Understanding Orphan Flows

Kevin Sam Tharayil*, Athanasios Avgetidis*, Zane Ma†, Manos Antonakakis*, Angelos D. Keromytis*
*Georgia Institute of Technology
{kevinsam, avgetidis, manos, angelos}@gatech.edu
†Oregon State University
zane.ma@oregonstate.edu

***Abstract*—Network operators require comprehensive and comprehensible network monitoring to manage resources, measure performance, ensure compliance, conduct audits, and detect/mitigate security threats. DNS labeling is one of the most useful techniques for tracking network activity, enabling operators to understand the web services that hosts communicate with. However, DNS-based monitoring has a blind spot: traffic to IPs with no associated DNS records, which we call *orphan flows/IPs*. Orphan flows can often indicate peer-to-peer or VPN communication, as well as bootstrap network services (e.g., root DNS / NTP servers), or software that is attempting to circumvent domain blocklisting systems.**

This work presents the first large-scale analysis of orphan flows to understand 1) the practical hurdles to measuring orphan flows, and 2) the potential utility of orphan flow identification for network operators and security analysts. Our seven-month study examines traffic from a large U.S. university with over 3.3 billion flows per day. We construct a robust multi-stage traffic analysis pipeline that accounts for practical challenges (e.g., data loss, clock skew) in order to hone in on true orphan flows. In total, we find communication to 26K orphan IPs, 63% of which we can categorize into behavior ranging from Windows update servers to malware-related traffic. Notably, we suspect 2.5% of the orphan IPs to be potentially malicious, but they do not appear in known threat intelligence sources. Ultimately, we shed light on a blind spot for network operators and highlight new monitoring opportunities.

I. Introduction

Network visibility is essential for network administrators. Without comprehensive awareness of devices, applications, and data flows, network operators cannot reliably perform resource and performance management, compliance and auditing, security monitoring, and threat detection. One common method for gaining network visibility is Domain Name System (DNS) monitoring. Since DNS is a ubiquitous protocol for mapping semantic domain names to IP addresses, most internet communications begin with a DNS lookup, making it a key element for monitoring network activity [1–3]. Furthermore, DNS monitoring can be combined with network flow monitoring—a popular traffic aggregation technique—to yield detailed, context-rich visibility into a wide range of network traffic [4, 5].

In an ideal network where every network flow corresponds to an IP with a DNS record, DNS-flow techniques would provide complete visibility into the named infrastructure accessed by a network. However, in less ideal real-world networks, the presence of flows that lack an associated DNS record creates visibility gaps. In this study, we refer to such network flows without an associated DNS record as ***orphan flows***. These flows may originate from legitimate applications, anonymization services such as VPNs and TOR, or malicious actors intentionally eschewing DNS [6, 7] due to DNS's prominence as an upstream choke-point for security controls. Since they are unobservable to network operators relying on DNS-based monitoring systems, it creates blind spots in network visibility, potentially allowing security events and malicious traffic to go unnoticed.

This work is the first to characterize the nature and prevalence of orphan flows in an enterprise network. Identifying and analyzing these flows is essential for understanding network operator visibility, evaluating the effectiveness of existing security controls, and determining whether additional monitoring strategies are necessary. Despite the conceptual simplicity of labeling flows based on their DNS lookup, operational obstacles associated with large-scale data collection, such as data loss, high noise levels, and the difficulty of orienting flows makes detection of orphan flows challenging at scale. To address this, we developed a multi-stage data processing pipeline taking into account these challenges which merges unidirectional flows into bidirectional flows and orients them in the correct direction before labeling flows as orphan or non-orphan using multiple DNS datasets. Our contributions in this work are as follows:

- We developed a pipeline for identifying orphan flows at scale, taking into account real-world considerations such as noisy data and applied it to seven months (over 3.32 billion flows per day) of data collected at a large US university with more than 50K members.
- We identified 10,651,552 orphan flows to 26,385 IPs over the measurement period. While over half of the orphan flows are due to explainable behavior (53.5%) such as peer-to-peer connections and VPNs, orphan traffic also consists of malicious traffic (7.27%), and even potentially malicious traffic (2.53%) not listed in any known blocklists or flagged by antivirus solutions, suggesting an overlooked security risk.
- We studied the persistence and malware associations of orphan flows, identifying the presence of threats that evade traditional DNS-based monitoring systems in the network.

Ultimately, we present the first large-scale measurement of

U.S. Government work not protected by U.S. copyright

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

orphan flows, shedding light on their characteristics, behavior, and security risks. We hope our findings can inform network operators and motivate future research into improving network traffic explainability.

II. Background and Related Work

A. NetFlow

NetFlow, introduced by Cisco, is the *de facto* standard for summarizing network communication between IPs. A flow represents a unidirectional sequence of packets that typically share the same source/destination IP addresses, source/destination port numbers, and IP protocol field. NetFlow monitoring will track aggregate metrics for each unique flow, such as the volume of data transferred, as well as the time and duration of the flow. While NetFlow has many applications [8, 9], its unidirectional nature limits its utility in certain use cases [10–12]. For instance, bidirectional communication between a client and server is represented as two separate flows, making it difficult to identify clients and servers without additional processing. Prior work has developed techniques to merge unidirectional flows into bidirectional flows [11, 12]. However, these methods do not effectively address the issue of inferring the overall direction of merged flows in a scalable manner, particularly in the presence of practical concerns, such as data loss. Additionally, in applications that correlate NetFlow and DNS data, if the focus is on flows that have corresponding DNS records, then the unidirectional nature of NetFlow is not problematic. This is because correlation attempts can be made regardless of whether the remote address is the source or destination, and flows lacking a corresponding DNS resolution are ignored. However, our objective is to identify orphan flows in a large enterprise network that hosts both clients and servers, which receive inbound requests, necessitating the exclusion of inbound flows as the concept of orphan flows do not apply to them. This requires merging unidirectional flows into bidirectional flows and determining the flow originator. No prior work has developed a robust and scalable technique for merging unidirectional flows into bidirectional flows and orienting them in the presence of practical challenges associated with large-scale network data.

B. Orphan Flows

Orphan flows are outbound network flows directed to IPs that lack an associated DNS record. Understanding orphan flows is important because they often go unmonitored, as most network monitoring systems depend on DNS for visibility. In modern enterprise networks that adhere to the principle of "default deny," where only essential interactions are permitted, blocking these unmonitored flows by default may disrupt critical services. Moreover, since many security mechanisms rely on DNS, orphan flows bypass these defenses. So at best, they represent unmonitored communication with benign services; at worst, they may indicate stealthy malware directly using IP addresses to evade DNS-based detection. Whyte *et. al.* [13] first introduced the concept of orphan flows in 2004, and identified orphan flows in a network with 63 devices that are indicative of worm propagation. Our study differs since it examines orphan flows in a modern enterprise network experiencing much larger traffic volume, significantly higher noise levels, and diverse malware threats, rendering prior techniques ineffective.

