Inside Residential IP Proxies: Lessons Learned from Large Measurement Campaigns

Elisa Chiapponi
EURECOM
Biot, France
elisa.chiapponi@eurecom.fr

Marc Dacier
RC3, CEMSE, KAUST
Thuwal, Kingdom of Saudi Arabia
marc.dacier@kaust.edu.sa

Olivier Thonnard
Amadeus IT Group
Villeneuve-Loubet, France
olivier.thonnard@amadeus.com

Abstract—**Residential IP Proxy (RESIP) providers represent a growing threat when used for web scraping and other malicious activities. RESIPs enable their customers to hide behind a vast network of residential IP addresses to perpetrate their actions. This helps the customers to evade detection. Thanks to two new large datasets of RESIP connections, we reveal new insights into RESIP inner functioning and modus operandi. We present the similarities and differences of the ecosystems associated with four RESIP providers (geographic distribution, types, management and amount of machines used). Moreover, we display how two of the providers have striking similarities and we propose a specific detection method to identify them. Furthermore, we show how to build a list of suspicious /24 blocks of IP addresses and use it to mitigate the actions of malicious parties behind RESIPs.**

1. Introduction

Residential IP Proxies (RESIP) are provided by legal companies that, for a fee, enable a user to proxy out requests through their vast network of residential devices. Some RESIP providers build their networks thanks to mobile SDKs included by developers in apps. Device owners voluntarily download these apps and give consent to be part of the network [21]. However, Frappier et al. [18] suggest that, in some cases, the provider lures these device owners. It let them install software that looks legitimate in exchange for a free VPN service and uses their devices to proxy out requests without their informed consent.

It has been shown that RESIP services also exploit infected IoT devices [20], leverage browser extensions and create dedicated clusters of SIM cards [31].

RESIP providers claim to have access to tens of millions of residential IPs[1]. Contrary to datacenter IPs, residential ones are dynamically assigned by Internet Service Providers, that typically reassign IP addresses [24]. Thus, a user proxying out a request through the same device, could end up sending requests with different IPs at different moments in time. Moreover, multiple devices could have the same IP in different moments in time or use the same IP at the same time when behind a Network Address Translation (NAT) device. For these reasons, there is no 1-1 correspondence between the number of devices and the number of IPs involved in a RESIP network.

RESIP services do not share details about their internal infrastructure. Generally, they work in, so called, backconnect mode. As discussed in previous works [8], [20] in a back connected RESIP, the client sends a request to the so-called SUPERPROXY. The SUPERPROXY forwards this request to a GATEWAY. In [20], the authors find that there is a series of backend servers intermediating between the SUPERPROXY and the GATEWAY for specific providers. However, it is not clear if this is a common behavior among other providers. Finally, the request arrives at the server with the IP of the GATEWAY as source IP and does not contain any application-level information about being proxied.

RESIP clients can specify the location of the GATEWAYs to use. Furthermore, they can decide if to use the same IP for a series of requests or instead obtain a new one for each new request they want to send. Different RESIP providers implement various proxy protocols, but generally, they all support HTTP/HTTPS. In this case, the client contacts the SUPERPROXY with a HTTP CONNECT to establish the communication.

In [35], the authors discovered that some RESIP providers offer a second operating mode called direct. In the case of direct RESIP, the client acquires the IPs of the gateways and contacts directly the one(s) chosen to proxy requests out. However, this type of RESIP looks popular mostly for Chinese providers and it does not appear to be widely used globally.

While customers can use RESIP for legitimate purposes, their features are appealing for malicious activities, such as providing anonymity when doing automatic ad clicks, generating new accounts, performing credential stuffing attacks and social media spam [13].

In recent years, RESIPs have been used more and more to perform web scraping [10]. Scraped websites usually put in place detection mechanisms against scrapers. However, they are hesitant to block residential IPs, for fear of preventing real users to access their content. In this way, scrapers using RESIP avoid being blocked [7]. Moreover, RESIP services usually offer advanced features (CAPTCHA solving [22], automatic fingerprint rotation) that help even non-experienced scrapers to easily achieve their goal.

RESIP IPs have been associated with other illicit operations. Mi et al. [20] show that these IPs have been involved in malicious website hosting and IoT botnet campaigns. They discover that all the providers they studied use "potentially unwanted programs" to relay their traffic and that part of the collected RESIP IPs served as fast flux proxies. In [35], 62.61% Chinese RESIP IPs have been detected conducting cryptojacking, while 21% were detected as

[1] In the rest of the paper, we use IP and IP address interchangeably.

bots. Finally, RESIP IPs have been exploited to masquerade the identity of cyber criminals [30].

Past works clearly reveal that different actors used RESIP for malicious purposes. Moreover, the RESIP shady device recruitment processes raise concerns. Our work aims at measuring and analyzing the RESIP ecosystem to characterize it better and mitigate the threats it represents when used by malicious parties.

To achieve this, we have collected and studied two new large datasets of RESIP connections. We obtained the first one during an intensive 4-months measurements campaign. In that campaign, machines around the world sent requests to each other through four RESIP providers. The second dataset records the modus operandi of three RESIP services when contacting a server that does not acknowledge SYN packets.

Thanks to the analysis performed on these two datasets, we have learned new pieces of information about RESIP providers and their infrastructures. We can summarize the contributions of our work as follows:

- We provide novel insights about the inner working of RESIP in terms of geographic distribution, types, management and amount of machines used. We show that RESIP services try to minimize the GATEWAY IP repetitions for the same client-server path and do not privilege the assignation of GATEWAYs close to the server. Concerning the SUPERPROXYs, we display how for one provider the SUPERPROXY domain name resolves in thousands of IPs, while for others this process ends up in less than 20 IPs. Moreover, we reveal the different distributions of GATEWAYs in terms of area of the world, operating system and availability when comparing the studied providers with each other. Furthermore, considering the volume of GATEWAY IPs, we see that our observations do not confirm the claimed pool sizes.
- We disclose the internal algorithms RESIPs use when their GATEWAYs need to send SYN packets to a non-acknowledging server. Multiple GATEWAYs, located in different areas of the world, are involved in this process. Moreover, two of the analyzed providers show a specific retransmission behavior that we can use to detect them.
- We arrive at the conclusion that, even if all RESIP providers share some features, not all of them have the same implementation and functioning. We can leverage the distinct characteristics of each provider to build detection and attribution methods for them. According to this, we build a new detection method for two RESIP providers.
- We recognize that two of the studied providers present almost identical results in all the performed analyses and share large portions of their pools of IPs.
- We suggest an approach similar to blocklists to find IPs that could potentially be part of RESIPs.

The rest of the paper is structured as follows. Section 2 presents the state of the art about RESIP investigations. In Section 3, we describe the datasets used for our analyses. Section 4 proposes eight findings that reveal new insights about the inner functioning of RESIP, thanks to the analysis of a 4-month-long collection campaign. Section 5 presents the study of the behavior of RESIP GATEWAYs when the server does not complete the TCP handshake. In Section 6, we discuss all the obtained results and summarize the lessons learned. Finally, Section 7 concludes our work.

2. State of the art

In recent years, we have witnessed the rise of Residential IP Proxies and researchers have started to analyze them. In this section, we provide a review of the state of the art about RESIP characterization and we ensure to position our work in comparison with the already published ones.

