K. Wehrle, “Website fingerprinting at internet scale,” in *Proceedings of the 23rd Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2016.

[34] J. Hayes and G. Danezis, “k-fingerprinting: A robust scalable website fingerprinting technique,” in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 1187–1203.

[35] Y. Xu, T. Wang, Q. Li, Q. Gong, Y. Chen, and Y. Jiang, “A multi-tab website fingerprinting attack,” in *Proceedings of the 34th Annual Computer Security Applications Conference*, 2018, pp. 327–341.

[36] P. Sirinam, M. Imani, M. Juarez, and M. Wright, “Deep fingerprinting: Undermining website fingerprinting defenses with deep learning,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1928–1943.

[37] P. Sirinam, N. Mathews, M. S. Rahman, and M. Wright, “Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1131–1148.

[38] D. Xue, M. Kallitsis, A. Houmansadr, and R. Ensafi, “Fingerprinting obfuscated proxy traffic with encapsulated TLS handshakes,” in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 2689–2706. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/xue-fingerprinting

[39] M. Yang, Y. Yu, X. Mi, S. Tang, S. Guo, Y. Li, X. Zheng, and H. Duan, “An extensive study of residential proxies in china,” in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’22. New York, NY, USA: Association for Computing Machinery, 2022, p. 3049–3062. [Online]. Available: https://doi.org/10.1145/3548606.3559377

[40] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, “Tranco: A research-oriented top sites ranking hardened against manipulation,” in *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS 2019)*, 2019.

[41] P. Dodia, M. AlSabah, O. Alrawi, and T. Wang, “Exposing the rat in the tunnel: Using traffic analysis for tor-based malware detection,” in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’22. Association for Computing Machinery, 2022, p. 875–889. [Online]. Available: https://doi.org/10.1145/3548606.3560604

[42] NCTA - The Internet & Television Association, “The asymmetric nature of internet traffic,” https://www.ncta.com/news/the-asymmetric-nature-of-internet-traffic, March 2021, accessed: May 14, 2025.

[43] Cartesian, “Us broadband - household bandwidth demand study,” https://www.cartesian.com/wp-content/uploads/2021/07/Cartesian_NCTA-US-Broadband-Household-Bandwidth-Demand-Study-July-2021.pdf, Cartesian, Tech. Rep., July 2021, accessed: May 14, 2025.

[44] CommScope, “Tracking bandwidth consumption – start of the roaring 20s,” https://www.commscope.com/blog/2021/tracking-bandwidth-consumption-start-of-the-roaring-20s/, May 2021, accessed: May 14, 2025. Original publication date on the blog may vary slightly but reflects 2020 data analysis.

[45] C. Schuhmann, R. Vencu, R. Beaumont, R. Kaczmarczyk, C. Mullis, A. Katta, T. Coombes, J. Jitsev, and A. Komatsuzaki, “Laion-400m: Open dataset of clip-filtered 400 million image-text pairs,” 2021. [Online]. Available: https://arxiv.org/abs/2111.02114

[46] M. Bain, A. Nagrani, G. Varol, and A. Zisserman, “Frozen in time: A joint video and image encoder for end-to-end retrieval,” in *IEEE International Conference on Computer Vision*, 2021.

[47] “Zeek, an open source network security monitoring tool,” https://zeek.org/.

[48] E. Faranda, “Pcapdroid,” Jan 2020. [Online]. Available: https://emanuele-f.github.io/PCAPdroid/

[49] “VirusTotal — virustotal.com,” https://www.virustotal.com/gui/home/url.

[50] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, “Autogluon-tabular: Robust and accurate automl for structured data,” *arXiv preprint arXiv:2003.06505*, 2020.

[51] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 646–661.

[52] K. P. Dyer, S. E. Coull, and T. Shrimpton, “Marionette: A programmable network traffic obfuscation system,” in *24th USENIX Security Symposium (USENIX Security 15)*, 2015, pp. 367–382.

[53] E. Choo, M. Nabeel, D. Kim, R. De Silva, T. Yu, and I. Khalil, “A large scale study and classification of virustotal reports on phishing and malware urls,” *ACM SIGMETRICS Performance Evaluation Review*, vol. 52, no. 1, pp. 55–56, 2024.

[54] N. Mehanna, W. Rudametkin, P. Laperdrix, and A. Vastel, “Free proxies unmasked: A vulnerability and longitudinal analysis of free proxy services,” in *Proceedings 2024 Workshop on Measurements, Attacks, and Defenses for the Web*. Reston, VA: Internet Society, 2024.