Orphan flows are relevant only to outbound traffic because inbound flows originate from external devices connecting to internal services, where the external IP represents a user and is not expected to have a DNS record. Hence we do not consider these flows. In the case of outbound flows, orphan flows are caused by applications communicating directly with IPs. These communications may stem from legitimate operations, such as routing protocols, network management traffic (e.g., SNMP, ICMP), and service discovery mechanisms (e.g., NetBIOS, LLMNR, UPnP), all of which can be easily identified by network operators based on their unique protocol characteristics. Since these flows are expected and do not obscure visibility into external traffic, they are also excluded from our analysis. Another significant portion of IP-only traffic expected in every network is traditional DNS traffic, one of the most widely used internet services. In our data, approximately 27% of flows were identified as UDP port 53 DNS traffic, consistent with prior research [14]. As these are unencrypted UDP traffic with port 53 as a unique defining characteristic, it is easy for network operators to identify and analyze. Moreover, operators maintain full control over their DNS servers, ensuring visibility into most DNS-related traffic in the network. Hence, these flows are also excluded from our analysis.

Other sources of orphan flows include peer-to-peer and anonymization services like VPNs and TOR. While not inherently malicious, these are undesirable in enterprise networks, as they may indicate users attempting to evade security controls and monitoring. Prior works have also found malware (e.g., RATS/trojans [6, 7], adware [15], worms [16]) that utilizes hard-coded IPs for command and control (C2) communications, while some malware like Mirai calculates random IP addresses [17]. As a rough estimate, we queried VirusTotal for 10K random malware samples from MalwareBazaar [18] and found that 20.3% of the 5.8K samples with IP communication did not perform any detectable DNS resolution. These malware-generated orphan flows will go unnoticed by network operators that rely on DNS-based monitoring systems. Such uncommon and hard-to-characterize orphan flows create monitoring blind spots while also evading DNS-based security controls.

C. Correlating NetFlow and DNS Data

While previous studies have demonstrated how network traffic data and DNS data can be correlated for various applications [4, 5, 12, 19–21], correlating DNS and flow data to discover orphan flows presents unique challenges. Whereas DNS-labeled (i.e., non-orphan) flows are easy to identify with high confidence, residual orphan flows are relatively noisy since there are many causes/categories of orphan flows mixed together. For example, a pseudo-cause of orphan flow in a

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

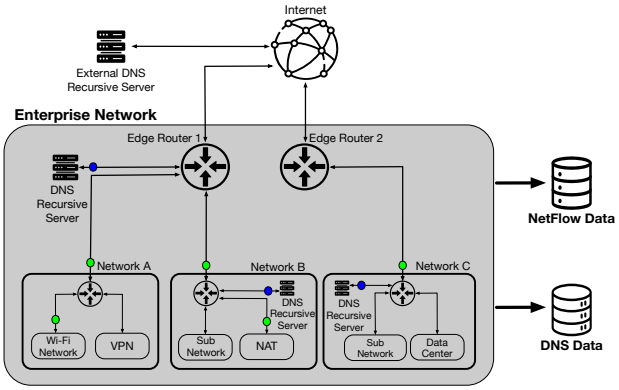

Fig. 1: An abstract representation of an enterprise network with multiple NetFlow (green) and DNS sensors (blue). Large-scale data collection often has issues like data duplication, clock skew, and data loss.

network is missing DNS data. This issue arises when the DNS record for an IP is missing in the DNS dataset, either due to a loss in data collection or if the user in the network uses encrypted DNS. In this study, we used multiple DNS datasets to identify absolute orphan flows and reduce cases of mislabeled orphan flows.

III. Datasets

To identify orphan flows, we leverage real-world NetFlow data and various DNS datasets to filter out non-orphan flows.

NetFlow Data We collected NetFlow data from a large US university with over 50K members. Sensors to gather NetFlows are located at various points within the university network, as shown in Figure 1. The data was collected from November 1, 2022 to May 31, 2023. Figure 2 shows the volume of NetFlow data we collected. Due to logistical difficulties in collecting and maintaining data at this scale, there are gaps in the NetFlow data in some of the months, which are not uncommon for large-scale network sensor deployments. Out of the 212-day (7-month) time span, we were unable to use the data from 42 days due to data collection issues. On average, our dataset consists of approximately 3.32 billion NetFlow records per day.

DNS Datasets In our implementation, the two main DNS datasets used are a passive DNS dataset collected from the university network and the Active DNS dataset [22]. The passive DNS data is collected from the university network using sensors placed at points above the recursive servers spread across the network as shown in Figure 1. The ActiveDNS dataset [22] is a comprehensive collection of DNS records gathered through daily active probing of the DNS namespace. On average, the passive and Active DNS datasets had 235 million and 356 million valid A records per day respectively.

A. Real-world data challenges

One of the key challenges in large-scale data collection is data loss [23–25]. When data is incomplete, we are unable to rely on timestamps for determining the overall direction of a merged flow, which is necessary for filtering out inbound flows. To quantify the extent of data loss within our datasets, we conducted a control experiment by generating known traffic. This experiment allows us to compare the data recorded by the NetFlow sensors with ground truth traffic collected at the source device. We sent out 10 HTTP requests to download data in various sizes from a known external server, repeating this process 125 times at varying intervals throughout the day. We carried out this experiment on four devices with globally routable IP addresses and four devices with private (NAT) IP addresses for three consecutive days.

Comparing the NetFlow data with the baseline traffic captured at the source, we observed 62.42% packet loss in the outbound traffic and an almost complete loss of inbound traffic. However, since we use NetFlow instead of individual packets, we can mitigate the potential impact of high packet loss. Since NetFlow aggregates many packets into a single flow, if at least one packet in a connection is captured, a flow will be created for it, which is sufficient for our aim of finding orphan flows. To interpret this packet loss in terms of the number of flows captured, we performed a simulation relating the number of packets per flow to the percentage of flows captured for a given percentage of packet loss. As shown in Figure 3, at 62.42% packet loss and an average of 8.37 packets per flow as observed from our data, we will be capturing 97.70% of flows, albeit often with incorrect values in the number of packets and number of bytes. This loss in data is not due to sampling at the NetFlow monitors, as no systematic sampling was in place. Rather, the loss is due to logistical challenges in data collection and is random and uncharacterizable across time and IP space.

IV. Identifying Orphan Flows

The goal of our analysis is to identify and evaluate orphan flows that hinder network visibility and may indicate DNS-evasive behavior, particularly those that are unexpected, less apparent, or difficult to characterize based on protocol properties. To identify such orphan flows, we first identify and exclude all expected and apparent cases of orphan flows from the network data (e.g., ICMP, DNS, etc.). We then correlate the remaining flows with passive DNS data collected within the network and the ActiveDNS data to label flows as orphan or non-orphan. This process is neither a detection system nor a classifier; rather, it is a framework for addressing a big data challenge by isolating orphan flows that impact network visibility. Furthermore, as the first study to investigate orphan flows, we took the most conservative approach in each step to be robust against practical challenges associated with large-scale network data to ensure a clean set of orphan flows.