In 2019, Mi et al. proposed the first comprehensive study of RESIP services [20]. They created an infiltration framework and they used it in 2017 to collect 6M+ unique IPs from 5 RESIP providers. To obtain one of our datasets, we use an infrastructure similar to theirs. With it, we obtain a much larger dataset than the one collected in their work (13M+ unique IPs).

Their analysis discloses internal features of RESIP services, but they focus on investigating the involvement in malicious activities of RESIP IPs and studying if these addresses are residential, as advertised by the providers. Differently from them, we examine with much more depth geographic distribution, types, management and amount of GATEWAYs IPs in our dataset. This enables to obtain original findings about the internal functioning of RESIPs.

After the work of Mi et al., other studies about RESIP services were proposed. In 2020, their dataset was used in [9] to compare RESIP GATEWAYs and open proxies. The authors compare the locality distribution of these parties and check the reputations of the collected IPs. In the same year, Hanzawa et al. [12] used the same dataset to better characterize RESIP GATEWAYs in Japan and identify connected malicious activities to those GATEWAYs. Differently from those two past works, we consider completely new datasets, we do not focus only on one specific region of the world and we perform new analyses about repetitions and assignation patterns of our IPs.

In [6] (2021), we collected a relatively small dataset of IPs performing web scraping. Our analysis suggested these IPs were part of a RESIP network. We performed mathematical modelings of these addresses to estimate the size of the IP pool of RESIP providers. Our results suggested that these numbers could be much smaller than the ones announced by these services. In Section 4.9, we apply one of these previously used models to our much larger dataset studied in this work. While the found pool sizes are much bigger than the ones seen in the previous modeling, in both cases the pool sizes claimed by RESIP providers do not correspond to our observations.

Between 2021 and 2022, studies on mobile devices as RESIP GATEWAYs have been published [21], [26], [29], [31]. The focus of these works is to understand how a device becomes part of a RESIP.

In 2022, Yang et al. [35] proposed a characterization of the RESIP ecosystem in China. They investigated how many different RESIP services exist and how they work. To study the back connected RESIP providers, they have used an infrastructure similar to the one in [20], as we do. With it, they have collected a smaller dataset with respect to ours (9M+ IP addresses). They focus on understanding

TABLE 1: Attributes of a connection in MAIN_DS.

Attribute	Attribute explanation
CLIENT_EPOCH	Epoch (UTC+0) in which the client sends the request (s)
CLIENT_CODE	Two-digits code identifying the client starting the connection. It ranges from 01 to 22
RESIP_CODE	One-digit code identifying the used RESIP provider. It ranges from 1 to 4
SERVER_CODE	Two-digits code identifying the server receiving the request. It ranges from 01 to 22
SUPERPROXY_IP	IP address seen by the client as the destination address of the request
GATEWAY_IP	IP address seen by the server as the source address of the request
TTL	Time-To-Live of the first client packet received by the server

the security risks connected to RESIP IPs. In comparison, we propose a worldwide approach and collection. Furthermore, we perform a much deeper analysis of the ecosystem.

In our past work [8] (2022), we have introduced a network measurement-based technique to detect RESIP connections. We have validated it using one of our datasets. In this paper, we perform additional analysis on this dataset focusing on the study of the IP addresses.

Furthermore, in this paper, we propose a novel dataset that has been collected with a brand-new setup with respect to all previous works. Thanks to this dataset, we disclose additional new insights about the algorithms used by RESIPs and propose a new detection method. While the previous technique we discovered ([8]) could detect any RESIP, our new method can recognize two specific RESIP providers. Used in combination with the previous technique, the method can help in attributing campaigns to specific providers.

In 2022, Vastel published in a blog post a machine learning detection method for RESIP connections [32]. This technique is based on the behavioral differences observed when the real owners use a device directly as opposed to when RESIP use the same device as GATEWAY. However, the author proposes only a high-level overview of the behavioral features and this method most likely requires a large dataset of connections to be effective. On the contrary, our method, proposed in Section 5, can identify a RESIP by only analyzing a single request.

3. Methodology and datasets

In this work, we study two large RESIP datasets that we collected through measurement campaigns. We refer to them as MAIN_DS and UNACKED_DS. Both datasets are available to the community on demand. Beyond studying the content of the two datasets, we leverage external datasets to enrich our analysis.

MAIN_DS This dataset was collected in the experiment we described in [8]. We refer to this publication for a detailed explanation of the collection and we recap here the important points to understand and discuss the new results we present in this work.

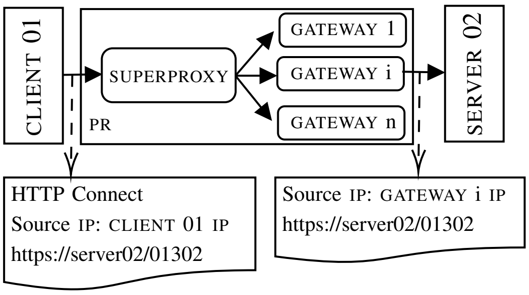


Figure 1: Simplified example of a connection in MAIN_DS.

In our experiment, we have machines that act at the same time as clients and servers. Each client keeps querying all the servers using 4 back connected HTTPS RESIP services, chosen among the most used ones by scrapers: Bright Data[2] (BR), Oxylabs (OL), Proxyrack (PR), Smartproxy (SP). We set the IP of the GATEWAY to be changed for each new request we proxy through a RESIP. We do not impose any localization constraint for the choice of the GATEWAYs.

Each client performs the DNS resolution for the SUPERPROXY domain name and sends a HTTP CONNECT to the obtained IP address. All the requests sent by our clients share the same customer ID. For this reason, it could be possible that the SUPERPROXY and GATEWAY IPs we collect are only taken from a partition of IPs assigned to our account. However, based on the available documentation and the interactions we had with the RESIP providers, there is no indication that this is a practice they follow. Because of this, we can safely assume that our datasets are representative samples of the global pool of IPs belonging to these parties.

We have acquired 22 machines to act as client/server. They are spread all over the world to study the impact on the RESIP functioning of the geolocalization of the source and the destination of requests. Two machines are located in each of the following locations: India, Australia, Japan, Germany, Ireland, Canada, USA (Virginia and Oregon), South Africa, United Arab Emirates and Brazil.

The queried URL encodes information on which client and RESIP provider we use. In this way, it is possible to link, on the server side, each request arriving with a GATEWAY source IP to its originating client and the used RESIP provider. Fig. 1 provides an example of a request from one client to one server.

For each request, we log application layer information as well as network measurements performed on the corresponding packets at the server side. Table 1 presents, with the corresponding explanation, the information we consider in this publication for each logged connection in our dataset.

We have collected data from 12/01/2022 at 15:00 UTC +0 to 01/05/2022 at 15:00 UTC +0 (110 days). In the first 12 days, only 16 client/server machines were active. After that, we had the opportunity to add more 6 machines from a different provider to our setup. We performed this addition to see if different providers would impact our study. For one of the analyzed RESIP providers (BR), we were forced to end the collection 13 days after the

[2] Previously known as Luminati.

TABLE 2: Attributes of a communication in UNACKED_DS.