Our process has two main parts: NetFlow processing and NetFlow labeling, as shown in Figure 4. The NetFlow processing step takes the raw NetFlow data and first removes noise (expected cases of orphan flows detailed in Section II-B). It then merges unidirectional flows into bidirectional flows and orients them in the correct direction. Finally, it removes all the

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

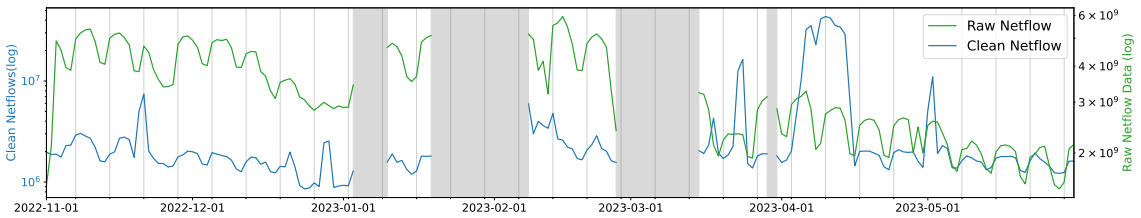

Fig. 2: Volume of raw (green line, right y-axis) and cleaned (blue line, left y-axis) NetFlow per day. The data volume follows an organic cyclic pattern of increasing from the beginning of the week, reaching a peak during the middle of the week, and falling towards the end of the week.

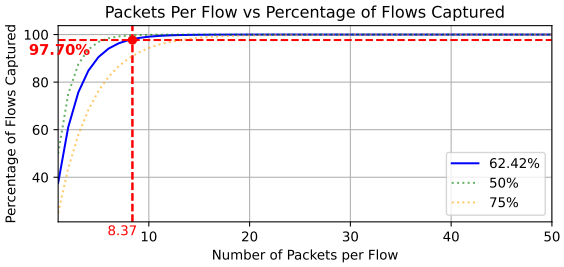

Fig. 3: Results of our statistics-based simulation showing the number of flows that would be recorded/unrecorded (*i.e.,* no. packet picked up by a NetFlow sensor) for three levels of packet loss as a function of flow size. Intuitively, flows with more packets are more likely to be recorded.

inbound flows, as the concept of orphan flows does not apply to them. Finally, in the NetFlow labeling step, we label and remove non-orphan flows using the DNS data from various sources. In the following subsections, we will describe each step in more detail.

A. NetFlow Processing

We implemented a 4-step process to clean the NetFlow data. The steps are: Pre-Merging filter, which removes noise and some expected cases of orphan flows, NetFlow merging and Netflow orienting, which is required to convert unidirectional flows to bidirectional flows, and final filter, as shown in Figure 4.

Pre-Merging filter In this step, we remove noise in our data due to misconfiguration and other expected and well-known cases of orphan traffic. This filter removes all ICMP flows, broadcast and multicast flows, port 53 DNS flows, flows to bogon IPs, organization-internal traffic, and private-address IP flows. For simplicity, we decided to limit the scope of our analysis to only IPv4 traffic and removed all IPv6 flows, which constituted 1% of the total traffic. In total, this filter removes 98.76% of our 3.32 billion average daily flows.

NetFlow Merging Since the concept of orphan flows does not apply to inbound flows, they need to be filtered out. Since flows are unidirectional, this creates a problem because, for a given unidirectional flow with an internal IP address as its source IP address, it could be either a communication that originated in our network or could be the return direction to a communication initiated by a host outside our network. In this scenario, we should only keep and attempt to label the first type of flows. Conversely, in the case of a flow that has an external IP address as the source and an internal IP address as the destination, we should only keep and attempt to label if this flow is the response to a flow that originated from our network.

In theory even though the communication between two hosts can be represented with exactly two flows (one flow for all the packets in each direction) we found contradicting evidence in our data where instances of traffic from a source to a destination i.e., the same *{src_ip, dest_ip, src_port, dest_port, protocol}* are logged multiple times. This could be because NetFlow fragments connections exceeding a predefined duration into smaller flows [12], or due to packets from the same connection taking different routes within the network and getting logged at multiple points, or because of NetFlow sensors at two different levels in the network hierarchy generating duplicate logs. To combine all the unidirectional flows that constitute the bidirectional communication between two hosts, we use the same five-tuple *{src_ip, dest_ip, src_port, dest_port, protocol}* that NetFlow protocol uses to aggregate packets into flows. The timestamp of the combined bidirectional flow will be the timestamp of the flow with the lowest timestamp among all the flows that were merged together.

NetFlow Orienting Data loss (Section III-A), along with clock skew between NetFlow sensors, means that we cannot use the timestamps of the constituent unidirectional flows in a merged flow to determine its overall direction. To address this issue, we used port numbers as the primary indicator to orient the merged flows in the correct direction. The port numbers are categorized into three ranges: well-known (0 - 1,023), registered (1,024 - 32,767), and ephemeral (32,768 - 65,535). Well-known ports are reserved for widely used processes and applications, such as HTTP and DNS. Registered ports are assigned by the Internet Assigned Numbers Authority to specific services upon request. Ephemeral ports, in contrast, are temporary and short-lived, automatically allocated by the operating system to client applications for outgoing connections. In a typical client-server communication, the client uses a random ephemeral port, and the server uses a well-known or registered port. Consequently, in a flow if one port is a well-

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

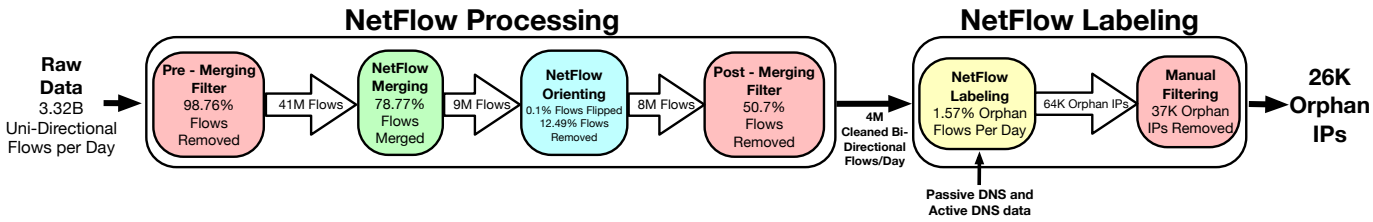

Fig. 4: We employ a two-step process to identify orphan flows. In the first step, NetFlow processing, raw NetFlow data is filtered and merged into bidirectional flows. In the second step, NetFlow labeling, diverse DNS datasets are used to identify orphan flows within the processed bidirectional flows.

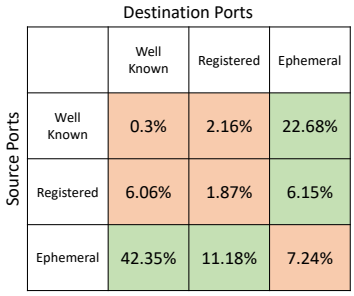

Fig. 5: Percentage of flows across all combinations of source and destination port types in our raw data. Green cells represent expected behavior flows, with one ephemeral port paired with a registered or well-known port.

known or registered port and the other port is a ephemeral port, the flow's direction can be inferred using this heuristic. In this case, the ephemeral port and its associated IP address will be the source that initiated the connection to an IP address running a service using a well-known port or a registered port [26]. In the merging step output, 87.51% of the flows followed this behavior and are oriented with the IP using the ephemeral port as the client that initiated the communication and the IP with well-known or registered port as the server.