Attribute	Attribute explanation
COM_START_EPOCH	Epoch (UTC+0) in which the client sends the request and thus starts the communication [s]
COM_END_EPOCH	Epoch (UTC+0) in which the client receives the signal by the SUPERPROXY to end the connection [s]
N_IPS	Number of different IP addresses (distinct GATEWAYs) that send SYN packets during the communication
AVG_SYN_PER_IP	Average number of SYN packets an IP sends in the communication
AVG_SYN_RET_PER_IP	Average number of retransmissions per IP performed in the communication
AVG_DIST_COUNTRIES	Average number of distinct countries associated with the IPs of the communication
AVG_DIST_CONTINENTS	Average number of distinct continents associated with the IPs of the communication
MEDIAN_NEW_SYN	Median delay between SYN packets the same IP issues from different ports [s]
INT_DIFF_IPSS	Elapsed intervals between a request from an IP and the next request with a new IP in the communication
BEHAVIORS_COUNTS	Counters for each type of behavior of an IP: only retransmissions, only sending a new SYN from a new port, doing both actions

beginning of the experiment. Hence, we have a much smaller number of connections from this provider in our dataset (2M+ against 22M+ each for the other providers). More detail about this is available in Appendix A.

In total, we have collected 69,913,009 requests sent by our clients through RESIP services (7.35 connections/second for 110 days).

UNACKED_DS This dataset has been collected with a different setup than the one of MAIN_DS. While studying the RESIP providers, we have noticed specific behaviors when a client uses a RESIP to contact a server that accepts SYN packets but does not answer them. Generally, when the server does not acknowledge a SYN packet, the kernel of the client keeps retransmitting the original SYN packet from the same port using the exponential backoff behavior of the retransmission timeout. This continues until a higher level timeout expires and the client closes the connection [27].

In the case of a connection passing through a RESIP, when a SYN packet is not acknowledged, we see that new GATEWAYs, thus machines different from the original one, starts to send SYN packets to the server. Moreover, every time the client sends a SYN packet, each GATEWAY can act in one or both of the following ways:

1) the kernel of the GATEWAY resends the SYN packet using the same port with exponential backoff, as in the normal case
2) the GATEWAY opens a new connection to the server from a different port and sends a new SYN packet to the server

We have studied in depth this scenario to characterize the RESIP specific behaviors.

To achieve this result, we consider two machines, SERVER_A and SERVER_B, located in Ireland. Contrary to the previous setup, preliminary analysis showed that the geolocalization of the clients and servers does not influence the behavior of RESIP when contacting a non-acknowledging server. Thus, for this experiment, we selected only one location for our machine among the available ones.

SERVER_A and SERVER_B are reachable from the public Internet on port 80 but never reply to SYN packets. They do not generate ICMP error messages either.

In this setup, SERVER_A tries to send HTTP GET requests to SERVER_B through a RESIP provider. Every time the client sends a request, it waits until the SUPERPROXY closes the connection. After that, it sends a request using another RESIP provider to be tested with this setup. On the contrary, we do not send any requests to SERVER_A. Thus, SERVER_A only receives SYN packets from external sources, if any.

When SERVER_B receives a SYN packet, it does not have information about the real initiator of the request (the machine behind the RESIP). Indeed, in the previous dataset, we encoded this information in the URL. In this new setup, we do not have this possibility since we send just one SYN packet. Moreover, SERVER_A does not have knowledge of which GATEWAY IPs the SUPERPROXY assigns to its outgoing requests. Hence, there is no direct way to match the requests sent by SERVER_A with the ones received by SERVER_B. Furthermore, SERVER_B is publicly reachable and scanning campaigns most likely produce some of the SYN packets received by it. To keep only the connections originating from RESIP GATEWAYs that reach SERVER_B, we can use the traffic that reaches SERVER_A as a reference. SERVER_A and SERVER_B share the same exact location. Hence, we can assume that they are frequently scanned at the same time by the same campaigns or, at least, that they witness such scans at a similar rate.

For 88 days (03/02/2022 at 09:00 UTC +0 to 01/05/2022 at 09:00 UTC +0) we recorded connections to SERVER_A and SERVER_B. We queried SERVER_B using three RESIP providers: Oxylabs (OL), Proxyrack (PR) and Smartproxy (SP). In total, we recorded 9,219 incoming connections to SERVER_A and 1,773,407 incoming connections to SERVER_B.

As explained above, we can see the connections to SERVER_A as a reference for the scanning activity on SERVER_B. We can then use them to clean the connections to SERVER_B and keep only the ones produced by the GATEWAYs. For each connection performed to SERVER_B, if the same IP address contacted SERVER_A in the same hour, we eliminate the entries associated with that IP address and time from the logs of SERVER_B and SERVER_A. These are very likely SYN packets generated by scanners, not by RESIPs. Thanks to this operation,

TABLE 3: IPs distribution statistics and repetitions per provider in MAIN_DS.

RESIP	# connections	# countries	# /32	# /24	# /16	# /8	# ASes	Repeated IPs	Repeated IPs per server	Repeated IPs per client
BR	2,413,405	226	1,546,886	712,274	23,274	193	17,026	31%	3±1.6%	3.3±1.8%
OL	22,387,788	226	6,660,452	846,165	15,230	194	19,370	49%	16.3%±0.5%	16.3%±1.3%
PR	22,523,876	234	3,982,149	411,949	14,145	201	9,871	61%	23%	23.4%±0.2%
SM	22,353,578	224	6,852,898	859,946	15,288	194	19,501	49%	15.7±0.4%	15.7%±0.4%

we can delete 1,666 connections from SERVER_B logs (0.09% of SERVER_B connections) and 1,840 connections (19.96% of SERVER_A connections).

Moreover, we can consider the number of connections per hour received by SERVER_A and by SERVER_B in these cleaned datasets. If the two values were similar, it would mean that the amount of scanning activities is comparable to the number of requests sent by GATEWAYs and thus it would be impossible for us to distinguish the two contributions.

On average, the connections to SERVER_A corresponds to 0.43% of the connections received by SERVER_B. This confirms that RESIP GATEWAYs, and not scanners, produced the vast majority of connections received by SERVER_B. Even if a small number of requests at SERVER_B are scanning ones, they are lost in the noise of the requests generated by SERVER_A and we can ignore them. Thus, this clean dataset enables us to have a good representation of the behavior of RESIP GATEWAYs when the TCP connection can be initiated but not completed.

For each connection initiated by our client, there are many SYN packets received at the server from different GATEWAYs. We match each connection started at SERVER_A with all the connections at SERVER_B that were received between when SERVER_A started the request and when the SUPERPROXY ended the request. We define this as a communication. In total, this dataset, which we call UNACKED_DS, contains 124,865 communications. This value corresponds to the number of connections initiated by SERVER_A. On SERVER_B, it corresponds to 1,734,351 incoming SYN packets. Table 2 shows the attributes we consider for each communication in the UNACKED_DS and the corresponding explanation.

GEOLOCALIZATION_DS and FINGERPRINTS_DS

We study the geo-localization of the IPs we collected thanks to the MaxMind GeoLite2 databases [19]. These databases contain information about the country, city and Autonomous System (AS) of IPs. We unify all the information for our IPs in the three databases into one called GEOLOCALIZATION_DS.