In practice, not all flows adhere to expected port behavior. Figure 5 illustrates the percentage distribution of flows across the nine possible combinations of source and destination port types as observed in our raw data. While the majority of flows exhibit the anticipated behavior (marked in green), 17.63% deviate from this pattern (marked in orange). These unexpected port combinations arise due to various applications and protocols that do not follow this convention [27–29]. Different operating systems and even different versions of the same OS define default ephemeral port ranges inconsistently [30, 31], which can also cause such unexpected port combinations. In this study, we designate port 32,768 as the lower bound of the ephemeral range, following the convention used in many Linux kernels [32]. Taking a conservative approach to ensure precise results, we excluded flows that had unexpected port combinations (combinations marked in orange in Figure 5) since we are unable to confidently determine the direction of such flows. Thus in this step, 12.49% flows output from the merging step are removed.

Post-Merging Filter Now that we know the overall direction of a merged bidirectional flow, the post-merging filter removes all the inbound flows. As recent works have shown, many organizations that conduct internet scanning employ wide-range scanners that scan a wide range of ports across the well-known, registered, and ephemeral port ranges [33]. As the heuristics we used in the NetFlow orienting step relied purely on port numbers, it is possible that any scanner flows that do not follow it will be misoriented and hence will not be filtered out. To handle any such flows, we used a list of known-scanner IPs published by *Maltrail* [34] to remove any remaining scanner flows. The final filter removes 50.7% flows in total.

Validation Due to the absence of ground truth data representing diverse enterprise network traffic, it is impractical to evaluate the accuracy of the pipeline in a traditional manner, which limited our evaluation to small-scale manual validation efforts using known behavior and data from controlled experiments. To validate the results of our NetFlow processing step, we passed the data generated for the data loss control experiment (Section III-A) through this process. The final clean merged output contained 100% of all the flows we generated from the test devices that were present in the raw NetFlow. To ensure that inbound flows are removed, we specifically examined flows associated with IP addresses from the Censys scanning subnets [35] in both the raw and cleaned NetFlow datasets. Analyzing the data from six randomly selected days, the raw NetFlow data contained an average of 1.814 million flows per day from Censys scanning IPs. In contrast, the final cleaned and merged dataset, spanning seven months, contained only 17 flows in total from these IPs.

B. NetFlow Labeling

To identify orphan flows by eliminating flows that have a corresponding DNS resolution in any of the DNS datasets, we first created a DNS cache using the Type A responses from the passive DNS data collected from the same network as well as the Active DNS dataset. Then, for each flow, we check if the destination IP is present in the DNS cache. If the IP is present in the cache, the flow is considered non-orphan, and we discard it. In this manner, we labeled seven months of NetFlow data at a large (50k+ members) US university and found that an average of 1.57% of outbound bidirectional flows per day were

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

orphan. At this stage, we found 64,061 unique orphan IPs, i.e., the destination IPs corresponding with the orphan flows.

Manual Filtering As mentioned before, due to the prevalence of wide-range scanners, it is possible that there are false orphan IPs in the orphan IPs we found in the previous steps. To further address this issue, we obtained the ASN and ISP information of the 64,061 orphan IPs using *Cymru* and *AbuseIPDB* APIs. Using this information we identified IPs belonging to Censys, Palo Alto, and Alphastrike Research, which are organizations that conduct large-scale internet scans [33]. After observing the NetFlow patterns of these IPs, we filtered out 123 IPs as scanners.

Finally, to avoid any erroneous labeling due to gaps in the passive DNS and ActiveDNS datasets, we queried the remaining orphan IPs on VirusTotal for any known DNS resolution associated with them. Orphan IPs that had a known DNS resolution on VirusTotal are also filtered out. We did not use VirusTotal as a third DNS dataset in the previous NetFlow labeling step because, unlike passive and active DNS data, the VirusTotal dataset is accessed via an API with quota limits, making it unscalable. After removing these false orphan IPs, we have 26,385 absolute orphan IPs from our 7-month experiment.

V. Characterizing Orphan IPs

In this section, we characterize the nature and potential security relevance of the 26,385 orphan IPs from our seven-month experiment. We first examined the behavior of the orphan flows associated with these IPs by analyzing their source and destination port distributions, as well as the frequency and density of the flows. We also collected information from external sources like IP Registry [36], VirusTotal [37], AbuseIPDB [38], Maxmind [39], and IP blocklists [40]. Using this information, we were able to group orphan IPs into four groups: explainable (53.47%), malicious (7.27%), potentially malicious (2.53%), and orphan IPs with indeterminate orphan behavior (36.73%). The following subsections describe each of these categories in more detail.

A. Explainable Orphan IPs

Out of the 26K orphan IPs we identified, we classified 14,190 (53.47%) IPs as explainable. Our strategy to try to explain the behavior of orphan IPs was to first use the information provided by IP intelligence services like IP Registry [36] and MaxMind [39]. For IPs not explained through these sources, we analyzed the most popular source and destination ports in the orphan flows associated with these IPs. For ports linked to unique protocols, we examined traffic patterns and used available information, such as the ASN, to confirm whether the observed behavior aligned with the expected behavior of the protocols associated with those ports. Table I shows the various categories of explainable orphan IPs. Using data from IP Registry and MaxMind, we identified 415 IPs associated with VPNs, 56 with proxies, and two as TOR exit nodes. This constitutes the first category of explainable orphan IPs, inferred directly from the data provided by these IP intelligence services.

Explanation	Category	# Orphan IPs
From IP Intelligence Datasets	VPN	415
	Proxy	56
	TOR	2
Based on Port number, flow behavior and IP info	Windows Update	7252
	Microsoft Teams	2365
	BitTorrent Peer	633
	NTP Time Server	632
	BitTorrent Tracker	99
	FaceTime & iMessages	77
	Blizzard Games	13
Misoriented Flows	Port 52869 Satori scan	1456
	Port 50000 scan from Korean IPs	947
	Amazon AWS IP scan from port 21345	162

TABLE I: Different categories of explainable orphan IP behavior.

For the remaining IPs, we investigated outliers in the distribution of the popular ports and looked for ports that we could attribute to specific uses. For example, one outlier was port 7680, a registered port used by Microsoft Windows Update Delivery Optimization for peer-to-peer connections [41]. We identified 7,252 orphan IPs with traffic on this port. Based on ISP information, these IPs belonged to telecom providers like Comcast, AT&T, T-Mobile, etc., suggesting they were update optimization-enabled Windows devices from home users. These devices formed peer-to-peer connections within our network for transferring Windows updates. Since such devices typically lack associated DNS records, we observed them as orphan IPs.