In MAIN_DS, we collect the TTL of each client connection and we use it to characterize the OSes of GATEWAYs, which should be residential devices, among different providers. To associate each TTL to the corresponding OS, we take advantage of FINGERPRINTS_DS. This dataset links TCP/IP fingerprinting information with the corresponding OS. It was built in [15] thanks to measurements performed in an academic wireless network and, thanks to this, it characterizes many classes of residential devices. The dataset is available at [16].

In the following sections, we present the analyses and results we obtained studying the above-mentioned datasets.

4. MAIN_DS analyses

4.1. General statistics

In this section, we study the connections collected in MAIN_DS. During the 110 days of collection, we registered 22M+ connections for OL, PR, SP, as shown in Table 3. BR count is lower than the other entries due to the discontinuation of the service imposed by the company (see Appendix A). Moreover, Table 3 shows the wide distribution of the IPs in terms of countries, ASes and subnets. We notice that BR shows high variability even if we have fewer connections from this provider. This tells us that potentially its real distribution presents even higher values. For PR, instead, we see the lowest values in terms of /32, /24 and /16 subnets count and AS variability. At the same time, it presents IPs in more countries than the other providers and it shows the highest value in /8 subnets diversity. This shows that this provider has a better presence on the global scene but lower shares in each region.

Thanks to the analysis of the above-mentioned connections, we introduce some novel findings into the inner working of RESIP providers. These findings, numbered between F1 and F8, are discussed in the next sections.

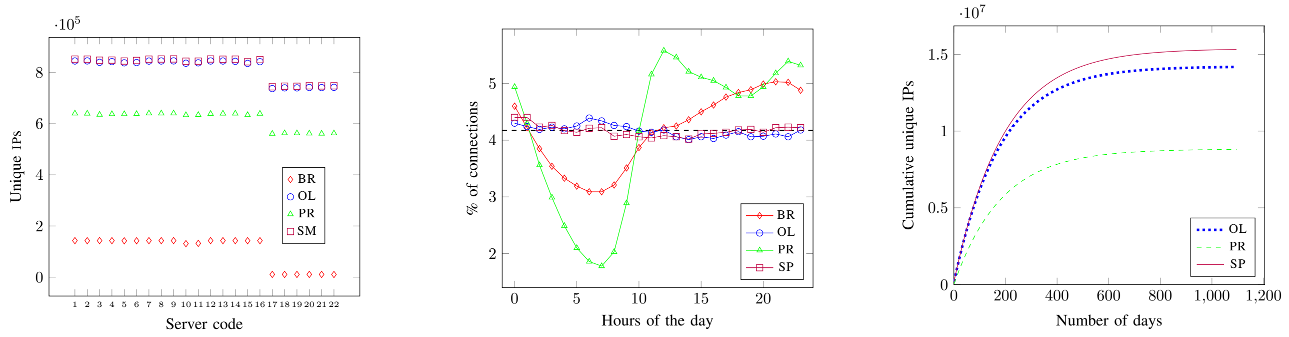
4.2. F1: Assignation of GATEWAYs to minimize repetitions per path

Fig. 2a shows the amount of unique GATEWAY IP addresses registered by each server for every provider.

On the X-axis, we see the codes (1-22) of our servers. On the y-axis, we find the count of unique IPs. Each mark represents the count of unique GATEWAY IPs seen by a server for a specific provider. Codes 17-22 correspond to the machines added in the second phase of the experiment. They have received fewer requests than the others and this is reflected in the count of unique IPs.

In general, we can see that, for each provider, the amount of unique IPs contacting each server is constant. This shows an inner strategy of RESIP providers. These parties try to assign unique GATEWAY IPs to each server with the same proportion.

Moreover, we can notice that if we sum the amount of unique IP addresses registered at each server for a specific provider we obtain much higher values than the total amount of unique IPs (# /32 in Table 3) seen for that provider across all servers. For instance, looking at

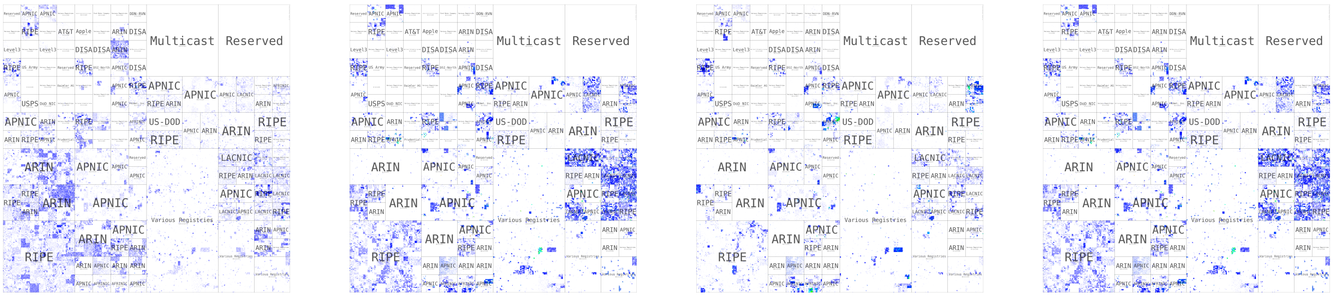


(a) Unique GATEWAY IPs registered by each server.

(b) Diurnal patterns of GATEWAYs connections.

(c) Projection in time of new unique GATEWAY IPs.

Figure 2: Characterization of GATEWAYs in MAIN_DS.



(a) BR

(b) OL

(c) PR

(d) SP

Figure 3: Hilbert curves of the GATEWAYs IPs in MAIN_DS.

PR, the average number of unique IPs in Fig. 2a is around 600,000. Multiplying this value by the number of servers (22) we obtain 13,200,000 which is much higher than the number of unique IPs seen for the same provider in the whole experiment (3,982,149).

Examining the count of unique IPs per path (combination of client-server), we can see that the number of unique IPs per path is 4971±4252, 45526±4247, 43164±3929, 45552±4252 for, respectively, BR, OL, PR and SP. If we multiply these values by the total number of paths (22*22), we obtain again values much bigger than the ones in the fourth column of Table 3 (e.g. for OL, 22,034,584 instead of 6,660,452).

These results tell us that RESIP providers increase the variability of IP addresses for a single path, and try to optimize their stealthiness by giving to each client a new IP for each of its requests, even if it queries a distinct server. At the same time, they reuse much more often the same IPs for other client-server combinations.

Moreover, the statistics about the repetitions of IPs, as shown in the last three columns of Table 3, validate this idea. We define a repetition when an IP address shows up in at least two connections. The third last column shows the percentage of repetitions with respect to the number of unique GATEWAY IP addresses.

PR presents a repetition percentage higher than 60%. However, it is in line with the fact that we almost reached the advertised size of its pool (4M). OL and SP percentage are similar and set to around 50%. In this case, since the claimed pool sizes are much bigger (100M and 40M respectively), we would have expected a lower percentage of repetitions.

The second last column of Table 3 displays the average and the standard deviation percentages of repeated IPs per server. In the last column of the table, we can see the same statistics per source client. These percentages are much smaller than the ones obtained considering all machines. Furthermore, the standard deviation is low, telling us that the frequency of IP repetitions is stable for each machine. We can see the biggest variation for BR. This is due to the company stopping our subscription just after the introduction of new machines. Because of this, the traffic in these machines and the repetition rate are lower and it influences the standard deviation.

The percentage of repetitions per path is even smaller. The obtained values are 0.2±0.1%, 1.5±0.3%, 6%±0.3%, 1.3±0.1% for, respectively, BR, OL, PR and SP.

This data, shows, once again, that RESIP providers, try to minimize the usage of the same GATEWAY IP for a single path. This is an incentive for the clients of RESIP services to use only one machine to send out many requests to one (or more) server(s) they want to contact. In this way, they maximize the IPs used for their requests and complicate the detection on the server-side.

4.3. F2: Non correlation between GATEWAY and destination server locations

In our setup, we have two machines per location. If GATEWAYS were chosen to minimize the additional distance between client and server introduced using a RESIP, we could expect higher percentages of repetitions of GATEWAY IPs between servers in the same location than among other ones. Indeed GATEWAYS close to the two machines should be chosen more often, increasing repetitions and diminishing the pool size of the provider seen by those servers.

However, examining the unique IPs per provider per couple of servers, we see that the percentage of repetitions is similar in all combinations. This suggests that RESIP providers do not choose a subset of GATEWAYs close to the destination server location.

4.4. F3: Non uniform distributions of GATEWAYs

As briefly mentioned in [8], we have discovered that the GATEWAYs of the studied RESIP services are not uniformly distributed around the world. Figure 3 shows the Hilbert curves [34], a continuous fractal space-filling curve, for the GATEWAY IP addresses of every single provider. We have created the curves with ipv4-heatmap [33]. Each pixel represents a single /24 block. The color of each pixel depends on the number of addresses of that block that we have collected during the experiment. A white pixel means that none of our IP addresses is in that /24 block. Colored pixels tell how many IPs are in the block. Colors range from blue (1 IP) to red (256 IPs).

The curves are annotated with IANA labels. We can see in the top right, the Multicast, and Reserved blocks. On the top left, we can notice the blocks that were assigned to private companies before Regional Internet Registries (RIRs) became in charge of IPv4 allocation. The majority of the rest of the blocks are divided among the world's five biggest RIRs: ARIN [4], RIPE [25], LACNIC [14], APNIC [3], AFRINIC [1].

We can clearly see that the majority of BR GATEWAYs are situated in the areas controlled by ARIN and RIPE. On the other hand, the distribution of PR GATEWAYs have peaks in AFRINIC. Most of the GATEWAYs of OL and SP are in LACNIC and RIPE registries. Moreover, these two providers exhibit similarities in terms of the respective Hilbert curves.

These data show that single providers do not have uniform distributions of their GATEWAYs around the world. Furthermore, it shows how SP and OL distributions are comparable, while the ones of PR and BR differ from each other and from the two previously mentioned ones.

4.5. F4: Different management of SUPERPROXYs among providers

The SUPERPROXY is identified through a domain name. Analyzing the SUPERPROXY IPs obtained from the Domain Name System (DNS), we see that PR SUPERPROXY is identified with the same 2 IP addresses from all locations. OL domain name resolution results in 18 IP addresses. 17 of them are shared among all our clients. One IP is observed only by both our machines in India and Canada and one among the two we have in India, Australia, and Tokyo. For SP, the domain name is resolved with 11 different IPs, seen by all our clients.

BR presents a different scenario. The total number of IPs is 5,603. For each machine, the resolution gives 3,539 IPs on average. The maximum number of shared IP addresses is 4,177. We have checked if BR is using its GATEWAYs network to play the role of SUPERPROXY as well. The data of our experiment reject this hypothesis since there is no intersection between the GATEWAYs and SUPERPROXYs sets of IPs of BR.

TABLE 4: Shared IPs among providers (MAIN_DS).

	BR	OL	PR	SP
BR	-	9%	5%	9%
OL	2%	-	8%	63%
PR	2%	13%	-	13%
SP	2%	61%	7%	-

From these results, we can see that BR has a more distributed network of SUPERPROXYs, while the other providers have a small number of addresses for this component. This lets us think that the infrastructure of OL, PR and SP is more similar with respect to the one of BR.

4.6. F5: OL and SP most likely share part (or all) their pools of GATEWAY IP addresses

Table 4 shows, for each couple of providers, the percentage of IP addresses shared by the two. Each cell represents the amount of IP addresses shared by the provider of the line and the one of the columns with respect to the amount of unique IP addresses of the provider of the line. PR shares more than 10% of its IP addresses with OL and SP. Moreover, OL and SP share more than half of their pool.

It is possible that the software of different RESIP services is installed on the same device and that, thus, the same IP address ends up being part of more than one RESIP pool. It is also possible that distinct devices behind a NAT can run the RESIP software of different providers. In this case, the providers share the same (NATed) IP address. Moreover, as previously said, the same IP can be used by distinct devices on different days. However, it is highly unlikely that these scenarios account for more than half of the IP addresses of these two providers. Our intuition is that SP and OL share a significant part of (or all) their pools of addresses, be it knowingly or not.

It would not be the first time that two providers shared a consistent part of their IPs. In the first RESIP study [20], the authors display how two other providers (Geosurf and Proxies Online) have major intersections of their pools. Moreover, in [35], the authors discover a high correlation among the pools of three Chinese RESIP providers. Thanks to further investigations, they disclosed that the three services are controlled by the same underlying operator.

Nowadays, there are more and more RESIP providers on the market claiming to have access to tens of millions of residential IPs. The above-mentioned data shows that even if these sizes were real, some pools of IPs are not uniquely used by a single provider. If this was true on a large scale, the global pool of RESIP IPs would be much smaller than the sum of the claimed numbers. This would be significant from a detection point of view. It would dramatically decrease the number of IPs to possibly detect and it would open the door to the use of blocklisting detection techniques.

TABLE 5: Distribution of GATEWAYs with respect to the initial TTL value and the associated OSes (MAIN_DS).

RESIP	Linux, Ubuntu, Android, MAC OS X, iOS, Solaris, openBSD, Debian (TTL = 64)	Windows, Windows Phone, Android (TTL = 128)	BlackBerry (TTL = 255)
OL	97.23%	0.51%	2.26%
PR	6.33%	92.79%	0.88%
SM	97.27%	0.51%	2.22%

4.7. F6: GATEWAYs of different providers support different OSes

As explained in Section 3, we have collected the Time-To-Live (TTL) of the first client TCP packet of each connection. The TTL is a value initially set by the OS kernel and included in every TCP packet. This value is decremented by one per each network element that it crosses on its path between sender and receiver. It was designed to prevent endless routing of packets.

This value has been widely used to perform passive fingerprinting in combination with other parameters [2], [15], [23], [36]. In this context, we use it to study the distribution on different OSes of the RESIP GATEWAYs of different providers. This is the only information that the TTL alone can give us.

To reconstruct the original TTL set at the sender, we round the value observed in a connection to the next higher power of two, as suggested by Lippman et al. [17], starting from 64. Table 5, shows the distribution of the original TTL of the three studied providers for a representative sample of 14 days of the data in MAIN_DS (14/04/2022-28/04/2022). Each initial TTL value is associated with the OSes found in FINGERPRINTS_DS[3] for that value.