We found 2,365 Microsoft delegated orphan IPs that had traffic on ports 3478-3481. These ports are used by Microsoft Teams for Session Traversal Utilities for Network Address Translators, audio, video, and screen sharing, respectively [42]. We hypothesize that these IPs appear as orphans because, in certain cases, Microsoft might be using hard-coded IPs instead of domain names for performance reasons like reducing latency or fault tolerance. Similarly, we found 77 orphan IPs belonging to Apple with traffic on ports ranging from 3478 to 3497, which Apple uses for FaceTime and iMessage [43]. Port 3724 is used by many games, especially Blizzard games, for various aspects of gameplay like updates, downloads and in-game communication [44]. In our data, we found 13 orphan IPs from Blizzard with traffic on port 3724.

We observed that 633 orphan IPs exclusively used ports in the range 6881–6889, the registered range for BitTorrent traffic, while 99 orphan IPs used only port 6969, the registered port for BitTorrent trackers [45]. These IPs were attributed to BitTorrent and BitTorrent trackers, as the peer-to-peer nature of BitTorrent inherently results in IPs lacking DNS resolution, leading to orphan flows. Another notable case involved port 123, which is used by Network Time Protocol (NTP) servers for time synchronization [46]. We identified 632 orphan IPs with UDP traffic solely on port 123, suggesting they were time servers. While time servers can have hostnames, they often lack DNS records because their IPs are directly provided via DHCP or manually configured (e.g., in */etc/ntp.conf* on Linux systems), leading to their appearance as orphan IPs.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

Several orphan IP flows appeared to exhibit likely scanning behavior. For example, 162 orphan IPs from Amazon AWS showed traffic originating from port 21345, a port not associated with any known service, targeting a range of ephemeral ports. Examining the popular source ports of orphan flows, 1,456 orphan IPs used port 52869, associated with the Satori botnet, a Mirai variant that exploits a command injection vulnerability in Realtek SDK's Universal Plug and Play (UPnP) SOAP interface [47]. Over 94% of these IPs exclusively used port 52869 as their source port, with flows directed to various internal IP addresses. Similarly, all flows from 947 orphan IPs belonging to KIXS-AS-KR, a Korean Telecom ASN, exclusively used port 50,000 as their source port. In all three cases, reports on AbuseIPDB described scanning behavior matching the observed source destination port patterns during our measurement period. These three sets of orphan IPs: Amazon AWS IPs scanning from port 21345, IPs scanning port 52869, and Korean IPs scanning port 50000 are likely misoriented inbound flows resulting in false orphan IPs. Based on flow patterns and corroborating data from external sources, it is improbable that internal IPs in our network initiated communication with these IPs. Instead, these unsolicited inbound scanning flows, which use well-known or registered source ports to target ephemeral ports, are likely false positives in our dataset.

Takeaways: *Explainable orphan IPs are primarily anonymizers like VPNs and proxies, or peer-to-peer communications. In tightly controlled networks, it is desirable to detect and disable these types of communications which can circumvent installed network protections. However, caution is required, as legitimate services like video conferencing and time servers also generate orphan flows. Additionally, despite efforts to filter out scanner flows, the prevalence of false positives, all identified as scanners, underscores the ongoing challenge of automatically and reliably distinguishing scanners in real-world network traffic at scale.*

B. Malicious Orphan IPs

Based on prior works [6, 7, 15, 16] that highlight the prevalence of IP-only communication in malware, we hypothesized that orphan flows can indicate malicious traffic trying to evade DNS-based detection methods. To find malicious IPs among the orphan traffic, we used two ground-truth datasets: the *BLAG* dataset [40] and malicious reports by security vendors on VirusTotal (VT). Figure 6 shows a Venn diagram indicating the number of malicious orphan IPs found. As the figure shows, 1,899 orphan IPs were in *BLAG*, and 130 orphan IPs had more than five malicious reports on VirusTotal. Of these, 112 IPs were common to both categories. Therefore, we classified a total of 1,917 orphan IPs (7.27%) as malicious.

Malicious Reports on VirusTotal VirusTotal is an online service that aggregates reports from various anti-virus (AV) engines and other security tools to provide a comprehensive assessment of potential threats associated with the different indicators of compromise (e.g., IP, URL, file hash). We queried

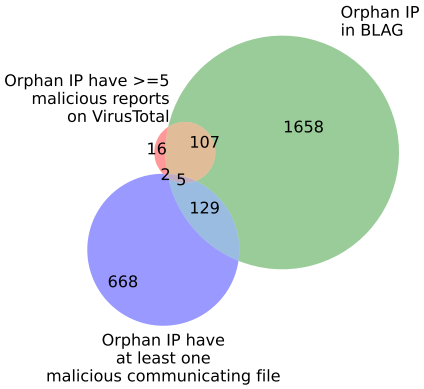

Fig. 6: Venn diagram of malicious and potentially malicious orphan IPs found using VirusTotal, BLAG, and malicious communicating files.

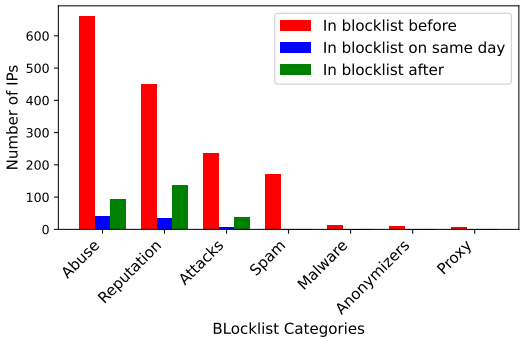

Fig. 7: Orphan IP distribution in our data by malicious category, and relative to the time of their first appearance on blocklists.

the 26,385 orphan IPs on VirusTotal. To take a conservative approach, we only labeled IPs that were flagged by at least five AV vendors, finding 130 malicious IPs. To augment the limited context of AV reports, we also enriched our data with 1) IP blocklists and 2) VT queries for malicious files that communicate with orphan IPs. These two methods provided more context for 114 out of 130 IPs found in VirusTotal AV reports and labeled an additional 2,455 orphan IPs, which we expand upon in the remaining parts of this section.

BLAG Blocklist Aggregator (BLAG) combines 157 publicly available IP blocklists [40] on a daily basis. Using data from the last six years from January 2019 to October 2024, we found that 1,899 orphan IPs were present in the BLAG dataset. To identify if orphan IPs are a timely indicator of malice, for each of these IPs, we calculated the days between the first orphan detection and the first day the IP appeared in BLAG. Out of the 1,899 IPs, 14.1% only appeared in the blocklist after they were first detected as orphan, 4.3% appeared in the blocklist on the same day they were first detected as orphan, and 81.7% first appeared in the blocklist before they were detected as orphan in our data. While a small percentage of IPs were detected as orphan first, a significant majority were already recognized as malicious prior to their detection

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

as orphan, indicating that deploying robust IP blocklists is effective in identifying malicious orphan IPs. Furthermore, to understand the type of malice associated with these malicious orphan IPs, we used the information provided in [40] and FireHOL [48] to categorize the various blocklists in BLAG into abuse, anonymizer, attack, malware, proxy, reputation and spam. Figure 7 shows the distribution of orphan IPs in BLAG across various categories of blocklists. It breaks down the number of IPs in each category that appeared in the blocklist before, on the same day as, or after the first day the IP appeared as an orphan in our data. Notably, a small percentage of IPs in the top three categories were detected as orphan before they appeared on the blocklist, whereas all the IPs in more acute categories (e.g., malware) were listed on the blocklist before they were detected as orphan ($\chi^2 = 88.37, p = 7.69 \times 10^{-13}$).

Takeaways: *Blocklists provide a valuable layer of security against malicious orphan IPs, as they are effective at identifying and flagging these many types of malicious orphan IPs before they pose a threat to the network. However, for IPs in abuse, reputation, and attack categories, studying orphan IPs in the network can provide an opportunity to augment existing threat intelligence.*

C. Potentially Malicious Orphan IPs

A potentially malicious IP is an IP that is not listed in blocklists or flagged by antivirus vendors but is associated with a malicious communicating file, unlike malicious IPs, which are explicitly listed or flagged. An IP address's communicating files are files that have generated traffic to the IP during execution. We queried VirusTotal to obtain the communicating files of the 26,385 orphan IPs, which yielded 16,924 unique file hashes. We further queried these files on VirusTotal to identify if they were malicious. Again taking a conservative approach, we classified files as malicious only if at least five antivirus (AV) vendors flagged them as malware. Using this criterion, we selected orphan IPs that had at least one malicious communicating file. Figure 6 shows that 804 orphan IPs were associated with at least one malicious communicating file (malware). Among the 804 IPs, 136 appeared in BLAG or were flagged by at least five AV vendors or both, indicating a stronger signal of malice. The remaining 668 IPs are categorized as potentially malicious orphan IPs. This classification stems from the fact that the presence of malware generating traffic to an IP is not definitive proof of the IP's maliciousness. For example, the IP could belong to a command-and-control (C2) server the malware is attempting to contact, or it could be a randomly generated IP used by the malware to check for connectivity. To conclusively identify these behaviors, it is necessary to execute the malware in a sandbox environment and analyze the resulting traffic. This we leave for future work.

Types and Families of Malware Using Orphan Flows Although the presence of a malicious communicating file alone is insufficient to confidently label an orphan IP as malicious, reports on these files provide insights into the types and families of malware that demonstrate some form of DNS evasion. We used VirusTotal to acquire malware labels from AV vendors and followed the methodology detailed by Faulkenberry et al. [49] to identify the specific malware type and family of each reported file.

Most orphan IPs were associated with grayware, viruses, and backdoors. Grayware can have a range of undesirable behaviors, from displaying intrusive ads to tracking user activity. We hypothesize that the grayware we detected may be using orphan IPs to remain undetected to ensure their continued operation. Viruses are malware that replicate and spread by attaching to host files or programs. For connectivity check or propagation it could be sending traffic directly to IP ranges, potentially creating orphan flows. Backdoors that provide attackers with unauthorized access are often associated with persistent threats, indicating adversaries seeking to establish stealthy and ongoing access to the target system [50, 51]. As such, it is in their best interest to bypass DNS-based security controls to avoid any operational disruptions.

The top 3 malware families associated with orphan IPs are *cerbu*, a trojan family [52], *pioneer*, a virus linked to the advanced persistent threat called DarkHotel [53], and *wapomi*, a virus family with trojan-like behavior [54]. The rest of the top 10 malware families are *mirai* botnet, *allapele* worm, *opencandy* adware, *wannacry* ransomware, *sality* virus and trojan families *gobot* and *razy*. To further substantiate the orphan direct IP communication behavior of these malware families, we queried VirusTotal to get the contacted domains of all the malware from the top 10 families we identified. Domains listed in the Tranco Top-1M [55] were excluded, as some malware are known to ping popular domains to verify Internet connectivity [56]. For five of the top 10 families—*Cerbu*, *Pioneer*, *Allaple*, *Opencandy*, and *Gobot*—over 90% of their samples did not contact any domains, aligning with their reported reliance on direct IP communication (Section II-B), which demonstrates and confirms the DNS-evasive behavior of certain malware.

Takeaways: *We found 668 orphan IPs that are associated with malware execution, which corroborates DNS evasion by malware, even if the traffic and orphan infrastructure themselves are not malicious. As these IPs are not listed in blocklists nor flagged by security vendors, they could either be false negatives that should be added to blocklists, or they are benign fingerprints that can help aid in malware detection. These potentially malicious orphan flows are present in a wide range of malware types and families.*

D. Orphan IPs with Indeterminate Orphan Behavior

We are unable to determine the orphan behavior of the remaining 9691 IPs (36.73%) as we were unable to attribute and validate their orphan behavior to specific causes based on IP intelligence sources, port numbers or any other observed characteristics. Table II shows the top 10 most popular destination ports for flows of these orphan IPs. Based on data from IP Registry, 56.86% of these IPs are owned by Internet Service Providers (ISPs), 38.42% are hosting provider IPs, 2.97%

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

Port #	Protocols and services	# Orphan IPs	Port #	Protocols and services	# Orphan IPs
443	HTTPS	2882	15000	Used by some games, Kaspersky	47
80	HTTP	1236	22	secure shell, file transfer	31
3724	Used by various games	91	25	Simple Mail Transfer Protocol	26
8090	HTTP alternate port	76	1024	IANA Reserved port, KDE Display manager	26
8080	HTTP alternate port	68	8999	Brodos Crypto Trade Protocol	20

TABLE II: Top destination ports for Orphan IP flows with indeterminate behavior.

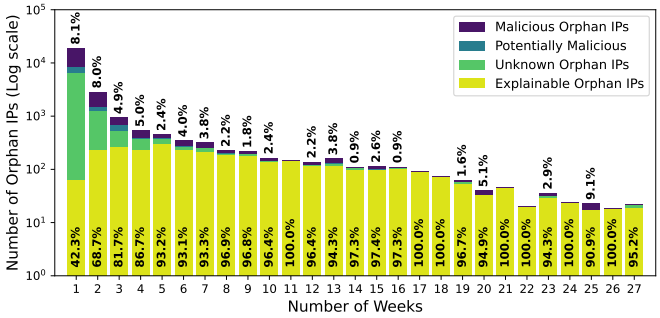

Fig. 8: Orphan IP Persistence. The percentage on the bottom of the bar indicates the percentage of explainable orphan IPs, and the percentage on top of the bar indicates the percentage of malicious orphan IPs in that week.

belong to educational institutions, and the remaining 1.48% are associated with private or government organizations.