We can see that, for all providers, the usage of BlackBerry OS (TTL = 255) is very limited. On the other hand, the distribution among the other TTL values differs among the three. Most of the GATEWAYs of PR have an initial TTL of 128, while the vast majority of GATEWAYs of OL and SP have the TTL set to 64. This shows that OL and SP providers have access to similar categories of devices while PR takes advantage of different types of machines.

4.8. F7: Diurnal patterns in the GATEWAY availability depend on provider

For each received connection, we use the GEOLOCALIZATION_DS to localize the GATEWAY that sends the request. Combining this information with the CLIENT_EPOCH (UTC+0) we can determine the time zone in which the GATEWAY operated and from that, we can retrieve the corresponding local time.

In Fig. 2b we can see the distribution of the number of requests with respect to the local time. On the x-axis, we have the hour of the day, and on the y-axis the percentage of connections sent by the GATEWAY in that local time, for each RESIP provider. A dashed line represents 4.17%, which should be the value of the percentage if the connections are equally distributed over the 24 hours. We can see that OL and SP present values around the dashed line, with a smaller prevalence of usage during the first hours of the day. On the other hand, BR and especially PR show to use devices as GATEWAY much more frequently in the second half of the day with respect to the first half. This second daily trend has been also found in [20] for all the studied RESIP GATEWAYs.

This shows that not all RESIP providers use the same strategies. Some of them use devices available at any time of the day. Others, instead, take advantage of different classes of devices whose availability has a different diurnal pattern (e.g. mobile phones could be turned off at night while desktop computers could remain available).

4.9. F8: Advertised IP pool sizes do not correspond to our observations

In our previous work [6], we collected a small dataset of 13,897 IPs used by scrapers against a major IT provider for the airline industry. Studying the reputation of such IPs we arrived at the conclusion that they were part of a RESIP pool. We studied the repetitions among those IP addresses. Our findings suggested that the size of the pool from which the RESIP service took those GATEWAY IPs was much smaller than the claimed ones.

We have now used the much larger MAIN_DS to revisit this idea. We have applied the modeling approach based on cumulative curves described in our previous work. We have excluded BR since we have limited connections for this provider and we cannot be conclusive.

As previously explained, there is no 1-1 correspondence between the number of devices and the number of IP addresses used by a RESIP. Thus, this analysis does not tell us the number of devices available to a RESIP provider at any point in time. However, counting the number of distinct IPs in a window of time, as we do, is usually considered to lead to an overestimation of the number of hosts [5]. Thus, our analysis likely provides an upper bound of this value.

For each RESIP provider, we build the cumulative curve of unique new IP addresses per day. This curve displays, for each day, the amount of distinct IP addresses that did requests on that day and were not seen before plus the amount of unique IP addresses seen since the beginning of the experiment. We fit the distribution with an exponentially decaying curve with the equation: $a * (1 - e^{-(x-b)/c})$. This approach, most likely, does not enable us to obtain the exact number of IPs used by RESIPs in time but gives us an idea of the correct order of magnitude of the size of the pool.

For each RESIP provider, we consider the parameters that give us a Pearson correlation factor [28] of 1.0 (total positive line correlation). We project the curves in time to find the value of the plateau. Fig. 2c shows our results. The x-axis represents the days of the experiment, the y-axis the amount of IP addresses.

The results indicate that the plateau for PR pool reaches values above 8M. This contrasts with the information given by the provider, which advertises a pool of half this size (4M). This could mean that they have more IPs at their disposal but they do not have them all available at the same time, or that they count one single IP per

[3] We consider only the measurements with 100% confidence; we are aware that Android appears with 2 distinct values.

TABLE 6: Mean value and standard deviation of attributes in UNACKED_DS.

RESIP	Duration (s)	N_IPs	AVG_SYN_PER_IP	AVG_SYN_RET_PER_IP	AVG_DIST_COUNTRIES	AVG_DIST_CONTINENTS
OL	52.39±97.02	3.18±0.76	3.07±0.85	1.8±1.14	2.87±0.78	2.24±0.71
PR	15.77±5.19	3.27±0.78	1.04±0.15	2.12±0.5	3.08±0.82	2.15±0.75
SM	51.87±96.33	3.98±0.87	2.66±0.7	1.66±0.88	3.61±0.87	2.62±0.77

TABLE 7: Median intervals between requests of different IPs in the same communication (UNACKED_DS).

RESIP	1st-2nd IPs	2nd-3rd IPs	3rd-4th IPs	4th-5th IPs
OL	7s	7s	6s	5s
PR	3s	6s	5s	5s
SM	0s	6s	7s	6s

device they can use, not considering that the device can change its address. OL and SP plateaus are around 14M and 15M respectively. These values largely differ from the available information about the size of the pools (100M for OL and 40M for SP). Considering once more the fact that there is no 1 to 1 correlation between devices and IPs, the providers seem to overestimate the number of IPs they have at their disposal at any moment in time.

The obtained values do not validate the pool size obtained in [6], where we estimated RESIP providers to have a pool of IPs in the low tens of thousands. However, in our previous work, we applied the technique on a server in a unique location, while in this case, we are considering IP addresses from different locations. To perform a more fair confrontation, we have conducted the same analysis also for every server of our infrastructure. Results show that for each server the plateau is bigger than 1M. In particular, we have values per server of 2,604,031±110,460, 1,818,972±33,534 and 2,737,814±145,454 for, respectively, OL, PR and SP. These results confirm that the pool sizes of RESIP providers differ from what they advertise on the Internet. Moreover, this tells us that the IPs in [6] were most likely belonging to a RESIP provider different from the analyzed ones and that that provider had a smaller pool of IPs at its disposal.

5. UNACKED_DS analysis

In this section, we analyze the communications in UNACKED_DS and we share the insights gathered from these analyses. Table 6 shows statistics about the attributes of the communications. It presents for each attribute the average plus or minus the standard deviation. The second column tells the duration, registered at the client, of a communication (COM_END_EPOCH-COM_START_EPOCH). This value informs us about the timeout set in the RESIP to determine when to abandon a connection. PR shows a low value and small variability. This tells us that most likely this interval is fixed for this provider. On the other hand OL and SP show higher values and higher variability.

The third column of Table 6 gives the number of different IPs that produced SYN packets per communication. We can notice that all the providers use on average 3 different IPs to contact the server. This shows an inner protocol of the RESIP providers. Apparently, when a GATEWAY is unable to establish a TCP connection, they believe the problem is with the GATEWAY. Thus, they retry from another machine. The last two columns of Table 6 show the mean value and standard deviation for the average number of distinct countries and continents of the IPs selected for each communication. We can notice that when the RESIP chooses a new machine, the IP belongs generally to a country different from one of the previously used machines for that communication. In most cases, the continent is also different. This indicates that the RESIPs check if there are regional connectivity problems and/or the server is blocking requests from specific locations.

Table 7 shows the median of the INT_DIFF_IPs, the interval of time between an IP contacting our server and the appearance of a new IP in the same communication. We can see that intervals of OL have similar values. For PR, the first interval is roughly half of the subsequent ones, which present similar values. SP shows similar values for the intervals except for the first one, in which the difference is set to zero. In the dataset, the granularity of the intervals is in seconds, so this means that, either the two SYN packets were issued from the provider at the same time, or the interval between them is less than one second. We believe the second to be a more plausible explanation, since sending two SYN packets for each connection would not be efficient for the providers.