Takeaways: *The inability to attribute the behavior of over a third of orphan IPs, despite leveraging all available open-source intelligence and manual analysis, underscores the challenges of fully demystifying orphan IPs. This difficulty reflects the inherent noise, complexities, and "wild west" nature of real-world network data. Resolving this issue remains an open problem, warranting future research with alternative approaches, such as analyzing network packets instead of flows or measuring orphan flows at the individual device level rather than across entire networks.*

E. Orphan IP Persistence

Understanding whether different categories of orphan IPs are short-lived or persistent will help network operators prioritize their responses and refine their monitoring strategies. Figure 8 shows a long-tailed distribution of orphan IP persistence, with over 19K (72.3%) orphan IPs appearing only during a single week in our 7-month study. Most orphan flow infrastructure, regardless of maliciousness, is contacted for short durations. Conversely, almost all the long-lasting orphan IPs are explainable. As our findings reveal, most malicious and potentially malicious orphan IPs were short-lived, with no persistent malicious activity observed. This transient behavior can be attributed to security systems improving over time and quickly incorporating intelligence on malicious IPs, blocking them before prolonged activity, and/or to attackers frequently rotating IPs to evade detection. Moreover, the behavior of orphan IPs can change over time; they may later be assigned domain names, or IP churn could occur with residential IPs, while hosting IPs may change users.

Takeaways: *Most orphan flows, regardless of maliciousness, are transient. The low persistence of malicious orphan IPs underscores the importance of maintaining up-to-date threat intelligence, such as IP blocklists and dynamic blocking mechanisms, to effectively mitigate their impact. The transient nature of potentially malicious IPs emphasizes the need for real-time monitoring of orphan IPs for timely detection and response, as they are not flagged by existing IP blocklists and antivirus systems.*

VI. Limitations

Due to high noise, data loss, and our conservative approach, orphan IP counts in some categories may be underestimated, but they serve as a robust lower bound. A significant challenge posed by data loss was determining the overall direction of merged flows, which forced us to rely on port-based heuristics. The limitations of this approach became especially apparent when we encountered scanners. Even though we used a static list of scanners and additional manual analysis to filter out wide-range scanner flows that do not conform to the port-based heuristics, as our results indicate, certain scanning IPs bypass these checks and are erroneously identified as orphan IPs (Sections V-A). Since we lacked direct access to the machines generating orphan traffic, we are unable to conclusively determine the orphan behavior across all orphan IPs. Instead, we had to infer orphan behavior using available information such as flow patterns, port numbers, and IP and threat intelligence datasets.

VII. Conclusion

Orphan flows create blind spots in network visibility since they are unobservable to operators using DNS-based monitoring systems. The inherent complexities of large-scale network data collection, such as data loss and clock skew, coupled with the unidirectional nature of NetFlows, make the identification of orphan IPs at scale challenging. We present the first large-scale study into the prevalence and implications of orphan flows by developing a data loss-resistant measurement pipeline using seven months of network flows from a large US university network. Our analysis reveals that most orphan flows can be attributed to explainable, and often overlooked traffic behaviors. Furthermore, a smaller set of orphan flows are associated with various malicious behaviors, highlighting security risks. While the decision depends on an organization's risk tolerance and cost-benefit considerations, incorporating orphan IP monitoring would not only enhance security but also assist in identifying benign orphan traffic, enabling operators to adjust controls and maintain operational integrity.

References

[1] M. Zain ul Abideen, S. Saleem, and M. Ejaz, "Vpn traffic detection in ssl-protected channel," *Security and Communication Networks*, vol. 2019, 2019.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

[2] S. Torabi, A. Boukhtouta, C. Assi, and M. Debbabi, "Detecting internet abuse by analyzing passive dns traffic: A survey of implemented systems," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3389–3415, 2018.

[3] A. Feldmann *et al.*, "The lockdown effect: Implications of the covid-19 pandemic on internet traffic," in *Proceedings of the ACM internet measurement conference*, 2020, pp. 1–18.

[4] A. Maghsoudlou, O. Gasser, I. Poese, and A. Feldmann, "Flowdns: Correlating netflow and dns streams at scale," in *Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies*, 2022.

[5] R. Hananto, C. Lim, and H. P. Ipung, "Detecting network security threats using domain name system and netflow traffic," in *Proceedings of the 2nd International Conference on Cryptography, Security and Privacy*, 2018, pp. 105–109.

[6] I. T. Intelligence. "Emotet: A malware family that keeps going." (2022), [Online]. Available: https : / / blogs . infoblox.com/threat-intelligence/cyber-threat-advisory/emotet-a-malware-family-that-keeps-going/.

[7] Sonatype. "There's a rat in my code: New npm malware with bladabindi trojan spotted." (2020), [Online]. Available: https://blog.sonatype.com/bladabindi-njrat-rat-in-jdb.js-npm-malware.

[8] C.-T. Yang, J.-C. Liu, E. Kristiani, M.-L. Liu, I. You, and G. Pau, "Netflow monitoring and cyberattack detection using deep learning with ceph," *IEEE Access*, vol. 8, pp. 7842–7850, 2020.

[9] Z. Long and W. Jinsong, "Network traffic classification based on a deep learning approach using netflow data," *The Computer Journal*, 2023.

[10] M. Fullmer and S. Romig, "The OSU flow-tools package and CISCO NetFlow logs," in *14th Systems Administration Conference (LISA 2000)*, 2000.

[11] R. Sommer and A. Feldmann, "Netflow: Information loss or win?" In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, 2002.

[12] M. Fejrskov, J. M. Pedersen, and E. Vasilomanolakis, "Using netflow to measure the impact of deploying dns-based blacklists," in *Security and Privacy in Communication Networks: 17th EAI International Conference, SecureComm 2021, Virtual Event, September 6–9, 2021, Proceedings, Part I 17*, Springer, 2021, pp. 476–496.

[13] D. Whyte, E. Kranakis, and P. C. Van Oorschot, "Dns-based detection of scanning worms in an enterprise network.," in *NDSS*, 2005.

[14] M. Čermák, P. Čeleda, and J. Vykopal, "Detection of dns traffic anomalies in large networks," in *Advances in Communication Networking: 20th EUNICE/IFIP EG 6.2, 6.6 International Workshop, Rennes, France, September 1-5, 2014, Revised Selected Papers 20*, Springer, 2014, pp. 215–226.

[15] JoeSandbox. "Windows analysis report: Securite-info.com.adware.opencandy.197.19.16636.exe, analysis report 1389553." (), [Online]. Available: https://www.joesandbox.com/analysis/1389553/1/html.

[16] C. I. R. C. L. (CIRCL). "Tr-40 - allaple worm activity in 2015 and long-term persistence of worm (malware) in local area networks." (), [Online]. Available: https://www.circl.lu/pub/tr-40/.

[17] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for DNS," in *19th USENIX Security Symposium (USENIX Security 10)*, 2010.

[18] *MalwareBazaar*, https://bazaar.abuse.ch/.

[19] K. S. Tharayil, P. Kintis, and A. D. Keromytis, "Augmenting dns-based security with netflow," in *2024 4th International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, IEEE, 2024, pp. 01–06.

[20] D. Plonka and P. Barford, "Context-aware clustering of dns query traffic," in *Proceedings of the 8th ACM SIGCOMM conference on Internet Measurement*, 2008, pp. 217–230.

[21] D. Plonka and P. Barford, "Flexible traffic and host profiling via dns rendezvous," in *Workshop Satin*, 2011.

[22] A. Kountouras *et al.*, "Enabling network security through active dns datasets," in *Research in Attacks, Intrusions, and Defenses: 19th International Symposium, RAID 2016, Paris, France, September 19-21, 2016, Proceedings 19*, Springer, 2016, pp. 188–208.

[23] R. Hofstede, I. Drago, A. Sperotto, R. Sadre, and A. Pras, "Measurement artifacts in netflow data," in *Passive and Active Measurement: 14th International Conference, PAM 2013, Hong Kong, China, March 18-19, 2013. Proceedings 14*, Springer, 2013, pp. 1–10.

[24] R. Hofstede *et al.*, "Flow monitoring explained: From packet capture to data analysis with netflow and ipfix," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.

[25] H. Lin, Z. Yan, Y. Chen, and L. Zhang, "A survey on network security-related data collection technologies," *IEEe Access*, vol. 6, pp. 18 345–18 365, 2018.

[26] M. Evangelou and N. M. Adams, "Predictability of netflow data," in *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, IEEE, 2016, pp. 67–72.

[27] S. U. IT, *Kerberos and firewalls*, Accessed: 2025-01-20, 2025. [Online]. Available: https://uit.stanford.edu/service/kerberos/firewalls.

[28] Z. Community, *Zoom client p2p calls port range*, Accessed: 2025-01-20, 2025. [Online]. Available: https://community.zoom.com/t5/Zoom-Phone-System/Zoom-client-P2P-calls-port-range/m-p/32223/highlight/true.

[29] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RFC 3550: Rtp: A transport protocol for real-time applications*, Accessed: 2025-01-20, 2003. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3550.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.

[30] D. Lee, B. E. Carpenter, and N. Brownlee, "Observations of udp to tcp ratio and port numbers," in *2010 Fifth International Conference on Internet Monitoring and Protection*, IEEE, 2010, pp. 99–104.

[31] NCFTP, *Ephemeral port numbers*, Accessed: 2025-01-20, 2025. [Online]. Available: https://www.ncftp.com/ncftpd/doc/misc/ephemeral_ports.html.

[32] The Linux Kernel Organization, Inc., *Ip sysctl variables*, https://www.kernel.org/doc/html/latest/networking/ip-sysctl.html, Accessed: 2024-12-19, 2024.

[33] J. Mayer *et al.*, "I know who you scanned last summer: Mapping the landscape of internet-wide scanners," in *2024 IFIP Networking Conference (IFIP Networking)*, IEEE, 2024, pp. 222–230.

[34] stamparm, *Maltrail - a malicious traffic detection system*, https://github.com/stamparm/maltrail/blob/master/trails/static/mass_scanner.txt.

[35] Censys Support, *Opt-out of data collection*, Accessed: 2025-01-09, 2025. [Online]. Available: https://support.censys.io/hc/en-us/articles/360043177092-Opt-Out-of-Data-Collection.

[36] IP Registry, *Ip geolocation api and database*. [Online]. Available: https://ipregistry.co/.

[37] VirusTotal, *Virustotal api documentation - overview*. [Online]. Available: https://docs.virustotal.com/reference/overview.

[38] AbuseIPDB, *Abuseipdb api documentation*. [Online]. Available: https://docs.abuseipdb.com/#introduction.

[39] MaxMind, *Geolite2 free geolocation data*. [Online]. Available: https://dev.maxmind.com/geoip/geolite2-free-geolocation-data/.

[40] S. Ramanathan, J. Mirkovic, and M. Yu, "Blag: Improving the accuracy of blacklists," in *NDSS*, 2020.

[41] Microsoft, *Windows as a service (waas) delivery optimization faq*, Accessed: 2025-01-20, 2025. [Online]. Available: https://learn.microsoft.com/en-us/windows/deployment/do/waas-delivery-optimization-faq.

[42] Microsoft, *Microsoft teams online call flows*, Accessed: 2025-01-20, 2025. [Online]. Available: https://learn.microsoft.com/en-us/microsoftteams/microsoft-teams-online-call-flows.

[43] A. Support, *Tcp and udp ports used by apple software products*, Accessed: 2025-01-20, 2025. [Online]. Available: https://support.apple.com/en-us/103229.

[44] P. Blog, *How to port forward for blizzard battle.net*, Accessed: 2025-01-20, 2025. [Online]. Available: https://www.purevpn.com/blog/port-forward-blizzard-battlenet/.

[45] W. Wiki, *Bittorrent - the wireshark wiki*, Accessed: 2025-01-20, 2025. [Online]. Available: https://wiki.wireshark.org/BitTorrent.

[46] D. L. Mills, *RFC 1305: Network time protocol (version 3) specification, implementation and analysis*, Accessed: 2025-01-20, 1992. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1305.

[47] 3. N. Blog, *Warning: Satori, a new mirai variant, is spreading in worm style on port 37215 and 52869*, Accessed: 2025-01-20, 2025. [Online]. Available: https://blog.netlab.360.com/warning-satori-a-new-mirai-variant-is-spreading-in-worm-style-on-port-37215-and-52869-en/.

[48] FireHOL, *Ip lists - a collaborative project to create a list of ips for firewalls*. [Online]. Available: http://iplists.firehol.org.

[49] A. Faulkenberry *et al.*, "View from above: Exploring the malware ecosystem from the upper dns hierarchy," in *Proceedings of the 38th Annual Computer Security Applications Conference*, 2022, pp. 240–250.

[50] S. Tomonaga, T. Tani, H. Soeda, and W. Takahashi, *Apt cases exploiting vulnerabilities in region-specific software*. [Online]. Available: https://www.virusbulletin.com/virusbulletin/2020/05/vb2019-paper-apt-cases-exploiting-vulnerabilities-regionspecific-software/.

[51] A. Zimba and Z. Wang, "Malware-free intrusions: Exploitation of built-in pre-authentication services for apt attack vectors," *International Journal of Computer Network and Information Security*, vol. 9, no. 7, p. 1, 2017.

[52] "W32/variant.cerbu!tr," FortiGuard Labs, Tech. Rep., May 2023. [Online]. Available: https://www.fortiguard.com/encyclopedia/virus/10137553.

[53] G. Research and A. Team, "The darkhotel apt: A story of unusual hospitality," Kaspersky Lab, Tech. Rep., Nov. 2014. [Online]. Available: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/08070903/darkhotel_kl_07.11.pdf.

[54] R. Alvarez, "Wapomi," *Virus Bulletin*, Jun. 2014. [Online]. Available: https://www.virusbulletin.com/virusbulletin/2014/06/wapomi.

[55] V. L. Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," *arXiv preprint arXiv:1806.01156*, 2018.

[56] Y. Nadji, M. Antonakakis, R. Perdisci, and W. Lee, "Understanding the prevalence and use of alternative plans in malware with network games," in *Proceedings of the 27th Annual Computer Security Applications Conference*, 2011, pp. 1–10.

APPENDIX

Ethical Concerns: This work does not raise any ethical issues. It was determined that the analysis we conducted did not require a human subject research protocol.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on March 17,2026 at 14:30:55 UTC from IEEE Xplore. Restrictions apply.