These values confirm the idea of an automation process at the RESIP provider. In one case, requests from each new IP are sent after a similar amount of time. In the other two, the interval between the first and second IPs is shorter. Subsequent intervals present constant values.

In the fourth column of Table 6, we find the average number of original SYN packets sent per IP in a communication. This corresponds to the average number of ports a device uses to send SYN packets for that communication. In the fifth column, we see the average number of retransmitted SYN packets for each of the IPs that contacted the server during the duration of the communication. As explained in Section 3, when a server does not send an ACK message, a generic client keeps retransmitting the SYN packet using the exponential backoff behavior. The GATEWAYs of PR appear to behave in this way. The average number of SYN packets is close to 1 and the same packet is retransmitted more than once.

This is not the case for OL and SP. In their communications, a single IP sends SYN packets to our server much more often by opening a new connection from a new port than retransmitting the same packet. Moreover, studying the distribution of the BEHAVIORS_COUNTS attribute, we noticed that there are no cases where an IP only retransmits packets and does not start also a new connection.

The MEDIAN_NEW_SYN parameter tells the median delay between new SYN packets from different ports of the same IP. The median of these values is 1.5s for OL and 1s for SP. In the kernel, the minimum interval before a retransmission is usually 1s. Thus, it is more probable that OL GATEWAYs and not SP ones retransmit SYN packets, as shown by our data (column 5 in Table 6).

The above-mentioned behavior is peculiar. It could be exploited to detect, server-side, when a connection is proxied through one of these two RESIP services or another one that adopts the same retransmission protocol. By delaying the sending of the SYN-ACK message, the server could check if it receives other requests from the same IP that are not retransmissions from the same port. If this was the case, the connection would likely go through such RESIPs and the server could decide to never answer. In this way, on top of the detection, this would provide mitigation. Indeed, the RESIPs would waste resources, having different devices in their network trying to connect, uselessly. The efficiency of this method on the server side remains to be assessed. Furthermore, RESIP could implement a countermeasure by changing the protocol these providers use or modifying the interval of time before sending a SYN packet. Depending on the setup, this could imply costs and/or reduce the RESIP efficiency.

6. Discussion

In this work, we have looked at the RESIP ecosystem from different angles. Hereafter, we show the lessons learned considering all our findings together.

We have seen that **all RESIP providers share part of their implementation and functioning strategy.** RESIP services try to maximize the number of IPs associated with a single client-server path and do not seek to choose GATEWAYs close to the contacted server (Sections 4.2-4.3). All the studied providers try to contact a non-acknowledging server from multiple GATEWAYs located in different areas of the world (Section 5). Finally, as described in our previous work [8], none of the providers breaks the TLS session in between client and server, but they do so for the TCP session.

However, even if we see these common features, we have many indications that **some of the studied parties have fundamental differences among themselves**. Sections 4.4 and 4.7-4.8 show that the providers recruit GATEWAYs with different characteristics (installed OS, availability during the day) and from different areas of the world. In Section 4.5, we can see that RESIP providers have different managements of their SUPERPROXYs. Finally, while all the providers use more than one GATEWAY to send SYN packets to non-acknowledging servers, PR keeps retransmitting packets from one port but OL and SP send SYN packets from newly opened ports (Section 5). These results tell us that even if the providers show similarities, we cannot consider them as one monolithic entity. This complicates the task of finding detection methods able to identify all of them. At the same time, it opens the door for attribution techniques thanks to the detection of specific features of providers, as proposed in Section 5.

Among the different tests we performed, we have noticed that **OL and SP have high similarities in almost all the results**. They share more than half of their GATEWAY IPs and they have similar values in the projection in time of the new unique IPs (Sections 4.6 and 4.9). They show the same distribution of /24 and OSes of their GATEWAYs (Sections 4.4 and 4.7). Neither of them shows a diurnal pattern in the availability of devices (Section 4.8) and they present similar distribution of the differences between TLS and TCP RTTs [8]. Moreover, both of them use a very similar algorithm to send new SYN packets in case they do not obtain a SYN-ACK packet from the server (Section 5). In this last scenario, however, we see that they have shown a difference. The median interval between the first and second GATEWAYs that contact the server is really short for SP (<1s) but not for OL (7s). While we can see striking similarities between these two parties, this difference and the fact that they have different SUPERPROXYs (there is no intersection among the two groups of IPs), does not enable us to say that these two companies are two different names for the same entity. The root cause of these similarities remains an open question.

Finally, **we could implement an approach similar to blocklists to detect RESIP**. We have seen that two out of three providers appear to have pool sizes much smaller than what they claim (Section 4.9). Moreover, Sections 4.4 and 4.6 tell us that two providers use the same /24s and that different providers share a high percentage of IPs. We need to remind that an IP registered on two different days, does not necessarily mean that the same device was part of a RESIP for the whole time between the two days. However, considering the corresponding /24 enables us to take into account devices that change address but remain in the same /24.

Based on this, our idea would be to build lists of suspicious /24. When an IP is seen as part of a RESIP, we mark the corresponding /24 as suspicious. When an IP from a suspicious /24 contacts a server, the server can answer with a more invasive technique, such as delaying the sending of the SYN-ACK packet (Section 5). Since we have seen that IPs are much more often reused for different paths and servers (Section 4.2), a collaborative protocol among different servers to build and share this list would enable to have a better detection.

This approach might not always be effective. As shown by Griffioen et al. [11], there is no general rule among ASes to maintain the previous prefix after reallocation. While some ASes reallocate only in the same /24 for operator policy, others present a more variegated scenario. Because of this, our idea could not work and a real-world measurement is necessary to assess the validity of the proposed approach.

7. Conclusions

In this paper, we provide novel insights about the RESIP ecosystem thanks to our two RESIP datasets. We show different aspects of the RESIP infrastructures, highlighting similarities and differences of the providers in the geographic distribution, types, management and amount of their GATEWAYs, as well as in the management of their SUPERPROXYs. We reveal internal algorithms used in specific situations and we leverage the characteristics of two providers to create a new detection technique for their connections. Finally, we offer arguments in favor of building a list of suspicious /24 blocks to identify RESIP connections.

Data Availability

The collected datasets and the code used to perform the collections are available upon request, except for the connections from Bright Data for which the authors agreed to keep the IPs confidential.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback.

References

[1] AFRINIC. African Network Information Centre. https://afrinic.net/.

[2] J. M. Allen. Os and application fingerprinting techniques. In *SANS Institute InfoSec Reading Room*, 2007.

[3] APNIC. Asia Pacific Network Information Centre. https://www.apnic.net/.

[4] ARIN. American Registry for Internet Numbers. https://www.arin.net/.

[5] L. Böck, D. Levin, R. Padmanabhan, C. Doerr, and M. Mühlhäuser. How to Count Bots in Longitudinal Datasets of IP Addresses. In *Proceedings 2023 Network and Distributed System Security Symposium*, San Diego, CA, USA, 2023. Internet Society.

[6] E. Chiapponi, M. Dacier, O. Catakoglu, O. Thonnard, and O. Todisco. Scraping Airlines Bots: Insights Obtained Studying Honeypot Data. *Intl. Journal of Cyber Forensics and Advanced Threat Investigations*, 2(1):3–28, 2021.

[7] E. Chiapponi, M. Dacier, O. Thonnard, M. Fangar, M. Mattsson, and V. Rigal. An industrial perspective on web scraping characteristics and open issues. In *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S)*, pages 5–8, 2022.

[8] E. Chiapponi, M. Dacier, O. Thonnard, M. Fangar, and V. Rigal. BADPASS: Bots Taking ADvantage of Proxy as a Service. In *Information Security Practice and Experience: 17th International Conference (ISPEC 2022)*, page 327–344, Berlin, Heidelberg, 2022. Springer-Verlag.

[9] J. Choi, M. Abuhamad, A. Abusnaina, A. Anwar, S. Alshamrani, J. Park, D. Nyang, and D. Mohaisen. Understanding the Proxy Ecosystem: A Comparative Analysis of Residential and Open Proxies on the Internet. *IEEE Access*, 8:111368–111380, 2020.

[10] DataDome. Blocking the IP is Not Enough—How to Stop Bots on Residential IPs, 2022. https://datadome.co/bot-management-protection/one-third-bad-bots-using-residential-ip-addresses/.

[11] H. Griffioen and C. Doerr. Quantifying Autonomous System IP Churn Using Attack Traffic of Botnets. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, ARES '20, New York, NY, USA, 2020. Association for Computing Machinery.

[12] A. Hanzawa and H. Kikuchi. Analysis on Malicious Residential Hosts Activities Exploited by Residential IP Proxy Services. In *Information Security Applications*, pages 349–361. Springer International Publishing, 2020.

[13] B. Krebs. The Rise of "Bulletproof" Residential Networks, 2019. https://krebsonsecurity.com/2019/08/the-rise-of-bulletproof-residential-networks/.

[14] LACNIC. Latin America and Caribbean Network Information Centre. https://www.lacnic.net/.

[15] M. Lastovicka, T. Jirsik, P. Celeda, S. Spacek, and D. Filakovsky. Passive os fingerprinting methods in the jungle of wireless networks. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, 2018.

[16] M. Lastovicka, T. Jirsik, P. Celeda, S. Spacek, and D. Filakovsky. PassiveOSFingerprint, 2018. https://github.com/CSIRT-MU/PassiveOSFingerprint.

[17] R. Lippmann, D. Fried, K. Piwowarski, and W. Streilein. Passive Operating System Identification From TCP / IP Packet Headers. In *Proceedings Workshop on Data Mining for Computer Security (DMSEC)*, 2003.

[18] Frappier M, P. Plante, and G. Joly. Illegitimate residential proxy services: the case of 911.re and its IOCs, 2022. https://gric.recherche.usherbrooke.ca/rpaas/.

[19] MaxMind. Maxmind geolite2. https://www.maxmind.com/.

[20] 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)*, pages 1185–1201, 2019.

[21] 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 *Proc. of NDSS 2021*, 2021.

[22] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage. Re: CAPTCHAs: Understanding CAPTCHA-Solving Services in an Economic Context. In *Proc. USENIX Security 2010*, page 28, USA, 2010.

[23] O. A. Osanaiye and M. Dlodlo. TCP/IP header classification for detecting spoofed DDoS attack in Cloud environment. In *IEEE EUROCON 2015 - International Conference on Computer as a Tool (EUROCON)*, pages 1–6, 2015.

[24] R. Padmanabhan, A. Dhamdhere, E. Aben, kc claffy, and N. Spring. Reasons Dynamic Addresses Change. In *Proceedings of the 2016 Internet Measurement Conference*, IMC '16, page 183–198, New York, NY, USA, 2016. Association for Computing Machinery.

[25] RIPE. Regional Internet Registry for Europe. www.ripe.net.

[26] M. Ryuichi, K. Takumi, F. Hikari, and K. Hiroaki. Investigating Potential Malicious Activities via Residential IP Proxy Services. *Research Report Computer Security (CSEC)*, 2023-CSEC-100(59):1–8, February 2023.

[27] W. R. Stevens and K. R. Fall. *TCP/IP Illustrated, Volume 1: The Protocols (Second edition)*. Addison-Wesley Longman Publishing Co., Inc., USA, 2012.

[28] S. M. Stigler. Francis Galton's Account of the Invention of Correlation. *Statistical Science*, 4(2):73–79, 1989.

[29] A. Tosun, M. De Donno, N. Dragoni, and X. Fafoutis. RESIP Host Detection: Identification of Malicious Residential IP Proxy Flows. In *2021 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6, 2021.

[30] W. Turton. Russian hackers used home networks to evade detection, 2021. https://www.bloomberg.com/news/articles/2021-10-26/suspectedrussian-hackers-use-home-networks-to-evade-detection.

[31] A. Vastel. Ever wonder how proxy providers & BaaS providers obtain residential proxies?, 2022. https://datadome.co/bot-detection/how-proxy-providers-get-residential-proxies/.

[32] A. Vastel. How to Use Machine Learning to Detect Residential Proxies, 2022. https://datadome.co/bot-management-protection/how-to-use-machine-learning-to-detect-residential-proxies/.

[33] D Wessels. ipv4-heatmap, 2011. https://github.com/measurement-factory/ipv4-heatmap.

[34] Wikipedia. Hilbert Curve. https://en.wikipedia.org/wiki/Hilbert_curve.

[35] M. Yang, Y. Yu, X. Mi, S. Tang, Y. Guo, S. 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*, 2022.

[36] M. Zalewski. p0f v3, 2016. https://lcamtuf.coredump.cx/p0f3/.

A. Ethical considerations

In our study, EURECOM legitimately subscribed to RESIP services to access their networks of the GATEWAYs. Subscriptions were made and paid online. For three out of for providers, there were no checks and, upon payment, we could start to use directly their RESIPs. On the other hand, BR asked us to participate in a recorded interview in which we had to explain the motivation for our work and how we wanted to use their infrastructure. We stated that our goal was to test connections between our clients and our servers using their infrastructure and that we would not disclose their IPs. Moreover, we communicated we were collaborating with a third-party organization that preferred to not be named. At this stage, BR agreed for us to perform our experiment and gave us access to the residential IPs.

However, 13 days after the beginning of the experiment, they paused our subscription. The shared motivation was that our scenario (targeting our own machines) could "expose their users IPs, which can become a privacy issue". They told us that we would need to disclose the name of the company that was collaborating with us.

We never misrepresented to BR what we were doing and we did exactly what they agreed with us in the first place. The organization collaborating with us is a real victim of bot scrapers. Disclosing its name could have triggered some reaction from the provider which could have biased our results. This is why we decided to not accept to disclose its name. At that point, they completely stopped our subscription and refunded it.

In our analyses, we discover inner functionalities of RESIP providers. The reader may wonder if we contacted the tested companies to comment on our results, especially the real size of their pools. We did not do so because we do not expect them to deny a piece of information publicly advertised on their websites. Moreover, we are customers of their services. Hence, it would not be legally advisable for them to admit that their marketing information is incorrect.